A STUDY OF HIGH-SPEED MULTIPLIERS

by

W. A. Klein

K. Olsen

A STUDY OF HIGH-SPEED MULTIPLIERS

by

W. A. Klein

DIGITAL COMPUTER LABORATORY

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

CAMBRIDGE 39, MASSACHUSETTS

JULY 1954

ACKNOWLEDGEMENTS

The study which this paper reports was made by J. F. Jacobs, R. C. Jeffrey and R. P. Meyer. S. L. Thompson contributed to the design of the acs D multiply-adder. The contribution of the author was the reorganization of the material of the study; the filling in of certain details, such as the proof that no carry overlap occurs in the multiply-adders which employ partial carry operations; and the extension of the study, such as the development of the Class II multipliers and the extension of the development of the Class III multipliers by inclusion of a second variable, p.

ABSTRACT

The origin of this study of high-speed multipliers, the results of which are recorded in this report, is discussed Chapter I. Two measures to be used in comparing arithmetic elements are also given here. In Chapter II the logical properties of the circuits envisioned for use in the realization of the multipliers are discussed; and symbols, values of time delays and equipment weights for the abstracted logical blocks are given. Certain drawing conventions, etc., are also listed. Chapter III gives a general framework within which all the multipliers presented can be discussed and results in a classification of them. This framework is illustrated with reference to the Class I multipliers. The purpose of Chapter IV is to catalogue the realizations of the Class I multipliers and to discuss their special features. Breakdowns of the equipment needed and estimates of the multiplication times are also given in this chapter. An explanation of the operation of the Class II and Class III multipliers will be found in Chapters V and VI. Specific examples of multipliers of these two classes are also discussed here, and their equipment counts and multiplication times are quoted. Furthermore, a definition is chosen for the meaning of "an optimum multiplier", and that multiplier of those studied in Chapter VI which satisfies the definition is specified. A comparison of most of the multipliers presented in the report is made in Chapter VII, and the overall optimum multiplier when defined as in Chapter VI is found. In addition some consideration is given to adapting the multipliers discussed to handle negative numbers and to perform addition, subtraction and division.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

The purpose of this report is to record and extend the results of a study of high-speed multipliers undertaken as a part of the design of a high speed, high reliability arithmetic element (a device for performing various arithmetic operations upon coded numbers). This problem was approached in the following manner:

All the available designs which appeared to possess the desired characteristics were to be considered in detail, and the "best" of these designs was to be chosen for the final system. (In the interest of restricting the number of the designs to be considered, two general statements can be made immediately: (1) High speed demands that only parallel systems to be considered. (2) If the basic circuits are to be required to distinguish only two states, then conservation of equipment and logical simplicity require that the system handle numbers in binary rather than decimal form.) As an aid to determining which was the "best" among the designs considered, two numerical measures were defined which would predict the speed and the relative reliability of any specific design. These are discussed in the two following sections.

1.1 A MEASURE OF SPEED: THE SINGLE ADDRESS INSTRUCTION TIME

The instructions proposed for the computer which would use this arithmetic element can be broken down into three groups according to whether they require one, two or three memory cycles.[1] For example: (1) "Shift right n digits" (sr n) requires only one access to the memory, namely that to extract the instruction sr n, and therefore requires only one memory cycle. (2) The "multiply" instruction requires two memory accesses: One to extract the instruction and one to extract the number to be operated upon. (3) The left store" instruction requires three memory cycles: one to extract the instruction, a second cycle during which a parity check and a new parity count[2] are being made, and a third during which the actual store operation takes place.

The instructions within each of these three groups can be classed according to the length of time which the arithmetic element requires to perform them; for instance, the instructions add, subtract, clear and add, and clear and subtract require approximately the same length of time for execution and hence are placed in the same class. The classes can be numbered consecutively from the first class of the first group to the last class of the third group.

---

1. A memory cycle must occur whenever a number or an instruction is extracted from or inserted into memory. A memory cycle requires a fixed amount of time.
2. Parity checks and parity counts are means for the computer to check the correct transfer of numbers to and from the memory. See.....

The definition of the single address instruction time (SAIT), which is to serve as a measure of the speed of an arithmetic element, is then as follows:

$$SAIT = \sum_{i=1}^{m} P_i (M + O_i) + \sum_{i=m+1}^{m+n} P_i (2M + O_i) + \sum_{i=m+n+1}^{m+n+p} P_i (3M + O_i)$$

where m is the number of classes in the group of instructions which require one memory access.

n is the number of classes in the group of instructions which require two memory accesses.

p is the number of classes in the group of instructions which require three memory accesses.

M is the memory cycle time. This is the minimum time which can elapse between the initiation of the extraction of successive words from the memory.

P_i is the fraction of the instructions in a typical program which fall in the class i. This fraction cannot be found by simply determining the frequency of occurrence of the instructions in the written program, but must take account of the possible repetition of certain sections of the program either as subprograms or as conditional subprograms.

O_i is the instruction overshoot time. It is the extra time, if any, required for the execution of any instruction of the class in question beyond the

end of the time required for the memory cycle or cycles.

1.2 A MEASURE OF RELIABILITY: THE WEIGHTED EQUIPMENT COUNT

Computers are not 100% reliable because of sudden failures or gradual deterioration of their physical components. Therefore, a measure of reliability must relate to the physical components of the system.

The effects on the reliability of an arithmetic element due to gradual deterioration of the physical components can be greatly reduced with the aid of marginal checking.[3] Moreover, conservative use of the physical components tends to prevent their over rapid deterioration and further stops this effect from adversely affecting the reliablility.

The occurrence of sudden failures cannot be avoided, but their frequency can be decreased by careful choice of components and by minimizing the number of components most subject to failure (tubes, diodes) which are necessary to construct the system. Thus if it can be assumed that the systems whose reliabilities are to be compared are constructed with carefully chosen and conservatively used components, then an approximate relative measure of reliability can be obtained by taking an equipment count. In order to make such an equipment count, the various circuits used in the realization must be assigned weights, and then the count is simply the sum of the weights of all the circuits in the system.

---

3. The truth of this statement has been established by experience with the Whirlwind I computer. See Digital Computer Laboratory Engineering Note E-536.

The problem of weight assignment is one which requires careful exercise of engineering judgement. The weights assigned to the various circuits used to realize the multipliers discussed in this report are given in Table II-1.

1.3 THE DECISION TO CONCENTRATE ON A STUDY OF MULTIPLIERS

The SAIT for the first few of the arithmetic elements proposed was calculated using a memory cycle time to be expected from a magnetic core memory. It was found that the instruction overshoot time (O_1) was approximately zero for all except the classes containing the multiply and divide instructions (these classes consisted of single instructions). Therefore, the SAIT could be decreased only by the discovery of systems which lowered the overshoot time of the multiply and divide instructions while maintaining zero O_i for the other instruction classes. Since P_i for the divide instrution is an order of magnitude smaller than P_i for the multiply instruction (0.01 as against 0.1 in the programs to be used with this computer), it was more important to minimize the overshoot time of the multiply instructions, even though that of the divide instruction might thereby remain constant or even slightly increase.

The design of an arithmetic element thus resolved itself into a study of high-speed multipliers. The remainder of this report will deal with the results of this study, although mention will be made in Chapter VII of methods for converting the multipliers discussed into arithmetic elements.

The study began with the investigation of previously suggested and successfully operating high-speed multipliers. It then expanded to consider ideas for other multipliers. Since the time allowed for the study was limited, only multipliers which upon a cursory examination showed fair promise of being high-speed and economical of equipment were investigated thoroughly. Thus the discussion to follow may suggest to the reader several schemes for the mechanization of multiplication which he will not find included here.

K. Olsen

CHAPTER II

THE LOGICAL BLOCKS

2.1 The logical blocks

The multiplier systems presented in this report were designed starting from certain basic blocks. If these basic blocks could actually be combined in any numbers and in any manner then systems design would be simplified. This is so because it would eliminate the problem of deciding how many, if any, "non-logical" blocks (such as isolating and amplifying circuits) should be used. Furthermore, the circuit delays could then be considered as properties of the circuits themselves. We shall invoke this simplification by formally neglecting the loading effect of one circuit on another as well as the effect of cascading circuits of less than unity gain. However, we shall try not to suggest designs which demand the impossible or require an excessive number of "non-logical" circuits to make good on this assumption.

The logical forms of the multipliers presented in this report are of course a result of the logical blocks available for their construction. These logical blocks are abstractions of forms of commonly known circuits, such as the flip-flop and the vacuum tube gate, which have been specially

developed by the laboratory for high speed and reliability.[1] In the abstraction of the logical blocks from these circuits only the following properties are retained. First, the number and type of inputs and outputs; second, the logical relation of outputs to inputs; and third, certain time delays. These properties, which are of importance to logical design, will be discussed below. Weights were also assigned to the various basic circuits in order to permit the equipment count mentioned in Chapter I. Some amplification of the three properties which affect logical design is required:

There are two types of inputs and outputs or lines, namely level and pulse. The distinction is necessary because of the electrical characteristics of the lines[2]; and these characteristics require that only lines of the same type be interconnected, thus restricting the possible combinations of the logical blocks. The circuitry feeding these lines from

1. The basic circuits whose use is envisioned in the realization of the multipliers presented here will be presented in the Circuit Applications Section of the proposed Military Reference Data Book.

2. The distinction is also necessary for logical reasons since the information carried by level lines is in the form of the state of the whole line (voltage high or low), a state which can be determined at any point along the line and can be retained only by the use of some holding device.

within the blocks has been designed to possess these characteristics as a result of the conception of the multipliers as systems clocked by pulses of short duration (0.1 μsec) and of very frequent occurrence. It happens that usually this requirement on the ability of the driver of an electrical line to handle frequent, short pulses is not compatible in circuit design with the requirement that a circuit be able to maintain an output at a constant level. Therefore, it is best to design different circuits for the different tasks, and with this decision a need for distinction of line types arises.

The notation to be used for pulse and level lines and their interconnections are shown in Figure II - 1(a) and (b). The interconnections indicated by (b) are simple electrical connections; the arrows are simply to indicate the type of line and the normal or logical direction of information flow. They do not imply the presence of crystal diodes. The logical arrangement of the blocks whose outputs are thus connected that the flow of information opposite to the arrow will cause no error in the system.

Specification of the logical relationship between inputs and outputs requires that a code to interpret the values of some electrical property

of the lines as a yes-no decision be specified. The electrical property chosen here is voltage; the logical entities used are the binary digits 0 and 1. The code is:

level lines
- relatively low voltage : 0
- relatively high voltage : 1

pulse lines
- absence of a positive voltage pulse over a specific time interval : 0
- presence of a positive voltage pulse over a specific time interval : 1

The time delays retained are: (1) those which occur between establishment of a particular condition on an input line and the resultant change in the outputs, and (2) those which must exist between input pulses to certain circuits in order that the proper effect upon the outputs is achieved. Any time delay of the first kind depends upon the number and kind of circuits which are connected to the output in question. However, it will be assumed, as was mentioned above, that the interconnections of the circuits are achieved thru the use of isolating circuits so that these time delays can be assigned constant values.

The literal symbols, logical block symbols, associated delays (defined below), and equipment weights for the various basic circuits are included in Table II-1. The label on the block diagram is the key to the

logical relationship between the inputs and the outputs, which are given further labels when necessary. These logical relationships can be expressed in terms of voltages (interpreted logically as in the above code) as follows: (1) The flip-flop block may be considered to have two sides: the "1" side and the "0" side according to the labelled outputs. A pulse input to the "1" side ('set' input) raises the "1" side output and lowers that of the "0" side output. A pulse input to the "0" side ('clear' input) has the opposite effect. A pulse input on the center line ('complement' input) reverses the states of the outputs. A flip-flop will be said to have the 'contents' 0 or 1 accordingly as the 0 or 1 output level is "high". (2) The pulse input to the gate tube circuit will appear as an output pulse only if the level input is "high". (3) The output level from the diode "and" circuit will be "high" only if all input levels are "high". (4) The output level from the diode "or" circuit is "low" only if all input levels are "low". (5) A pulse will appear on the output line from the diode pulse mixer circuit whenever a pulse appears on an input line. (6) The output level from an inverter circuit is "high" if the input is "low" and vice-versa. (7) The delay line unit simply delays the input pulse.

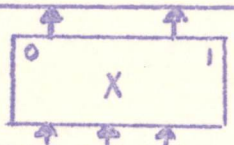
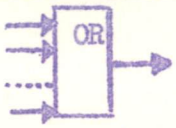
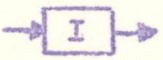
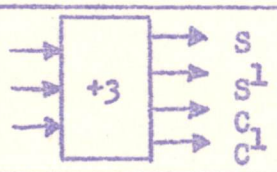
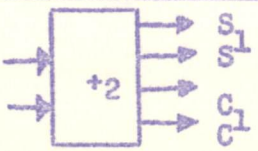
The time delays for these circuits are defined as follows: (1) Two time delays are associated with the flip-flop circuit. The transition time, t, is defined as the time from the occurrence of the peak value of an input pulse till the slowest changing output level reaches a value which differs from its final value by 10% of the total change occurring. (When this value is reached, the level is established.) The flip-flop resolution time, ρ_f, is defined as the time which must be allowed between the occurrence of any two input pulses in order that the second pulse shall have its proper effect upon the outputs. (2) The delay of a gate tube circuit, γ, is the time which elapses between the input pulse and the output pulse (when the input level is "high"). The input pulses must be separated by a resolution time ρ_g. (3) The delay of the diode "and" circuit, α, is the time between the establishment of the input level and the establishment of the output level, when this level is changing. The time α is assumed to independent of the number of inputs. (4) The delay of the diode "or" circuit, ω, is defined exactly as was α. (5) The delay of the pulse mixer circuit, μ, is that experienced by a pulse in passing thru the circuit and is assumed independent of the number of inputs. A resolution time ρ_m is required between input pulses. (6) The delay of

The inverter circuit, i, is the time between the establishment of the input level and the establishment of the output level. (7) The delay of a delay line unit will be symbolized by δ. The unit possesses a resolution time ρ_d.

The values given for the delays of the "and" and "or" circuits in Table II-1 are based on the assumption that the "and" circuit is designed so that a diode never assumes its high resistance value (reverses) when an input level is rising and that the "or" circuit is used so that a diode never reverses when an input level is falling. Thus the voltage waveforms at the outputs of these circuits follow the waveforms at the inputs (when logically they are supposed to) except for an inevitable, very small delay. The values of α and ω are also considered to include the delay which will be introduced when it is necessary to include cathode follower and buffer amplifier circuits. The values assigned to these delays in Table II-1 are debatable, but at the present time they appear reasonable. The value chosen for the delay of the pulse mixer is also questionable, but some value must be assigned. The value given for χ is a conservative estimate. Since the estimates for the delays of these four circuits are so small, they will not be considered in estimating multiplica-

Table II-1

Logical Blocks

CIRCUIT	LITERAL SYMBOL	BLOCK SYMBOL	ASSOCIATED DELAYS (μsec)	Equipment WEIGHT
flip-flop	FF	 X:FF,A,B,C,Ac	$\tau = 0.5$ $P_f = 0.2$	40
gate tube	GT		$\gamma = 0.04$ $P_g = 0.5$	10
input diode "and" gate	δ_k	 (k inputs)	$\alpha = 0.02$	2k
input diode "or" gate	or_k	 (k inputs)	$\omega = 0.02$	2k
input diode pulse mixer	M_k	 (k inputs)	$\mu = 0.01$ $P_m = 0.5$	k
inverter	I		$\iota = 0.1$	10
delay line unit	D		δ = delay $P_d = \delta$ or 0.5 μsec whichever is greater	20
three input adder	+ 3, S^1		-	58
	+ 3			48
two input adder	+2, S^1		-	32
	+2			22
Not present for +3 and +2.				

tion times of multipliers except when they occur in appreciable aggregates.

In addition to the basic circuits discussed above, three logical units composed of several of these circuits occur frequently enough to be assigned block symbols. These are two adder circuits and a step counter circuit. The first adder circuit is capable of summing three binary digits, presented to it coded as voltage levels. It is shown in Figure II-2. The "s" output is the coded right-hand digit of the binary sum, while the "c" output is the coded left-hand digit of the binary sum. The "s^1" and "c^1" outputs are the inversions of the "s" and "c" outputs. The inputs are interchangeable, as would be expected, and hence require no labels. This circuit is known as the three input adder; the literal and block symbols for it and its equipment count are given in Table II-1. Three delays associated with this circuit are of importance. They are the delays from the establishment of an input level to the establishment of: (1) the "c" output level, (2) the "c^1" output level and (3) the "s^1" output level. These delays can readily be expressed in terms of those already given for the basic circuits and are assigned no separte symbols.

The second of these composite circuits is a two input adder and is merely a simplification of the three input adder. It is shown in Figure

II-3; its literal symbol, bolck symbol and equipment weight are given in Table II-1.

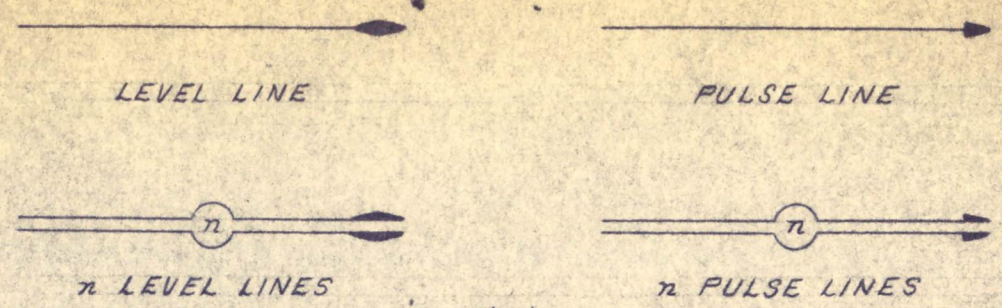
The step counter circuit and its logical block symbol are shown in Figure II-4. A step counter consisting of m flip-flops will provide an end-carry pulse on the kth add-one pulse, where k is any integer which satisfies $1 \leqslant k \leqslant 2^m$ and depends on the setting of the preset toggle switches. The equipment count cannot be fixed in general, but one of the two delays associated with the circuit is independent of the number of flip-flops. This is the time which must be allowed between the occurrence of the preset pulse and the first add-one pulse or between any two add-one pulses. It may be thought of as the step counter resolution time and clearly must be equal to the flip-flop transition time, τ . The delay between the occurrence of the k^{th} add-one pulse and the resultant end-carry output pulse is $m\gamma$.

2.2 Conventions

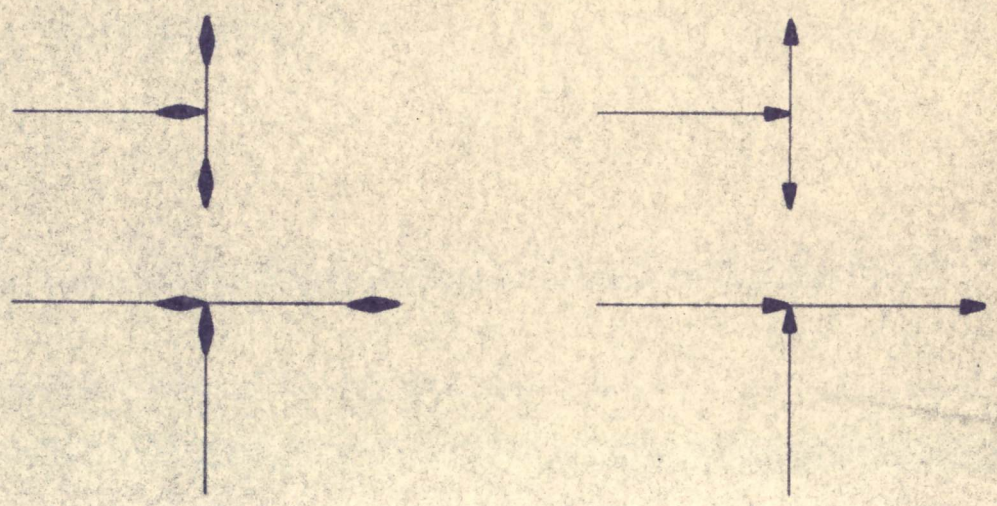
Some conventions which have to do with the drawing of block diagrams are shown in Figure II-1. Pulse and level lines and their interconnections, shown in parts (a) and (b) of this figure, have already been discussed. The adoption of the convention of Figure II-1 (c) reduces the number of

lines and arrow-heads on many block diagrams and thereby gives them a neater appearance.

The letter "n" will be used thmuout the report for the maximum number of digits ("digit length") of the binary numbers to be multiplied. These numbers will always be considered to be of length n, zeros being inserted at the left end of the number when necessary. The digits will be numbered from 1 to n from left to right (most significant to least significant). The binary numbers will be stored in flip-flop registers (a string of flip-flops) in this same order; that is, the leftmost flip-flops in the registers as shown in the block diagrams will contain the most significant digits. The flip-flops of all the registers will also be numbered from left to right.

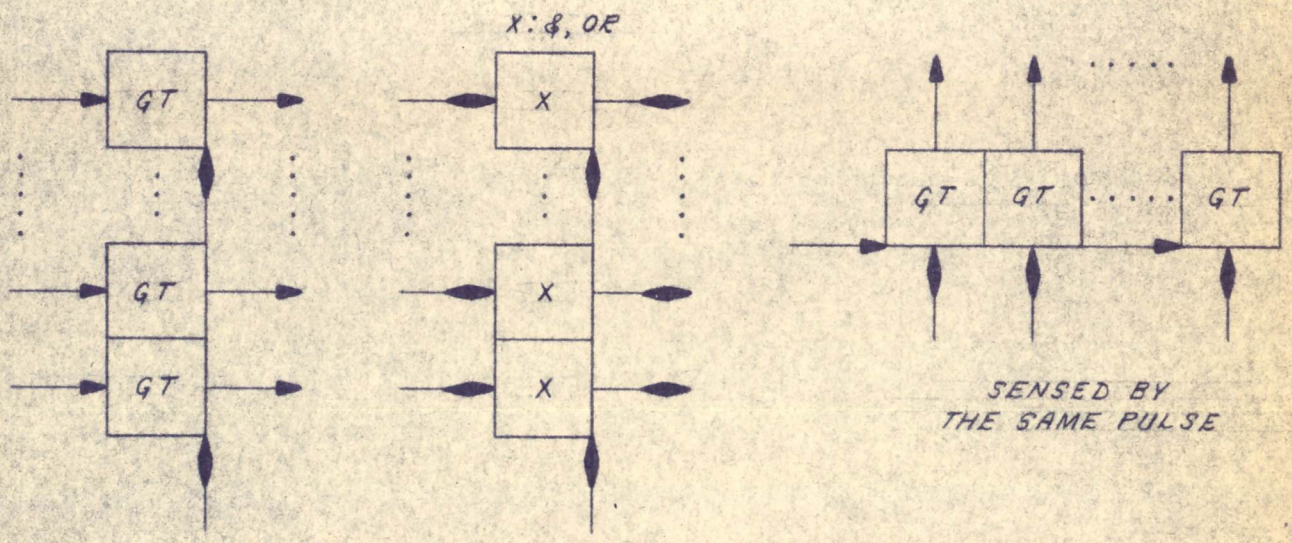


(a)



INTERCONNECTIONS

(b)



CONTROLLED BY THE SAME LEVEL

BLOCK AGGREGATES

(c)

FIG. II-1

CONVENTIONS USED ON THE BLOCK DIAGRAMS

A-57307

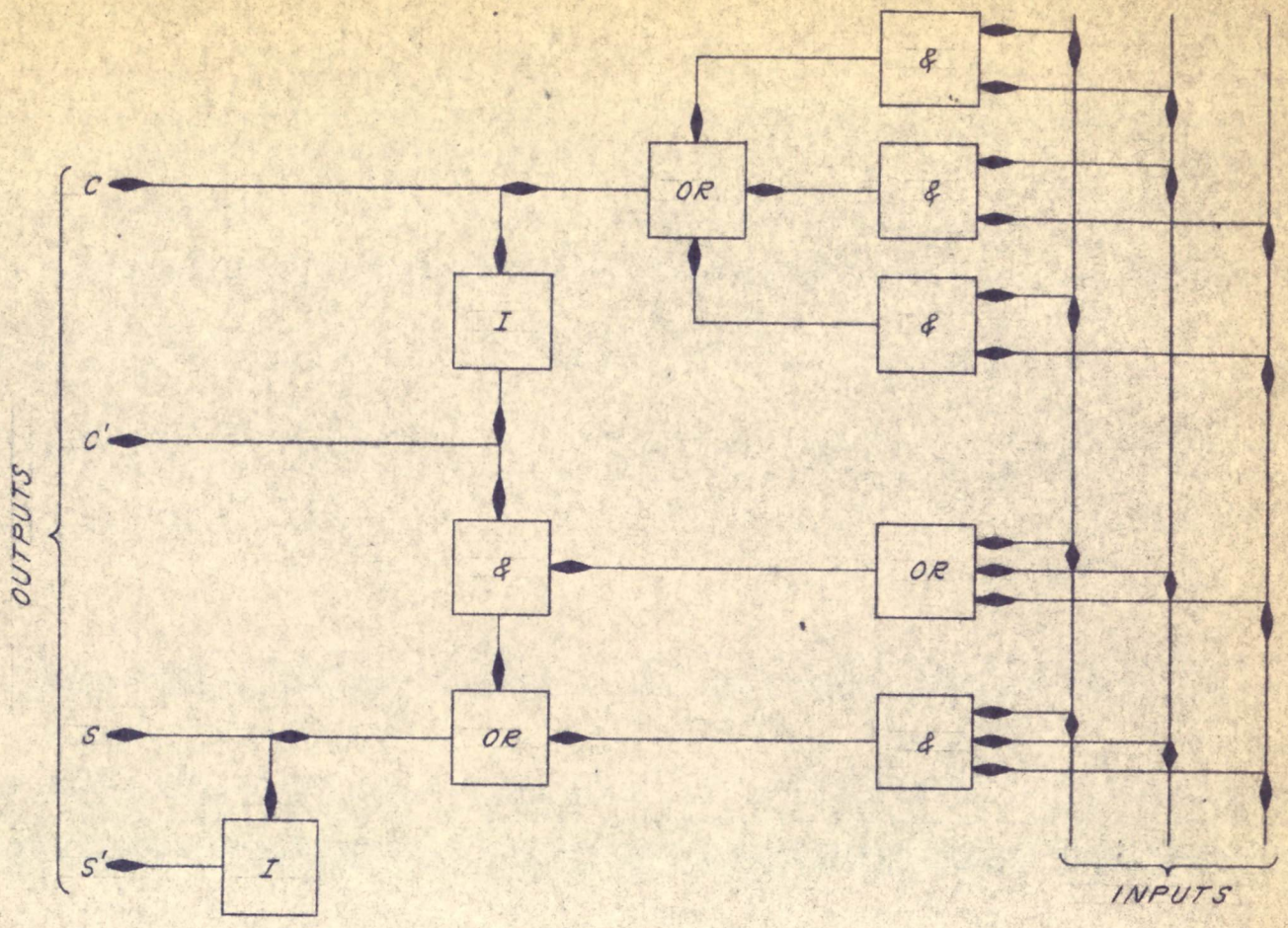


FIG. II-2

THE THREE INPUT ADDER

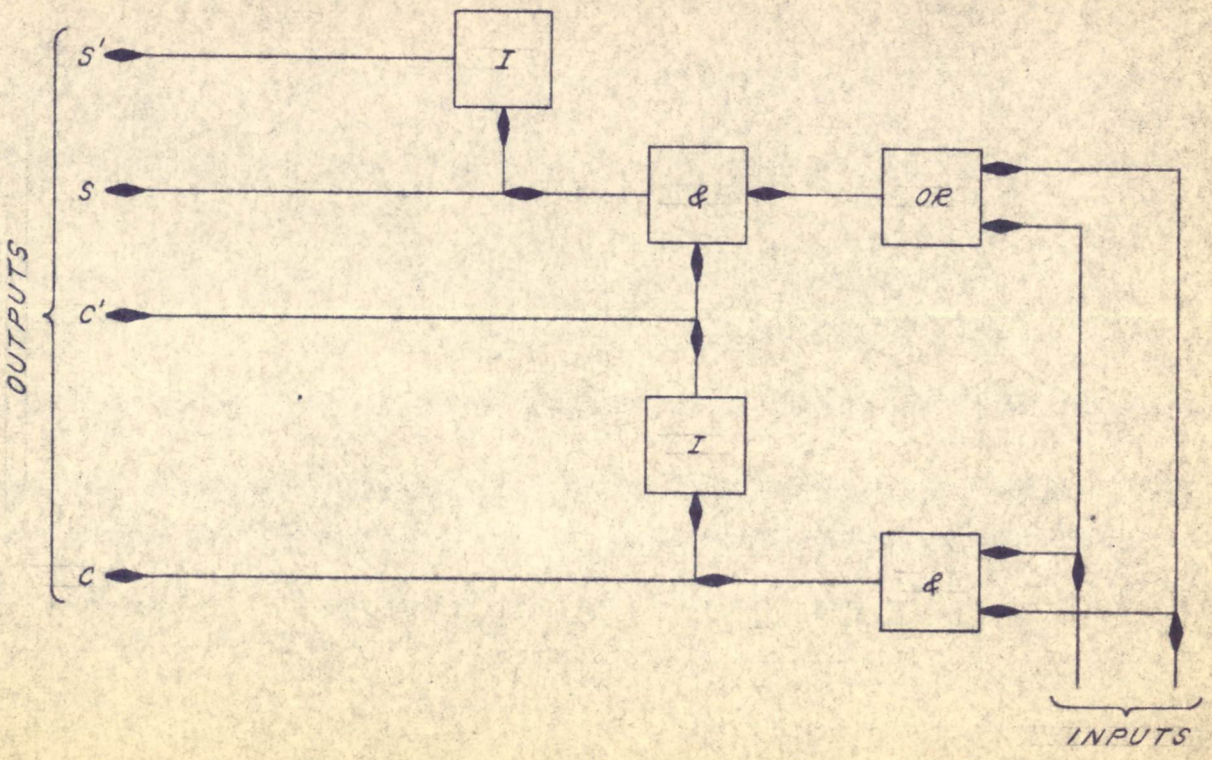


FIG. II-3

THE TWO INPUT ADDER

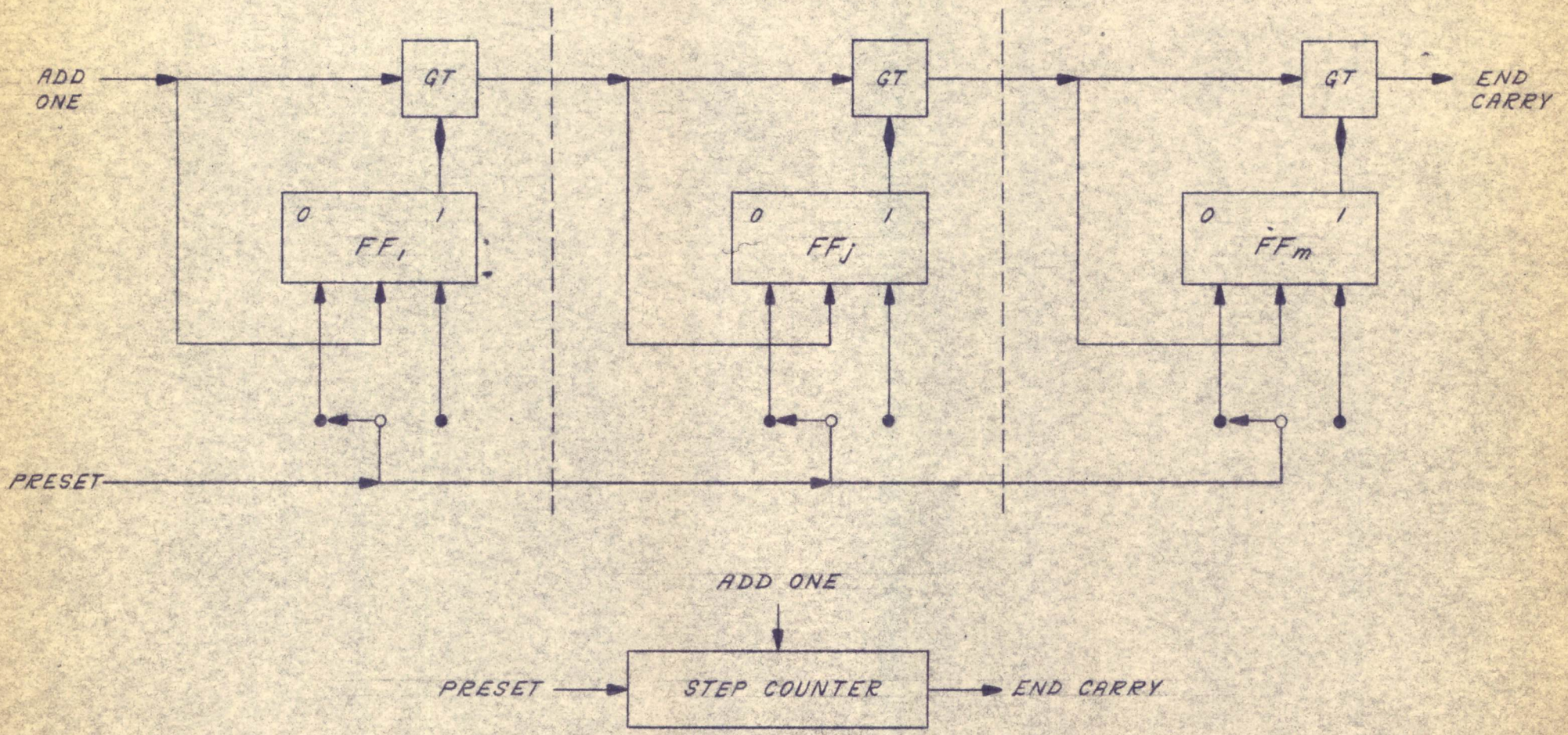


FIG. II-4

THE STEP COUNTER AND ITS BLOCK SYMBOL

K. Olsen

Chapter III

Classification of the Multipliers and General Considerations

3.1 Introduction to the classes of multipliers

The concept of binary notation for real numbers and the rules for performing the various arithmetic operations upon binary numbers have been discussed in other reports of this laboratory.[1] Therefore, understanding of the operation of binary multiplication will be assumed.

The arrangements of digits for long-hand binary multiplication of n digit numbers is shown in the example of Fig. III-1 for n = 5. No digits need be carried when forming the partial products shown in this figure as in decimal multiplication, and hence the digits of these numbers can be completely specified as shown, i.e., as products of two binary digits. The main divisions in the classification of the multipliers discussed in this report are based on the manner in which the digits of the partial products shown in this figure are grouped during an iterative summation process. These main classes of the multipliers are three in number. The multipliers of the first class utilize an obvious grouping of the digits of the partial products. The grouping used by the Class II multipliers is slightly more involved, while that of the Class III multipliers is the most complex.[2]

---

1. See Digital Computer Laboratory Report R-127, p.21, article 4.1; or Report R-90.
2. Other groupings have been considered, but have not as yet led to any other worthwhile classes of multipliers.

3.2 The anatomy of the multipliers

In general, the problem of realizing any multiplier is three-fold: First, an arrangement of logical blocks must be made which will supply digit groups, one by one, to a multiply-adder. Second, this multiply-adder must be designed to sum properly[3] the digits of the digit group supplied to it at a particular instant with the sum of the foregoing groups. Third, a circuit to control the operation of the entire system must be designed. The equipment for forming the digit groups, the multiply-adder, and the flip-flop registers which must be provided to store the multiplier, the multiplicand, the product and other numbers--in short, that part of the multiplier which handles the numerical quantities--will be known as the arithmetic section. The remainder of the multiplier will be called the control section.

The discussion in the following two sections will be illustrated by remarks which apply to a specific case. An introduction to this specific case follows. An obvious way to group the digits of the partial products is simply to take the groups to be the partial products themselves. A simple order for summing these numbers is to sum the first partial product with the

---

3. By "sum properly" is meant addition with the two numbers shifted appropriately as in the description of the preceding paragraph. The inclusion of the ability to shift is all that distinguishes a multiply-adder from an adder.

second partial product shifted one digit to the left (see Fig. III-1), then the result with the third partial product shifted two digits to the left, etc. The Class I multipliers proceed in the manner just described.

3.2.1 The arithmetic section

The problem of the design of the arithmetic section may be divided into two design problems, as was indicated above. The multiply-adder will be considered first.

3.2.1.1 The multiply-adder

The general multiply-adder must be provided with inputs which indicate the values of the digits of a particular one of the digit groups, and with other inputs which indicate the contents of the register (hereafter known as the accumulator register) used to store the sum of the preceding digit groups--see Fig. III-2. It must also be provided with command pulse inputs which cause the multiply-adder to perform whatever operations are necessary in order to add the i^{th} digit group to the contents of the accumulator register, and to store the result shifted appropriately. Some sort of shift is necessary with every multiplier discussed. (The appropriate shift in the case of the Class I multipliers is one digit to the right. This achieves the shifts described in the last paragraph of section 3.2, since a one digit shift right of the result

of each summation is equivalent to a one digit shift left of all the digit groups which remain to be added in.) Such a set of operations will be said to correspond to a step in the multiplication procedure. Thus the number of steps required to complete the multiplication will also be the number of digit groups which must added together to form the product.

The operation or operations which any multiply-adder of those discussed in this report must perform to accomplish the addition mentioned in the preceding paragraph arise from two different logical procedures for binary summation. These are discussed in the following two sections.

3.2.1.1.1 The first method of binary summation

A possible breakdown of the operation of binary addition of two n-digit numbers is illustrated by an example in Figure III-3(a). The rule of binary addition which determines the partial sum (or sum) and the carry-row digits can be expressed as a table:

Table III-1

The Partial-Carry and Partial-Add Operations[4]

$k-1^{st}$ Carry Row (or addend)	$k-1^{st}$ Partial Sum (or Augend)	k^{th} Carry Row	k^{th} Partial Sum
j^{th} digit	j^{th} digit	$j-1^{st}$ digit	j^{th} digit
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

4. The contents of this table are symbolized by two operations of Boolean algebra. Thus the carry row digit corresponding to two digits A and B is symbolized by $A \cdot B$, while the partial sum digit is symbolized by $A \oplus B$. For example if A = 1, B = 1 then referring to the above Table $A \cdot B = 1$ and $A \oplus B = 0$. Boolean algebra is discussed, among other places, in The Design of Switching Circuits, S. H. Caldwell, Chapter III, to be published.

The operation of forming the first partial sum and the first carry row from the addend and the augend--these names are implicitly defined in Figure II-3(a)--is called a partial-add operation (symbolized by "a") while the subsequent, otherwise identical, operations are called partial-carries (symbolized by "c"). If n is the digit length of the original pair of numbers, then no more than n partial-carry operations need be performed before the partial sum becomes the sum.

If this method were used for every addition which must be performed during a multiplication, a great deal of time would be consumed in performing the partial carries. There are at least two ways to remedy this difficulty. First, a realization can be achieved which will perform all the partial-carries in one operation. This operation is ordinarily known as a high-speed carry (symbolized by "c"). It is displayed in the following table and illustrated by an example in Fig. III-4.

Table III-2

The High-Speed Carry Operation

1^{st} Carry Row j^{th} digit	1^{st} Partial Sum j^{th} digit	High-Speed Carry to j^{th} digit	Sum j^{th} digit	High-Speed Carry to $j-1^{st}$ digit
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

It should be noted that when a high-speed carry operation is performed, information in the form of a binary digit (called the high-speed carry digit) is provided to the $j-1^{st}$ digit column from the j^{th} digit column; and that the high-speed carry digit put out depends upon the one received from the $j+1^{st}$ column. Thus the operation must take place in a sequence from right to left. The propagation of this digit, whether it is coded as a pulse or a level, requires time and will affect the speed of any multiplier which uses this operation.

Second, the multiplication operation can be successfully realized even if only one partial carry operation is performed for the summation at the i^{th} step. A series of partial carries or a high-speed carry must be performed at the

final step of such a procedure in order to obtain the product.

3.2.1.1.2 The second method of binary summation

Another possible approach to the problem of summation is to introduce an operation which will be called a predictor carry (symbolized by "$\bar{c}$"). This operation acts upon the original addend and augend to produce a number which can be considered to be an altered addend. The operation essentially predicts where carries will occur and alters the addend so that the complete sum can be obtained by means of a partial add operation upon the augend and altered addend (the carry row produced by this operation is ignored). It is displayed in the following table, and a mumerical example is given in Fig. III-3(b).

Table III-3

The Predictor Carry Operation

Augend j^{th} digit	Addend j^{th} digit	Predictor Carry to j^{th} digit	Altered Addend j^{th} digit	Predictor Carry to $j-1^{st}$ digit
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	0	1

Like the high-speed carry operation, the predictor carry operation must take place in a sequence from right to left. The propagation time for the digits will affect the speed of any multiplier which uses this operation.

The discussion of methods of binary addition has shown that the operations of addition initiated by the command inputs to the multiply-adder can be those of any one of three groups: (a, c) (a, $\underline{c}$) or ($\overline{c}$, a). In the multiplication procedure, each of these groups must be followed by a shift operation (symbolized by "s") so that a set of operations making up a step can be either (a, c, s) ; (a, $\underline{c}$, s) or ($\overline{c}$, a, s).

A saving in the number of operations required for a step can be obtained by building a multiply-adder which combines two or more of the operations of the above sets to obtain such operations as "cs" (partial carry and shift combined), or "ac" (partial add and high-speed carry combined). In the case of the Class I multipliers, it will be noted that an entire digit group will be all zero's if the corresponding digit of B is zero (see Fig. III-1). Thus only the shift operation need be performed at the step which adds in this digit group.

If the addition and shift are combined into just one operation, one might expect the speed of multiplication to be increased since each separate operation in a step requires that the control section supply a command pulse to the multiply-adder. However, this is not necessarily so because of the time delays of the equipment which must be added; furthermore, speed is not the only characteristic being sought: the reliability is also of utmost importance. Therefore, multipliers using more than one operation per step are included in the study.

The multiply-adder for a Class I multiplier can be designed by first designing an elemental unit used for extension of the multiply-adder to handle numbers of any digit length. This unit will be called a stage. The multiply-adder is built up of a string of these stages terminated at each end by special forms of themselves. A stage may handle digits from any number of digit columns, but usually it involves those of only one digit column. The steps in the design of the stages of the multiply-adders presented in this report are not given; only the final design is shown.

The multiply-adders for the Class II and Class III multipliers were developed from those designed for the Class I multipliers (see Chapters V and VI).

3.2.1.2 Formation of the digit groups

In the case of the first class of multipliers the problem of designing a system of logical blocks for forming the digit groups, i.e., the partial products, is comparatively simple. This is so since the k^{th} partial product is either zero or A (see Fig. III-1) depending on whether $B_i = 0$ or 1 (where $i \triangleq n - k + 1$), a fact which was pointed out in the preceding section. Thus the correct partial product will be added in if an add and shift group of command inputs to the multiply-adder are pulsed when $B_i = 1$; otherwise, only the shift-input is pulsed.[5]

The arrangement used to provide these pulses for all but two of the seventeen Class I multipliers studied is to store B in a shifting register (see Fig. IV-1), hereafter known as the B register, and to provide the n^{th} flip-flop in this register with "1" and "0" gate tubes which are sensed by a pulse at each step. The outputs of these gate tubes are fed to the control of the multiplier which then sends them to the proper command inputs of the multiply-adder. Of course the B register must be shifted after each step in order that the $(n - k + 1)^{st} = i^{th}$ digit of B will be in the n^{th} flip-flop for the k^{th} step. The shift gates of the B register and the two gates on the n^{th} flip-flop of this register may be considered to be the system of

---

[5] If the multiply-adder does not otherwise perform a separate, simple shift operation, then it must usually be provided with the ability to do so. The exceptions are the multipliers which perform a "cs" operation.

logical blocks necessary to form the k^{th} digit group for this sub-class of the Class I multipliers.

The two class I multipliers which do not fall into the above sub-class utilize a similar but somewhat more complicated method for forming the k^{th} partial product. They are similar to the multipliers of the first-discussed sub-class in that they require shift gates on the B register, but more complicated in that they use diode "and" circuits to aid in the formation. These multipliers are discussed in sections 4.4 and 4.14.

The equipment for forming the digit groups for the Class II and Class III multipliers is more complex, and discussion of it is postponed to Chapters V and VI.

3.2.2 The control section

Any of the multipliers to be discussed is visualized as just a part of a larger system, e.g., a computer. When the system requires that a multiplication be formed, it will supply the control section of the multiplier with a multiply-command pulse. When the multiplication is complete--i.e., when the output levels from the accumulator flip-flops have stabilized after the last command pulse--the control section must supply a multiplication-complete pulse to the system. Between the occurrence of these pulses, the control will perform its function of distributing the proper number and kinds of

command pulses to the arithmetic section. In order to control the number of commands which it supplies, the control section must contain a step counter. As to the kinds of commands to be supplied, the sequence of these required by the arithmetic section is fixed except for the Class I multipliers. Here the control section requires information from the n^{th} B register flip-flop in determining what sequence of operations is to be performed at a specific step. This situation is the direct result of the method chosen for forming the digit groups of the Class I multipliers.

The control section must be designed carefully in order that it will add as little time to the operation of the multiplier as possible and still not depend upon extremely precise time for its correct operation or require more equipment than is absolutely necessary.

The control sections used in the realizations to be presented are basically of two types, namely those which use a clock pulse generator to supply the command pulses (hereafter called the K type) and those which use a circulating pulse to supply them (hereafter called the C type). Multipliers which use control sections of the first-mentioned type must be synchronous, which here means a "fixed rate," that is, the command pulses are supplied at the

fixed rate of the clock pulses. Multipliers which use the circulating pulse type of control may operate synchroncusly or asynchronously.

3.2.3 Combinations of arithmetic and control section types

The arithmetic sections tobe introduced in later chapters may be typed according to the type of multiply-adder which they use. These multiply-adders utilize two ways of initiation the "a," "ac," "s", etc., operations: (1) An initiating pulse is sent to every stage simultaneously, or (2) an initiating pulse is sent to the rightmost stage of the multiply-adder and propagates through the multiply-adder stage by stage. All the operations of each of the multiply-adders are initiated in either one or the other of the above manners. Thus the multiply-adders and also the arithmetic sections fall into two types: Those whose operations are initiated in the first manner will be called the block type (B type), and the others the domino type (D type) from an anlogy with the knocking over of a row of dominos (the stage becomes a domino in this analogy).

Either type of arithmetic section may be used with either type of control. A C type control used with A B type arithmetic section uses delay units to establish the intervals between the commands, while a C type control

used with a D type arithmetic section may use either delay units or the delay of command through the multiply-adder (with delay units added if necessary) to establish these intervals. A C type control is usually faster than a K type control when both are used with the same arithmetic section since the time intervals between its command pulses are not restricted by a clock pulse period. For D type arithmetic sections, a C type control which uses delay units is usually faster than one which utilizes the delay through the multiply-adder since time intervals between commands are less restricted in the first case than in the second.

It may, from the standpoint of speed of operation, be more desirable to use one rather than the other type of control with any specific arithmetic section. Further discussion of the choice of the control section for use with a specific arithmetic section must be postponed to section 3.4.

3.3 The multiplication time

The memory overlap time is measured from the instant the output levels of the A register have stabilized after reading the multiplicand into it from the memory (by means of the read-in pulse). Therefore, the time required for a multiplication should also be measured from this instant.

The multiplication time can be divided into three parts: (1) The time which elapses between the stabilization of the A register outputs and the occurrence of the first command pulse of the first step. This is the initial time. (2) The time required from the first command pulse of the first step to the first command pulse of the last step. The time required for a given step will be known as the step time; and, for the Class I multipliers, it may have any one of several values depending on the contents of the n^{th} B register flip-flop when the step is to occur. (3) The time required for the stabilization of the accumulator flip-flop outputs from the first command pulse of the last step. This final time will include the time for a final carry operation if one is necessary.

In order to allow for the initial time and yet hold it to a minimum, the time between the read-in pulse and the multiply-command pulse must be as follows:

The C type controls presented for all the multipliers send the first command pulse to the multiply-adder almost immediately upon receiving the multiply-command pulse. Thus for C type controls the multiply-command must occur approximately τ plus the minimum allowable initial time, t_i, min, after the read-in pulse. Then the initial time $t_i = t_i$, min.

The K type controls presented for all the multipliers send the first command pulse to the multiply-adder one clock pulse period (the letter T will be used throughout the report to denote the clock pulse period in microseconds) after receiving the multiply-command pulse (since this pulse is assumed assumed to be synchronized with the clock pulses. If $T \leqslant \gamma + t_{i,\,min}$, then the read-in pulse can occur simultaneously with the multiply-command pulse. Then $t_i = T - \gamma$. Otherwise, the multiply-command pulse must occur $\gamma + t_{i,\,min} - T$ microseconds after the read-in pulse and $t_i = t_{i,\,min}$.

The definitions introduced above and in the next section are intended to provide a standard basis for and an aid to estimating the multiplication time. It should be emphasized that they do not change the fact that the values given in later chapters for multiplication times are merely estimates.

3.3.1 The step time

The time required for each step of the multiplication is fixed for the Class II and Class III multipliers, but varies according to the contents of the n^{th} B register flip-flop (and in certain cases the other digits in the A and B registers) for the Class I multipliers. The following discussion applies primarily to the Class I multipliers, but is of use in determining the constant step times of the Class III and Class III multipliers.

The step time of the Class I multipliers varies because (1) the number and/or kind of operation to be performed at the step depends on the contents of the n^{th} B register flip-flop, and (2) the time to "perform" an operation depends upon the operation itself and the operation which is to follow it. This latter consideration leads to the following definitions.

operation-pair: Any sequence of two operations which can occur in the functioning of a given arithmetic section.

time-lapse: The time from the occurrence of the first operation of an operation-pair to the instant when the arithmetic section is prepared for the second operation of the pair. (Clearly every time-lapse is associated with a specific operation-pair. The time-lapse may also depend on the numerical contents of the various registers in the arithmetic section, as when a carry propagation occurs.)

maximum time-lapse: The largest value which the time-lapse required by a particular operation-pair can assume. (This time will be symbolized by the abbreviations for the operations of the pair separated by a colon, e.g., a:cs.) The maximum time-lapse differs from the particular time-lapses of a specific operation-pair for B type arithmetic sections because of the irregular times required for carry propagation. Since in this case there is no simple way to

indicate to the control what time-lapse is required when the operation-pair occurs, the maximum time-lapse must be allowed at its every occurrence. The determination of the maximum time-lapses of D type arithmetic sections requires further discussion, which in turn requires the following definitions.

per-stage delay (σ): The delay experienced by a propagating command pulse is passing through a stage. (This delay may vary for any stage of a multiply-adder according to the contents of the corresponding sections of the A and accumulator registers. Each per-stage delay is of course associated with a particular command.)

maximum-stage time-lapse: For every stage of a D type arithmetic section there is a time which must be allowed between the instant at which the stage receives the pulse of the first operation of a pair and the instant at which that stage is prepared for the second of the operations. This can be called the stage time-lapse. The maximum stage time-lapse is the largest of these values. (This time is dependent upon the particular operation-pair and will be symbolized by the abbreviations for the operations of the pair separated by a colon and followed by a subscript "S", e.g., $(a{:}cs)_S$.)

The determination of the maximum time-lapse for D type arithmetic sections is made from the binary digit length, n, of the numbers multiplied (this determines the number of stages in the multiply-adder), the per-stage delay, and the maximum stage time-lapse. In particular, if the per-stage delays for the commands which occur in the operation of a particular D type multiply-adder are independent of the numerical contents of the controlling registers, and moreover if they are equal, then the maximum time-lapses are equal to the minimum stage time-lapses for the respective operation-pairs. This situation occurs for all of the D type arithmetic sections introduced in Chapter IV except one (see section 4.11.1).

The maximum time-lapse differs from particular time-lapses for specific command-pairs of D type arithmetic sections either because of variable carry propagation time or because the per-stage delay is dependent upon the numerical contents of the controlling registers. In the latter case an indication of the time-lapse required at a particular occurrence of the operation-pair is automatically given to a C type control which uses the completion pulses (the command pulses as they appear from the final stage of the multiply-adder) in place of the commands delayed by fixed delay units. This situation of variable per-stage delays arises for the multiply-adder mentioned above as being discussed in section 4.11.1. The step time is a sum of maximum time-lapses.

A given step with a Class I multiplier requires one of at most two possible operation sequences (corresponding to the fact that the contents of the n^{th} B register flip-flop is either 0 or 1. Thus there are at most four step times, and this number occurs only when each operation sequence can be followed either by itself or by the other sequence in the operation of the arithmetic section.

3.3.2 The measure for the speed of a multiplier

The fact that the multiplication time of a multiplier depends in certain cases upon the numbers to be multiplied makes it necessary to introduce this section in order to specify how to measure the speed of a multiplier: The multiplication time itself will be used when constant. Otherwise, the maximum and minimum multiplication times will be found, and the average of these two values used as the measure. If we assume that a zero or a one has equal probability of occurrence in every digit position of the number B, this average multiplication time will also be the most likely multiplication time for most of the multipliers of this type.

3.3.2.1 Choice of type of control section for use with a specific arithmetic section

The criteria for the choice of the control section will be based solely on the speed with which the resultant multiplier operates since alterations

in the control section can affect the equipment count for the whole multiplier only slightly.

The clock pulses of a K type control may be used to initiate either each step or each of the commands of a step. (In cases where there is only one command perstep the either or has no distinction associated with it.) In the first case, a K type control would be used only when the step times, if more than one, are all very nearly equal. In the second case, it would be used only when all maximum time-lapses are all very nearly equal and it is possible to arrange a K type control to supply the proper kinds of commands.[6] It is more desirable to use K type controls in the second of the above manners if there is a choice, for otherwise delay line units will be necessary to establish the intervals between the command pulses. The two cases are the only ones in which K type controls can compete with C type controls for speed of operation. Even here, because of the effect of the control itself on the multiplication time, the C type control may be faster. Thus if a

---

6. It has been found possible to design K type controls which can supply the proper kinds of commands when these are initiated separately by the clock pulses only when either just one command is required per step or when at most two commands are required, the last of which is a shift or a combined carry and shift. There is no trick to the design in the first case, but in the second a special device is necessary. It is explained in section 4.1.3.

a particular arithmetic section falls into either of the above cases, its multiplication time with both C and K type controls will be considered and that control which gives the faster operation will be chosen; otherwise, C type controls will be used.

Chapter IV

Realizations of Class I Multipliers

4.1 Introduction

4.1.1 Outline of the multipliers to be discussed in this chapter

It is desirable to have a symbolism for naming the Class I multipliers which will also indicate something of their structure. The symbol for a given multiplier should at least indicate the type of control and the type of arithmetic section used. In addition, the operations performed by the multiply-adder when $B_i = 1$ can be indicated by use of the symbols introduced in Chapter III, separating the separate operations by periods. For example, a.cs stands for the operation partial add followed by a combined complete carry and shift. Finally, the different physical realizations which are presented for certain of the multiply-adders are distinguished by numbering them. Thus, the second scheme for realizing the multiplier which utilizes a clock pulse type of control and a block type of arithmetic section which performs a combined partial add and complete carry followed by a shift when $B_i = 1$ will be K-ac.s B2. This symbolism will serve for the Class I multipliers and will be extended in later chapters to include the Class II and Class III multipliers.

In the following outline of the Class I multipliers to be discussed, the multipliers are first classified according to their logical properties and then according to physical differences.

The Class I Multipliers

A. First method of summation used

1. Final high-speed carry needed

a. Two operations when $B_i = 1$

K=a.cs Bl[1].

C=a.cs Bl

K=a.cs B2

C=a.cs B2

b. One operation when $B_i = 1$

K=acs Bl

K=acs B2

2. No final high-speed carry needed

a. Three operations when $B_i = 1$

C=a.c.s B

b. Two operations when $B_i = 1$

(i) C=as.c B

(ii) C=ac.s Bl

C=ac.s B2

K=ac.s Dl

C=ac.s D2

K=ac.s D3

c. One operation when $B_i = 1$

(i) C=asc B

(ii) C=acs Bl

K=acs D[2].

B. Second method of summation used

C=$\bar{C}$.a.s B

---

1. This is the scheme used by the Whirlwind I computer.

2. This is the scheme chosen for the computer of whose design this study was a part.

Previous to the development of the system used in this outline for naming the Class I multipliers they were named differently. The correspondence between the old and the new terminology is given below.

Old	New
2 sc (WWI)	C-a.cs B1
-	C-a.cs B2[3].
3 asc	K-acs B1
2 asc	K-acs B2
2 C	C-a.c.s B
2 as (with carry flip-flops)	C-as.c B
3	C-ac.s B1
2D	C-ac.s B2
A 1	K-ac.s D1
A 2	C-ac.s D2
A 5	K-ac.s D3
2 as (with delay lines)	C-asc B
3 as	C-acs B1
A slight alteration of the 3 as multiplier	C-acs B2
A as	K-acs D
2 a	C-$\bar{c}$.a.s B

The old S = 2 (and 2 x 2) multipliers are of Class III. These are discussed in Chapter VI.

3. The idea for this Class I multiplier came from one of the two realizations presented for the old S = 2 multiplier in the original study.

4.1.2 The storage registers

The arithmetic sections of the Class I multipliers each have at least three storage registers, namely the A, B and accumulator registers. The accumulator register was introduced in section 3.2.1.1, and the B register in section 3.2.1.2. In addition, some multipliers have a "C" register for carry storage. The A and B registers are used to store the multiplicand and multiplier respectively and must therefore contain n flip-flops each. Consecutive sensing of the digits of the multiplier, required for formation of the digit groups for the Class I multipliers, is achieved by shifting it through the rightmost flip-flop of the B register, as was mentioned in section 3.2.1.2. No provision is made to store the digits shifted off the end of the B register and hence the multiplier is lost during the multiplication.

It can be shown that the product of two n digit binary numbers can at most be 2n binary digits in length. Thus the accumulator register, which is used to store this number, should contain 2n flip-flops. However, the B register may be used for the rightmost n flip-flops of the accumulator because of the following facts: (1) After the k^{th} step of the multiplication, k partial products will have been summed, and the result will be a number of length n + k. (This is the general case when a new leftmost digit may be formed by a carry). The rightmost k digits of this number are digits of the final product, (cf. Fig. III-1) and therefore may be stored in a register whose contents are not operated upon by the multiply-adder except for shift right operations. (2) The B register is a shifting register which must shift at each step of the multiplication.

The fact that the B register also serves as the right half of the accumulator register means that formally speaking, the associated shift gates are part of the multiply-adder, as well as part of the equipment for forming the digit groups. It is assumed that the B register is shifted before or at the same time that the accumulator is shifted, for otherwise the left-most digit of B would be lost.

The shifting register shown in Figure IV-1 is used for the B register in all the Class I multipliers. Therefore, it is not redrawn for each multiplier.

Neither read-in nor read-out gates are shown for any of the registers in the figures to follow since the result would be to needlessly complicate the diagrams. Neither are means shown for clearing the registers, although in the discussions to follow the registers are assumed to have been initially cleared.

4.1.3 A remark on the design of K type controls for Class I multipliers

If an arithmetic section requires two commands when $B_i = 1$, the last of which is a shift or a combined carry and shift, then a K type control can be used to operate this arithmetic section if it is built to clear the $n^{\underline{th}}$ flip-flop of the B register when the first command of the $B_i = 1$ sequence is performed and then to again sense the contents of this flip-flop (which must now be 0). This will certainly operate correctly as long as the second operation of the $B_i = 1$ sequence is a shift. If instead it is a combined carry and shift then the control will perform a cs whenever the contents of the $n^{\underline{th}}$ flip-flop of the B register is 0, i.e., even if $B_i = 0$. This scheme for control will still cause the arithmetic section to perform the multiplication correctly as is established

in section 4.2. In addition, a considerable equipment saving will be effected by eliminating the need for shift gates in the arithmetic section.

4.2 The K-a.cs B1 and C-a.cs B1 multipliers -- Figs. IV-2, 3 and 4

4.2.1 The operation

The maximum time-lapses are found from Fig. IV-4 to be:[4]

a:cs	2γ	$+ \tau$
cs:a	γ	$+ \tau$
cs:cs	2γ	$+ \tau$
cs:c	2γ	$+ \tau$

These values are substantially the same and therefore C and K type controls will operate with about the same speeds. The block diagram of an appropriate K type control is shown in Fig. IV-2. The clock pulse input is assumed to be synchronized with the multiply command pulse. If this is not so, then a synchronizing circuit using two flip-flops must be inserted in the control block diagram; for otherwise, a clock pulse might occur at such a time after the multiply command to result in a degenerate output pulse from the first gate tube in the path of the clock pulses. This degenerate pulse might perform some but not all of the functions required of it and hence produce an error.

The multiply command pulse causes clock pulses to be sent to sense the gates on the $n^{\underline{th}}$ flip-flop of the B register. If this flip-flop holds a one then the clock pulse is sent to the partial add input of the multiply-adder,

4. The Greek letter symbols used here are defined in section 2.1.

of the multiply-adder, and at the same time it clears the n^{th} flip-flop of the B register (see section 4.1.3). Thus the next clock pulse is sent to the partial carry and shift input of the multiply-adder, and at the same time it shifts the B register. If the n^{th} B register flip-flop holds a zero, then the clock pulse is sent to the partial carry and shift line. Thus the required shift of the accumulator (and the B register) is performed, while the additional partial carry has no harmful effect as is established below.

The K-a.cs B1 multiplier requires n clock pulses to complete the multiplication, and therefore the step counter must be preset to produce and end-carry when it receives the n^{th} add-one pulse. The end-carry pulse turns off the clock pulses and, after a delay to allow the completion of the last partial carry and shift operation, is sent to the high-speed carry input of the multiply-adder. After a further delay to allow the completion of the high-speed carry operation, the end-carry appears as the multiplication complete pulse.

The block diagram of an appropriate C type control is shown in Figure IV-3. This control also performs a combined partial carry and shift whenever the contents of the n^{th} B register flip-flop is a zero and thus also saves the set of shifting gates in the multiply-adder.

The a.cs B1 arithmetic section is shown in Fig. IV-4. An illustration of the procedure of multiplication used by this multiplier is given in Fig. IV-5. Such a multiplier can be successfully mechanized if its multiply-adder is provided with a register for storage of the carry row. This statement requires proof, as may be seen from the example of Fig. IV-5. For if the boxed-in zero in this figure were a one, then a second carry digit would have to be stored in

the next digit column to the left after the third partial add operation. Thus a single register could not be used for storage of the carry row, for this carry overlap would cause one carry digit to be lost. It is not intuitively obvious that such a situation can never arise. If it could, then at least two registers for storage of carry digits would be required in the realization of the a.cs B1 arithmetic section. The proof that this situation can never occur follows:

Let: 1. The flip-flops of the AR, AcR and CR (carry register) be numbered according to the digit column in which they fall (see Fig. IV-4).

2. A_j be the $j^{\underline{th}}$ digit of A. (It will be stored in the $j^{\underline{th}}$ flip-flop of AR.)

3. $Ac_j(k)$ be the digit stored in the $j^{\underline{th}}$ flip-flop of AcR just prior to the $k^{\underline{th}}$ operation of the multiplication procedure.

4. $C_j(k)$ be the digit stored in the $j^{\underline{th}}$ flop-flop of CR just prior to the $k^{\underline{th}}$ operation.

Suppose that in the number B (the multiplier), there is a sequence of digits: 1, (m - 1) zeros, 1 (e.g. if m = 3 than the sequence is 1001, if m = 1 then the sequence is 11. Then, beginning with that step of the multiplication corresponding to the rightmost digit and assuming that k operations have preceded this step, the following operations are performed for the portion of the number B which is under consideration:

For the rightmost 1:

"a"
$$\begin{cases} Ac_j(k+1) = Ac_j(k) \oplus A_j \\ C_j(k+1) = Ac_{j+1}(k) \cdot A_{j+1} \end{cases}$$

"cs"
$$\begin{cases} Ac_j(k+2) = C_{j-1}(k+1) \oplus Ac_{j-1}(k+1) \\ C_j(k+2) = C_j(k+1) \cdot Ac_j(k+1) \end{cases}$$

For the (m - 1) zeros

$(m-1)^{\underline{th}}$ "cs"
$$\begin{cases} Ac_j(k+m+1) = C_{j-1}(k+m) \oplus Ac_{j-1}(k+m) \\ C_j(k+m+1) = C_j(k+m) \cdot Ac_j(k+m) \end{cases}$$

5. For the meaning of these Boolean algebraic expressions see the Table III-1 footnote 3 on p. _____.

For the left-most 1:

"a"

$$Ac_j\ (k + m + 2) = Ac_j\ (k + m + 1) \oplus A_j$$

"cs"

$$C_j\ (k + m + 2) = Ac_{j+1}\ (k + m + 1) \cdot A_j + 1$$

We are interested in the two digits which occupy the $j^{\underline{th}}$ flip-flop of the carry register before and after the second partial add operation. Specifically, we would like to show that both of these digits cannot be unity. This is so if $C_j\ (k + m + 2) \cdot C_j\ (k + m + 1) = 0$. Making use of the above relationships, it follows that

$$C_j\ (k + m + 2) = Ac_{j+1}\ (k + m + 1) \cdot A_{j+1} = \left[C_j\ (k + m) \oplus Ac_j (k + m)\right] \cdot A_j + 1$$

and

$$C_j\ (k + m + 1) = C_j\ (k + m) \cdot Ac_j\ (k + m)$$

Thus $C_j\ (k + m + 2) \cdot C_j\ (k + m + 1) = 0$ as can be seen from the following table in which all possible combinations of the values of the Ac_j and C_j variables are displayed.

$C_j\ (k+m)$	$Ac_j\ (k + m)$	$C_j\ (k + m + 1)$	$C_j\ (k + m + 2)$	$C_j\ (k + m + 1) \cdot C_j (k + m+2)$
0	0	0	0	0
0	1	0	$A_j + 1$	0
1	0	0	$A_j + 1$	0
1	1	1	0	0

The end digit columns are shown separately in Figure IV-4 because they differ from the general case. The $n^{\underline{th}}$ digit column does not require the carry flip-flop with its seven associated gates since the carry-digit here is always zero. For the same reason no high-speed carry line and associated gate on the accumulator flip-flop, the high-speed carry gate are necessary. Gates must be added to perform the shift right function of those gates which would have been associated with a carry flip-flop.

The Ac_1 flip-flop will always contain a zero before a partial add, since

this operation is always preceded by a shift (combined with a partial carry). Therefore, there cannot possibly be a carry generated by a partial add pulse and hence the associated gate and carry flip-flop (which would have been C_o) are not necessary. Since the final high-speed carry operation is also preceded by a shift, which clears the Ac_1 flip-flop, the high-speed carry gate can be eliminated from the first digit column and any carry arriving on the high-speed carry line from the second digit column or from C_1 can be sent to set 1 in the Ac_1 flip-flop. That there cannot be two carries arriving, one from C_1 and one from the second digit column, is proved by the following argument.

A carry can be present on the high-speed carry line from the second digit column after a high-speed carry command only if $Ac_2 = 1$, while a carry can arrive from C_1 only if that digit is unity. These two digits, $Ac_2(k)$ and $C_1(k)$, are formed by a partial carry and shift operation from the previous contents of Ac_1 and C_1 -- $Ac_1(k-1)$ and $C_1(k-1)$ -- according to the Boolean expressions:[6]

$$C_1(k) = Ac_1(k-1) \cdot C_1(k-1)$$

$$Ac_2(k) = Ac_1(k-1) \oplus C_1(k-1)$$

Thus both $Ac_2(k)$ and $C_1(k)$ cannot be unity simultaneously and hence two carries cannot arrive at the set input of Ac_1 simultaneously. This argument can immediately be generalized to show that the high-speed carry gate of the j^{th} digit will never receive two pulses during the execution of a high-speed carry operation. A similar argument applies to other multipliers which perform a high-speed carry operation.

6. See Table III-1 and footnote 3, p.____.

4.2.2 Equipment count

Basic Circuit	Number Used - n digits: Control Section K	Control Section C	Arithmetic Section	Total Used n = 16: K	C
FF	$\geq 1 + \log_2 n$	$\geq 1 + \log_2 (n-1)$	$4n - 1$	68	68
GT	$\geq 1 + \log_2 n$	$\geq 2 + \log_2 (n-1)$	$12n - 8$	189	190
M_2	0	2	$3n - 4$	44	46
D	2	3	0	2	3

Total Weighted Equipment Count = $\begin{cases} 4738 \text{ for K-a.cs Bl} \\ 4772 \text{ for C-a.cs Bl} \end{cases}$

4.2.3 Multiplication time

A. K-a.cs Bl (Control of Fig. IV-2)

Let the multiply-command pulse occur simultaneously with the read-in pulse. Then:

1. $t_{max} = n2T + \delta_1 + \delta_2 - \tau$

   $t_{min} = nT + \delta_1 + \delta_2 - \tau$

2. T must be determined so that the multiplication process can be stopped reliably:

   $T \geq 6\gamma + \tau$: Let T = .75 μsec. $(16 \geq n \geq 9)$

   $\delta_1 \geq cs - c = 2\gamma + \tau$: Let δ_1 = .6 μsec.

   $\delta_2 \geq (n-1)(\gamma + \mu) + \tau$: Let δ_2 = 1.3 μsec. if n = 16

3. If n = 16, t_{ave} = 19.6 μsec.

B. C-a.cs Bl (Control of Fig. IV-3)

Let the multiply comman pulse occur τ seconds after the read-in pulse.

Then:

1. $t_{max} = n(2\gamma + 2\mu + \delta_1 + \delta_2) + \delta_3$

   $t_{min} = n(2\gamma + 2\mu + \delta_2) + \delta_3$

2. $\delta_1 \geq 2\gamma + \tau \cong .58$ μsec. : Let δ_1 = 0.6 μsec.

   $\delta_2 \geq 4\gamma + \tau - 2\gamma$: Let δ_2 = 0.6 μsec. $(16 \geq n \geq 9)$

   $\delta_3 = (n-1)(\gamma + \mu) + \tau$: Let δ_3 = 1.3 μsec. if n = 16

3. If n = 16, $t_{ave} \cong$ 17.3 μsec.

4.3 The K-a.cs B2 and C-a.cs B2 multipliers -- Figs. IV-2, 3, and 6.

4.3.1 The operation

The arithmetic section of a.cs B2 (omitting the B register) is shown in Fig. IV-6. The partial add operation is performed in a standard manner except that any carry generated is passed through a delay line before being stored in the carry flip-flop. This delay is to be of such a magnitude that its output will reach the carry flip-flop, ρ microseconds after the arrival of a pulse generated by the "cs" which occurs after the partial add. Thus this "cs" will be performed on the carry row before the carries from the immediately preceding partial add have been included. Since a shift intervenes before the storage of the carries generated by this partial add, they must be placed as shown, i.e., one digit column to the right of where they would otherwise have gone.

The reason for including the delay before carry storage is not evident here, since it would certainly be much simpler to omit the delay and send any carries directly to the carry flip-flop of the next digit column to the left. However, the delay is necessary if this multiply-adder is to be used in the realization of certain of the Class III multipliers (namely those for which $p < q$, where p and q are defined in Chapter VI). The correct operation of the multiply-adder of Fig. IV-6 will be justified below, but the multiplication time (and thus necessarily the maximum time-lapses) will be figured from a multiply-adder which does not use these delay lines.

The "cs", intervening before the storage of the carries, alters the order of summation of carry digits from that which would have occured in the a.cs B1 multiplier, but the final product can still be correct. This is so because the only chance for an incorrect product lies in the possibility that a carry digit will be lost. This might occur in either of two ways: The first and most serious

is the possibility of a pulse arriving from a delay line unit at a carry flip-flop which already holds a "1", thus causing carry overlap.

It can be seen with the aid of Figs. IV-6 and IV-7 that carry overlap in digit column j is impossible if and only if

$$\left[A_j \cdot Ac_j\ (k)\right] \cdot \left[C_j\ (k+1) \cdot \left\{Ac_j\ (k+1) \oplus C_j\ (k+1)\right\}\right] = 0\ ^{7}.$$

This simplifies to

$$\left[A_j \cdot Ac_j\ (k)\right] \cdot \left[C_j\ (k+1) \cdot Ac_j\ (k+1)\right] = 0$$

Again from Figs. IV-6 and 7, $Ac_j\ (k+1) = Ac_j\ (k) \oplus A_j$. Therefore the original relation is satisfied for every possible combination of digits.

The other way in which a carry digit might be lost is that the set and clear sides of a carry flip-flop may receive pulses "simultaneously" (logically speaking). Examples can be found in which this situation will occur, but from the above proof it follows that the contents of the carry flip-flop will always be 0 before the arrival of the two pulses. Thus if the flip-flop circuit will always complement in this situation, no carries will be lost. However, if the pulses arrive in an adverse order or in the right order but separated by less than ρ_f μsec. (which could occur if the delay δ were not long enough or if it proved desirable to decrease δ in the interests of faster operation) it is obvious that the flip-flop circuit will not complement. This situation can be avoided entirely by an alteration of the multiply-adder as shown in Fig. IV-8. It will be noted that this requires more equipment. Decision between the two forms of the multiply-adder depends on the physical situation and must include consideration of many diverse factors such as the period of the circuit used

7. See Table III-1 and footnote 3, P._____.

for a clock pulse generator (if a K type control is to be used), the stability of delay lines, etc., as opposed to the susceptibility of gate tube circuits to intermittent errors, etc.

It should be noted that the delay in a control section before a c pulse would have to be great enough to allow any possible carries from a partial add which may occur before the final cs to pass thru the delay lines and to set up the carry flip-flop outputs.

The maximum time-lapses for the a.cs B2 arithmetic section of Fig. IV-6 modified as discussed above are found to be:

a:cs	$2\gamma + \tau + \alpha + \omega$
cs: a	τ
cs:cs	$\gamma + \tau + \alpha + \omega$
cs:c	τ

These values are approximately the same and therefore C and K type controls will operate with about the same speeds. The control sections of Figs. IV-2 and 3, designed for the a.cs B1 multiplier and discussed in the preceding section, are applicable here.

4.3.2 Equipment count

Basic Circuit	Number Used - n digits: Control Section K	Control Section C	Arithmetic section	Total Used n = 16: K	C
FF	$\geq 1 + \log_2 n$	$\geq 1 + \log_2 (n-1)$	4n	69	69
GT	$\geq 1 + \log_2 n$	$\geq 2 + \log_2 (n-1)$	8n	133	134
$\&_2$	0	0	4n	64	64
OR_2	0	0	2n	32	32
M_2	0	2	n - 1	15	17
D	2	3	0	2	3

Total Weighted Equipment Count = $\begin{cases} 4544 \text{ for K-a.cs B2} \\ 4578 \text{ for C-a.cs B2} \end{cases}$

4.3.3 Multiplication time

A. K-2.cs B2 (Control of Fig. IV-2)

Let the multiply command pulse occur simultaneously with the read-in pulse. Then:

1. $t_{ave} = (3/2)nT + \delta_1 + \delta_2 - \tau$

2. T must be determined so that the multiplication process can be stopped reliably.

$T \geq 6\gamma + \tau$: Let T = .75 μsec. ($16 \geq n \geq 9$)

δ_1 - cs:c = τ: Let δ_1 = 0.5 μsec.

$\delta_2 \geq (n-1)\gamma + \tau$: Let δ_2 = 1.1 μsec. if n = 16

3. If n = 16, $t_{ave} \cong$ 19.1 μsec.

B. C-a.cs B2 (Control of Fig. IV-3)

Let the multiply command pulse occur $\tau + \alpha + \omega$ μsec. after the read-in pulse. Then:

1. $t_{max} = n(2\gamma + 2\mu + \delta_1 + \delta_2) + \delta_3$

$t_{min} = n(2\gamma + 2\mu + \delta_2) + \delta_3$

2. $\delta_1 \geq 2\gamma + \tau + \alpha + \omega$: Let δ_1 = 0.6μsec.

$\delta_2 \geq 2\gamma + \tau$: δ_2 = 0.6μsec. ($16 \geq n \geq 9$)

$\delta_3 \geq (n-1)\gamma + \tau$: δ_3 = 1.1 μsec. (n = 16)

3. If n = 16, $t_{ave} \cong$ 17.1μsec.

4.4 The K-acs B1 multiplier - Fig. IV-9

4.4.1 The operation

The maximum time-lapses for the acs B1 arithmetic section are found from Fig. IV-9 to be:

acs:acs	$\gamma + \tau + 2i + 3(\alpha + \omega)$
acs:cs	$\gamma + \tau + 2_i + 5\alpha + 4\omega$
cs:acs	"
cs:cs	$\gamma + \tau + 2i + 3(\alpha + \omega)$
acs:c	$4\gamma + \tau + 2i + (2N-1)(\alpha + \omega)$ $(16 \geq n \geq 9)$
cs:c	"

These maximum time-lapses are all essentially the same except those for acs:c and cs:c. The clock pulse period of a K-type control may be chosen disregarding these latter, longer time-lapses since only one of them can occur during a multiplication and then only at the end of the operation. Therefore, they can be taken care of by a delay line after the clock pulse flow has been stopped. Hence either a K or a C-type control can be used. A C-type control offers very little possibility of increasing the speed of operation over that of a K-type control here since it will be found below that the value of the clock pulse period need not be increased to ensure reliable stopping of the multiplication process. Hense only a K-type control will be considered.

This multiplier realization is somewhat different from the others to be considered. The difference lies in the use of levels as well as pulses from control. Change of the control levels effectively changes the interconnection of logical blocks within the multiply-adder. This different type of control makes the sensing gates of the $n^{\underline{th}}$ digit section of the B register unnecessary, but they must be replaced by the diode "and" and "or" circuits shown in Fig. IV-9.

It will be noted that no separate shift pulse is used, but that the "acs" pulse is used with the effect of a "cs" in place of it. The "acs" input is made to perform a "cs" simply by effectively disconnecting the A register flip-flops from the adders by means of the B_i level.

The high-speed carry which must be performed to complete the formation of the product is accomplished by another interchange of connections within the multiply-adder, again achieved by the change of two control levels: the lowering of the multiply level and the raising of the high-speed carry level. Before the c line can be pulsed to record the final product, sufficient time must be allowed for the carry levels to propagate thru the

multiply-adder. The maximum time-lapse (cs:c) occurs when a level must travel the entire length of the multiply-adder. Knowledge of the elapsed time in this case is necessary for the determination of δ_1 in the control section.

If the separate "acs" and "acs + c" lines were not used than an additional pulse mixer would be necessary in each digit column.

4.4.2 Equipment count

Basic Circuit	Number Used - n digits: Control Section	Arithmetic Section	Total Used n = 16
FF	$\geq 1 + \log_2 n$	4n - 1	68
GT	$\geq 1 + \log_2 n$	6n - 5	94
	0	n - 1	15
$\frac{1}{2}$	1	6n - 4	93
OR_2	0	3(n - 2)	52
D	2	0	2

Total Weighted Equipment Count = 5150

4.4.3 Multiplication time

Let the multiply command pulse occur simltaneously with the read-in pulse. Then:

1. $t_{max} = t_{min} = nT + \delta_1 + \delta_2 + 5\gamma - \tau \quad (16 \geq n \geq 9)$

2. T must be chosen so that the flow of clock pulses will be reliably stopped, and so that sufficient time will be allowed for the maximum time-lapses. In this case it is the latter requirement which determines T:

$T \geq \gamma + \tau + 2i + 5\alpha + 4\omega$: Let T = 0.85 μsec

$\delta_1 \geq 4\gamma + \tau + 2i + (2n - 1)(\alpha + \omega) - 5\gamma$: Let δ_1 = 1.3 μsec if n = 16

$\delta_2 \geq \gamma + \tau$: Let δ_2 = 0.55 μsec

3. If n = 16, $t_{ave} \cong$ 15.2 μsec.

4.5 The K-acs B2 multiplier - Fig. IV-10

4.5.1 The operation

The maximum time-lapses are found from Fig. IV-10 to be:

acs:acs	$\gamma + \tau + \alpha + \omega$
acs:cs	$2\gamma + \tau + \alpha + \omega$
cs:acs	$\tau + \alpha + \omega$
cs:cs	$\gamma + \tau + \alpha + \omega$
acs:c	$2\gamma + \tau$
cs:c	$\gamma + \tau$

A K type control is included in Fig. IV-10. The delay unit D_3 in this control allows the multiplication process to be stopped at the proper step without increasing the clock pulse period, T. Since it is not necessary to make T larger than is necessary for the maximum time-lapses and since these latter are very nearly equal, a C type control will not be considered.

The multiplier is quite straightforward in its operation; there are no unusual details which require discussion.

4.5.2 Equipment count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\geq 1 + \log_2 (n-1)$	$4n - 1$	68
GT	$\geq 1 + \log_2 (n-1)$	$10n - 6$	154
$\&_2$	0	$4(n-1)$	60
OR_2	0	$2(n-1)$	30
M_2	0	$3n - 2$	46
D	2	0	2

Total Weighted Equipment Count = 4752

4.5.3 Multiplication time

Let the multiply command pulse occur simultaneously with the read-in pulse. Then:

1. $t_{max} = t_{min} = nT + \delta_1 + \delta_2 - \tau$
2. $T \geq 2\gamma + \tau + \alpha + \omega$: Let $T = 0.6\mu sec$.

   $\delta_1 \geq T + 2\gamma + \tau - 5\gamma$: Let $\delta_1 = 1.1$ μsec $(16 \geq n \geq 9)$

   $\delta_2 \geq (n-1)\gamma + \tau$: Let $\delta_2 = 1.1$ μsec if $n = 16$

$$\begin{cases} 5\gamma + \delta_3 \geq T \\ 5\gamma + \delta_3 + \tau \geq 2T \end{cases}$$

$$\text{Let } \delta_3 = \frac{\delta_M + \delta_m}{2} = \frac{2T - 5\gamma - \tau + T - 5\gamma}{2} = 0.45 \text{ μsec}$$

3. If $n = 16$, $t_{ave} = 11.3$ μsec

4.6 The C-a.c.s B multiplier -- Figs. IV-11 and IV-12

4.6.1 The operation

The maximum time-lapses are:

a:c	$2\gamma + \tau$
c:s	$(n-1)\gamma + \tau$
s:a	τ
s:s	$\gamma + \tau$

There is a significant difference between the maximum time-lapse values if n has any reasonable value. Therefore a C type control is dictated. The control to be used is shown in Fig. IV-11.

The gate on the "1" side of the control flip-flop is used to supply the multiplication complete pulse in place of using a delayed end-carry since in half the cases this will save $\delta_1 + \delta_2$ μsec.

The arithmetic section (omitting the B register) is shown in Fig. IV-12. This realization operates in a straightforward manner. It might be mentioned, though, that the shift pulse need not operate on the carry register since the preceding high-speed carry operation will always leave this register cleared.

4.6.2 Equipment count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\geq 1 + \log_2 (n-1)$	$4n - 1$	68
BT	$\geq 2 + \log_2 (n-1)$	$8n - 4$	130
M_2	2	$3n - 4$	46
D	3	0	3

Total Weighted Equipment Count = 4172

4.6.3 Multiplication time

Let the multiply command pulse occur τ μsec. after the read-in pulse.

Then:

1. $t_{max} = n(2\gamma + 2\mu + \int_1 + \int_2 + \int_3)$

   $t_{min} = n(2\gamma + 2\mu + \int_3)$

2. $\int_1 \geq a{:}\underline{c} = 2\gamma + \tau$: Let $\int_1 = 0.6$μsec.

   $\int_2 \geq \underline{c}{:}s = (n-1)\gamma + \tau$: Let $\int_2 = 1.1$ μsec if $n = 16$

   $\int_3 \geq 2\gamma + \tau$: Let $\int_3 = 0.6$ μsec. $(16 \geq n \geq 9)$

3. If $n = 16$, $t_{ave} \cong 24.8$ μsec.

4.7 The C-as.c B multiplier -- Figs. IV-14 and IV-15

4.7.1 The operation

The maximum time-lapses are:

$as{:}\underline{c}$	$2\gamma + \tau$
$\underline{c}{:}as$	$(n-1)\gamma + \tau$
$\underline{c}{:}s$	$n\gamma + \tau$
$s{:}as$	$\mu + \tau$
$s{:}s$	$\mu + \gamma + \tau$

The significant differences between these maximum time-lapses forces the use of a C type control.

The operation of the control shown in Fig. IV-14 is very similar to that of the control discussed in section 4.6.1. The only change is that required to supply the commands in proper sequence.

An alternate mixed "K-C type" control is shown in Fig. IV-13. The

major disadvantage of this type of control is that to obtain a maximum speed when using it, close timing between the reset pulse to the control flip-flop and the next clock pulse is required.

The arithmetic section of C-as.c B (omitting the B register) is shown in Fig. IV-15. Since whenever $B_i = 1$ a shift operation (combined with a partial add) is performed before the c operation, the carries generated by this operation must be added into the accumulator flip-flop shown rather than into the next one to the left, as in the ordinary case.

Whenever a shift occurs, a "c" has preceded it and cleared the carry register. Therefore, the "s" operation need not act upon the carry register.

A flip-flop is not needed for carry storage in digit column number one since the Ac_1 flip-flop can be used for this purpose. Any high-speed carry pulse can also be sent directly to this flip-flop without danger of carry overlap. This follows from the known fact that the product of two numbers of n and p digits respectively cannot exceed n + p digits, and from the fact that p shifts have been performed before the "c" which forms the product of $A \cdot \left[B_{n-(p-1)} \cdots B_n\right]$ is performed.

4.7.2 Equipment count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\geq 1 + \log_2 (n-1)$	4n - 1	68
GT	$\geq 2 + \log_2 (n-1)$	10n - 2	164
M_2	2	2n + 1	35
D	3	0	3

Total Weighted Equipment Count = 4522

4.7.3 Multiplication time

Let the multiply command pulse occur μsec. after the read-in pulse.

Then:

1. $t_{max} = n(2\gamma + 2\mu + \delta_1 + \delta_2$

   $t_{min} = n(2\gamma + 2\mu + \delta_3$

2. $\delta_1 \geq 2\gamma + \tau$: Let $\delta_1 = 0.6$ μsec.

   $\delta_2 \geq (n-2)\gamma + \tau$: Let $\delta_2 = 1.1$ μsec if $n = 16$

   $\delta_3 \geq 2\gamma + \tau$: Let $\delta_3 = 0.6$ μsec $(16 \geq n \geq 9)$

3. If $n = 16$, $t_{ave} \cong$ 20 μsec.

4.8 The C-ac.s Bl multiplier -- Figs. IV-16 and 17

4.8.1 The operation

The maximum time-lapses are:

ac:s	$\gamma + \tau$
s:ac	$\gamma + \tau + n\alpha + (n-1)w + i$
s:s	$\gamma + \tau$

A C type control is necessary for maximum speed. A possible control is shown in Fig. IV-16. It is very similar to the control of Fig. IV-14 (discussed in section 4.7.1) except for the additional delay in the $B_i = 1$ input line. This delay is necessary since s:ac is longer than s:s.

The arithmetic section (omitting the B register) is shown in Fig. IV-17. Its operation is straightforward.

4.8.2 Equipment count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\geq 1 + \log_2 (n-1)$	$3n + 1$	54
GT	$\geq 2 + \log_2 (n-1)$	$5n + 3$	89
$\&_2$	0	$5n - 4$	76
OR_3	0	$n - 1$	15
OR_2	0	$n - 1$	15
I	0	$n - 1$	15
M_2	2	0	2
D	3	0	3

Total Weighted Equipment Count = 3868

4.8.3 Multiplication time

Let the multiply command pulse occur $\tau + \alpha + \omega$ μsec. after the read-in pulse. Then:

1. $t_{max} = n (2\gamma + 2\mu + \int_1 + \int_2 + \int_3)$

   $t_{min} = n (2\gamma + 2\mu + \int_2)$

2. $\int_1 \geq \gamma + \tau$: Let $\int_1 = 0.55$ μsec.

   $\int_2 \geq 2\gamma + \tau$: Let $\int_2 = 0.6$ μsec $(16 \geq n \geq 9)$

   $\int_3 \geq \gamma + \tau + i + (n-1)(\alpha + \omega) + \alpha - 2\gamma - \int_2$: Let $\int_3 = 0.3$ μsec if n = 16

3. If n = 16, $t_{ave} \cong$ 18 μsec

4.9 The C-ac.s B2 multiplier -- Figs. IV-16 and 18

4.9.1 The operation

The maximum time-lapses, obtained from Fig. IV-18 are:

ac:s	$n\gamma + \tau + \int$
s:ac	τ
s:s	$\gamma + \tau$

A C type control is necessary; a possible one is shown in Fig. IV-16. Note that the D_3 delay line unit is not needed when this control is used for the C-ac.s B2 multiplier.

The arithmetic section (omitting the B register) is shown in Fig. IV-18. This arithmetic section is identical to that of the K-a.c.s B multiplier (Fig. IV-12) except that here each carry flip-flop (except C_o) with its associated gate tube has been replaced by a delay line. This change permits the high-speed carry tobe performed without the need of aseparate "c" pulse. More important, it results in a considerable reduction in the equipment count. However, this change will not allow a very great decrease in the multiplication time of C-ac.s B2 over that of C-a.c.s B since the flip-flop transition time, τ, must still elapse before the carry pulses can emerge from the delay lines. (Therefore $\int = \tau$.)

The C_o flip-flop cannot be replaced by a single delay unit for a rather complicated reason. The argument is as follows. It is easy to find an example of multiplication for which $Ac_1 = 1$ and $A_1 = 1$ at a particular time. Suppose an "ac" operation is now performed. A carry digit to column 0 will then be generated, and if the carry flip-flop has been replaced by a delay $\int_o$, whose output goes to the set input of Ac_1, then $\int_o$ must equal $\int_1$ of Fig. IV-16 since: (1) the carry must not be permitted to destroy the content of Ac_1 until this content has been shifted to Ac_2, and (2) the carry must arrive soon enough to permit the flip-flop to stabilize before the shift pulse occurs. On the other hand suppose that $A_1 = 0$ and $Ac_1 = 1$ and that a carry comes into the first digit column $\int$ microseconds after the "ac" pulse. Again a carry to digit column zero will be generated, but now this carry must experience a delay $\int_o = \int_1 /2$ for the same two reasons as before. Since the delay thru a delay line unit is just one specific value, the C_o flip-flop cannot be replaced by a single delay line unit. However, it can be replaced by two delay units, a gate tube and a two input pulse mixer, as shown in Fig. IV-19. That this circuit will operate correctly can be checked by considering all the cases which can occur. These are outlined as follows.

I. $Ac_1 = 0$

A. The next pulse is on the shift line.

B. The next pulse is on the "ac" line.

1. a carry into the 1st digit column from the "ac" pulse.

a. $A_1 = 1$

b. $A_1 = 0$

2. no carry into the 1st digit column from the "ac" pulse.

a. $A_1 = 1$

b. $A_1 = 0$

II. $Ac_1 = 1$

A. The next pulse is on the shift line.

B. The next pulse is on the "ac" line.

1. a carry into the 1st digit column from the "ac" pulse.

a. $A_1 = 1$

b. $A_1 = 0$

2. no carry into the 1st digit column from the "ac"pulse.

a. $A_1 = 1$

b. $A_1 = 0$

If the operation of the circuit in Fig. IV-19 is checked in each one of these cases it will be found that the correct result must occur in every case except case II. 1. a. However, it can easily be established from the given initial condition that the accumulator register contains all zeros that this case can never occur. The argument follows.

Case II . 1. a. can arise from the "ac" pulse only if the configuration of the Ac and A digits in the first two columns previous to the pulse is as follows:

r \ c	1	2
Ac	1	1
A	1	1

c: digit column

r: flip-flop register

The initial configuration of Ac is always:

r \ c	1	2
Ac	0	0

The initial configuration of A cannot be changed during the multiply operation and therefore since we wish to end up with an A configuration of

r \ c	1	2
A	1	1

This must also be the initial configuration. Now consider what can happen within the multiplier as the multiply operation begins. A shift could occur, but this is of no interest since it will not change Ac_1 or Ac_2. Suppose then that an "ac" pulse occurs. The result will be :

r \ c	1	2
Ac	1	1

At least one shift must intervene before another "ac" pulse can occur. After one shift:

r \ c	1	2.
Ac	0	1

This is not case II. 1. a. Another shift would return us to the initial situation, so assume that an "ac" pulse is now performed:

r \ c	1	2
Ac	0	0

and $C_o = 1$. After one shift:

r \ c	1	2
Ac	1	0

. Again this is not case II. 1. a. Another shift would return us to the immediately preceding situation, and so the only chance for case II. 1. a. to arise must come from another "ac" operation being performed at this time. Then:

r \ c	1	2
Ac	0	1

and $C_o = 1$. After one shift:

r \ c	1	2
Ac	1	0

. Clearly we have entered a cycle in which Case II. 1. a. will never occur.

4.9.2 Equipment count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Contrd Section	Arithmetic Section	
FF	$= 1 + \log_2 (n - 1)$	$3n + 1$	54
GT	$= 2 + \log_2 (n - 1)$	$7n + 1$	119
M_2	2	$n - 1$	17
D	2	$n - 1$	17

Total Weighted Equipment Count = 3724

4.9.3 Multiplication time

Let the multiply-command pulse occur τ microseconds after the read-in pulse. Then:

1. From section 4.8.3 with $\int_3 = 0$:

   $t_{ave} = n\,(2\gamma + 2\mu + 1/2\,\int_1 + \int_2)$

2. $\int \geq \tau$: Let $\int = 0.5$ μsec

   $\int_1 \geq n\gamma + \tau + D$: Let $\int_1 = 1.65$ μsec if $n = 16$

   $\int_2 \geq 2\gamma + \tau$: Let $\int_2 = 0.6$ μsec $(16 \geq n \geq 9)$

3. If $n = 16$, $t_{ave} \cong$ 24.4 μsec

4.10 The K-ac.s D1 and C-ac.s D1 multipliers -- Figs. IV-20, 21 and 22

4.10.1 The operation

The maximum stage time-lapses are found from Fig. IV-22 to be:

$(\underline{ac}{:}s)_S$	$\gamma + \tau$
$(s{:}\underline{ac})_S$	$2\gamma + \tau + \alpha$
$(s{:}s)_S$	$2\gamma + \tau$

Also from Fig. IV-22, the per-stage delays are:

	σ
$\underline{ac}$	γ
s	$\gamma + \mu$

Since the per-stage delays are very nearly equal and the maximum stage time-lapses are essentially equal, the maximum time-lapses are very nearly equal and therefore both K and C types controls must be considered. A K type control is shown in Fig. IV-20 and a C type in Fig. IV-21.

The arithmetic section (omitting the B register) is shown in Fig. IV-22. The operation is straightforward.

4.10.2 Equipment count

Basic Circuit	Number Used - n digits			Total Used n = 16	
	Control Section		Arithmetic Section	K	C
FF	$\geq 1 + \log_2 n$		3n + 1	54	
GT	$\geq 1 + \log_2 n$		10n - 3	162	
OR_2	0		2n - 1	31	
$\&_2$	0		2n - 1	31	
M_2	0	2	n + 1	17	19
D	1	3	0	1	3

Total Weighted Equipment Count = { 4082 for K-acs D1 ; 4126 for C-acs D1

4.10.3 Multiplication time

A. K-acs D1 (Control of Fig. IV-20)

Let the multiply command pulse occur simultaneously with the read-in pulse:

1. $t_{max} = 2nT + \int - \tau$

   $t_{min} = nT + \int - \tau$

2. T must be determined so that the multiplication process will be stopped reliably.

   $T \geq 6\gamma + \tau$: Let T = .75 μsec. $(16 \geq n \geq 9)$

   $\int \geq n\gamma + (n-1)\mu + \tau - 4\gamma$: Let $\int$ = 1.05 μsec. if n = 16

3. If n = 16, $t_{ave} \cong$ 18.5 μsec.

B. C-acs D1 (Control of Fig. IV-21)

Let the multiply command pulse occur $\tau + \alpha$ μsec. after the read-in pulse. Then:

1. $t_{max} = n(2\gamma + 2\mu + \int_1 + \int_2) + 3\gamma + \int_3$ $(16 \geq n \geq 9)$

   $t_{min} = n(2\gamma + 2\mu + \int_2) + 3\gamma + \int_3$ $(16 \geq n \geq 9)$

2. $\int_1 \geq$ acs = $\gamma + \tau$: Let $\int_1$ = 0.55 μsec.

   $\int_2 \geq 4\gamma + \tau$: Let $\int_2$ = 0.7μsec. $(16 \geq n \geq 9)$

   $\int_3 \geq n\gamma + (n-1)\mu + \tau - 4\gamma$: Let $\int_3$ = 1.05 μsec. if n = 16

3. If n = 16, $t_{ave} \cong$ 18.4 μsec.

Since the two controls give nearly equal multiplication times, let us choose the K type in order to avoid the critically located delay line units of the C type.

4.11 The C-ac.s D2 multiplier -- Figs. IV-23 and 24

4.11.1 The operation

An inspection of the arithmetic section shown in Fig. IV-24 shows that per-stage delay for the "ac" command has no fixed value, but may be either γ or 2γ. This indicates that the C type control which uses the completion pulses from the arithmetic section (instead of outputs from delay lines) may provide the fastest operation. The value of t_{ave} for the two C type controls of Figs. IV-16 and IV-23 are calculated below. The maximum stage time-lapses and the per-stage delays will be needed:

$(ac{:}s)_S$	$\gamma + \tau$
$(s{:}ac)_S$	$2\gamma + \tau$
$(s{:}s)_S$	$2\gamma + \tau$

	σ
ac	γ or 2γ
s	$\gamma + \mu$

If the C type control of Fig. IV-16 is used, then from the expressions developed in section 4.8.3:

$$t_{ave} = n\left(2\gamma + 2\mu + \frac{\int_1 + \int_3}{2} + \int_2\right)$$

It remains to be determined whether or not the $\int_3$ delay is necessary. The maximum time-lapses are

$$ac{:}s = n(^{\sigma}_{ac\ max}) + \tau\,(n-1)^{\sigma}_{S} = (n+1)\gamma + \tau$$

$$s{:}ac = 2\gamma + \tau$$

$$s{:}s = 2\gamma + \tau$$

Since $s{:}s = s{:}ac$, $\int_3 = 0$. The other delay line values are:

$\int_1 \geq (n+1)\gamma + \tau$: Let $\int_1 = 1.2$ μsec. if $n = 16$

$\int_2 \geq 2\gamma + \tau$: Let $\int_2 = 0.6$μsec. $(16 \geq n \geq 9)$

Then $t_{ave} \approx 20.7$ μsec when n = 16.

If the control of Fig. IV-23 is used, then n must satisfy:

minimum total delay of the "ac" pulse $\geq (s\text{-}ac)_S$ and $(s - s)_S$

$$n\gamma \geq 2\gamma + \tau$$

$$n \geq 2 + \frac{\tau}{\gamma}$$

$$n \geq 15$$

If n<15, then the "ac" pulse must be delayed an additional amount before emerging from the multiply-adder.

Let $\int_{ac}$ denote the delay experienced by the "ac" pulse in passing thru the arithmetic section and $\int_S$ denote the delay experienced by the "S" pulse. Then:

$$t_{max} = (2\gamma + 2\mu + \int_{ac\ max} + \int_S)\, n$$

$$t_{min} = (2\gamma + 2\mu + \int_S)\, n$$

where $\int_{ac\ max} = 2\gamma n$

$$\int_S = \gamma(n - 1)$$

Every time that an "ac" pulse occurs in the multiplication procedure the value of $\int_{ac}$ is likely to be different. The maximum value of $\int_{ac}$ is $2\gamma n$, while the minimum is γn. If we first take the average of the extremes of the multiplication time when B is all one's, and then average this with the multiplication time when B is all zero's (which does not depend on $\int_{ac}$) the result is:

$$t_{ave,\ ave} = (2\gamma + 2\mu + \int_s + \frac{\int_{ac\ max} + \int_{ac\ min}}{4})\, n$$

$$= (2 + n - 1 + \frac{2n + n}{4})\, \gamma n + 2\mu n$$

$$= (\frac{7}{4} n + 1)\gamma n + 2\mu n$$

If n = 16, then

$$t_{ave,\ ave} \approx 18.8 \text{ μsec.}$$

Although this figure may not have the same significance as the t_{ave} for the control of Fig. IV-16, namely the multiplication time most likely to occur, it is approximately (if not exactly) what this figure would be for the control of Fig. IV-23. At any rate this latter control is most likely slightly faster than the first, and if not then the overall improvement in speed which would be due to using the first rather than the second would certainly be insignificant.

The arithmetic section (Fig. IV-24) is simple in its logical construction and nothing further need be said of its operation.

4.11.2 Equipment count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\geq 1 + \log_2 (n-1)$	$3n + 1$	54
GT	$\geq 2 + \log_2 (n-1)$	$10n$	166
M_2	2	n	18

Total Weighted Equipment Count = 3856

4.11.3 Multiplication time

If the multiply command pulse occurs τ microseconds after the read-in pulse, then the value calculated in section 4.11.1 can be used, namely 18.8 μsec.

4.12 The K-ac.s D3 and C-ac.s D3 multipliers -- Figs. IV-20, 21, and 25

4.12.1 The operation

The maximum stage time-lapses are found from Fig. IV-25 to be:

$(\underline{ac}:s)_S$	$\gamma + \tau$
$(s:\underline{ac})_S$	$2\gamma + \tau + \alpha\, 2\omega$
$(s:s)_S$	$2\gamma + \tau$

Also from Fig. IV-25, the per-stage delays are:

	σ
ac	γ
s	$\gamma + \mu$

Since the per-stage delays are very nearly equal and the maximum stage time-lapses are essentially equal, the multiplication times provided by K and C type controls (Figs. IV-20 and 21) must be calculated and compared before a choice can be made.

The arithmetic section (Fig. IV-25) is different from any other D type section of the Class I multipliers in that two digits are handled simultaneously, i.e., a stage consists of two digit columns. This requires a rather involved arrangement of gates, although the logical design of this circuit is entirely straightforward when Boolean techniques are used.

4.12.2 Equipment count

Basic Circuit	Number Used - n digits			Total Used n = 16	
	Control Section		Arithmetic Section		
	K	C		K	C
FF	$\geq 1 + 16g_2\, n$		$3n + 1$	54	
GT	$\geq 1 + \log_2 n$		$8n - 2$	131	
OR_2	0		$5n - 3$	77	
$\&_3$	0		$2\,(n - 1)$	30	
$\&_2$	0		$3n - 2$	46	
M_2	0	2	$n/2 + 1$	9	11
D	1	3	0	1	3

Total Weighted Equipment Count = 4180 for K-ac.s D3
4224 for C-ac.s D3

4.12.3 Multiplication time

A. K-ac.s D3 (Control of Fig. IV-20)

Let the multiply command pulse occur simultaneously with the read-in pulse. Then:

1. $t_{ave} = 3/2\, nT + \int - \tau$

2. T must be determined so that the multiplication process will be stopped reliably.

$T \geq 6\mathcal{V} + \tau$: Let T = .75 μsec ($16 \geq n \geq 9$)

$\int \geq (n/2 - 1)\,(\mathcal{V} + \mu) + \tau - 3\mathcal{V}$: Let $\int$ = .75 μsec. if n = 16

3. If n = 16, $t_{ave} \cong$ 18.1 μsec

B. C-ac.s D3 (Control of Fig. IV-21)

Let the multiply command pulse occur $\tau + 2\omega + \alpha$ after the read-in pulse. Then:

1. $t_{ave} = n(2\gamma + 2\mu - 1/2\int_1 + \int_2) + 3\gamma + \int_3 \quad (16 \geq n \geq 9)$
2. $\int_1 \geq$ ac:s $= \gamma + \tau$: Let $\int_1 = .55$ μsec.
   $\int_2 \geq 4\gamma + \tau$: Let $\int_2 = .7$ μsec. $(16 \geq n \geq 9)$
   $\int_3 \geq (n/2 - 1)(\gamma + \mu) + \tau - 3\gamma$: Let $\int_3 = .75$ μsec if $n = 16$
3. If $n = 16$, $t_{ave} \cong 18.1$ μsec.

4.13 The C-asc B multiplier: Figs. IV-26 and IV-27

4.13.1 The operation

The maximum time-lapses are deduced from Fig. IV-27 to be:

asc:asc	$(n+1)\gamma + \int + \mu + \tau$
asc:s	$n\gamma + \int + \mu + \tau$
s:asc	$\mu + \tau$
s:s	$\mu + \gamma + \tau$

In addition, it is found that the delay, $\int$, should equal the flip-flop transistion time, τ. Since the maximum time-lapses differ by a considerable amount, a C type control will allow the faster operation. A control which can be used with this arithmetic section is shown in Fig. IV-26.

It should be noted that the arithmetic section for this multiplier (Fig. IV-27) is the same as that for the C-as.c B multiplier (Fig. IV-14) with all the carry flip-flops and their associated gate tubes replaced by delay lines. This same device was used to obtain the C-ac.s B2 multiplier from the C-a.c.s B multiplier. As there, the new multiplier will operate slightly faster than the original one and a considerable amount of equipment will be saved.

4.13.2 Equipment count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\cong 1 + \log_2 (n-1)$	3n	53
GT	$\cong 2 + \log_2 (n-1)$	9n - 1	149
M_2	2	2n - 1	33
D	2	n	18

Total Weighted Equipment Count = 4036

4.13.3 Multiplication time

Let the multiply command pulse occur τ microseconds after the read-in pulse. Then:

1. $t_{max} = n(2\gamma + 2\mu + \int_1)$

   $t_{min} = n(2\gamma + 2\mu + \int_2)$

2. $\int_1 \geq \tau$: Let $\int = 0.5$ μsec.

   $\geq (n+1)\gamma + \int + \mu + \tau$: Let $\int_1 = 1.7$ μsec if $n = 16$

   $\int_2 \geq 2\gamma + \tau$: Let $\int_2 = 0.6$ μsec. $(16 \geq n \geq 9)$

3. If $n = 16$, $t_{ave} \cong$ 20 μsec.

4.14 The C-acs B1 multiplier: Figs. IV-28 and 29

4.14.1 The operation

The maximum time-lapses are found from Fig. IV-29 to be as follows (provided that the control used initiates the shift of the B register simultaneously with the shift command to the multiply-adder).

acs:acs	$\gamma + \tau + n(\alpha + \omega) + 2i$
acs:s	$\gamma + \tau + n(\alpha + \omega) + \alpha + 2i$[7]
s:acs	"
s:s	$\gamma + \tau + 2i + 2(\alpha + \omega)$

The fact that the maximum time-lapses for two command-pairs which begin with a shift command differ significantly shows that a C type control would be speedier than a K type. However, the very fact that it is these two command-pairs whose time lapses differ requires an extra flip-flop and two extra gate tubes for the C type control to take advantage of the shorter time required by the shift, shift command-pair. The necessary arrangement is shown in Fig. IV-28.

The arithmetic section of this multiplier is somewhat similar to that of the K-acs B1 multiplier (Fig. IV-9) in that it requires a level from the n^{th} flip-flop of the B register in order to effectively change the interconnections within the multiply-adder. The result of this interchange is to convert the "acs" command input into an "s" input. The two gate tubes on the n^{th} B register

[7] This is based on the assumption that it takes just about as long for a fall in the carry level to propagate thru the multiply-adder as it does for a rise to propagate thru.

flip-flop are again exchanged for the diode "and" circuits of Fig. IV-29 as part of the digit forming equipment.

4.14.2 Equipment count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\geq 2 + \log_2 (n-1)$	3n	54
GT	$\geq 3 + \log_2 (n-1)$	4n + 2	73
+3,5'	0	n - 1	15
+2,5'	0	1	1
ℓ_2	0	n	16
M_2	4	0	4
D	3	0	3

Total Weighted Equipment Count = 3924

4.14.3 Multiplication time

Let the multiply command pulse occur $\tau + (n+1)(\alpha + \omega) + 2i + \alpha$ after the read-in pulse. Then:

1. $t_{max} = (2\gamma + 2\mu + \int_1)(n-1) + \int_3 + (n/2)\delta + (n+1)(\alpha+\omega) + 2i$ [8]
   $t_{min} = (3\gamma + 2\mu + \int_2)(n-1) + \int_3 + (n+1)(\alpha+\omega) + 2i$
2. $\int_1 \geq 2\gamma + \tau + 2i + n(\alpha+\omega) + \alpha - 2$: Let $\int_1$ = 1.0 μsec. if n = 16
   $\int_2 \geq \gamma + \tau + 2i + 2(\alpha+\omega) - 3\delta$: Let $\int_2$ = 0.7 μsec.
   $\int_3 \geq \int_1 + \tau - \gamma$: Let $\int_3$ = 1.5 μsec. if n = 16
3. If n = 16, $t_{ave} \geq$ 16.2 μsec.

4.15 The C-acs B2 multiplier: Figs. IV-30 and 31

4.15.1 The operation

The maximum time-lapses are found from Fig. IV-31 to be:

acs:acs	$\gamma + \tau + 2i + (n-1)(\alpha+\omega) + \alpha$
acs:s	$\gamma + \tau$
s:acs	$\gamma + \tau + 2i + (n-1)(\alpha+\omega) + \alpha$
s:s	$\gamma + \tau$

8. This value occurs when B = 10101..... and n is even.

A C type control must be used for optimum speed because of the significant differences in these values. The control of Fig. IV-30 will take advantage of the shorter maximum time-lapses.

The arithmetic section for this multiplier is identical to that for the C-acs B1 multiplier except that shift gates have been added. This addition means that the multiply-adder no longer needs an add level from control not the associated diode "and" gates. The result of this change is to decrease the multiplication time at the expense of increasing the equipment count.

4.15.2 Equipment count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\geq 1 + \log_2 (n-1)$	3n	53
GT	$\geq 2 + \log_2 (n-1)$	6n + 2	104
+3,5'	0	n - 1	15
+2,5'	0	1	1
M_2	2	2n + 1	35
D	2	0	2

Total Weighted Equipment Count = 4172

4.15.3 Multiplication time

Let the multiply command pulse occur $\tau + n(\alpha + \omega) + 2i + 2\alpha + \omega$ after the read-in pulse. Then:

1. $t_{max} = n(2\gamma + 2\mu + \delta_1 + \delta_2) + (n+1)(\alpha + \omega) 2i$

   $t_{min} = n(2\gamma + 2\mu + \delta_2 + (n+1)(\alpha + \omega) + 2i$

2. $\delta_1 \geq \gamma + \tau + 2i + (n-1)(\alpha + \omega) + \alpha - \delta_2 - 2\gamma - 2\mu$; Let $\delta_1 = 0.35$ μsec. if n = 16

   $\delta_2 \geq 2\gamma + \tau$; Let $\delta_2 = 0.6$ μsec. ($16 \geq n \geq 9$)

3. If n = 16, $t_{ave} \approx$ 14.9 μsec.

4.16 The K-acs D multiplier: Figs. IV-32 and 33

4.16.1 The operation

The maximum stage time-lapses and the per-stage delays are found from Fig. IV-33 to be:

$(\underline{acs}:\underline{acs})_S$	$2\delta + \tau + 2\mu + \alpha + \omega$
$(\underline{acs}:s)_S$	$2\delta + \tau + 2\mu$
$(s:\underline{acs})_S$	$2\delta + \tau + 2\mu + \alpha + \omega$
$(s:s)_S$	$2\delta + \tau + 2\mu$

	σ
$\underline{acs}$	$\gamma + \mu$
s	$\gamma + \mu$

Since the per-stage delays are equal and the maximum stage time-lapses are essentially equal, either a C or a K type control can be used. However, since the K type control of Fig. IV-32 allows the clock pulse period to be chosen without regard to reliable stopping of the multiplication, a C type control does not appear to be of sufficient promise to be considered.

The arithmetic section (Fig. IV-33) is straightforward in its design and its operation.

4.16.2 Equipment Count

Basic Circuit	Number used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\geq 1 + \log_2 (n-1)$	3n	53
GT	$\geq 1 + \log_2 (n-1)$	10n - 3	162
δ_2	0	4n	64
CR_2	0	n	16
M_4	0	2n - 1	31
M_3	0	2n - 1	31
M_2	0	n + 4	20
D	2	0	2

Total Weighted Equipment Count = 4347

4.16.3 Multiplication time

Let the multiply command pulse occur simultaneously with the read-in pulse. Then:

1. $t_{max} = t_{min} = t_{ave} = nT + \int_1 + 4\gamma - \tau \quad (16 \geq n \geq 9)$
2. $T \geq 2\gamma + \tau + 2\mu + \alpha + \varepsilon$: Let $T = 0.6$ μsec.

$$\begin{cases} \int_1 \geq \tau + 2\mu & : \text{Let } \int_1 = 0.55 \text{ μsec.} \quad (16 \geq n \geq 9) \\ \int_2 \geq \tau - 5\gamma & \\ \int_2 \geq 2T - 5\gamma - \tau & : \text{Let } \int_2 = 0.45 \text{ μsec.} \quad (16 \geq n \geq 9) \end{cases}$$

3. If $n = 16$, $t_{ave} \simeq$ 9.8 μsec.

4.17 The C-$\bar{c}$.a.s multiplier: Figs. IV-11 and 34

4.17.1 The operation

The maximum time-lapses are found from Fig. IV-34 to be:

$\bar{c}$:a	$(n-2)\gamma + \rho^{9}_{f}$
a:s	$\gamma + \tau$
s:$\bar{c}$	$\gamma + \tau + \alpha$[10]
s:s	$\gamma + \tau$

Since there is a significant difference in these values, the C type control of Fig. IV-11 should be used.

The arithmetic section (Fig. IV-34) was designed from Table III-3, p._____ through the use of Boolean techniques. The control of the two gate tubes feeding the $\bar{c}$ line in each digit column is such that both can pass a pulse at a particular step of the multiplication.

---

9 A full-length carry is possible; e.g., when A = 101011 and B = ...11.

10 Since $\alpha \ll \gamma$: however, both are so small that this point is trivial.

4.17.2 Equipment Count

Basic Circuit	Number Used - n digits		Total Used n = 16
	Control Section	Arithmetic Section	
FF	$\geq 1 + \log_2 (n-1)$	$3n + 1$	54
GT	$\geq 2 + \log_2 (n-1)$	$7n + 1$	119
$\&_2$	0	$3n - 2$	46
OR_2	0	$n - 1$	15
M_2	2	$n - 1$	17
D	3	0	3

Total Weighted Equipment Count = 3684

4.17.3 Multiplication time

Let the multiply command pulse occur $\tau + \alpha + \varepsilon$ microseconds after the read-in pulse. Then:

1. $t_{max} = n(2\beta + 2\mu + \delta_1 + \delta_2 + \delta_3)$

   $t_{min} = n(2\beta + 2\mu + \delta_3)$

2. $\delta_1 \geq (n-2)\gamma + \rho$: Let $\delta_1 = 0.8$ μsec. if $n = 16$

   $\delta_2 \geq \gamma + \tau$: Let $\delta_2 = 0.55$ μsec.

   $\delta_3 \geq 2\beta + \tau$: Let $\delta_3 = 0.6$ μsec. $(16 \geq n \geq 9)$

3. $t_{ave} \cong 22$ μsec.

A-56157

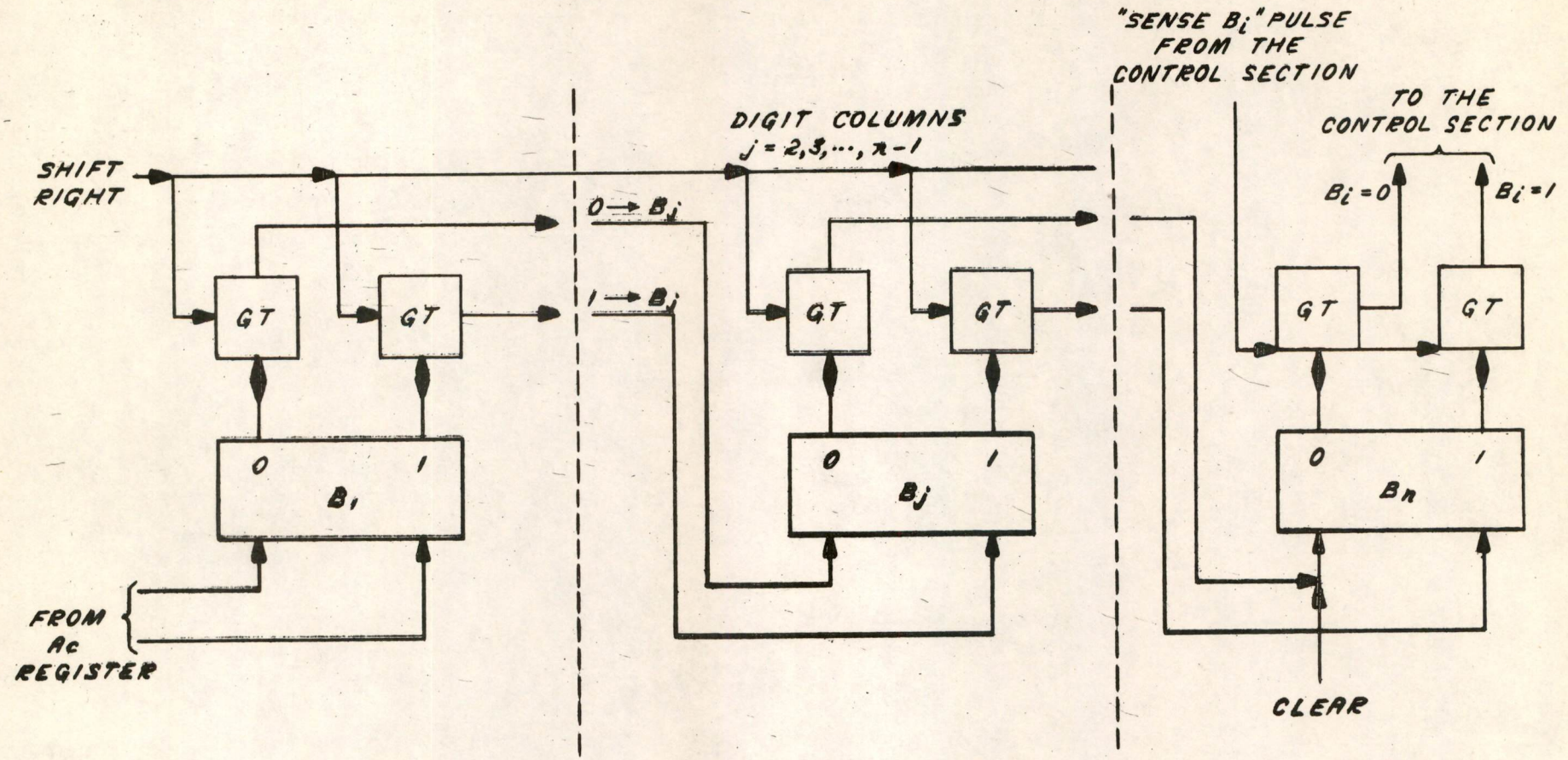


FIG. IV-1

THE B REGISTER FOR THE CLASS I MULTIPLIERS

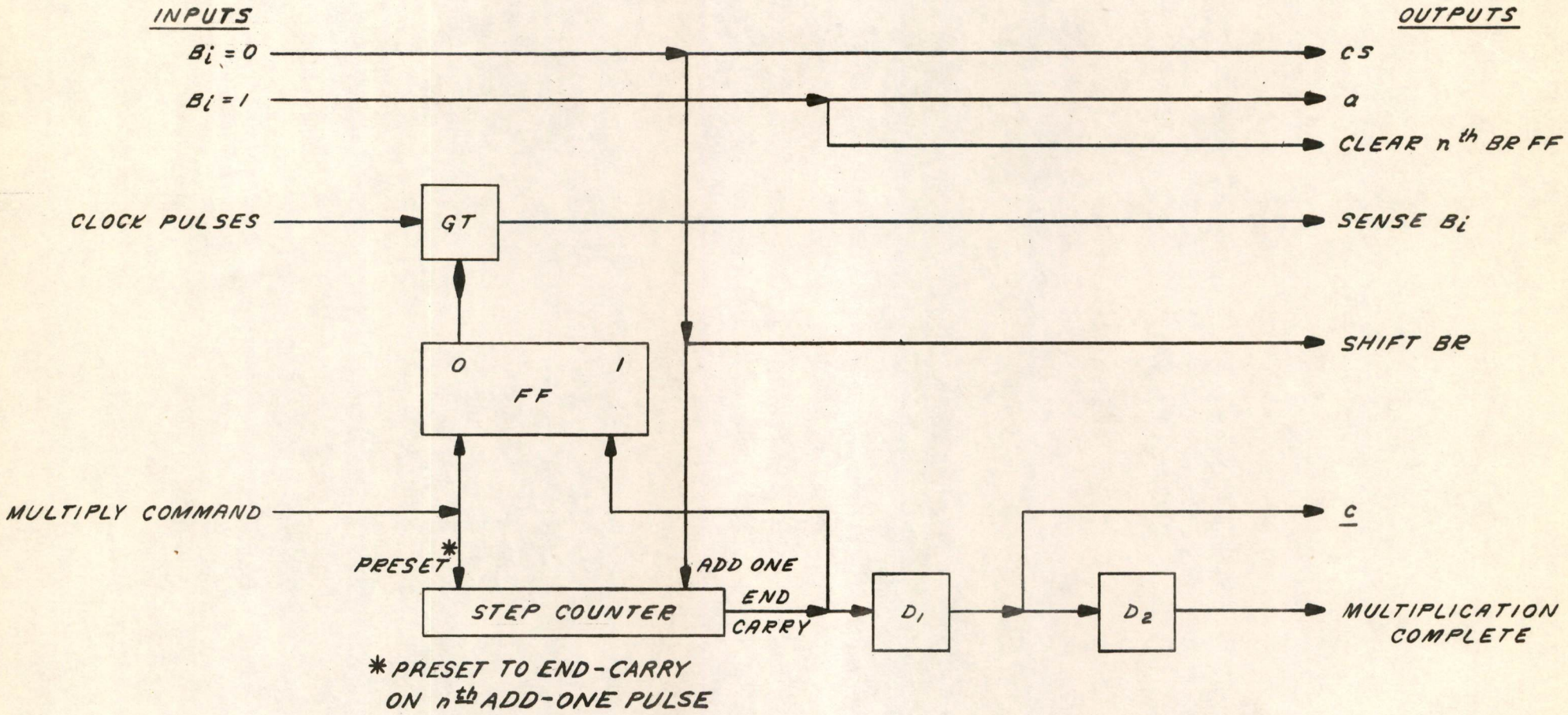
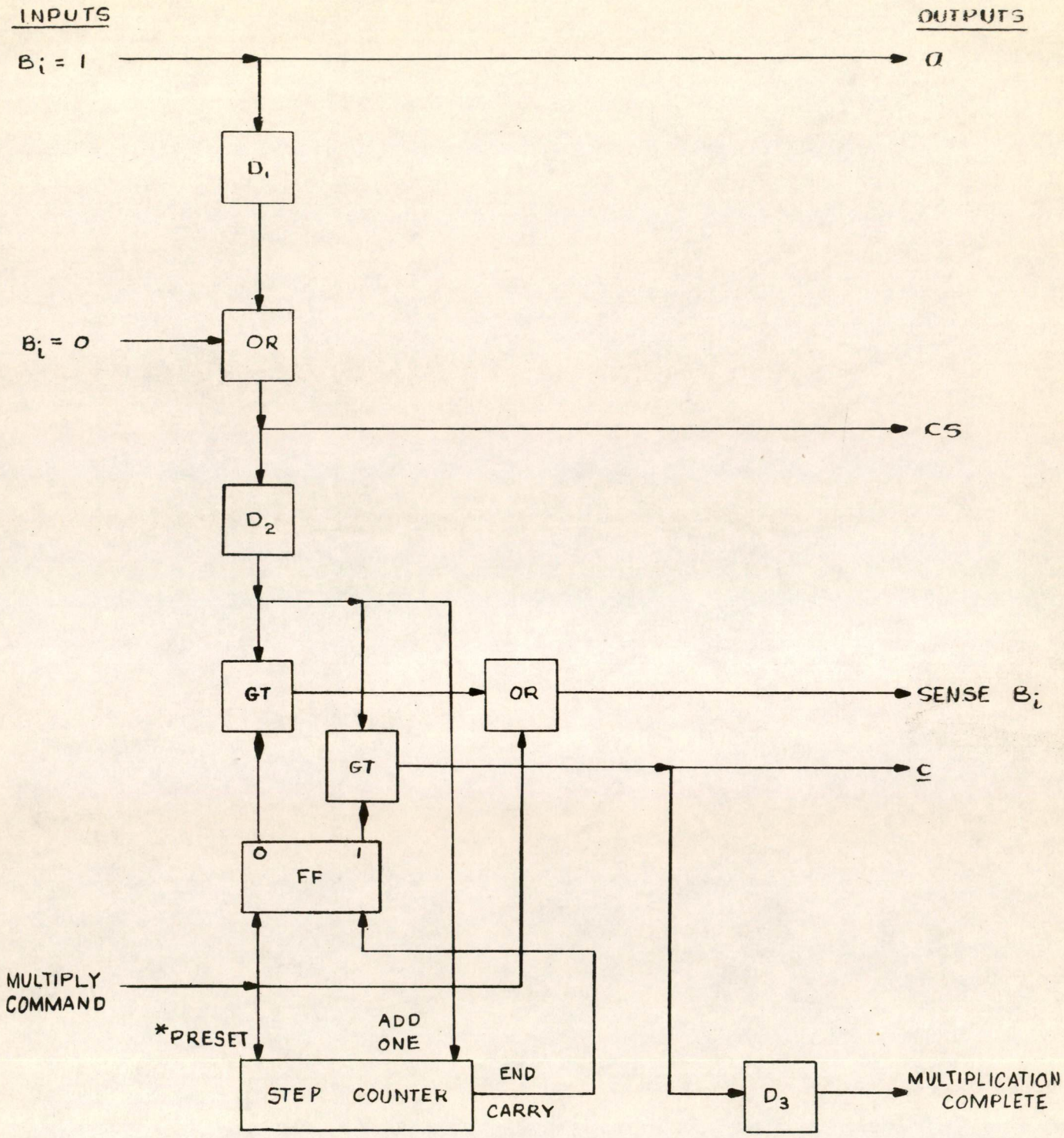


FIG. IV-2

THE CONTROL FOR THE K-a.cs B1 AND K-a.cs B2 MULTIPLIERS



*PRESET TO END-CARRY ON n-1ST ADD-ONE PULSE.

FIG. IV-3

CONTROL FOR C-a.cs B1 AND C-a.cs B2

A-56592

C-56164

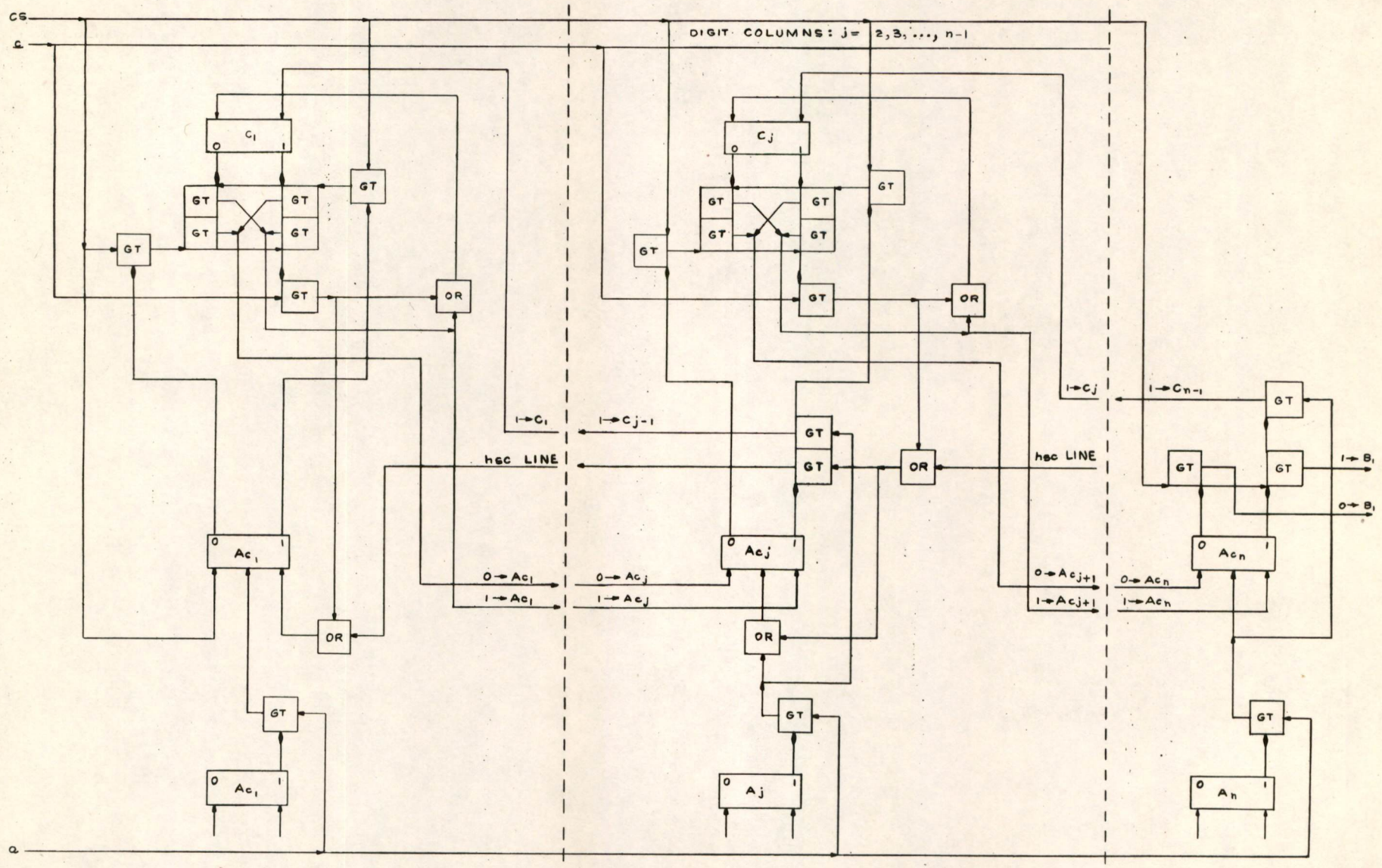
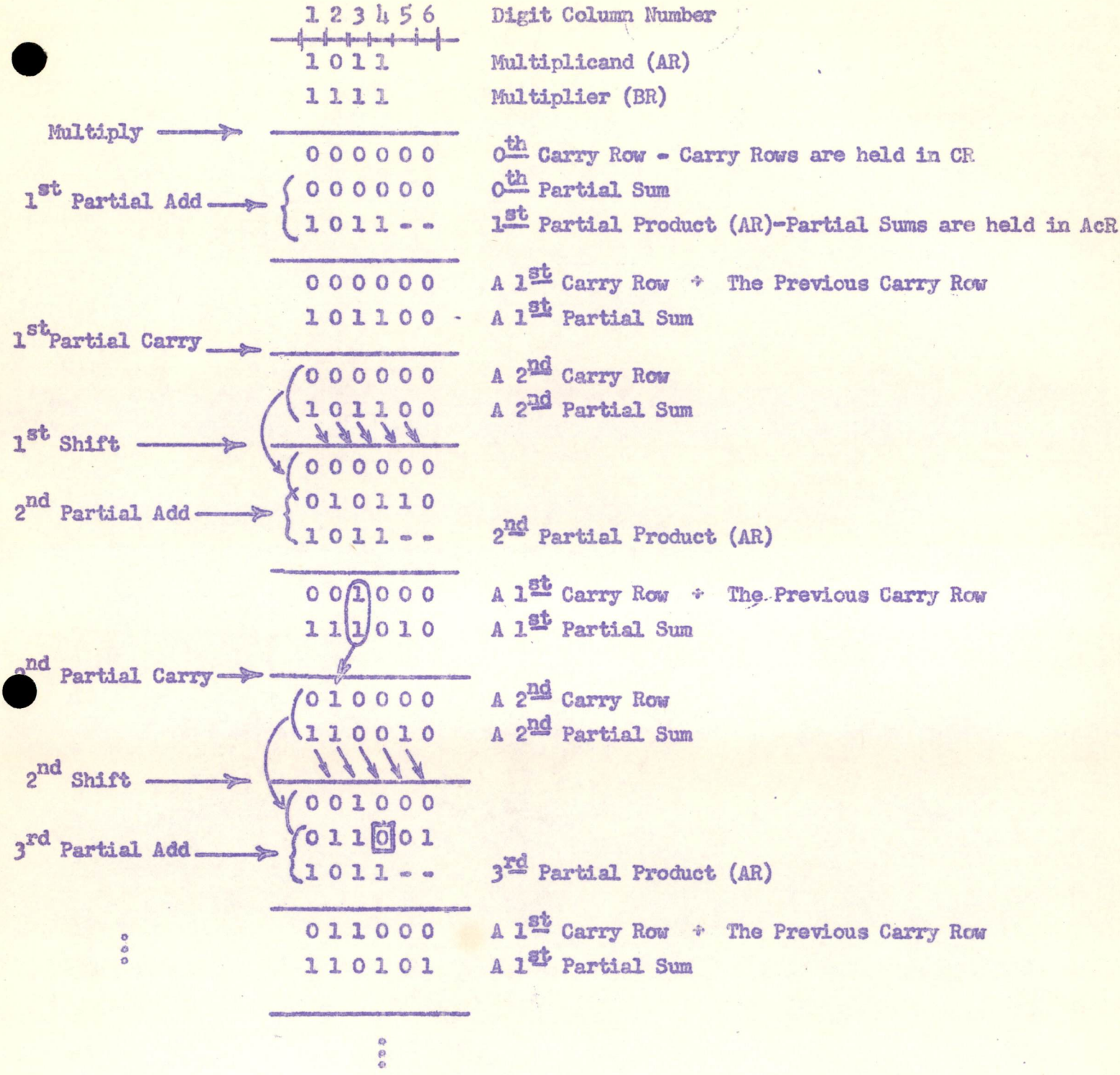


FIG. IV-4

THE a.cs B1 MULTIPLY—ADDER WITH AR & AcR



An Illustration of the Procedure used by the a.cs B1 Multiply-Adder

Fig. IV - 5

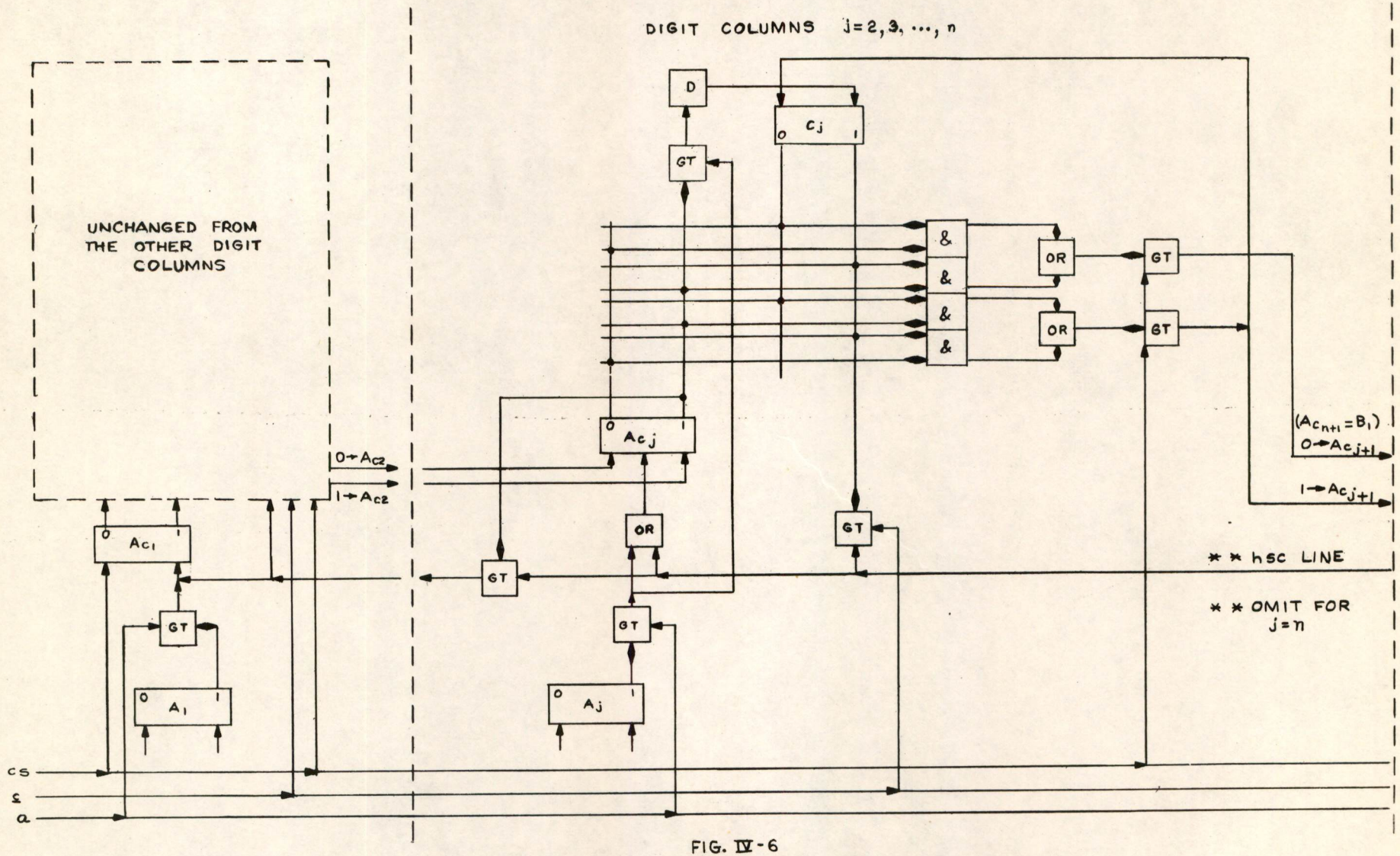


FIG. IV-6

THE a.cs B2 MULTIPLY—ADDER WITH AR & AcR

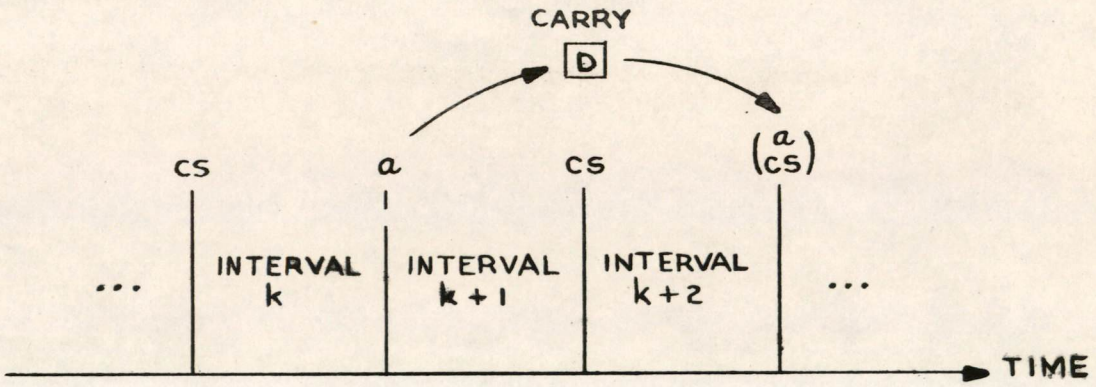


FIG. IV-7

HANDLING OF CARRIES IN THE a.cs B_2 MULTIPLY–ADDER

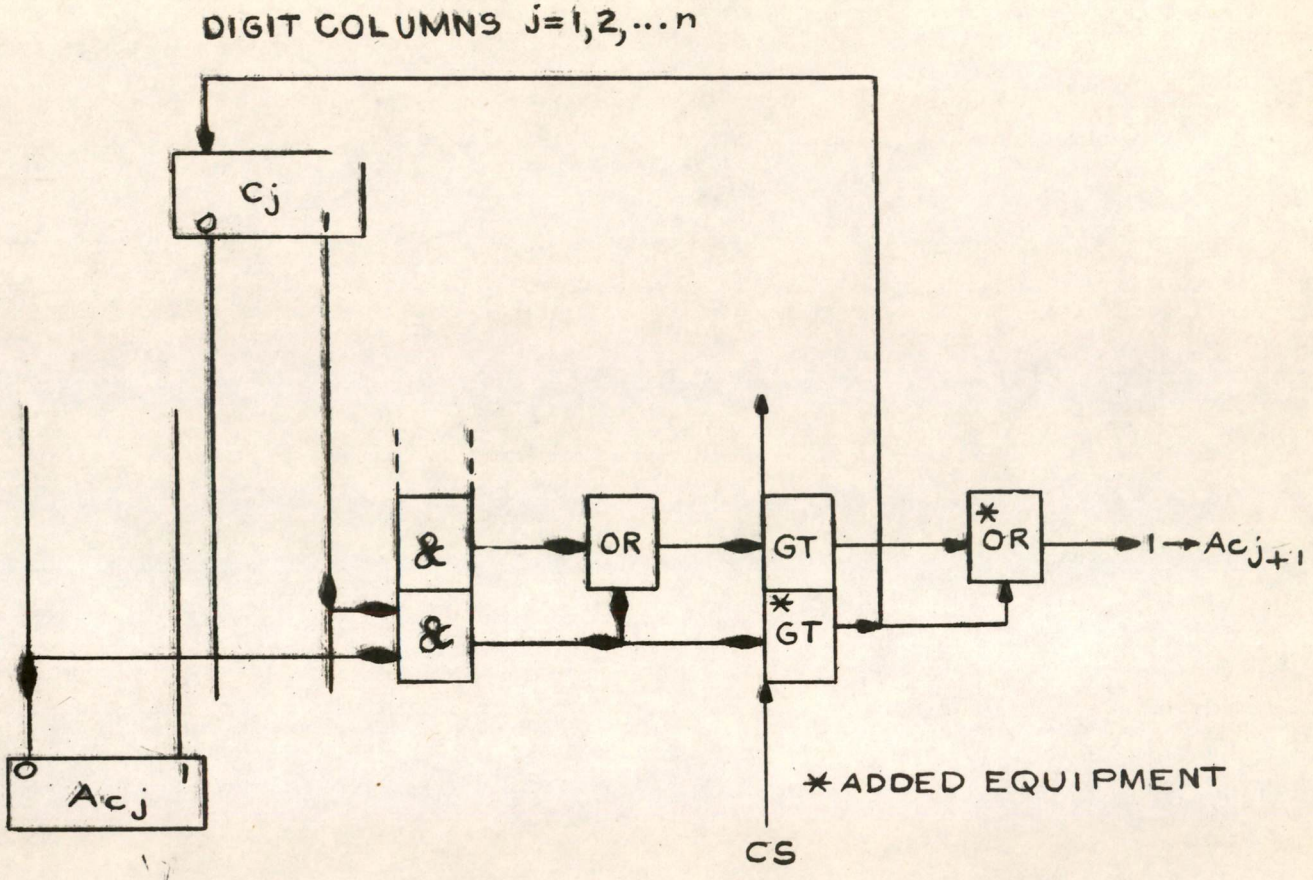


FIG. IV-8

AN ALTERATION IN THE ac-s B2 MULTIPLY-ADDER

(ONLY THE ALTERED PORTION IS INDICATED)

A-56193

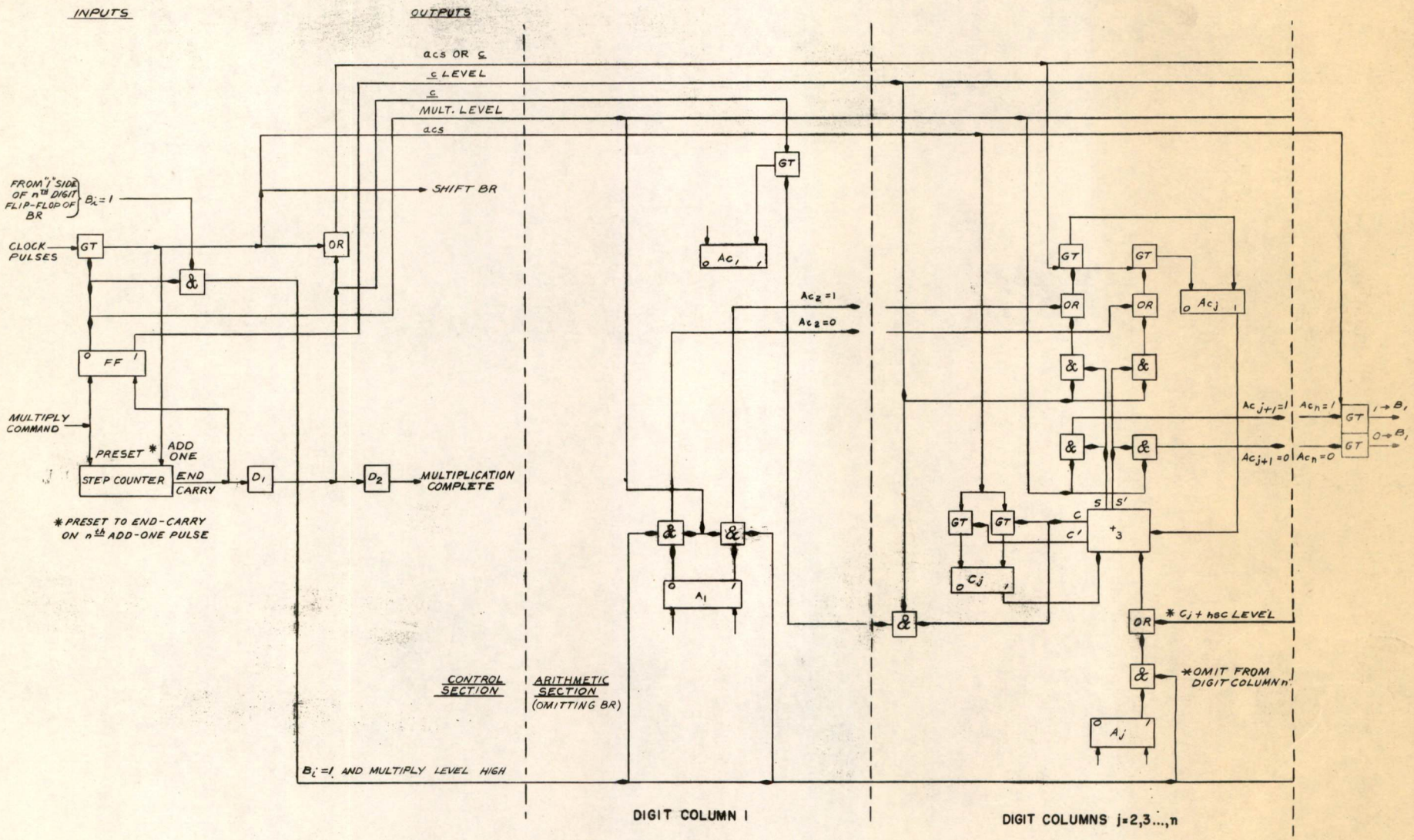


FIG. IV-9

THE K-acs B1 MULTIPLY ADDER OMITTING BR

C-47013

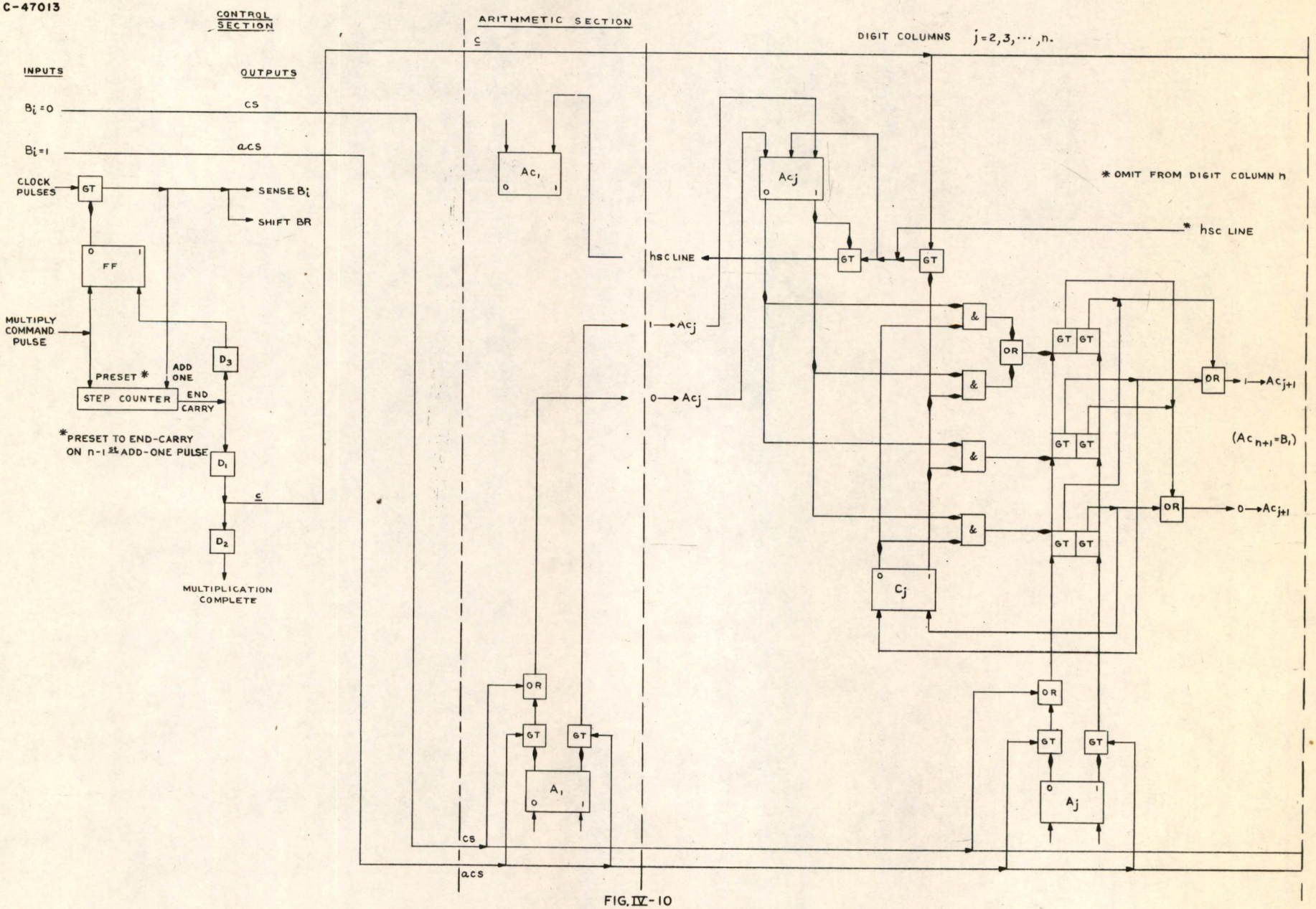
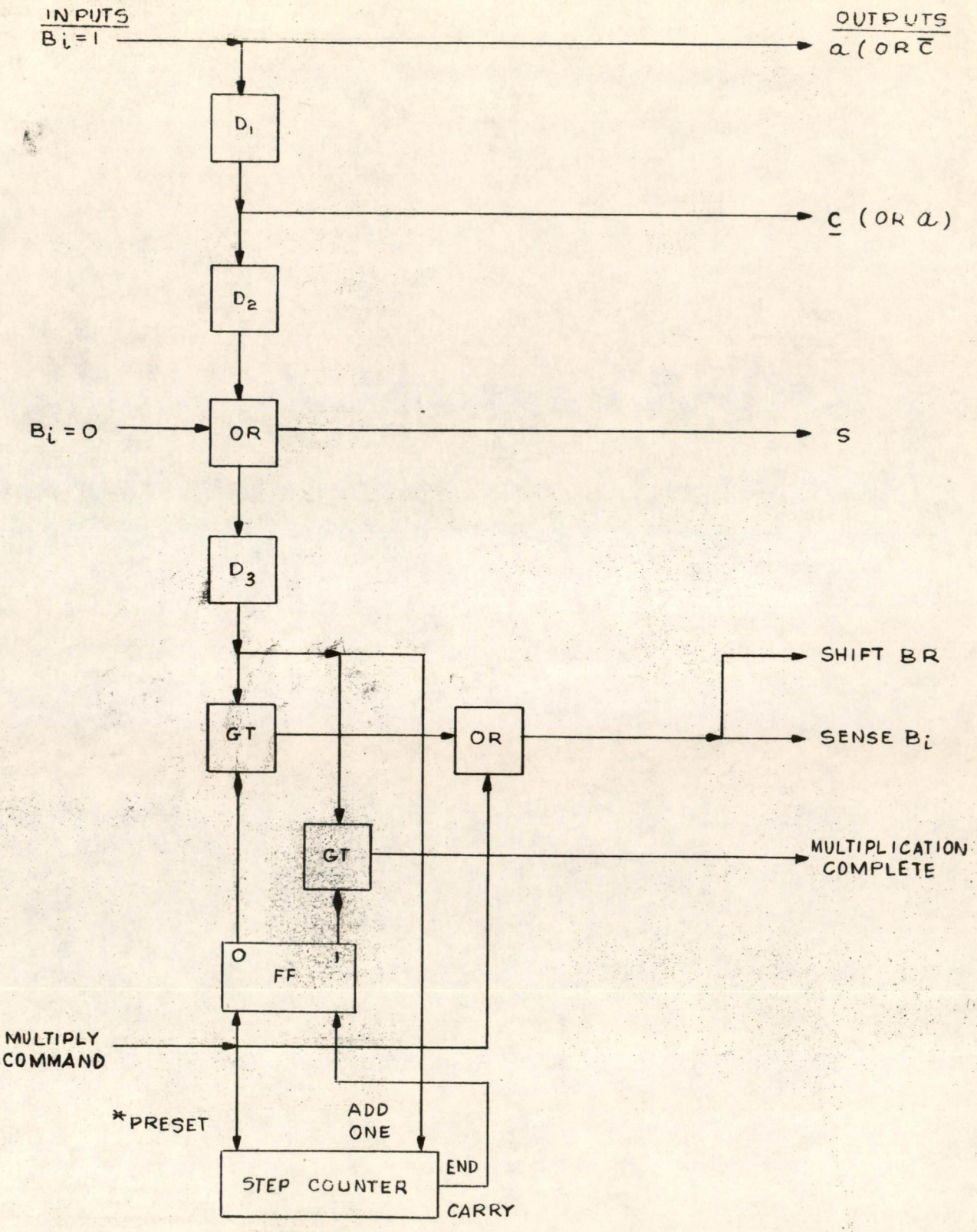


FIG. IV-10

THE K-acs B2 MULTIPLIER OMITTING BR



* PRESET TO END-CARRY ON n-1 ST ADD-ONE PULSE.

FIG. IV - 11

CONTROL FOR C- a.c.s B AND C- $\bar{c}$.a.s B

A-56593

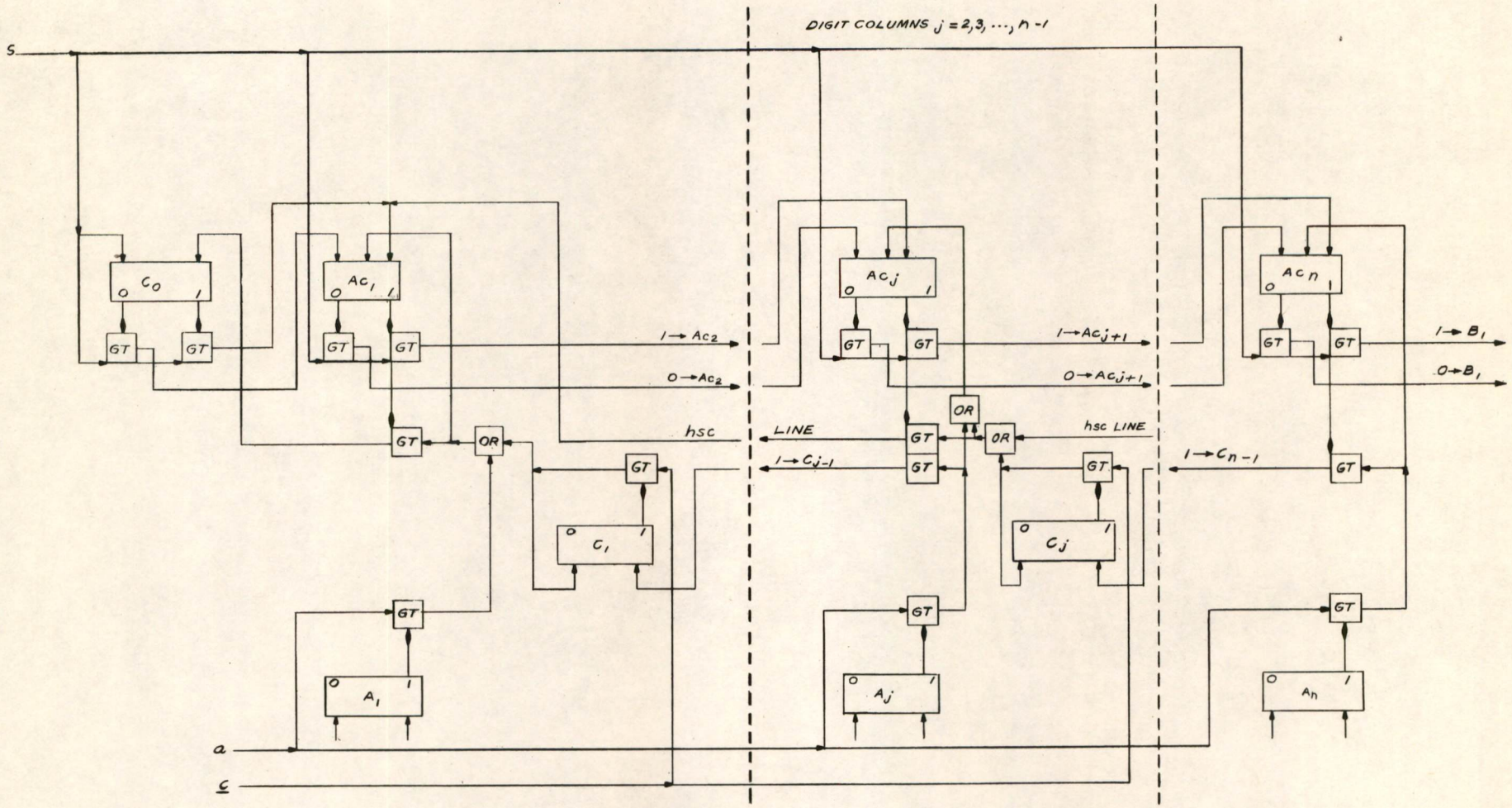
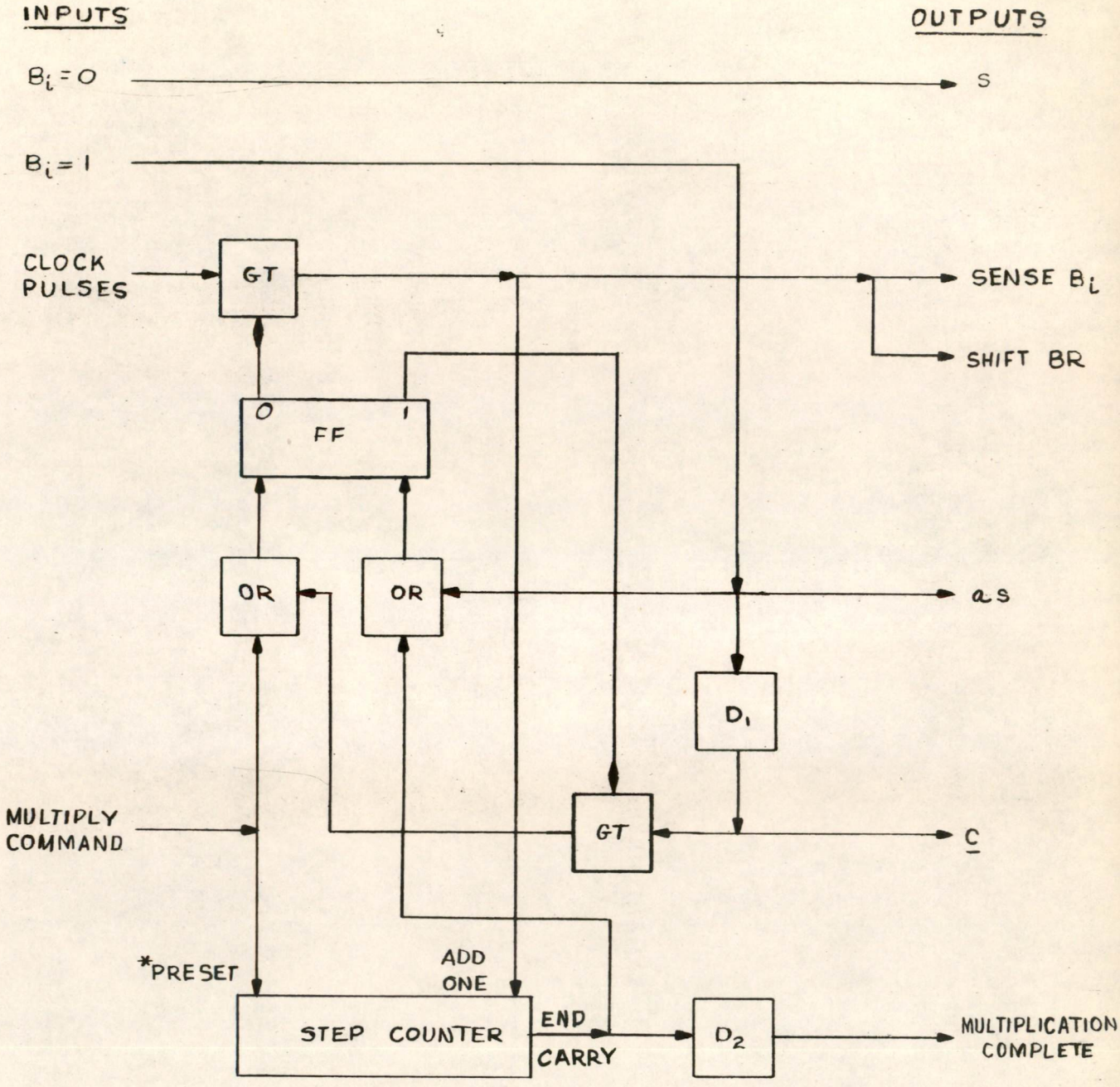


FIG. IX-12

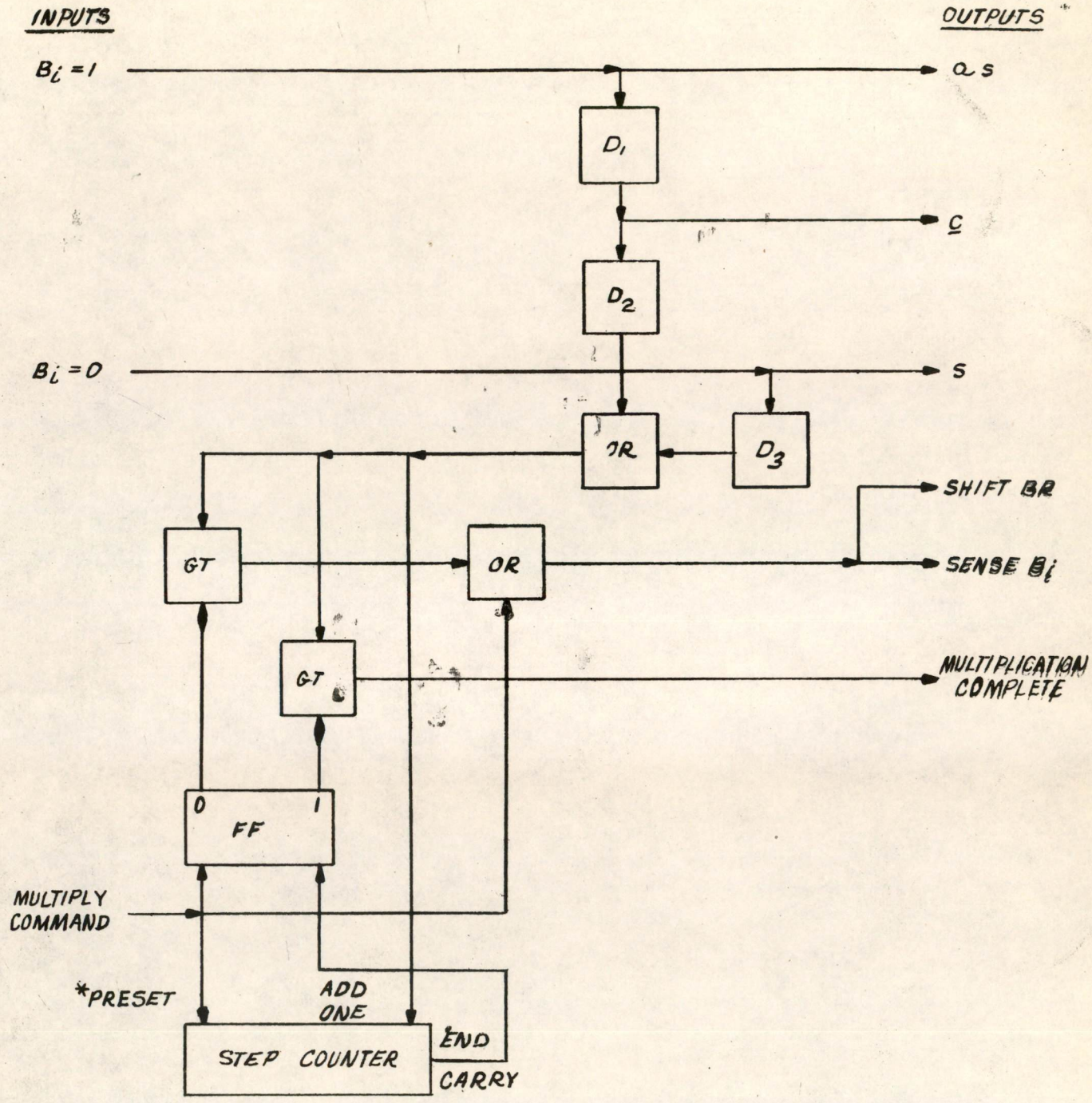
THE a.c.s B MULTIPLY-ADDER WITH AR & AcR



* PRESET TO END-CARRY ON n^{th} ADD-ONE PULSE

FIG. IV - 13

A MIXED K-C TYPE CONTROL FOR as.c B



*PRESET TO END-CARRY ON h-1ST ADD-ONE PULSE

FIG. IV-14

CONTROL FOR C-as.c B

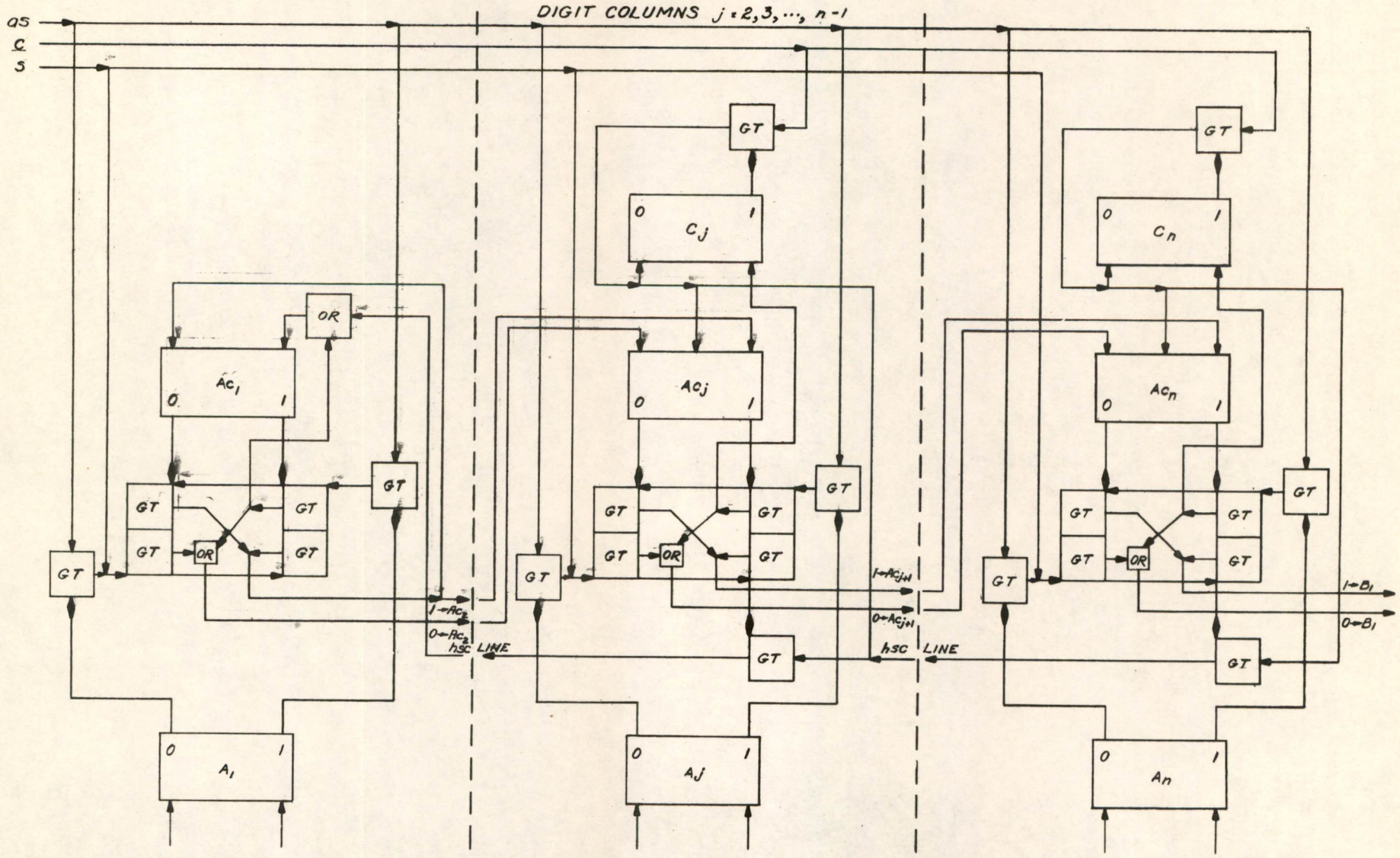
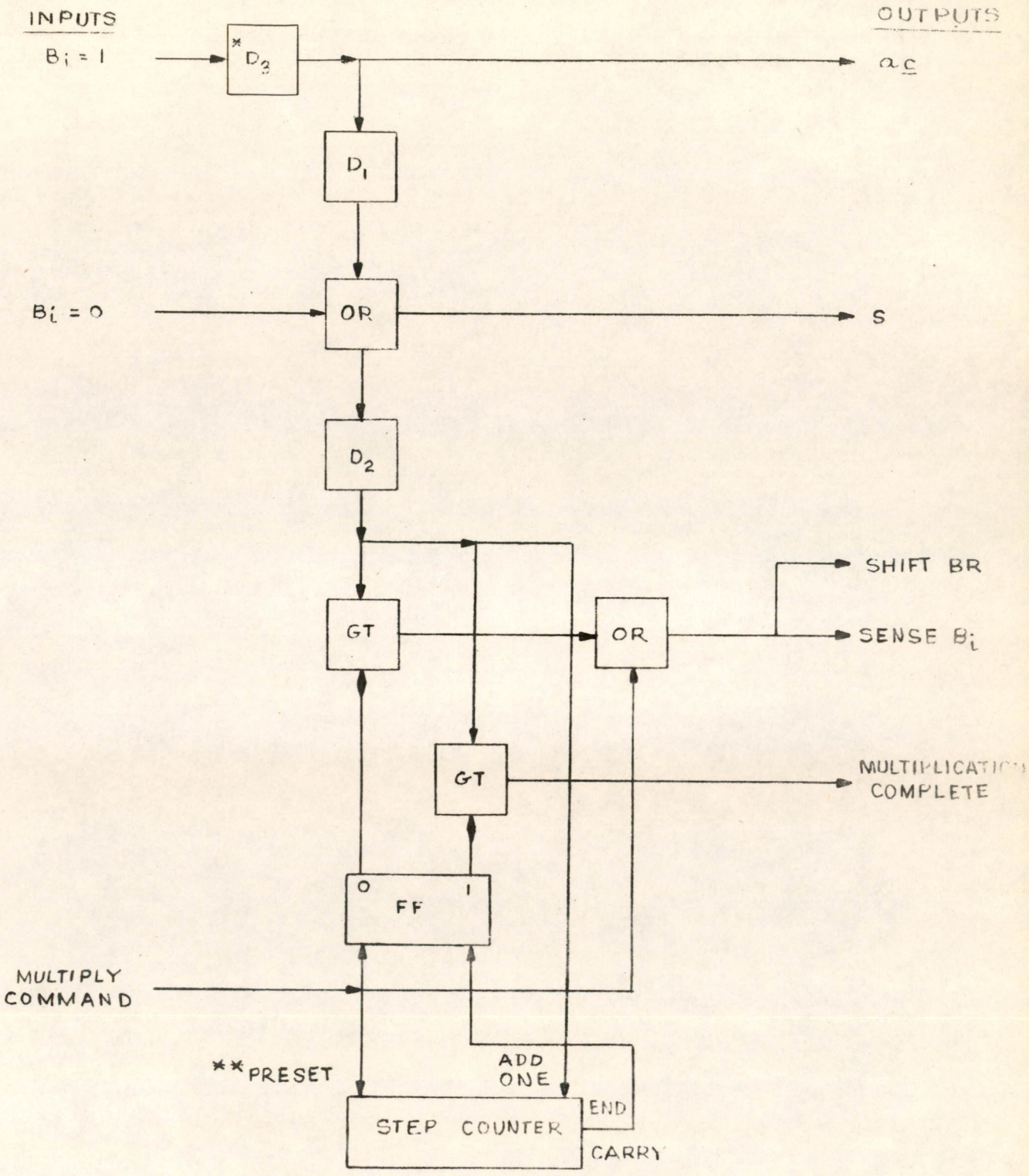


FIG. IV-15

THE as.c B MULTIPLY-ADDER WITH AR & AcR



*OMIT WHEN USED FOR THE C-ac.s B2 MULTIPLIER

**PRESET TO END-CARRY ON THE n-1ST ADD-ONE PULSE

FIG. IV-16

CONTROL FOR C-ac.s BI AND C-ac.s B2

A-56613

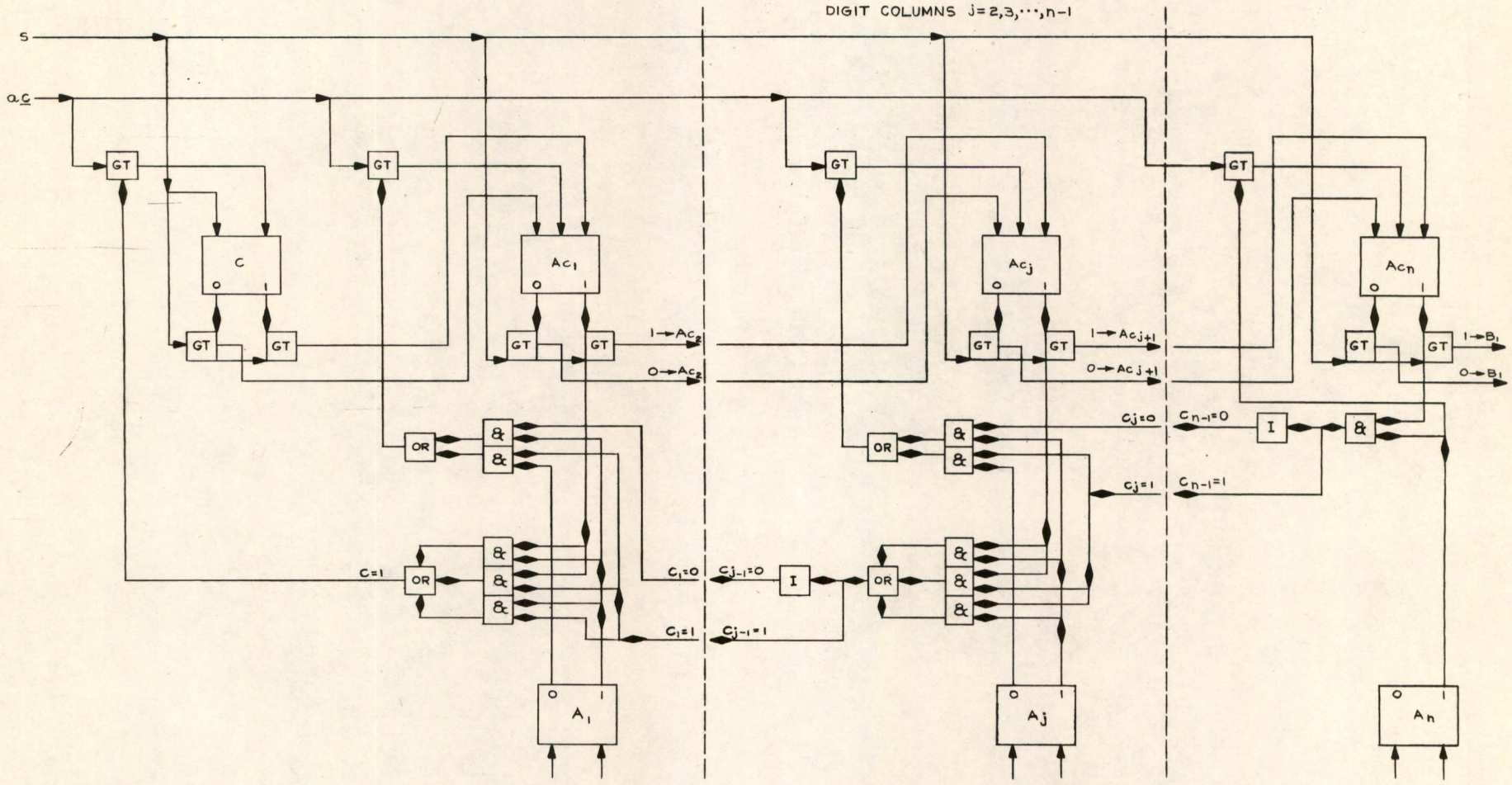


FIG. IV-17

THE ac.s B1 MULTIPLY-ADDER WITH AR & AcR

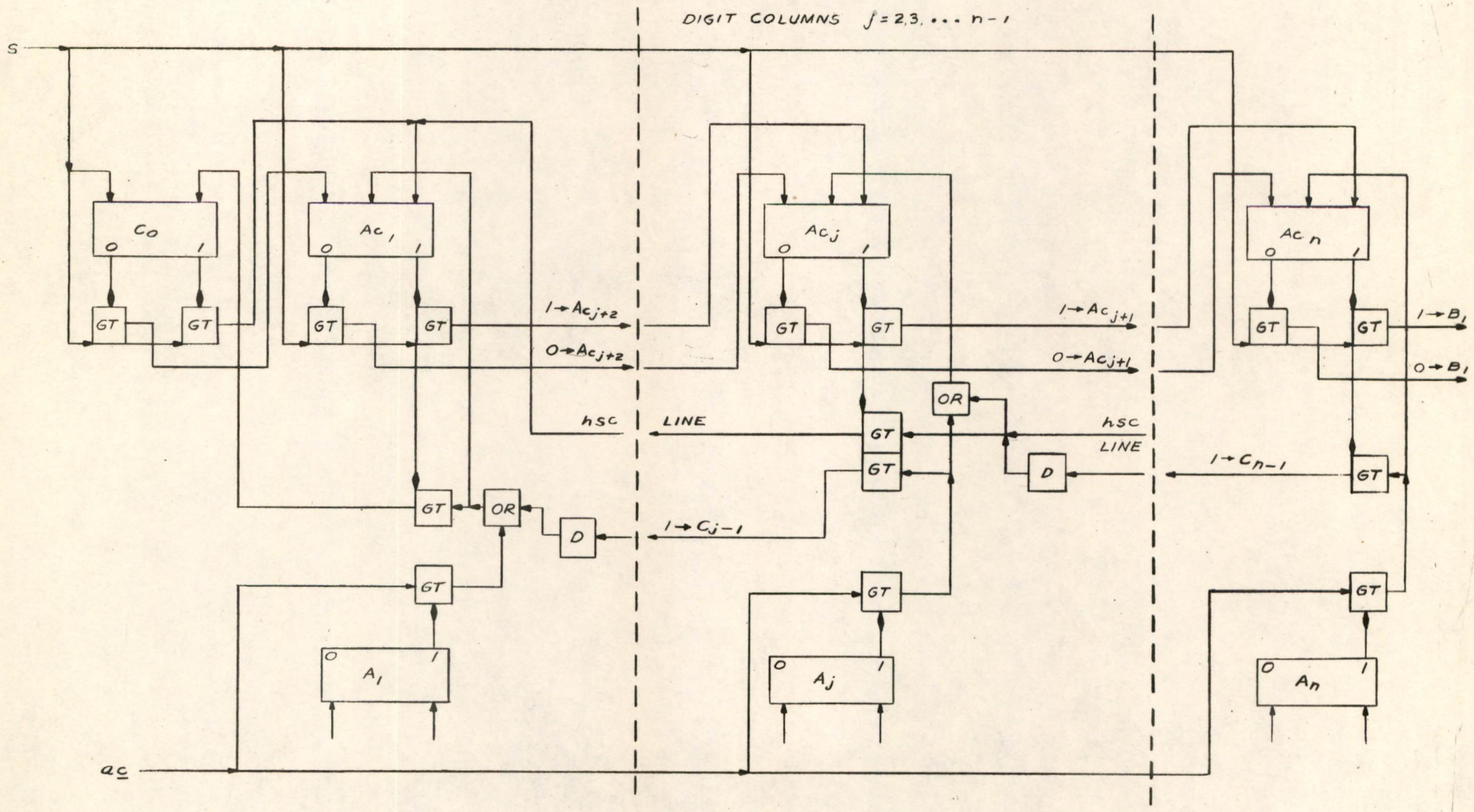


FIG. IV-18

THE ac.s B2 MULTIPLY-ADDER WITH AR & AcR

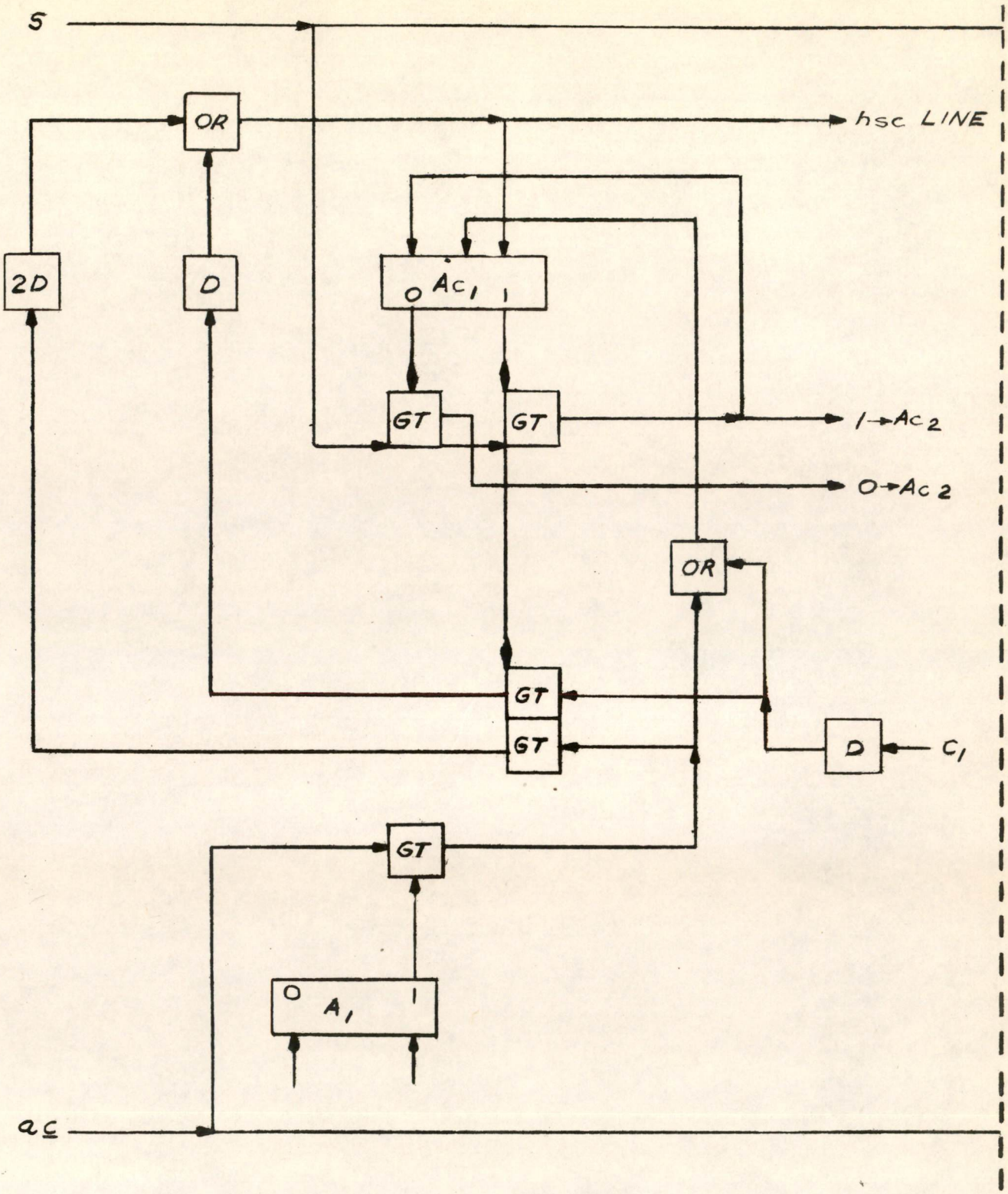


FIG. IV-19

AN ALTERNATIVE FOR THE FIRST DIGIT COLUMN OF ac.s B2

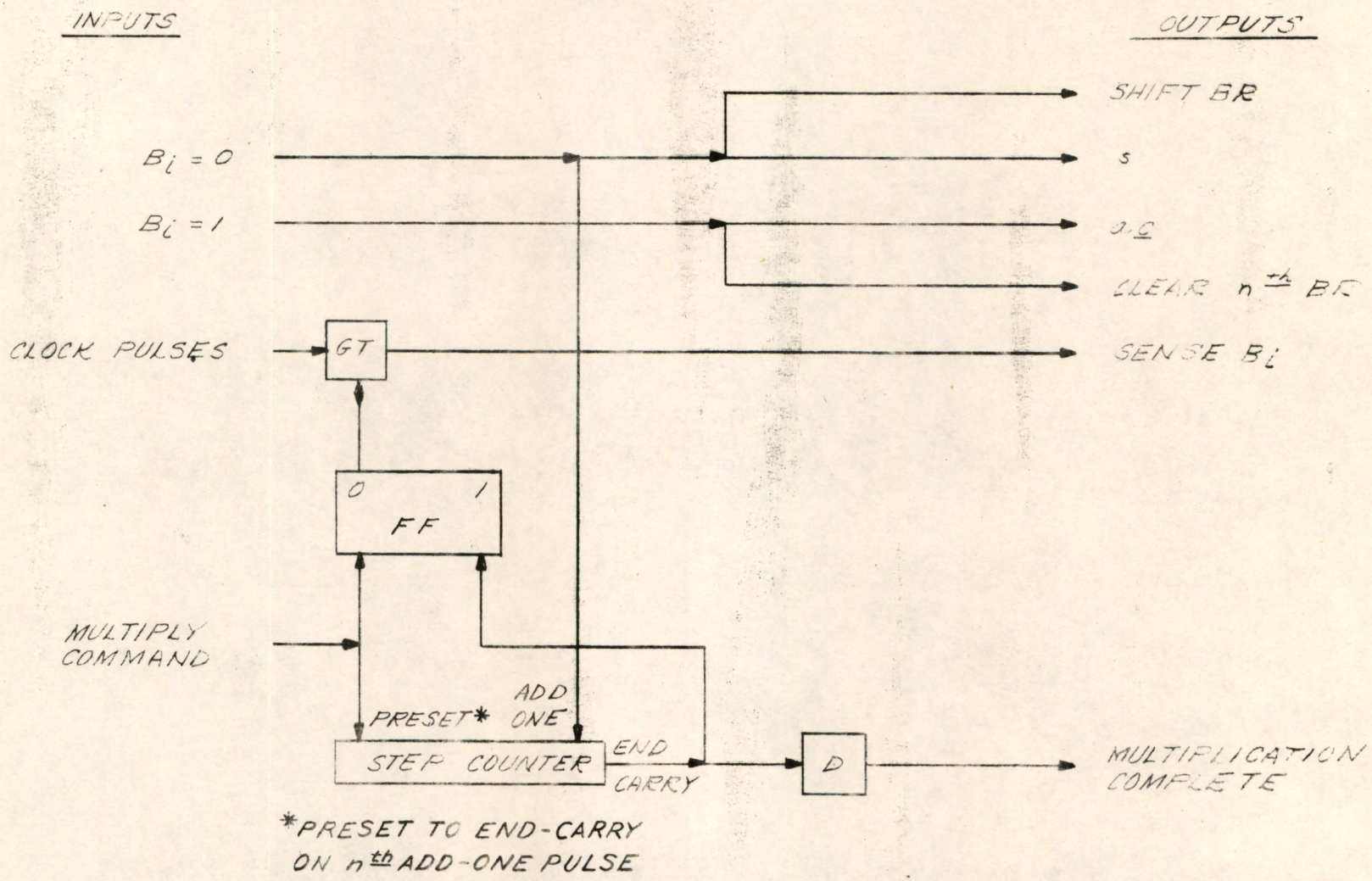


FIG. IV-20

CONTROL FOR K-ac.s DI AND K-ac.s D3

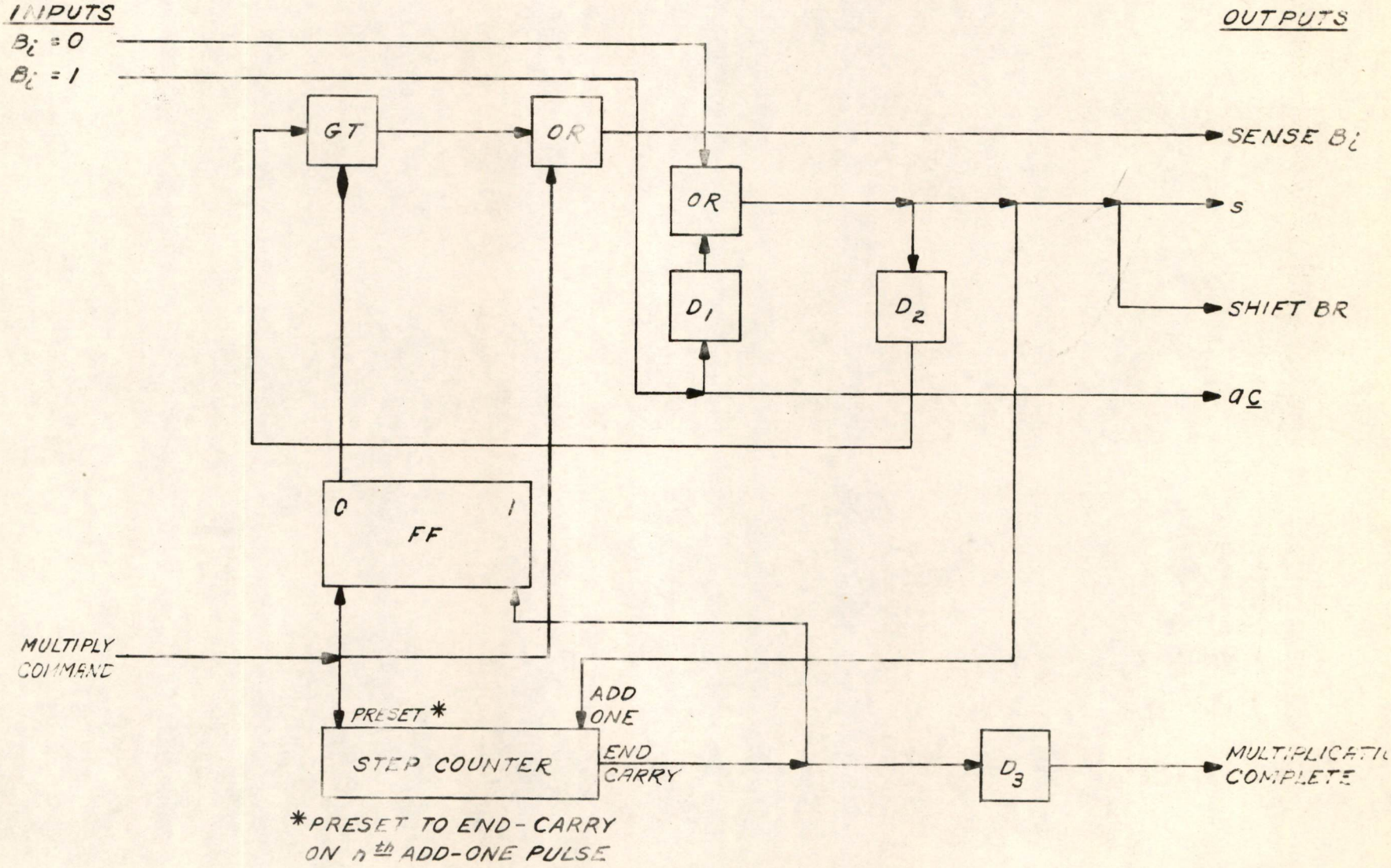


FIG. IV-21

CONTROL FOR C-ac.s DI AND C-ac.s D3

C-56322

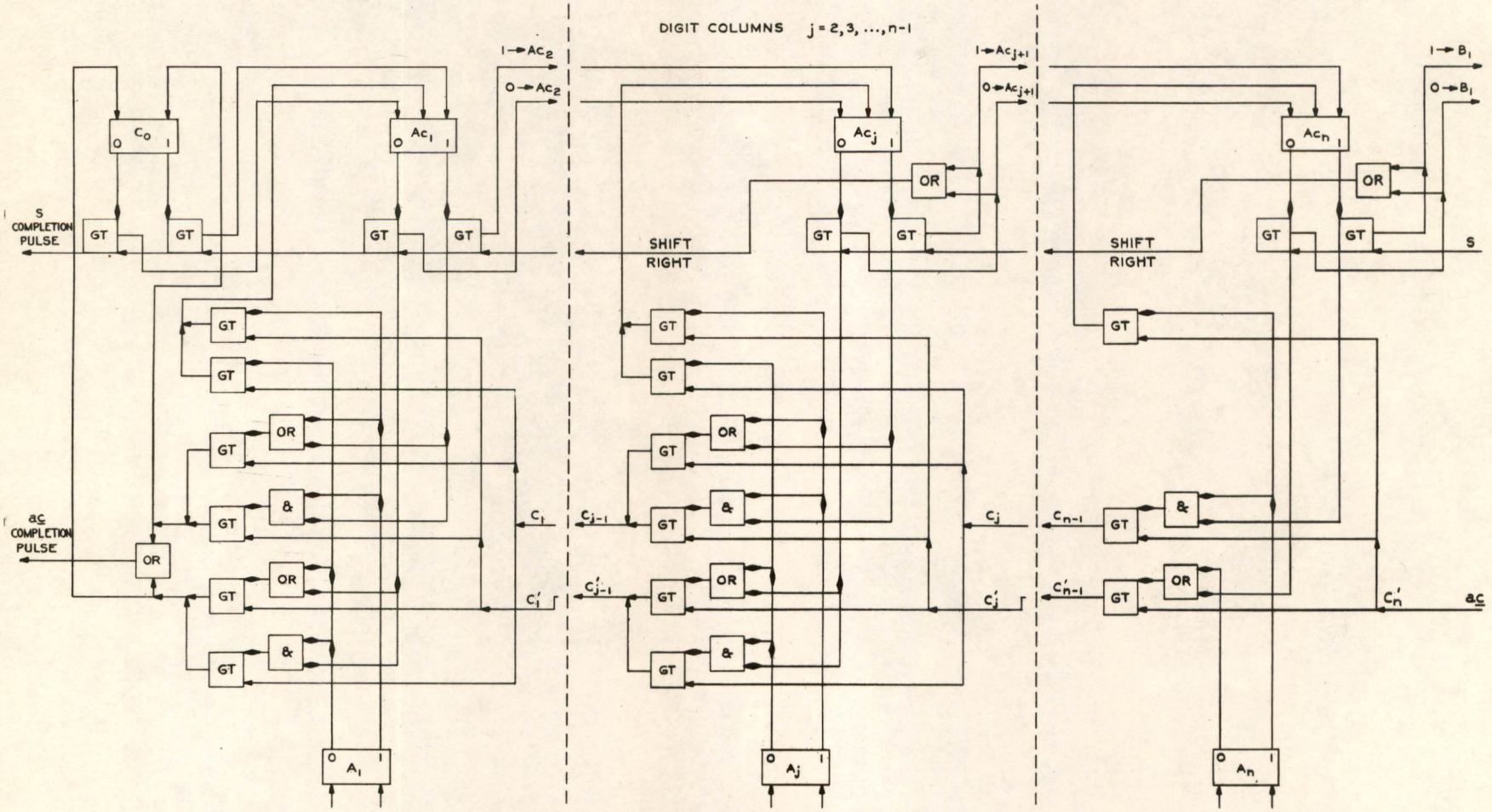
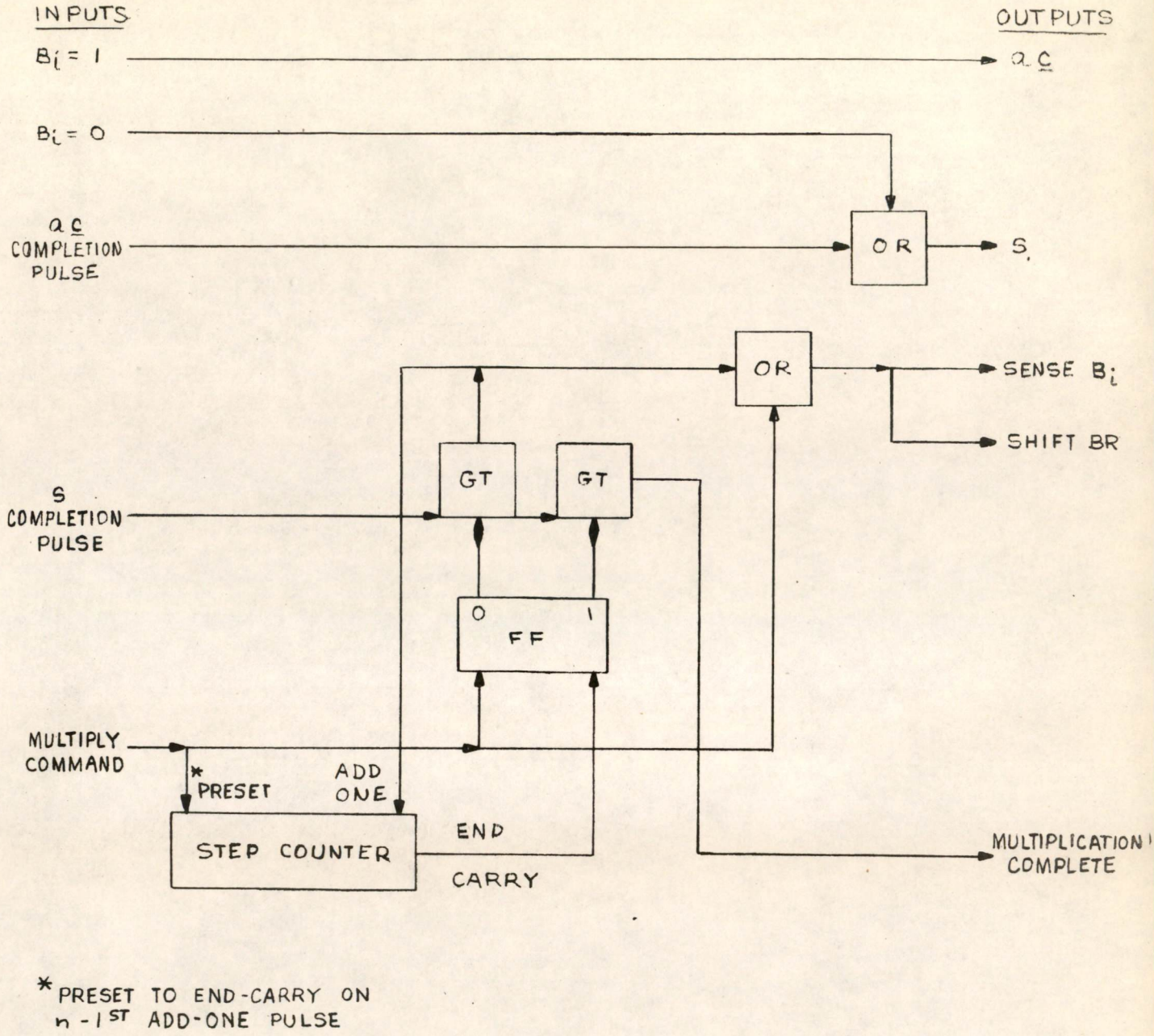


FIG. IV-22

THE ac.s D1 MULTIPLY-ADDER WITH AR AND AcR



* PRESET TO END-CARRY ON n - 1ST ADD-ONE PULSE

FIG. IV-23

CONTROL FOR C- ac.s D2

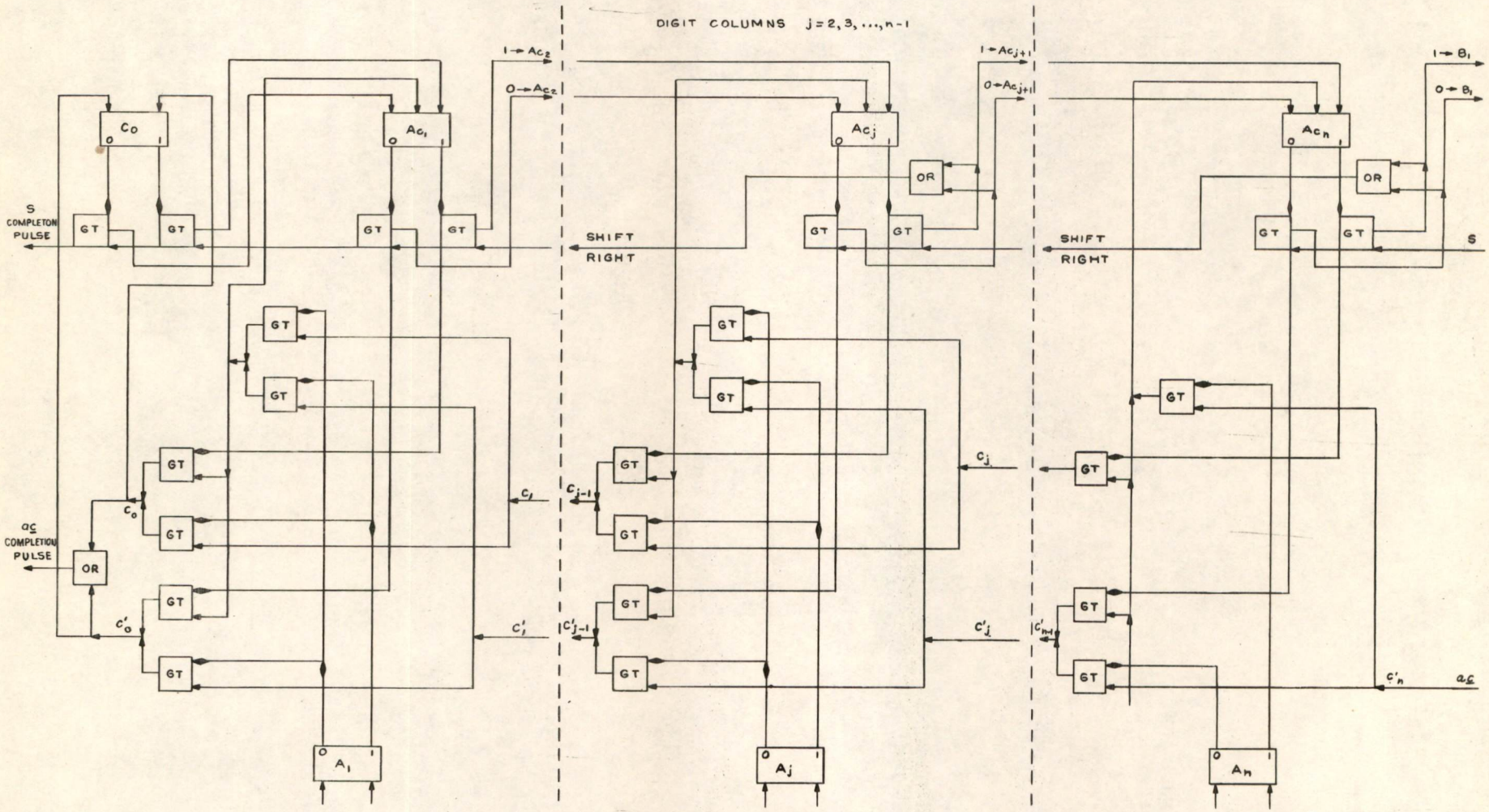


FIG. IV-24
THE ac.s D2 MULTIPLY—ADDER WITH AR AND AcR

C-56233

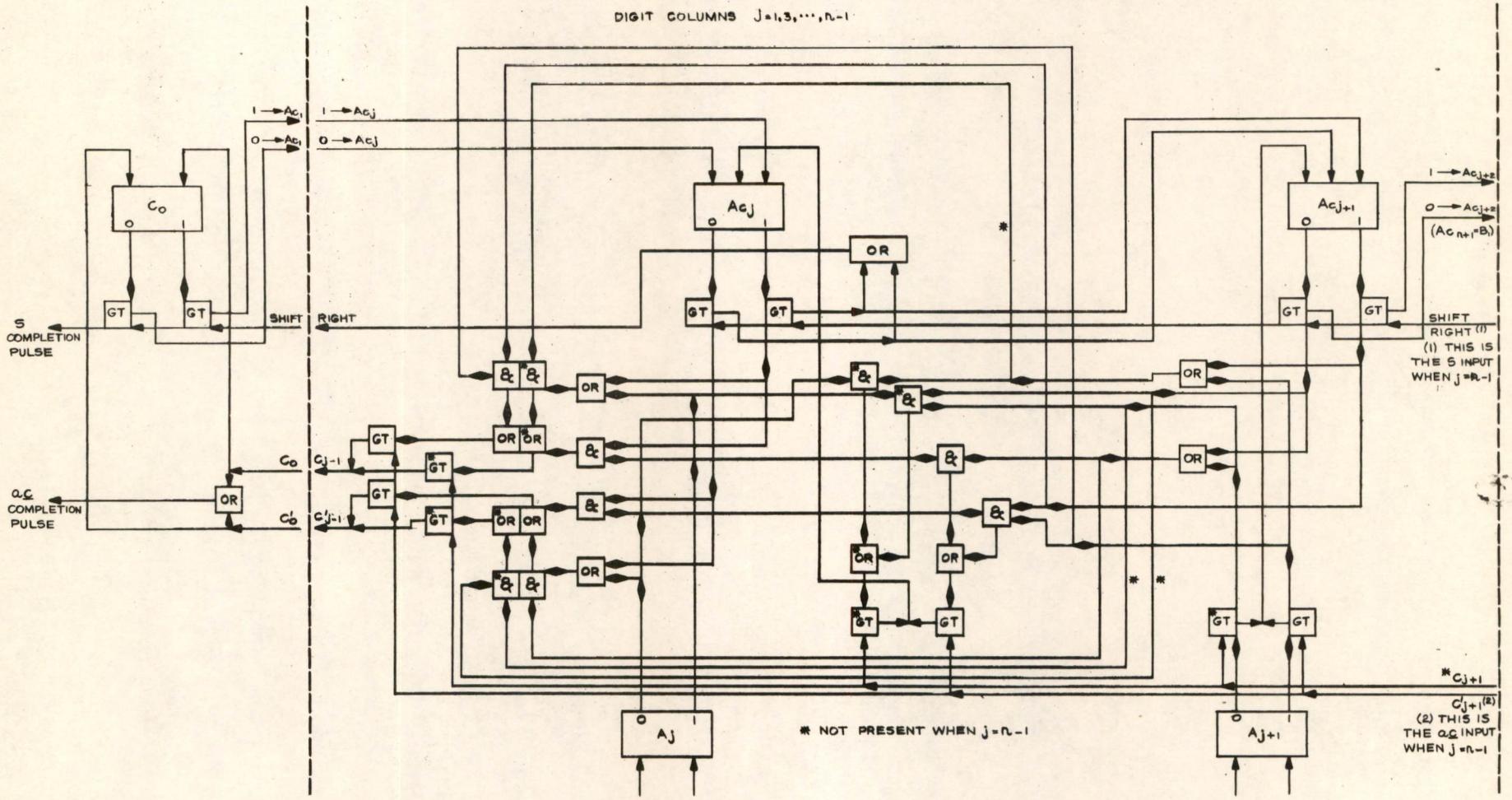


FIG IV-25

THE a.c.s D3 MULTIPLY — ADDER WITH AR & AcR

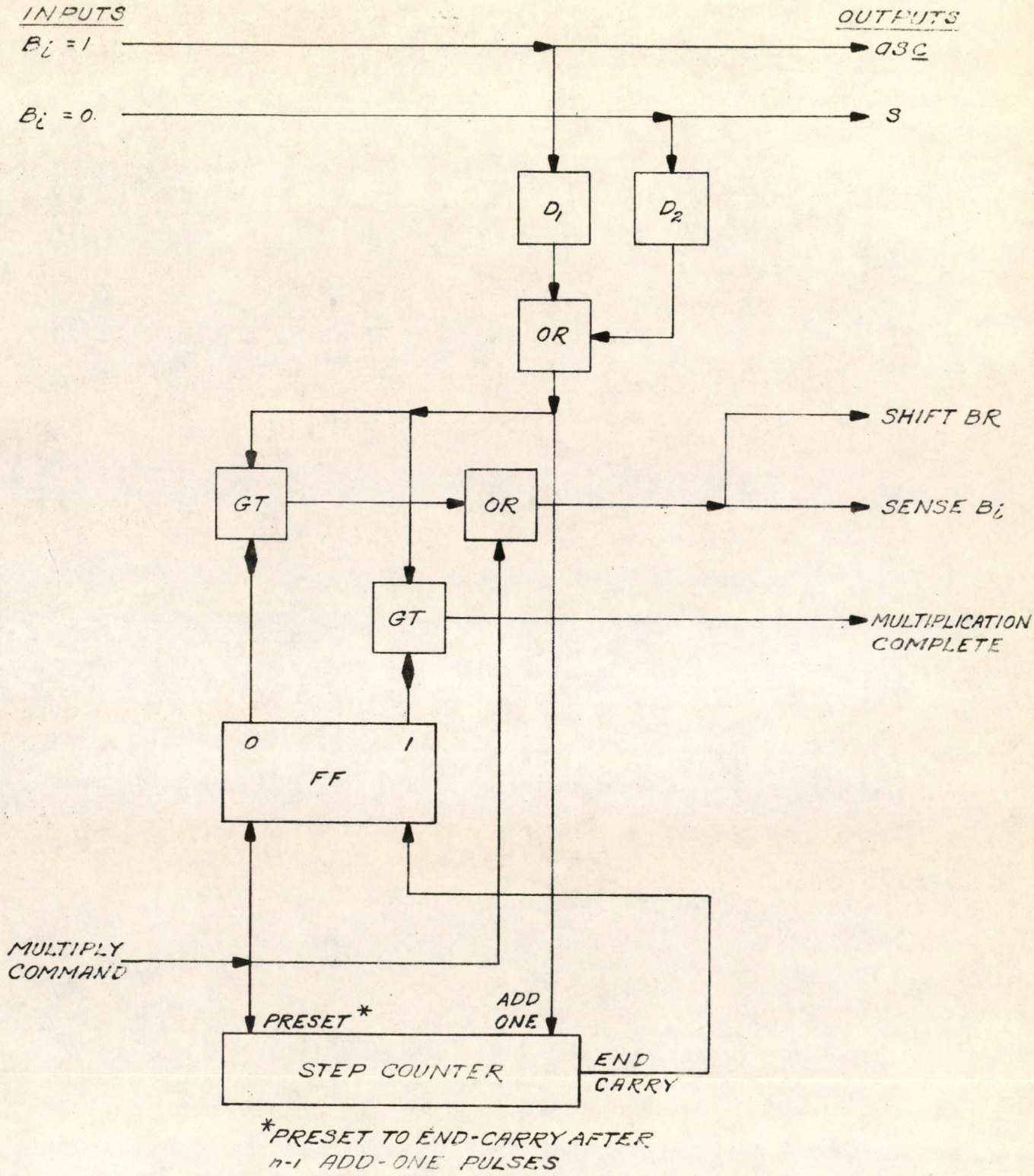


FIG. IV-26

THE CONTROL FOR C - asc B

A-56245

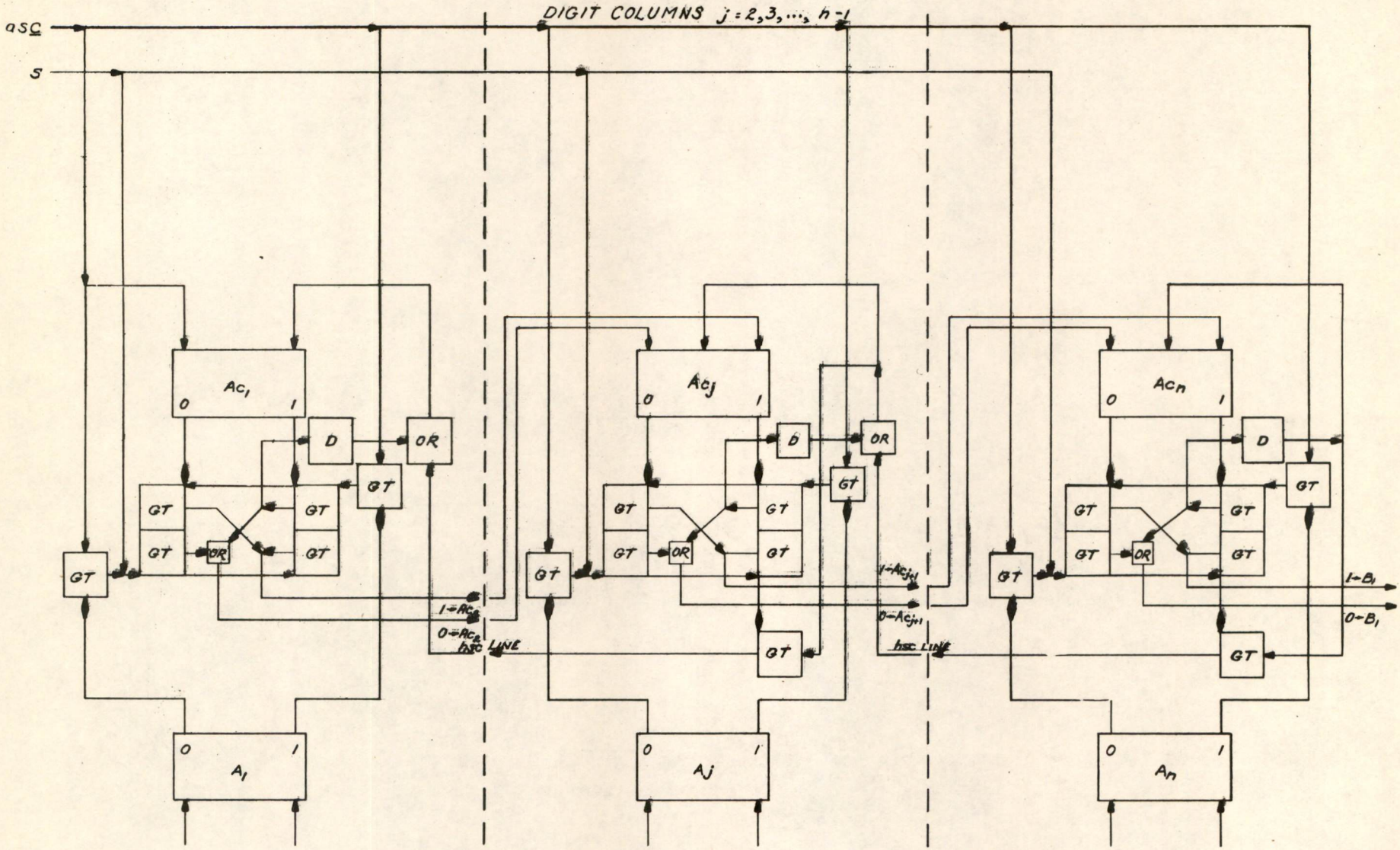
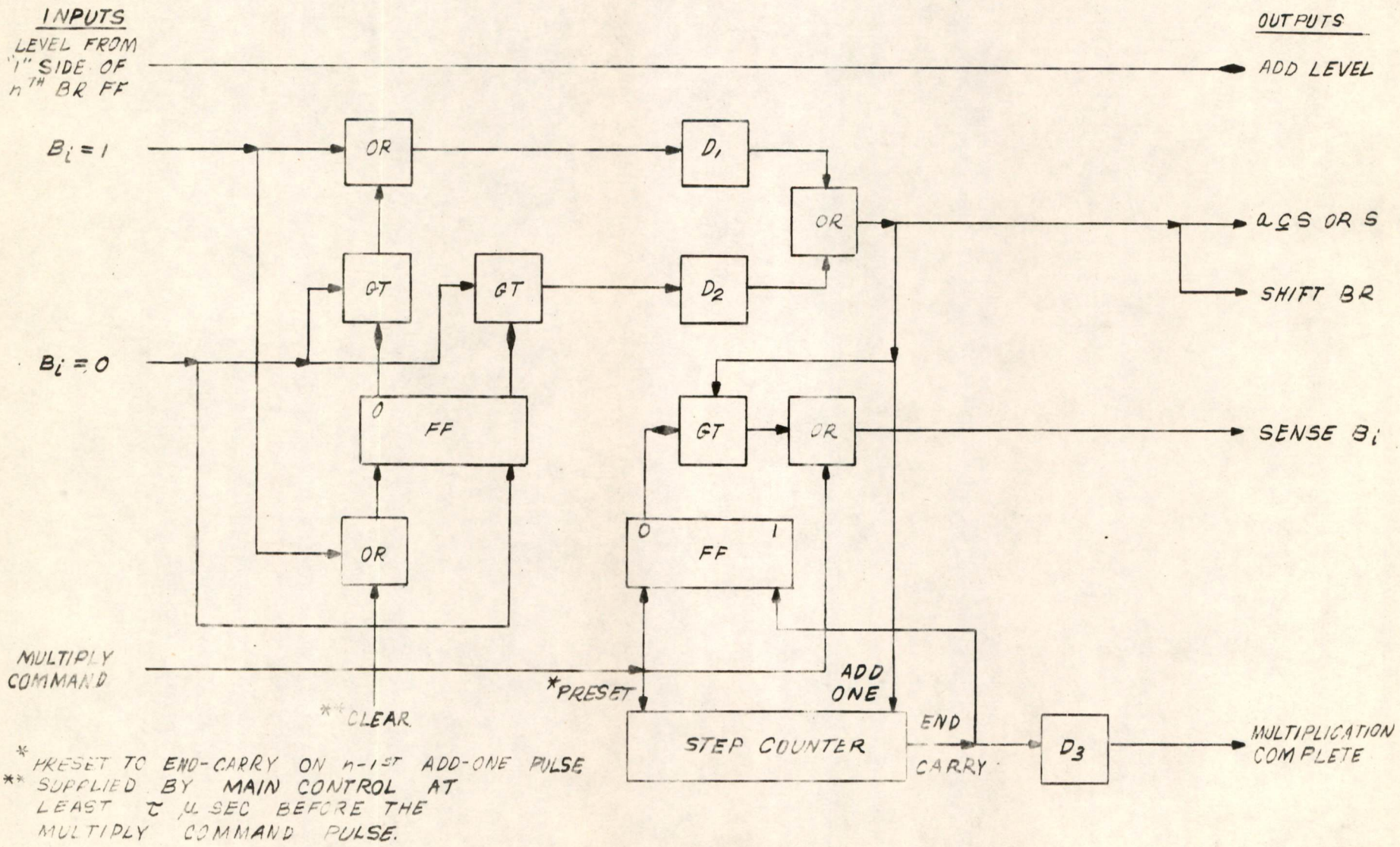


FIG. IV-27

THE asc B MULTIPLY-ADDER WITH AR & AcR

A-56224



* PRESET TO END-CARRY ON n-1ST ADD-ONE PULSE
** SUPPLIED BY MAIN CONTROL AT LEAST τ μ SEC BEFORE THE MULTIPLY COMMAND PULSE.

FIG. IV - 28

CONTROL FOR C-acs BI

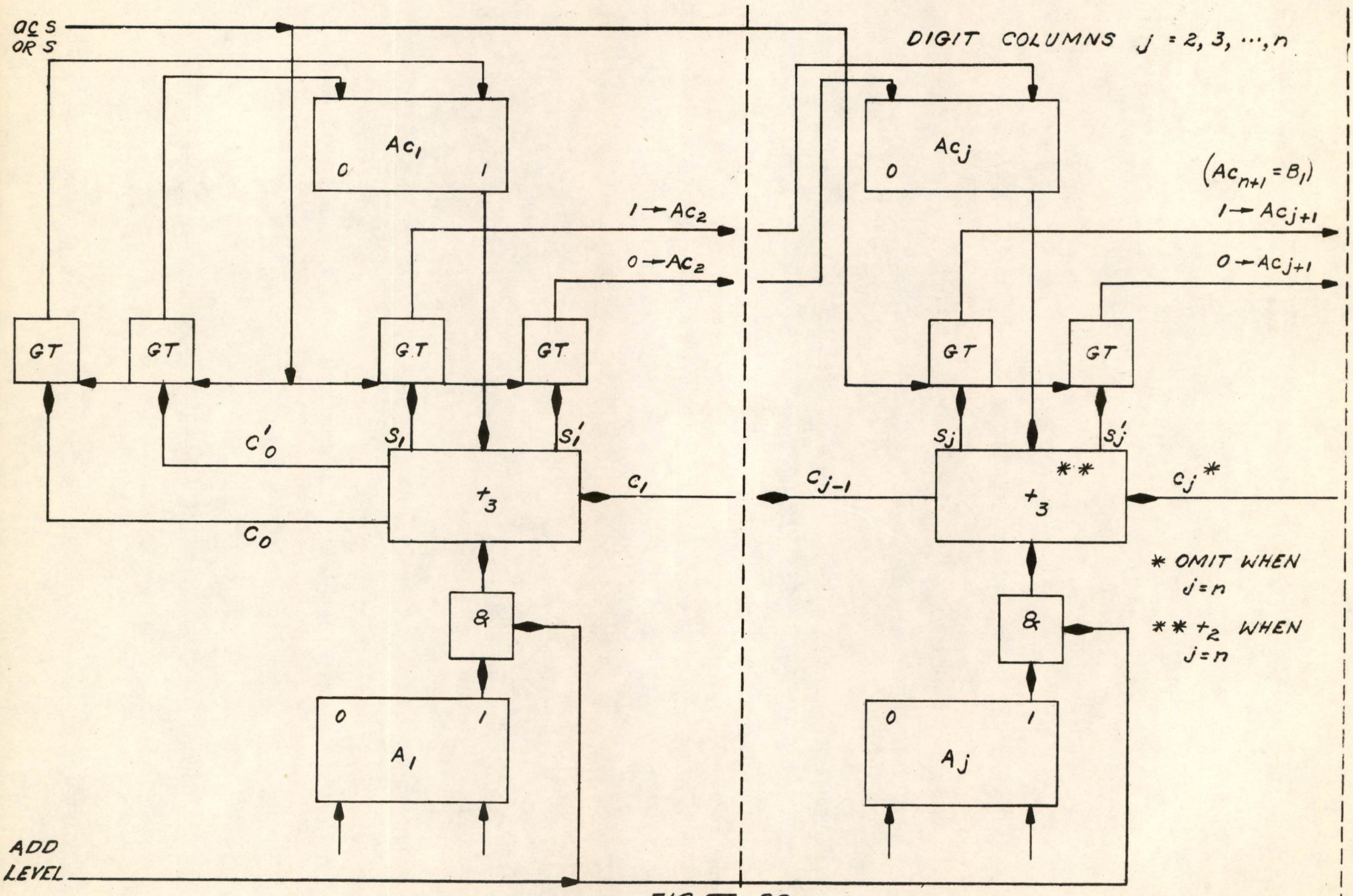


FIG. IV-29

THE acs BI MULTIPLY-ADDER WITH AR & AcR

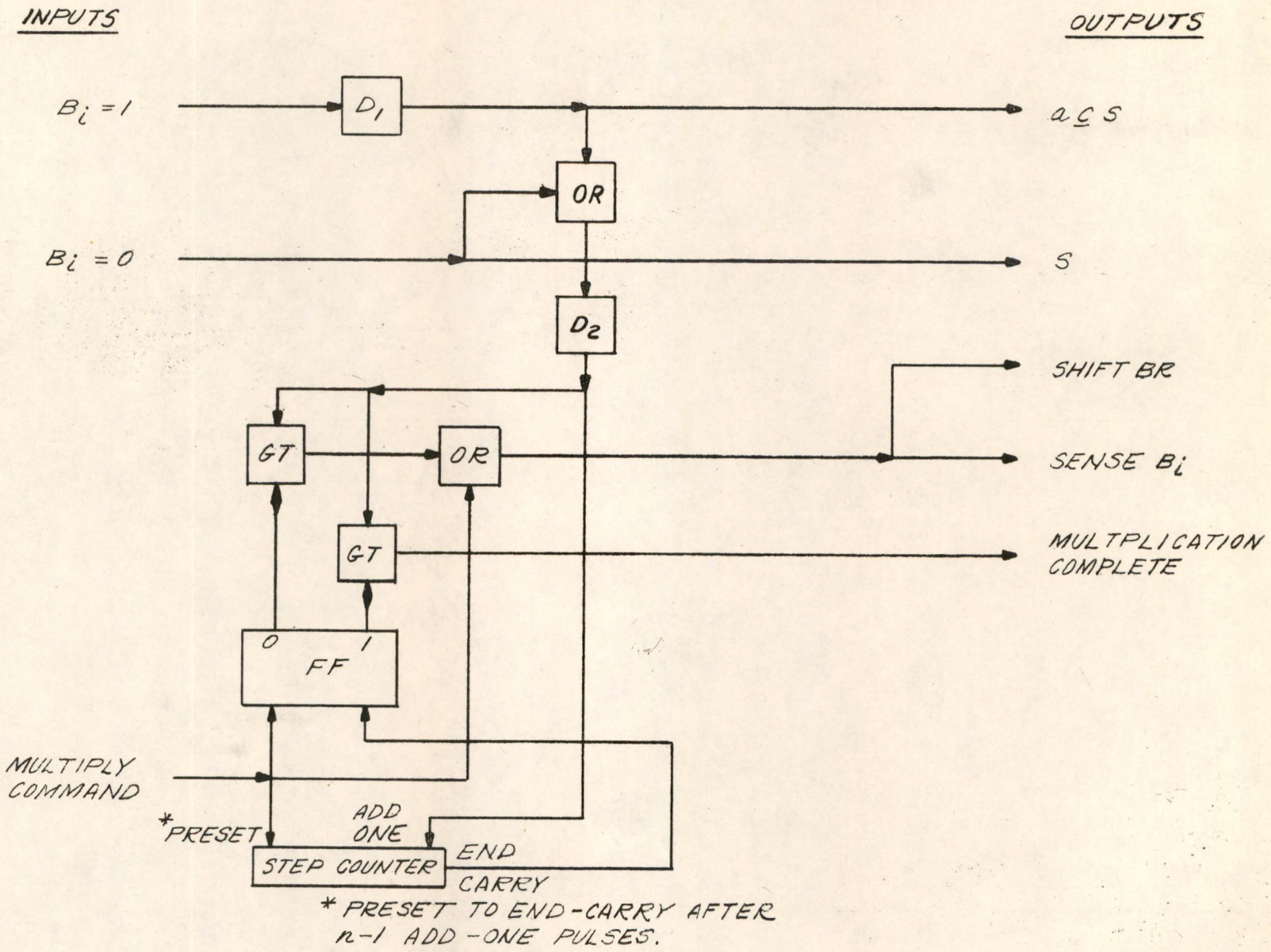


FIG IV-30

THE CONTROL FOR THE C-acs B2 MULTIPLIER

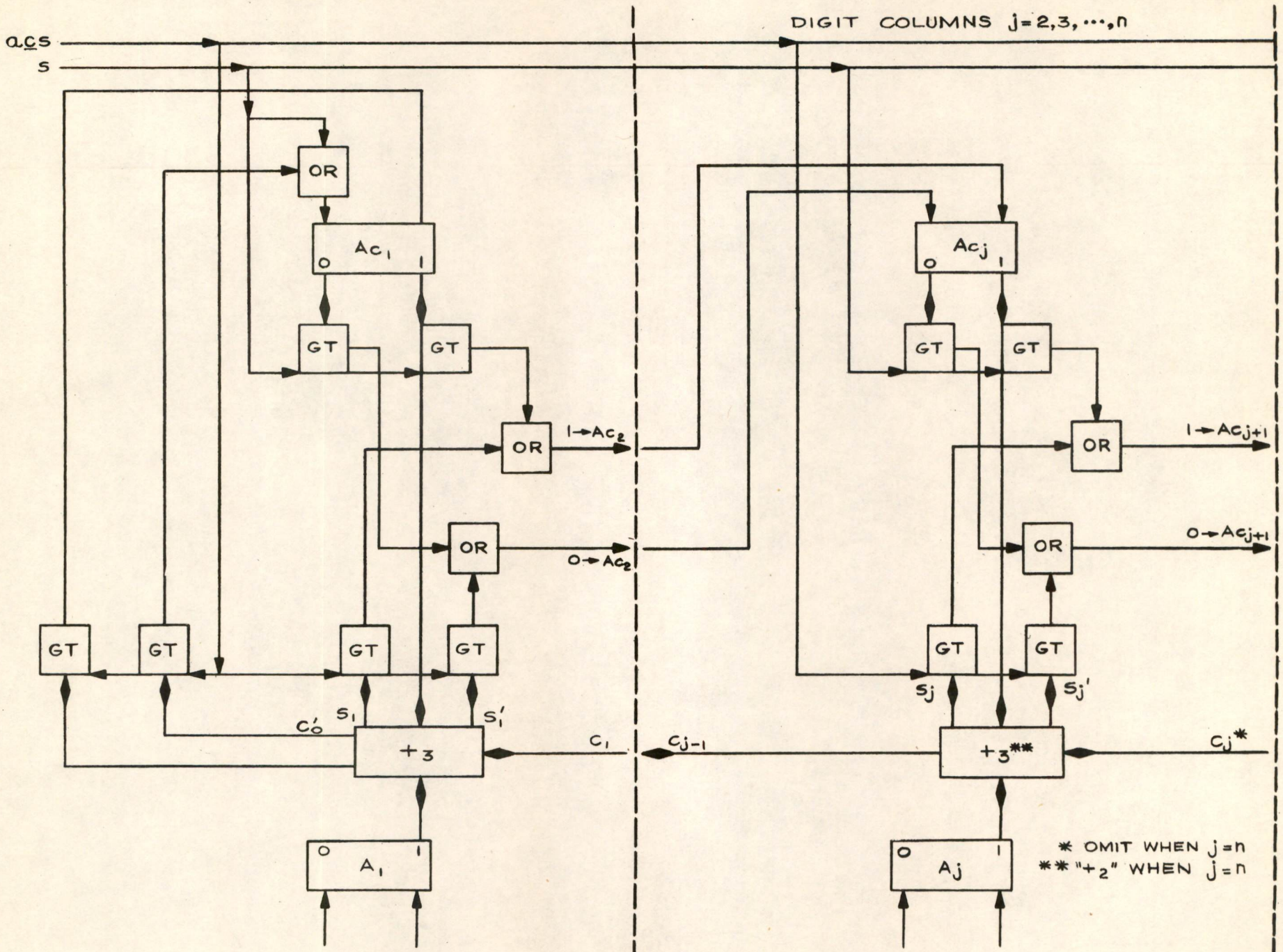
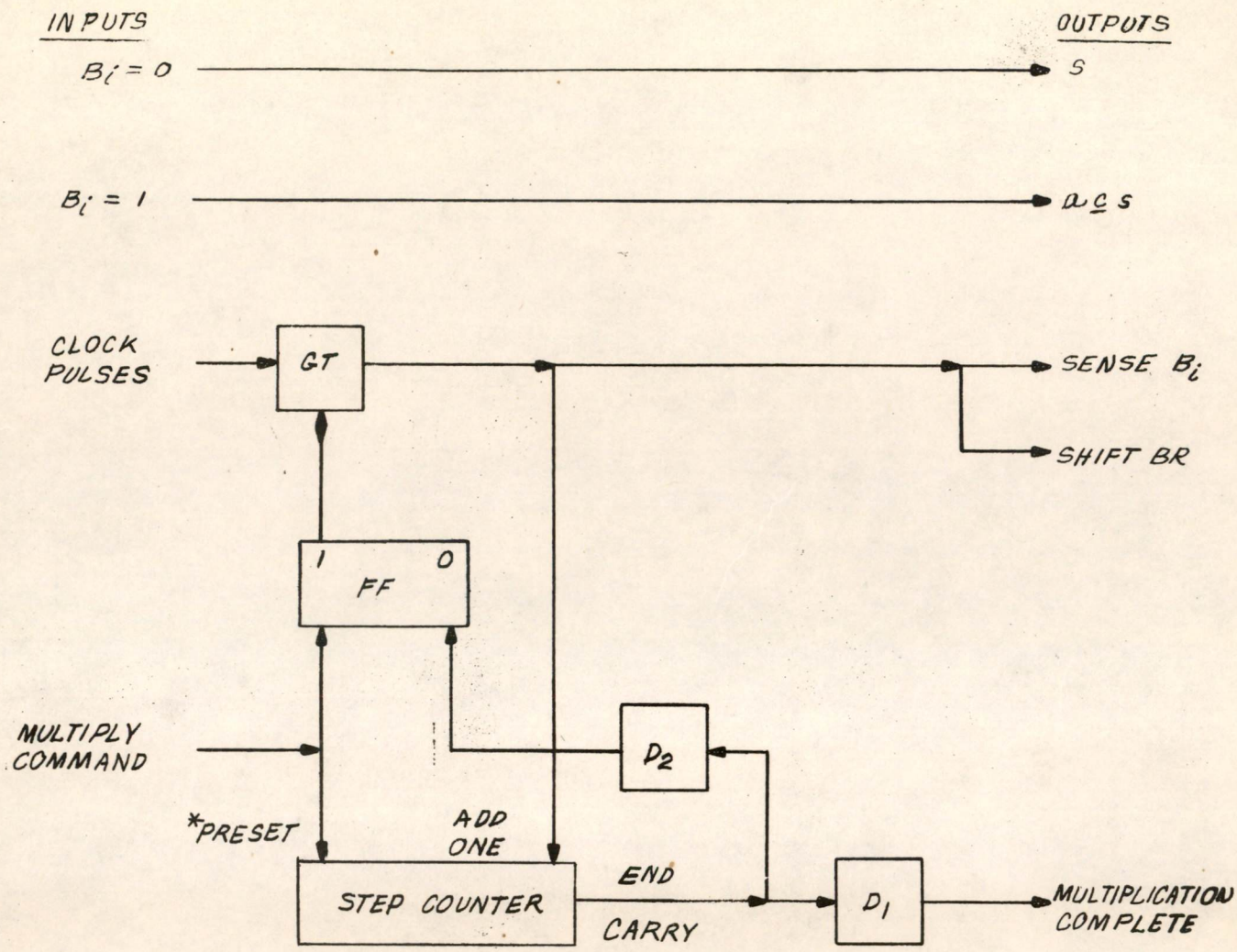


FIG IV -31

THE acs B2 MULTIPLY — ADDER WITH AR AND AcR

B-56242



* PRESET TO END-CARRY ON n-1st ADD-ONE PULSE

FIG. IV-32

CONTROL FOR K- a c s D

A-56221

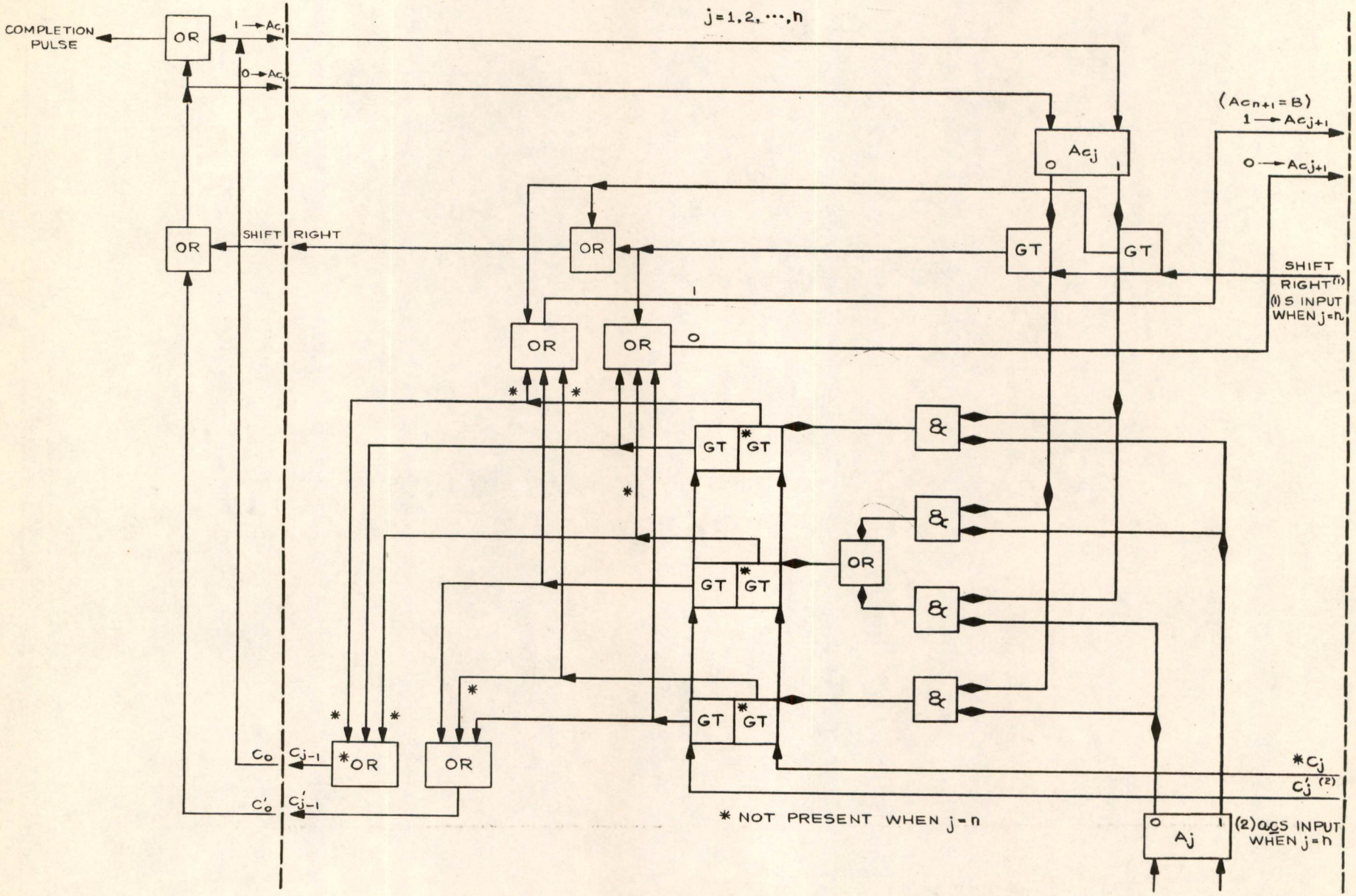


FIG IV -33

THE acs D MULTIPLY — ADDER WITH AR AND AcR

Drawings required for Chapter IV

1. A-56157
2. A-56158
3. A-56592
4. C-56164
5. Typed
6. B-56163
7. A-56200
8. A-56193
9. C-56180
10. C-47013
11. A-56593
12. C-56207
13. A-56591
14. A-56611
15. B-56228
16. A-56613
17. C-56165
18. B-56252
19. A-56192
20. A-56223
21. A-56229
22. C-56322
23. A-56612
24. C-56271
25. C-56233
26. A-56245
27. B-56243
28. A-56224
29. A-56244
30. A-56222
31. B-56242
32. A-56221
33. B-56241
34. C-56233

Chapter V

The Class II Multipliers

5.1 Introduction

Consider the possibility of using the sum of q partial products as the i^{th} digit group and summing these groups in the same order as the partial products were summed by the Class I multipliers. The multipliers which proceed in this manner will be called the Class II multipliers as long as q 1; if q = 1 then the Class I multipliers are obtained. Thus the Class II multipliers are a direct generalization of the Class I multipliers. The process of adding in one digit group will be termed a step in the multiplication. Thus I_q steps will be required to complete the multiplication, where I_q is the integer satisfying $n/q + 1 \quad I_q = n/q$.

The problem of forming the i^{th} digit group (termed a q-order partial product) is more complicated than it was for the Class I multipliers. In fact if q = n then the entire problem of performing the multiplication becomes one of designing the equipment to form the digit group. However, the design presented below for this equipment requires a tremendous number of diodes (and hence of isolating and amplifying circuits) when q = n if n has a value such as 16. It will also be found, however, that the multiplication time decreases as q increases. Therefore, some compromise is desirable in choosing the value of q. Whatever compromise is chosen, variations are possible according to the method used for summing the digit groups, i.e., according to the specific multiply-adder used. Many of the multiply-adders developed for the Class I multipliers can be used in the realization of Class II multipliers with the main alteration that the shift right performed at each step must be for q digits instead of one digit. This alteration may or may not require added equipment.

5.2 Realization of the Class II multipliers

5.2.1 The arithmetic section

The interconnections of the blocks which constitute a general arithmetic section are shown in Fig. V-1. A discussion of the various blocks in this figure is given below.

5.2.1.1 Forming the digit groups

A general description of the method used for forming the digit groups follows. The first digit group is the sum of the first of partial products which in turn is the product of the rightmost of digits of the multiplier (B in Fig. III-1) with the multiplicand (A in Fig. III-1). This number will be at most n + q digits in length. It can be found by first devising a circuit which will form all of the $2^q - 1$ possible products of a q digit number with the n digits of A (omitting the zero product). Such a circuit will be called a product generator. These $2^q - 1$ products can then be supplied to an electronic switch controlled by the rightmost q digits of B, i.e., the outputs of the rightmost q flip-flops of the B register, which will select the correct product. This latter circuit will be called a product selector. In order to form the next digit group, it is only necessary to shift the contents of the B register q digits to the right so that the product selector will be controlled by the next group of q digits of B. Thus the equipment for forming the digit groups consists of the product generator, the product selector and a set of shift gates used with B register. The interconnections of the flip-flops of the B register with these gates are shown in Fig. V-2. With the arrangement shown it is possible to shift the contents of the B register q digits to the right with a single pulse. It should be noticed that n is assumed divisible by q in this figure. The situation when n is not divisible by will be discussed in section 5.2.1.3.

The product generator is to supply the $2^q - 1$ products of any q digit number with the n digits of A. Therefore it must have inputs from each of the n flip-flops of the A register, and it must supply $2^q - 1$ groups of at most n + q lines per group (one line per digit) as outputs. The circuits designed to perform the desired function is shown in general form in Figure V-3. It utilizes the fact that 2^k times a binary number is that number shifted k digits to the left, with zeros inserted in the k rightmost digits. No output lines are provided for digits which must be zero. The equations at the bottom of the figure indicate how the circuit would be extended. The adder chains which occur in this product generator will require some sort of isolating circuits in the carry lines between the adders. The delay in the propagation of the carry level thru the chain must be considered when finding the speed of the multiplier. Furthermore, the output of one set of adders may be used to drive other adders. For example, if q = 2, then the "3A lines" will be used in forming 7A. Additional isolating circuits will be needed here, and the magnitude of the additional delay must not be neglected.

The product selector to be used is a simple combination of diode "and" and "or" circuits; it is illustrated in Fig. V-4 by an example for which q = 2. A general circuit is difficult to draw because it is desirable to omit lines and associated diodes which must always transmit a zero, but it is simple to derive a product selector for any value of q from the basic plan presented in this figure. Certain multiply-adders require the inverses of as well as the direct digit inputs. Inverters must be added at the outputs of the product selector in these cases.

5.2.1.2 The multiply-adder

The multiply-adders developed for the Class I multipliers can be adapted for use here. All multiply-adders will require n + q digit columns with varying degrees of simplification possible in the end sections. The extra q digit columns may be considered as added to the left end of the multiply-adder to accommodate new digit columns numbered 0, -1, ..., - (q - 1).

It is highly desirable that the multiply-adders be altered so that a q digit instead of a single digit shift right is performed by any command pulse which requires a shift. Besides requiring changes in the internal connections of the multiply-adders, this alteration may also require additional equipment in each digit column, as the case of a.cs B1 for example. Furthermore, the q digit shift will require that cirtain multiply-adders be extended to the right. Specifically, multiply-adders which do not perform a complete addition but perform only a single partial carry at each step must be extended $(q - 1)(I_q - 1)$ digit columns to the right, while those which shift before the addition is complete e.g., as.c B must be extended by q digit columns. In any case, these added multiply-adder stages require less equipment than a "normal" stage. The extension of the multiply-adder must be accompanied by an extension of the accumulator register.

In addition to a possible effect on the equipment count of the change to a q digit shift, a change in the multiplication time may result. In particular, the necessary change of connections within the D type multiply-adders may increase a maximum stage time-lapse slightly, which may in turn affect the multiplication time.

It will be possible to eliminate some equipment from certain of the multiply-adders which required extra separate shift inputs when used in the Class I multipliers, e.g., acs B2. These extra inputs were required because of the way in which these multipliers formed the digit groups and are not necessary when the multiply-adders are used in Class II multipliers.

5.2.1.3 Use of the B register to store part of the product

If we allow the multiplier to be lost during the multiplication then a number of flip-flops and gate tubes can be saved by using the B register to store part of the product. In order that it be possible to send the outputs of the rightmost q digit columns of the n + q digit column multiply-adder directly to the B register, several conditions must be satisfied. First the leftmost section of the B register must contain q flip-flops. If n is not divisible by q then $qI_q - n$ flip-flops must be added to this section to fulfill this condition. The other conditions which must be satisfied are: (1) The multiply-adder used performs a complete addition at each step and does not shift before completing this addition. (2) The B register is shifted before or at the same time that the q completed digits of the product arrive at the B register from the multiply-adder. This condition is necessary in order that the contents of the first section of the B register are not lost.

If the last two conditions of the preceding paragraph are not satisfied, then various alterations must be made. If the first condition is not satisfied either because the multiply-adder performs only a partial carry at each step, or because the multiply-adder shifts before completing the addition, then, as was remarked in the preceding section, the multiply-adder and the accumulator must be extended to the right. The q rightmost outputs of the extended multiply-adders may be sent directly to the B register provided that the second condition is met thus still saving some flip-flops. It becomes impossible to satisfy the second condition if acs D is the multiply-adder to be used. In this case the q digits begin arriving at the leftmost section of the B register before the acs command pulse has finished propagating thru the multiply-adder and hence before the B register can be shifted (since the digit group which controls the action of the propagating command pulse is selected by the contents of the rightmost section of the B register). In this case a

buffer section consisting of q flip-flops and q gate tube pairs must be inserted before the B register[1]. This requires that the shift of the B register be delayed after the arrival of the q digits from the multiply -adder at the buffer section for approximately τ microseconds.

5.2.2 The control section

There will be just one step time for any Class II multiplier. Hence a K type control whose clock pulses initiate the steps or a C type control will give approximately the same speeds of operation, as discussed in section 3.2.3.1. Experience with the design of the controls presented in Chapter IV shows that as long as the clock pulse period required is great enough to ensure that the flow of clock pulses will be stopped reliabl , then the K type control will usually be preferable. K type controls were chosen for the examples of Class II multipliers presented in section 5.4.

5.3 The multiplication time

The time for the formation of the digit groups and delays in the multiply-adder will determine the initial time((see section 3.3). The time for the formation of the digit groups will be measured from the instant when the outputs from the A register flip-flops stabilize after the read-in pulse to the time when all the outputs from the product selector have stab lized. This time must be considered in two parts: (1) A time which need be allowed only at the beginning of the multiplication and which will be known as the initial formation time (IFT). This is the time until the stabilization of the outputs from the product generator. (2) A time which elapses between a shift pulse to the B register until the stabilization of the outputs from the product selector and which must be allowed for at each step of the multiplication.

It may be permissible to overlap intervals taking advantage of flip-flop delays and thus eliminate the need for a buffer section. However, the safe solution is chosen here.

This will be known as the final formation time (FFT).[2] The multiply-adder time delay which must be added to the total digit group formation time to obtain the initial time is easily estimated in each particular case.

The estimation of the step time is simple when no buffer section is required. It is then only a matter of choosing the greater of two sums: the sum of the appropriate maximum time-lapses or the sum of the final formation time with the delay which may be necessary between the occurrence of the first command pulse of the step and the time when the B register shift can be commanded. This delay will be termed the digit group hold time (DGHT). It is zero for all B type multiply-adders except $\bar{c}$.a.s B for which it is equal to $\bar{c}$:a, and non-zero for all D type multiply-adders (for these presented in Chapter IV, it is essentially the propagate time of the first command pulse of a step). If a buffer section is required, then the estimation of the step time is complicated by the need for allowing the stabilization of the buffer section flip-flops before commanding the B register shift.

The ingredients of the final time (see section 3.3) can be readily estimated in each specific case.

5.4 Some Class II multipliers

Only the faster of the Class II multipliers which can be developed from the multiply-adders of Chapter IV will be discussed here. As an aid in discovering which multiply-adders to use in the construction of such multipliers, a comparison of their step times when n = 16 and q = 2 was made and is shown in Table V-1. This table indicates that either acs B2 or ac.s D3 would be a good choice in order to obtain a fast Class II multiplier.

[2] These times for the special case q = 2 and n = 16 are found from Figs. V-3 and 4 to be:

IFT = 16 (α +) + i = 0.74 μsec.
FFT = τ + + 2α + = 0.6 μsec. if

inverters are not necessary on the outputs of the product selector. If they are necessary, then

FFT = 0.7 μsec.

If a multiplier is built using ac.s D3 with q = 2 and a K type control as shown in Fig. V-5 with:

$$T = ac{:}s + s{:}ac = 1.2 \ \mu sec.$$

$$\int_1 = ac{:}s = 0.55 \ \mu sec$$

$$\int_2 = ac{:}s + \gamma + \tau - 3\delta \cong 1.0 \ \mu sec.$$

then, assuming that the read-in pulse occurs (ITL) - T μsec. before the multiply-command pulse, the multiplication time if n = 16 is:

t = initial time + (no. of steps - 1) (step time) + final time

$t \cong 0.74 + 0.7 + 0.06 + (8 - 1)\ (1.2) + 1.0 = \underline{10.9 \ \mu sec.}$

The equipment count is found to be 5730, taking into account the fact that several diode level gates can be omitted in the Ac_{-1}, Ac_0 section of the multiply-adder (since the Ac_{-1} flip-flop must always hold a zero after the shift command).

A multiplier using a slight modification of acs B2 and with values of n and q of 16 and 2 is shown in Figs. IV-6 and 7. The multiplication time of the multiplier when T = 0.7 μsec., $\int_1$ = 0.5 μsec. and $\int_2$ = 1.4 μsec. is approximately 8.4 μsec. The equipment count is 7310.

The multiplication time of Class II multipliers using either acs B2 or acs D3 could be improved beyond the values quoted above by using a greater value of q; however, the equipment count, which is already larger in both cases than any encountered in Chapter IV, is thereby increased rapidly while the multiplication time decreases slowly. Furthermore, the Class III multipliers give better results than can be expected from the Class II multipliers with any value of q.

Table V-1

Step Time

(n = 16, q = 2)

Multiply Adder	Sum of Max. time-lapses for one-step (μsec.)	DGHT + FFT (μsec.)	Step Time (μsec.)
a.cs B1	0.6 + 0.55	0 + 0.6	1.15 *
a.cs B2	0.6 + 0.5	0 + 0.6	1.1 *
acs B1	0.86	0 + 0.7	0.86 *
acs B2	0.6	0 + 0.7	0.7 *
a.c.s B	0.6 + 1.1 + 0.5	0 + 0.6	2.1
as.c B1	0.6 + 1.1	0 + 0.7	1.7
ac.s B1	0.55 + 1.26	0 + 0.7	1.81
ac.s B2	1.65 + 0.5	0 + 0.6	2.15
ac.s D1	0.55 + 0.65 **	0.6 + 0.7	1.3
ac.s D2	-	1.16 + 0.7	≥ 1.86
ac.s D3	0.55 + 0.65	0.28 + 0.7	1.2
asc B	1.6	0 + 0.7	1.6
acs B1	1.34	0 + 0.6	1.34
acs D	0.66	0.6 + 0.7	1.3
c.a.s B	0.8 + 0.55 + 0.55	0.8 + 0.7	1.9

* It must be remembered that these arithmetic sections require a final carry operation.

** Increased maximum time-lapses must be used because of the two digit shift.

Drawings for Chapter V

1. A - 47064
2. A - 57303
3. D - 57278
4. B - 57304
5. A - 57431
6. A - 57430
7. D - 47066

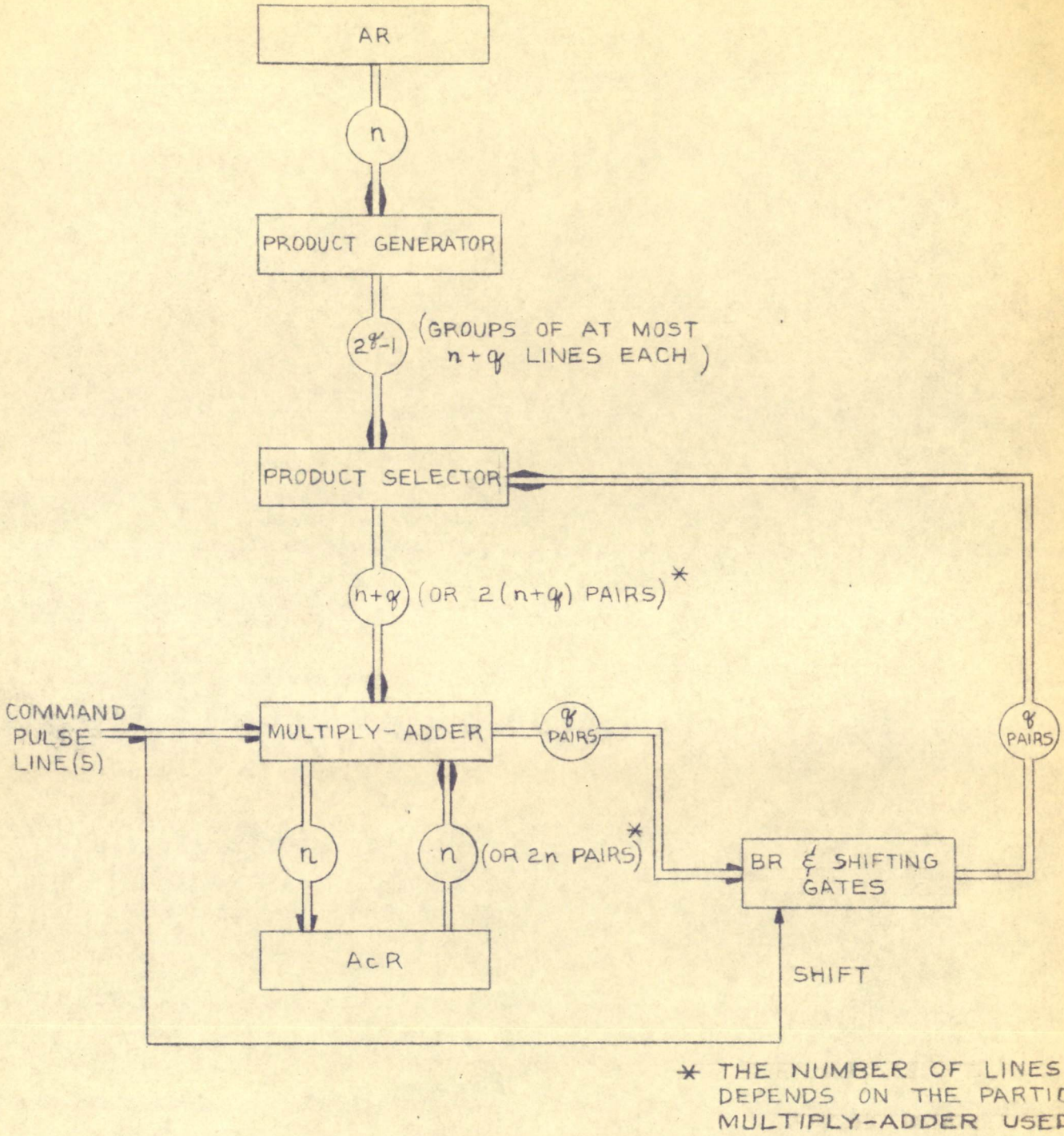


FIG. V-1

ARITHMETIC SECTION OF CLASS II MULTIPLIERS

A-47064

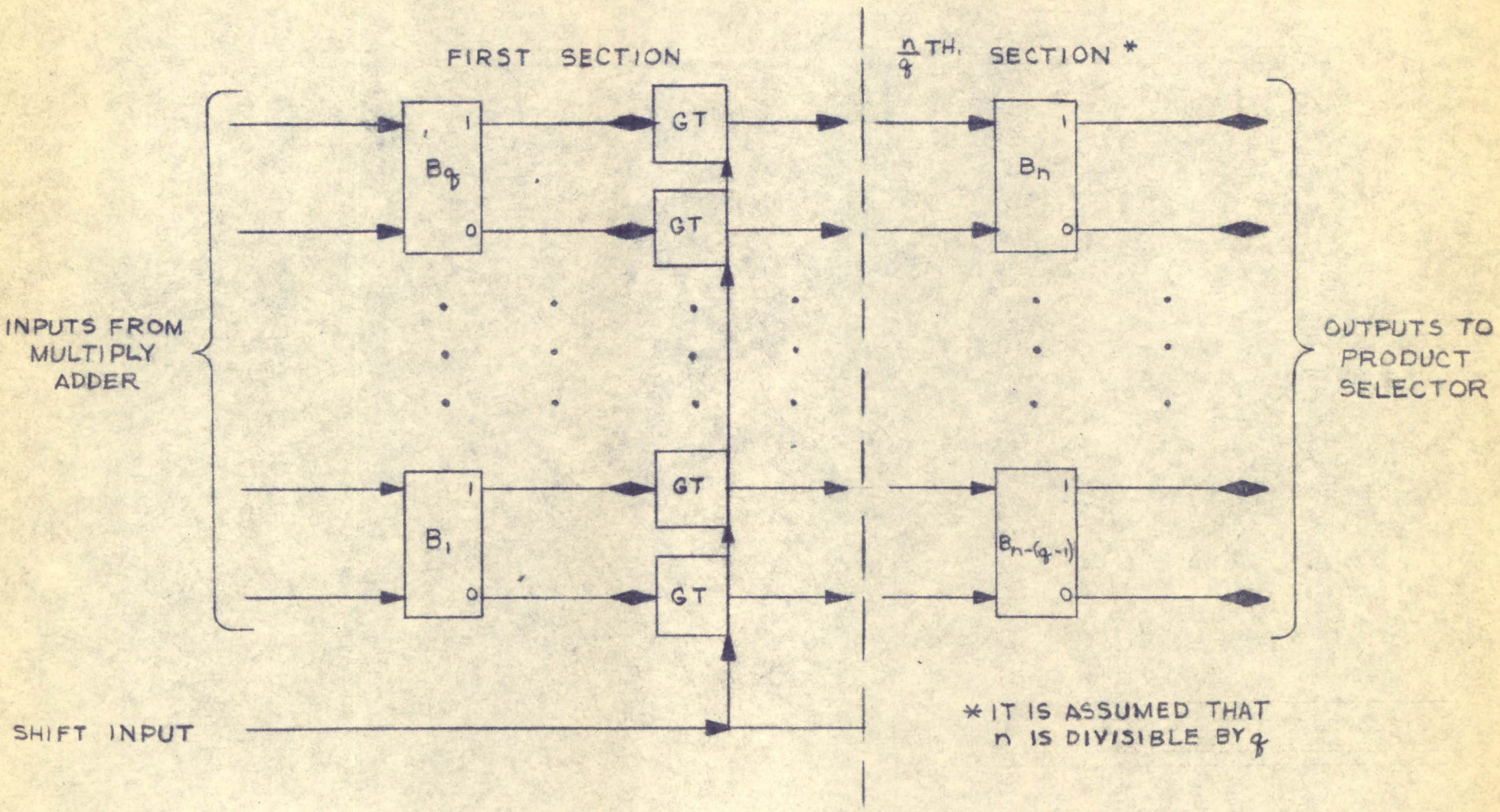
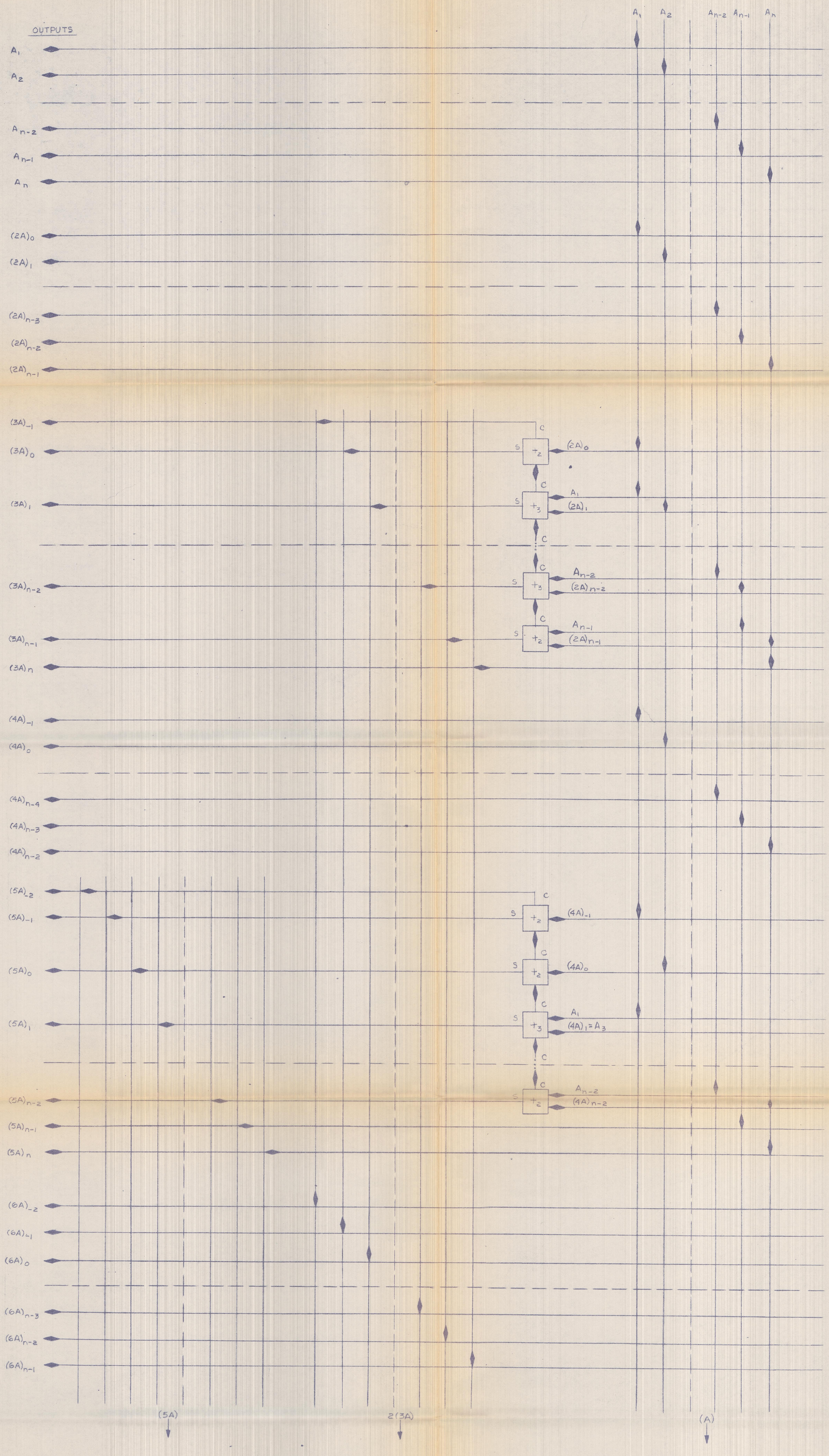


FIG. V-2

THE B-REGISTER FOR CLASS II MULTIPLIERS



$7A = 2(3A) + A$

$8A = 2^3 A$

$\vdots$

$(2^q - 1)A = 2(2^{q-1} - 1)A + A$

FIG. V-3

A PRODUCT GENERATOR FOR THE CLASS II MULTIPLIERS

-57304

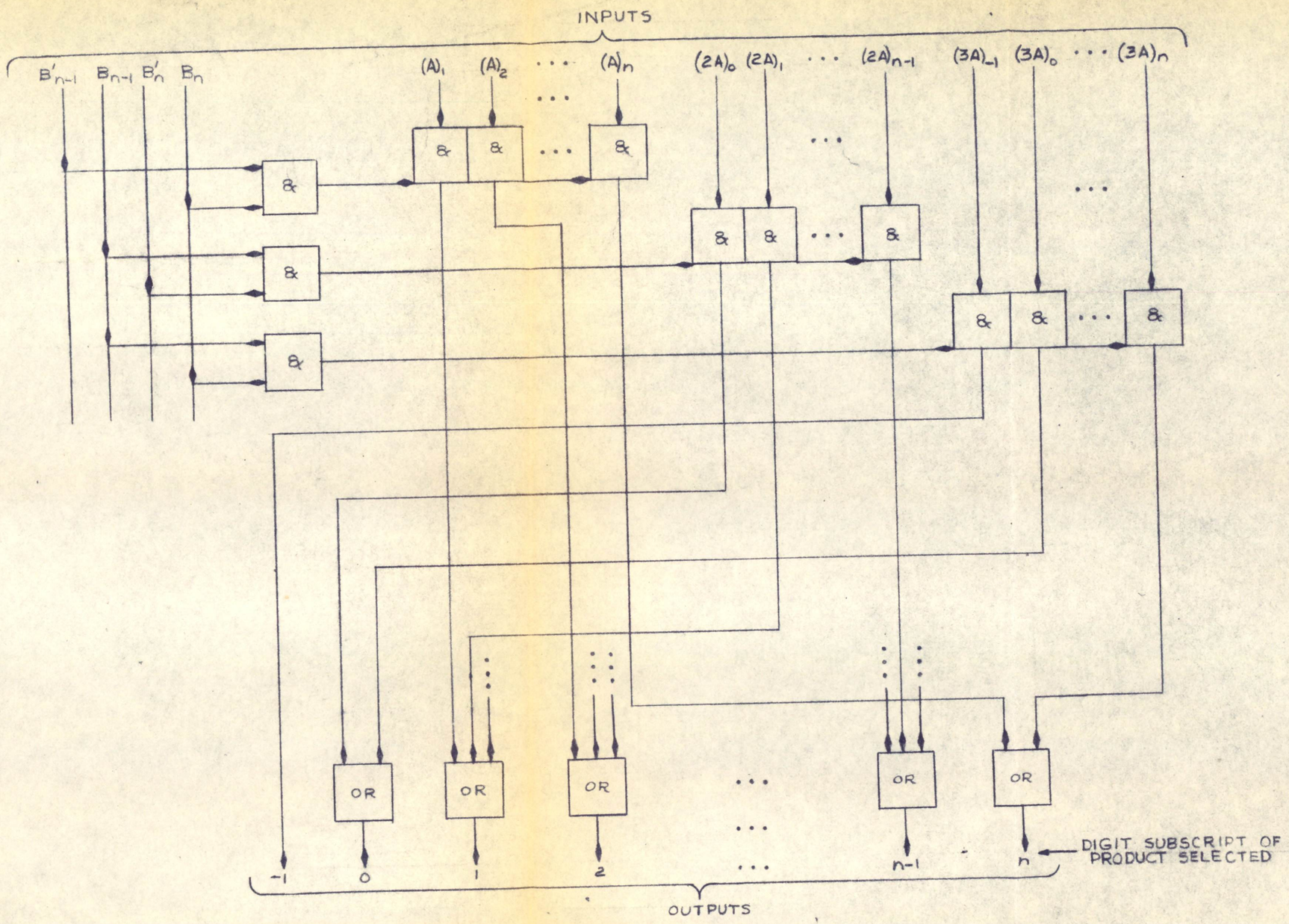


FIG. V-4

THE PRODUCT SELECTOR CIRCUIT FOR $q=2$

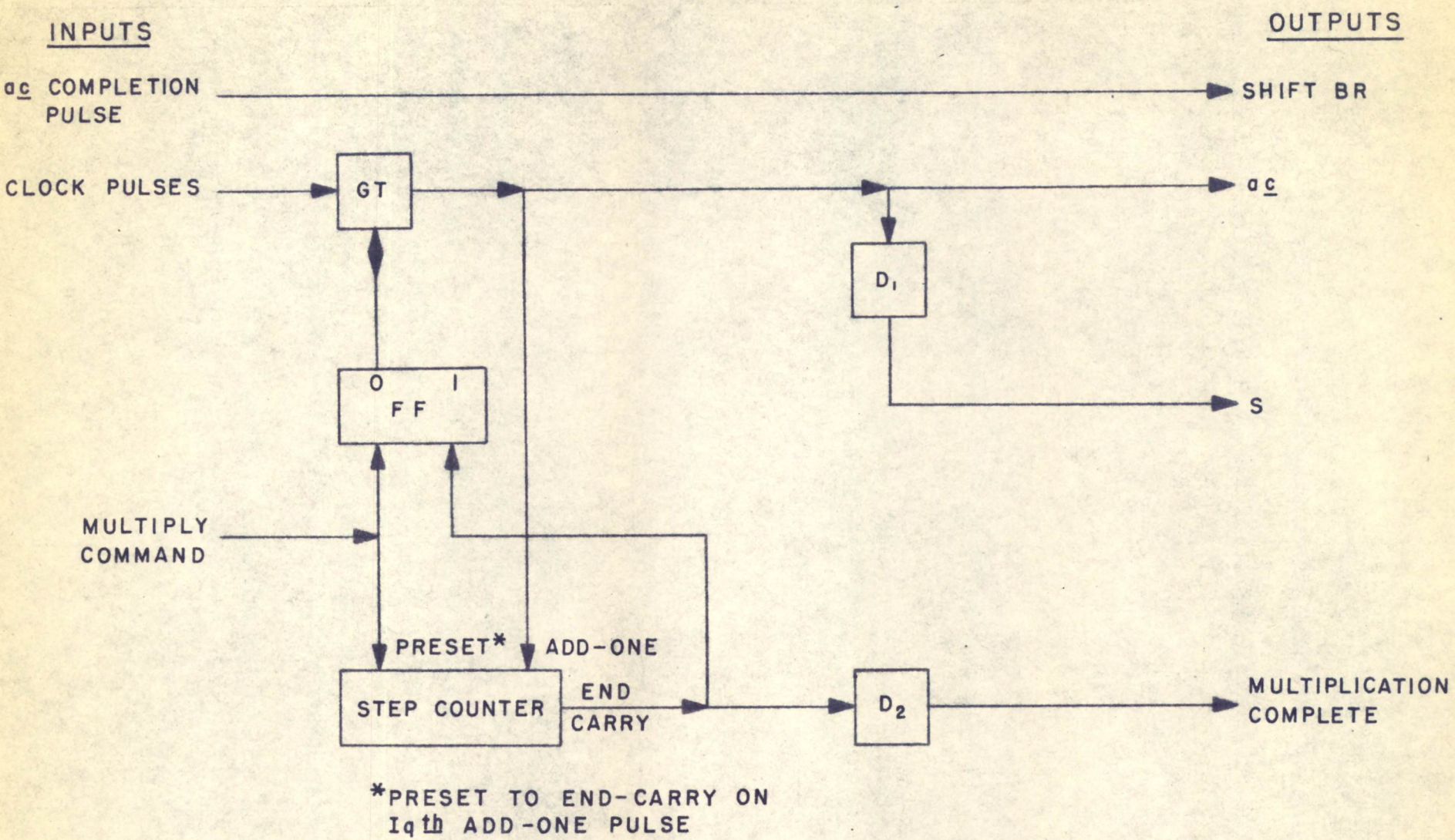


FIG. V-5

CONTROL FOR K-ac.s D3,q

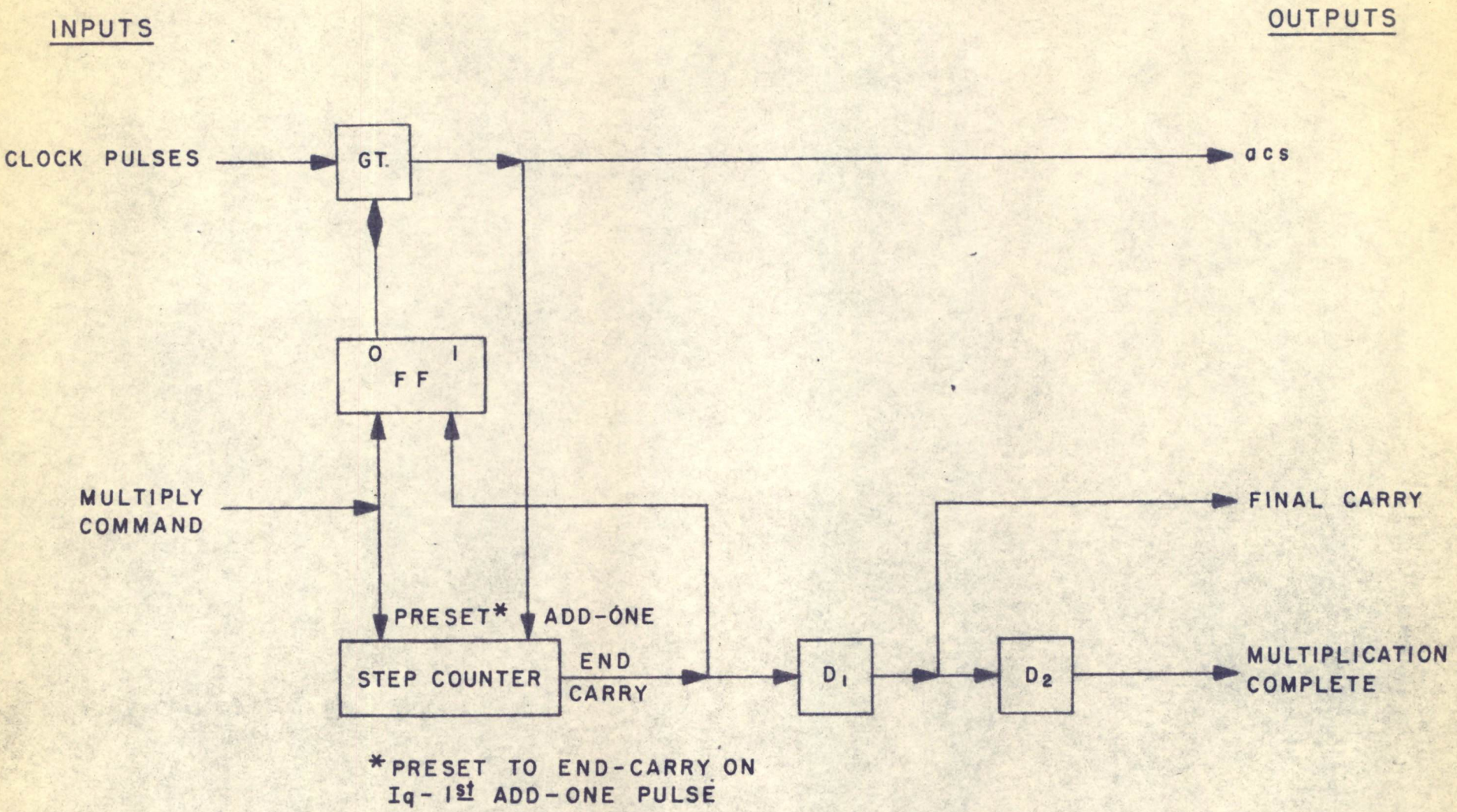


FIG. V-6

CONTROL FOR K—acs B2,q

D-47066

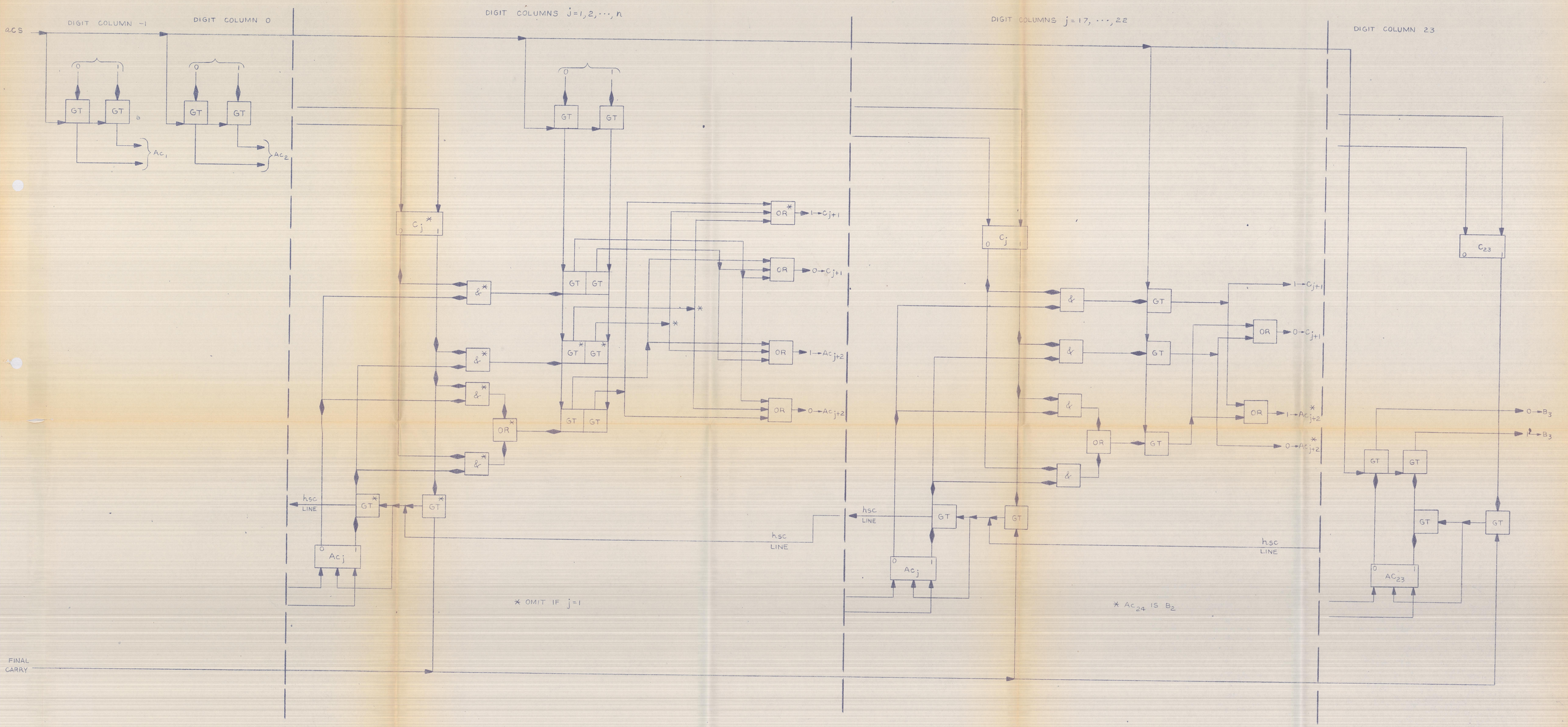


FIG. V-7

ARITHMETIC SECTION FOR acs B2, q=2 AND n=16 (OMITTING BR)

Chapter VI

The Class III Multipliers

6.1 Introduction

Instead of forming and summing the q-order partial products one by one as in the Class I and Class II multipliers, the following procedure is possible: The q-order partial products may be imagined to be divided lengthwise into groups of p digits each, as illustrated in the example of Fig. VI-1. The way in which the digit groups are chosen is indicated in this example, and these groups are formed and summed in the order indicated by the numbers associated with them. If $q = 1$ then the q-order partial products are in general $n + q$ digits in length, and there will be I digit groups, where I is the integer which satisfies $(n + q)/p + 1 > I \geq (n + q)/p$. If $q = 1$ then I is found from $n/p + 1 > I \geq n/p$. A realization of this method can be achieved with the use of adaptations of the previously developed product generator, product selector and acs B1 multiply-adder circuits. Other multiply-adders of Chapter IV can be adapted for use in a Class III multiplier, but the ones developed from the acs B1 type are particularly adaptable to the case where $p > q$, as will be seen later. Furthermore, the acs B1 has a relatively small per digit column equipment count.

The Class III multipliers may be denoted by the type of control and multiply-adder used followed by the values of p and q. Thus K-acs B3, $p = 2$, $q = 3$, or more briefly K-acs B3, 2×3.

6.2 The realization of Class III multipliers

6.2.1 The arithmetic section

The general arrangement of the blocks making up the arithmetic section of a Class III multiplier is shown in Fig. VI-2. The most striking difference between this block diagram and that for the arithmetic section of

The Class II multipliers (Fig. V-1) is that more than one product selector is used. Each product selector will be associated with a particular q-order partial product and will supply p digits of that product in forming a digit group. The number of q-order partial products is I_q, which is the integer which satisfies $n/q + 1 > I_q \geq n/q$. If the product selectors are controlled by the B register as shown, then the first product selector will supply digits of the I_qth q-order partial product, etc. It will be noted that the B register must not be shifted during the multiplication and hence there will be no possibility of using it to store part of the product. However, in the formation of the digit groups it will be found necessary to shift the A register. It will further be found that we can use the A register for storage of part of the product (as is indicated by the p pairs of lines joining the multiply-adder to the A register) if we are willing to lose the multiplicand during the multiplication.

A single pulse at each will be needed by the multiply-adder since we are going to consider only multiply-adders developed from acs B1. This step pulse will also be used to shift the A register and to advance the product generator (see section 6.2.1.1). In one form of multiply-adder developed from acs B1, a final carry pulse is required at the end of the multiplication.

Fig. VI-3(b) shows an example of Fig. VI-2 for the particular case of $n = 6$, $p = 3$, $q = 2$. The labels on the A register flip-flops correspond to the digits which they hold at the beginning of the multiplication. By the end of the process these digits have been shifted into the product generator (and thereby lost) and replaced by the rightmost n digits of the product. The six gate tubes between the two sections of the A register serve both as part of the equipment for forming the digit groups and as the multiply-adder,

but are placed as shown for symmetry.

Certain of the level lines in Fig. VI-3(b) are labelled with the digits which they will carry when the product selectors are supplying the digits of the $t^{\underline{th}}$ group. The position and source of each digit in relation to the long-hand scheme for binary multiplication is indicated in Fig. VI-3(a). The outputs of the product selectors are changed from the digits of one group to the digits of the next digit group by the step pulse. The step pulse is also needed by the multiply-adder in order to add the digit groups with the contents of the accumulator.

6.2.1.1 Formation of the digit groups

The product generator for the Class III multipliers is similar to that used by the Class II multipliers since they both utilize a k digit shift left in order to multiply a binary number by 2^k. The Class III product generator is more complicated, however, because it must produce p digits of each q-order partial product at a time and it must supply these successively the same set of lines. It is found that the circuit shown in Fig. VI-4 will operate correctly. The details of the "D_c" (for "controlled delay") unit of this figure is shown in Fig. VI-5. When there is only one output from a D_c unit it is understood to come from the "1" side of the flip-flop. These units are supplied with a pulse just before the inputs to the product generator circuit are changed with the purpose of retaining certain digits which will be needed in the formation of the next digit group.

The product generator circuit requires as inputs one level from each of the rightmost p digit sections of the A register. Proper operation requires that the A register be shifted p digits at a time. This may be thought of as "sensing" p digits of A at a time.

The shift of the A register changes the inputs to the product generator

circuit. Since the flip-flop outputs do not change immediately after the shift pulse, this pulse can also be sent to the D_c units and the carry units. The step pulse can be used for this purpose, and its occurrence advances the outputs of the product generator so that the product selectors can select the next digit group.

Each product selector will receive $2^q - 1$ groups of p lines each and will put out p lines. The product selector circuits are all identical to or degenerate cases[1] of the one shown in Fig. VI-6. The circuit operates exactly like that used by the Class II multipliers.

6.2.1.2 The multiply-adder

The details of the summation procedure and two multiply-adder which have been devised for its realization will now be described. This description will proceed thru three cases: $p = q$, $p < q$ and $p > q$ discussed in terms of the specific examples $p = q = 2$; $p = 1$, $q = 2$; and $p = 3$, $q = 2$ with $n = 16$ throughout. A multiply-adder for a multiplier with any values of p and q can be designed using one of these examples as a model.

$p = q$

The example of Fig. VI-7[2], where $p = q = 2$ and $n = 16$ will serve as a basis for this discussion. Fig. VI-7(a) is meant to represent the arrangement of the 2-order partial products. The digits, represented by the smallest boxes, are divided into sub-groups of $p = 2$ digits each by the heavier

---

1. Only the product selector associated with the leftmost section of the B register can be degenerate, and this occurs when n is not divisible by q, i.e., when some of the digits of the leftmost group of q digits of B are always zero.
2. The multiplier based on this multiply-adder was previously known as the "2 by 2" multiplier.

lines. Fig. VI-7(b) shows the arrangement of blocks for a possible multiply-adder and the accumulator. The two lines from each product selector are assumed to supply the two digits of each sub-group in the same left-to-right order as the two digits possess in Fig. VI-7(a). The flip-flops of the solid string of 16 D_c units constitute the left half of the accumulator. The flip-flop of the remaining D_c unit is used for storage of carries.

We shall discuss the logical operation of this scheme by considering what happens to the first pair of digits put out by the first product selector - the outline of this sub-group is emphasized in Fig. VI-7(a). Since the contents of the accumulator and carry flip-flops before the first step pulse are all zero's, there can be no carries into the leftmost pair of adders, and hence the digits under consideration will be stored unchanged in the leftmost pair of flip-flops of the accumulator by the first step pulse. At the second step pulse, the complete sum of these digits with (1) the second pair of digits put out by the second product selector and with (2) any carry from adder number five added in will be stored in the next pair of accumulator flip-flops. We shall now see where the carry which was added in originated and that it was added into the correct digit column: This carry was generated from the sum of (1) the first pair of digits put out by the second product selector with (2) the second pair of digits put out by the third product selector and (3) any carry entering from the seventh adder. A check of Fig. VI-7(a) will show that this carry was added into the proper digit column. Similarly the carry outputs of the seventh, ninth, etc. adders may be checked for their origin and proper inclusion in the sum. The carry output from the third adder is summed with the second pair of digits put out by the first product selector, which is a correct procedure according to Fig. VI-7(a). The carry

output of the first adder will be stored by the second step pulse, and will then proceed to add with the rightmost digit of the third pair of digits put out by the first product selector. Thus all the carries are correctly handled, and the accumulator flip-flops and the carry flip-flop, considered as the leftmost digit after the contents of the accumulator, contain the leftmost n + 1 digits of the complete sum of the first two digit groups.

The third step pulse will now cause the sum of our original pair of digits, which have been altered by summation and which are now stored in the second pair of flip-flops, and with (1) the third pair of digits put out by the third product selector (which is as it should be according to Fig. VI-7(a) and with any (2) incoming carry, to be stored in the third pair of accumulator flip-flops. Again all the carries can be checked and again it will be found that the accumulator flip-flops plus the carry flip-flop contain the leftmost n + 1 digits of the sum of the first three digit groups. Thus after the eighth step pulse, the sum of the whole column of digits situated above our original pair of digits, taking account of all carries arriving from the digits to the right, has been formed and the result stored in the eithth pair of accumulator flip-flops, i.e., the rightmost pair. Upon the next step pulse, these digits will be shifted off the end of the register and sent to the A register, which is also being shifted p = 2 digits at a time by each step pulse. The summation of the columns to the right of the one just considered can be similarly traced. It is found that each step pulse beginning with the second one transfers a pair of digits to the A register.

There are I = q pairs of digits to be put out by each product selector. After the ninth stepping pulse, the continued repetition of the above argument shows that the leftmost n + 1 digits of the sum of the nine digit groups are stored in the accumulator and carry flip-flops. But these include all of the non-zero digits of the 2-order partial products and hence the left-most n + 1 digits of the final product must be those stored. Since the final product cannot exceed 2n digits in length, the n + 1^{st} digit must be zero, i.e., the

contents of the carry flip-flop must be zero. The rightmost n digits of the product will be in the A register.

The method of summation discussed above and illustrated in Fig. VI-7 requires a chain of 16 adders. This is undesirable if utmost speed of operation is desired and indicated that an effort should be made to modify the multiply-adder. Furthermore, the modification used provides a stepping-stone to multiply-adders which can be used by the multipliers for which $p < q$ and $p > q$. It is a fairly simple modification of the system just discussed, but requires the addition of a considerable amount of equipment. However, it does break down the chain of 16 adders into 7 chains of two adders each. The modification is that the carry outputs of the third, fifth, etc. adders instead of being sent to the next adder to the left are sent to control gate tubes which in turn read into carry flip-flops upon the occurrence of the step pulse, i.e., D_c type units pulsed by the step pulse. This system is illustrated in Fig. VI-8 (which does not shown the complete multiply-adder) and justified below.

The effect of introducing the carry storage is simply to delay the inclusion of these carries into the summation process by one step. This one step delay necessitates that the carry flip-flop outputs be returned one pair of adders to the right of the position to which the carry would otherwise have been sent. The leftmost pair of adders can be eliminated since there will never be a carry input to these and hence they will always transmit the output of the first product selected or unchanged. After the $I = 9^{\underline{th}}$ stepping pulse, all the digits of the partial products will have been added in; however, some digits may remain in the carry flip-flops. (It may be noted that after any step pulse the digits stored in the accumulator flip-flops will not necessarily be the leftmost $n + 1$ digits of the sum of several groups of digits of the q-order partial products because of the carry storage.) This necessitates a

final carry operation which is simply a high-speed carry. The blocks and the control pulse for this operation are not shown in Fig. VI-8, but are included in the drawing of Fig. VI-9.

The two schemes which have been discussed may be distinguished as the acs B3, 2 x 2 and the acs B4, 2 x 2 multiply-adders respectively.

p < q

The example of Fig. VI-10, where p = 1, q = 2 and n = 16, is used to illustrate this case[3]. Fig. VI-10(a) represents the arrangement of the q-order partial products. The digits and the sub-groups are in this case identical. Fig. VI-10(b) shows the arrangement of blocks for the major portion of the multiply-adder and for the accumulator. The multiply-adder used here is taken directly from the second type discussed in the p = q case and can be designated as acs B4, 1 x 2. The flip-flops of the solid string of 16 D_c units constitute the accumulator, while the flip-flops of the remaining D_c units are for carry storage.

The only blocks omitted from Fig. VI-10(b) are those necessary for the final carry operation. Thus the operation of the circuit for the I = 18 step pulses which preceed the final carry can be checked thru using this figure in a manner similar to that used above in the p = q = 2 case. The blocks for the final carry operation are included in the more detailed Fig. VI-11.

p > q

The example of Fig. VI-12, where p = 3, q = 2 and n = 16, is used to illustrate this case. As before Fig. VI-12(a) represents the arrangement of the q-order partial products, which are shown divided into sub-groups of three digits each. Fig. VI-12(b) shows the arrangement of blocks for the multiply-adder and the accumulator, omitting those necessary for the final carry operation. This multiply-adder may be designated acs B4, 3 x 2. Again the flip-flops of the solid string of 16 D_c units constitute the

[3] The multiplier based on the multiply-adder discussed here was previously known as "s = 2".

accumulator, while the flip-flops of the remaining D_c units are for carry storage. The operation of the circuit for the I = 6 step pulses which precede the final carry can be checked thru using this figure in a manner similar to that used in the p = q case.

It must be noticed that adders are cascaded vertically, that is several adders occur in the same digit column in the multiply-adder given for this case. Since diode adders can handle this situation with less complication of control pulses and equipment, it was stated in the first paragraph of this chapter that multiply-adders developed from acs B1 are more adaptable to the case where p>>q. An alternative to the form of the acs B4 multiply-adder used in Fig. VI-12, but which is not as good as this arrangement, is as follows: The three input adder and controlled delay groups could be arranged in a diagonal proceeding from the upper right corner to the lower left corner. In the specific example given here, this would involve a carry propogate thru 21 three input adders instead of a maximum of three in the preferred arrangement. Mention of this alternative is made for the sake of completeness.

The blocks for the final carry operation are included in the block diagram of Fig. VI-13. The A register has been included in this figure to clarify the connection between it and the multiply-adder.

In each of the foregoing examples a multiply-adder was introduced which required a final carry operation. Nothing was said in these examples to assure that the systems presented would perform this final carry without error. An error would result if the pulse mixers in these multiply-adders and hence the complement inputs to certain accumulator flip-flops could receive two pulses separated by less than the resolution time of the flip-flop.

It can be shown that if $q > 1$ these certain accumulator flip-flops will receive at most one complement pulse during the final carry operation and therefore no error can occur. This demonstration makes use of the fact that the two leftmost digits of the $n + q$ digit product of any n digit binary number with any q digit binary number cannot both be unity.

6.2.1.3 Use of the A register to store part of the product

The A register must be arranged in I_p groups, where $n/p + 1 > I_p \geq n/p$, of p flip-flops each as indicated in Fig. VI-13. If the outputs of the rightmost p adders of the multiply-adder (which provide p digits of the completed product) are sent directly to the leftmost group of the A register flip-flops by the step pulse (assuming that this group contains p flip-flops - extra flip-flops and gate tubes are added when necessary), then an A register will be filled with digits of the final product after the I_p th step pulse. Since the number of step pulses I is $\geq I_p$, a number equal to $I - I_p$ of buffer sections consisting of p controlled delay units each must be included between the outputs of the rightmost p adders and the leftmost group of A register flip-flops in order that no digits of the product be lost.

6.2.2 The control section

The acs B4 multiply-adders described above require only one pulse per step, and, since it will be necessary to allow the same time for each step, a clocked type control can be used for optimum speed. A possible control is shown in Fig. VI-14.

6.3 The multiplication time

A general expression can be developed for the multiplication time of the K-acs B4, p x q multipliers. This expression is:

the K-acs B4, p x q multipliers. This expression is:

$$t \triangleq t_{max} = t_{min} = (2n - pI + 1)\gamma + \tau + \begin{cases} I\left[\gamma + \tau + (p+1)(\alpha+\omega) + (q+1)(\alpha+\omega+i)\right] & (p+1 \geq q > 1) \\ I\left[\gamma + \tau + (p+1)(2\alpha+2\omega+1) + \alpha + \omega + i\right] & (p+1 < q > 1) \\ I\ (\gamma + \tau + \alpha + \omega) & (q = 1) \end{cases}$$

where $(n+q)/p + 1 > I \geq (n+q)/p$ as before.

The clock pulse period must equal the step time. The step time is evident as the coefficient of I in the above expression and includes the full time for the formation of the digit groups since this must be allowed at each step. When developing the above expression the values of the delays in the control section of Fig. VI-14 are assumed to be:

$$\delta_1 = \tau + (1 - I_{sc})\gamma \text{ where } 1 + \log_2 I > I_{sc} \geq \log_2 I$$

$$\delta_2 = (2n - pI)\gamma + \tau$$

6.4 The equipment count

A general expression for the equipment count can be found. Some of the facts used in the derivation follow: The number of product selectors is I_q. The multiply-adder contains $I_q - 1$ chains of p three input adders each, and two gate tubes and a flip-flop are associated with each chain. The A register combined with the flip-flops necessary to store the outputs of the product selectors and the subsequent adder chains (exclusive of the carry flip-flops) number $pI + q(I_q - 1)$. If $pI + q(I_q - 1) - 2n > 0$, then this number of flip-flops will be fed directly from the leftmost outputs of the first product selector and must contain zero's upon the completion of the multiplication operation. The general expression, making use of the equipment weights assigned in Chapter II, is:

$$E = p\left[30I_p + (b + 6)\,n - b + c\right] + (c + 110)I_q + 50I_{sc} + 102n - 2c - 20$$

if $q = 1$ and $I_q \geq 3$, or

$$E = \begin{cases} p\left[6(3 + I_q + I_q + 2^n - qIq)\,2^q + 50I + (b - 6)I_q - b + c + d - 48\right] \\ + q\left[2(I_q - 1)(1 - 2^n - q\,Iq)\,2^q + (c + 60)I_q - 2c + e - 60\right] \\ + (1/2a + 2n2^n - qIq)2^q + 70I_q + 50I_{sc} + 58n - a + c + 10 \end{cases}$$

if $q > 1$ and $I_q \geq 3$ and where if $\begin{cases} q > 2,\ a = 130 \\ q = 2,\ a = 60 \end{cases}$

$p > q$; $b = 48$, $c = 10$, $d = 0$, $e = 10$

$p \leq q$; $b = 58$, $c = 0$, $d = 10$, $e = 0$

and the intergers I_p, I, I_q and I_{sc} satisfy

$$n/p + 1 > I_p \geq n/p$$

$$(n + q)/p + 1 > I \geq (n + q)/p$$

$$n/q + 1 > I_q \geq n/q$$

$$1 + \log_2 I > I_{sc} \geq \log_2 I$$

It will be noticed that all possible cases are not covered by the above expressions, but it is felt that all practical cases will satisfy the restrictions given, namely $I_q \geq 3$.

6.5 An optimum multiplier

There is no advantage to having a multiplier operate with a smaller multiplication time than that required to reduce the instruction overshoot time, O_1 (introduced in section 1.1), to zero. Thus if the multiplication time which just reduces O_1 to zero is specified and if n, the digit length of the numbers to be handled, is also specified, then the K-acs B4, p x q multiplier which has a minimum equipment count while giving a multiplication time less than or equal to the specified value may be considered to be the

optimum multiplier of this class. The equations to be handled in finding the values of p and q for this optimum multiplier are extremely awkward for two reasons: first they are not continuous, and second they are transcendental because of the 2^q terms in the equipment count expression. The first difficulty can be removed by replacing the integers I, I_q, etc. by their lower bounds and then letting p and q vary continuously. A curve can then be drawn in the p-q plane for any specified multiplication time using the new continuous time expression and the values for the time delays given in Table II-1. Furthermore, it will be found that the multiplication time increases as p decreases then q is constant, or as q increases when p is constant. Thus a set of points having integral coefficients can be determined on the p-q plane such that the multiplication time associated with each point is either less than or near (it may be slightly greater) the specified value. Substituting the time delay values from Chapter II into the new continuous equipment count expression which holds when $q>1$, it is found that the count given by this expression increases when p is increased and q is held constant and vice-versa. Thus only a few of those points of the above mentioned set which are nearest the origin need be singled out for more detailed attention since the equipment counts for the remainder will be too large. Other points for $q = 1$ can easily be found and included in this group. The "exact" multiplication time for each point of this group should now be calculated, since some of these values may exceed the specified value and thus be eliminated. The exact equipment count for the remaining points can then be calculated and the multiplier with the minimum count discovered. In this way the second of the above mentioned difficulties, namely the solution of transcendental equations, is avoided.

Pertinent facts concerning the optimum multiplier when n = 16 and $t \leq 6.5$ μsec., the value which will be specified for the multiplication time required for zero O_1 when a magnetic core memory is used, are:

p = 2, q = 1

t = 6.3 μsec.

E = 5794

T = 0.58 μsec. or clock frequency = 1.72 mc

It should be emphasized that this is merely the optimum K-acs B4, p x q multiplier.

6.6 Two Other Class III multipliers

The K-acs B4, p x q multiplier having minimum equipment count when $t \leq 10$ μsec. and n = 16 was determined, along with the consequent values of t and E, in order to have a Class III multiplier to be compared with the previously developed multipliers in this time interval. The pertinent values are p = 3, q = 2, t = 8.9 μsec and E = 5330. The clock frequency required for the stated multiplication time is 894 kilocycles per second.

The K-acs B4, p x q multiplier with the least equipment count when n = 16 can be determined. For this multiplier p = 1, q = 2, t = 19.8 μsec., E = 4314 and the clock frequency required is 1.0 megacycle per second.

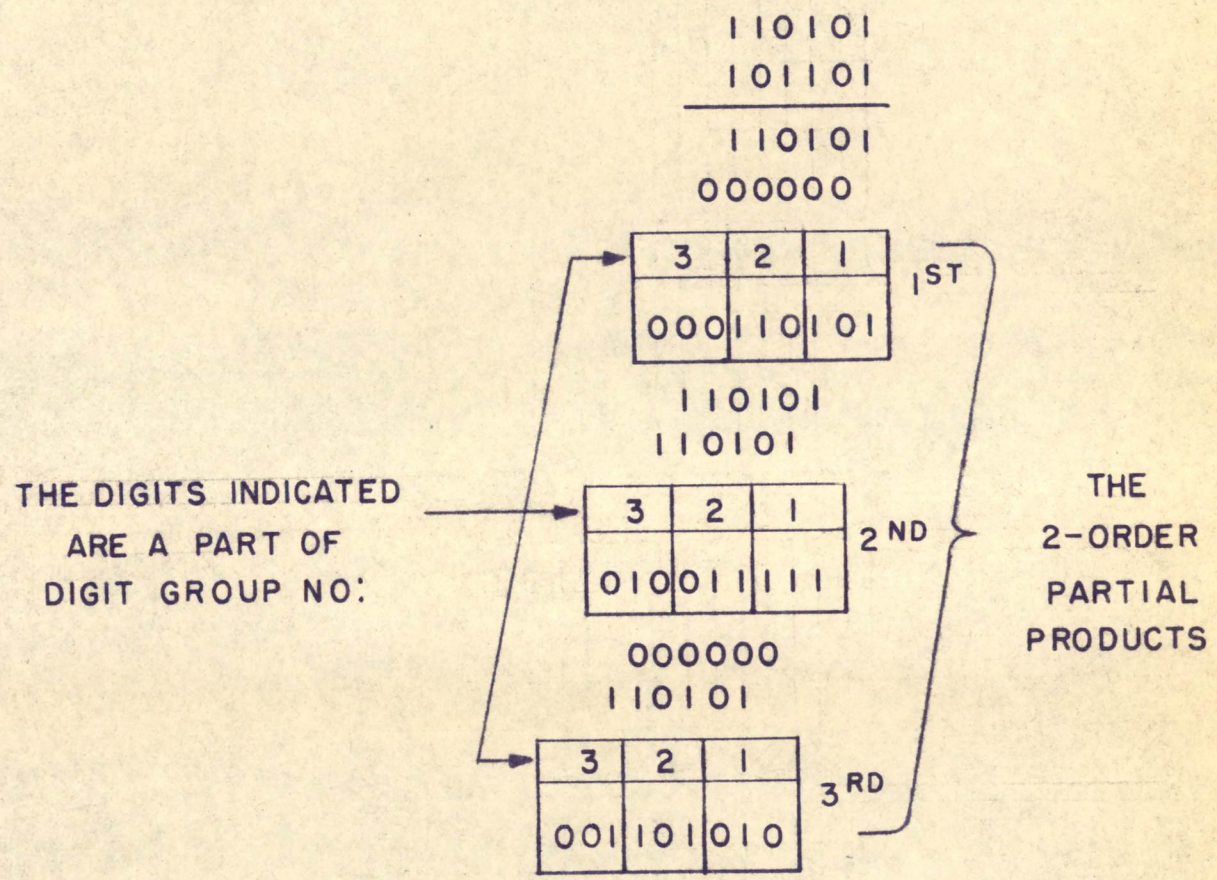


FIG. VI - 1

EXAMPLE OF DIGIT GROUPS FOR CLASS III MULTIPLIERS

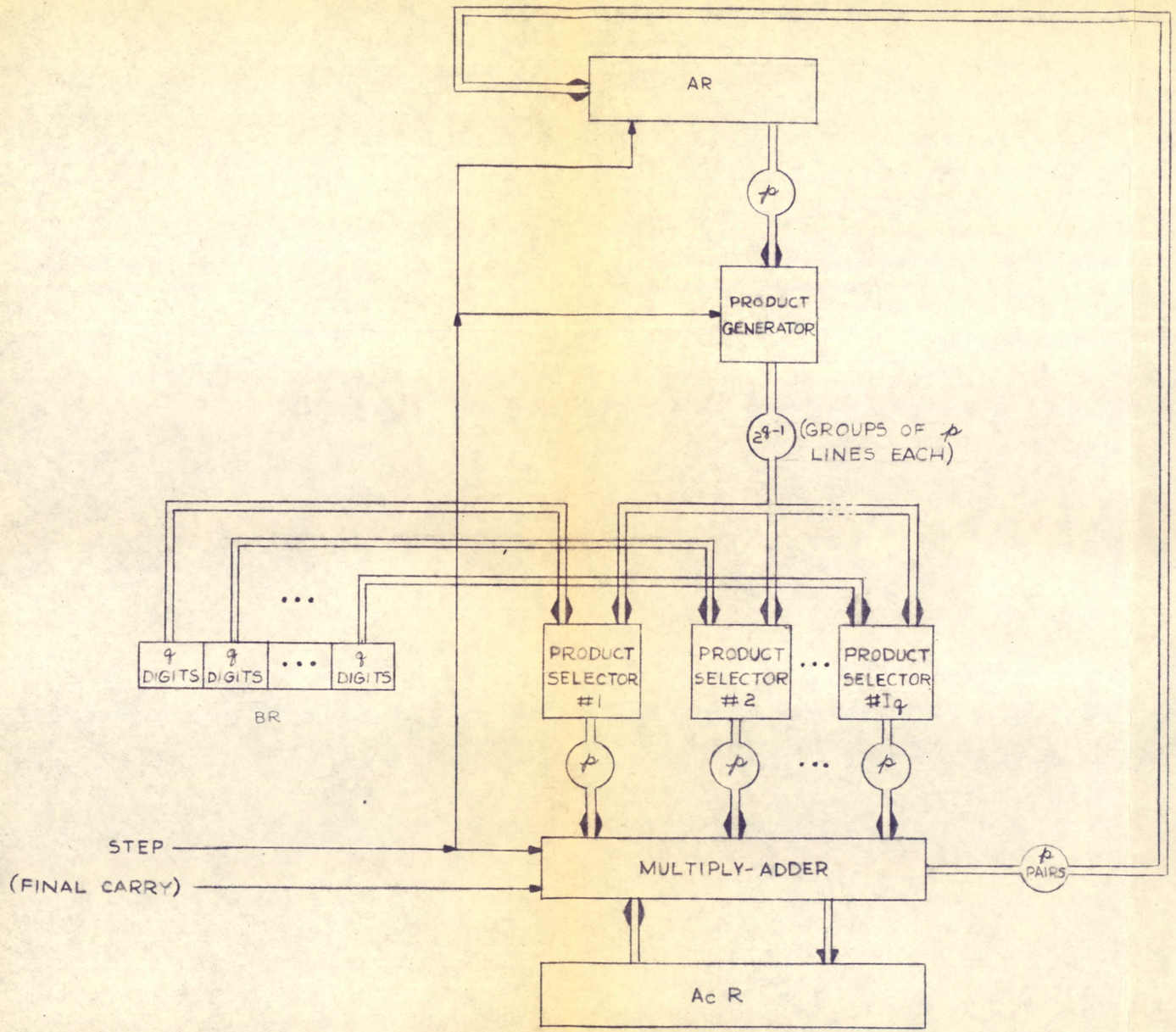


FIG. VI-2

ARITHMETIC SECTION FOR A CLASS III MULTIPLIER

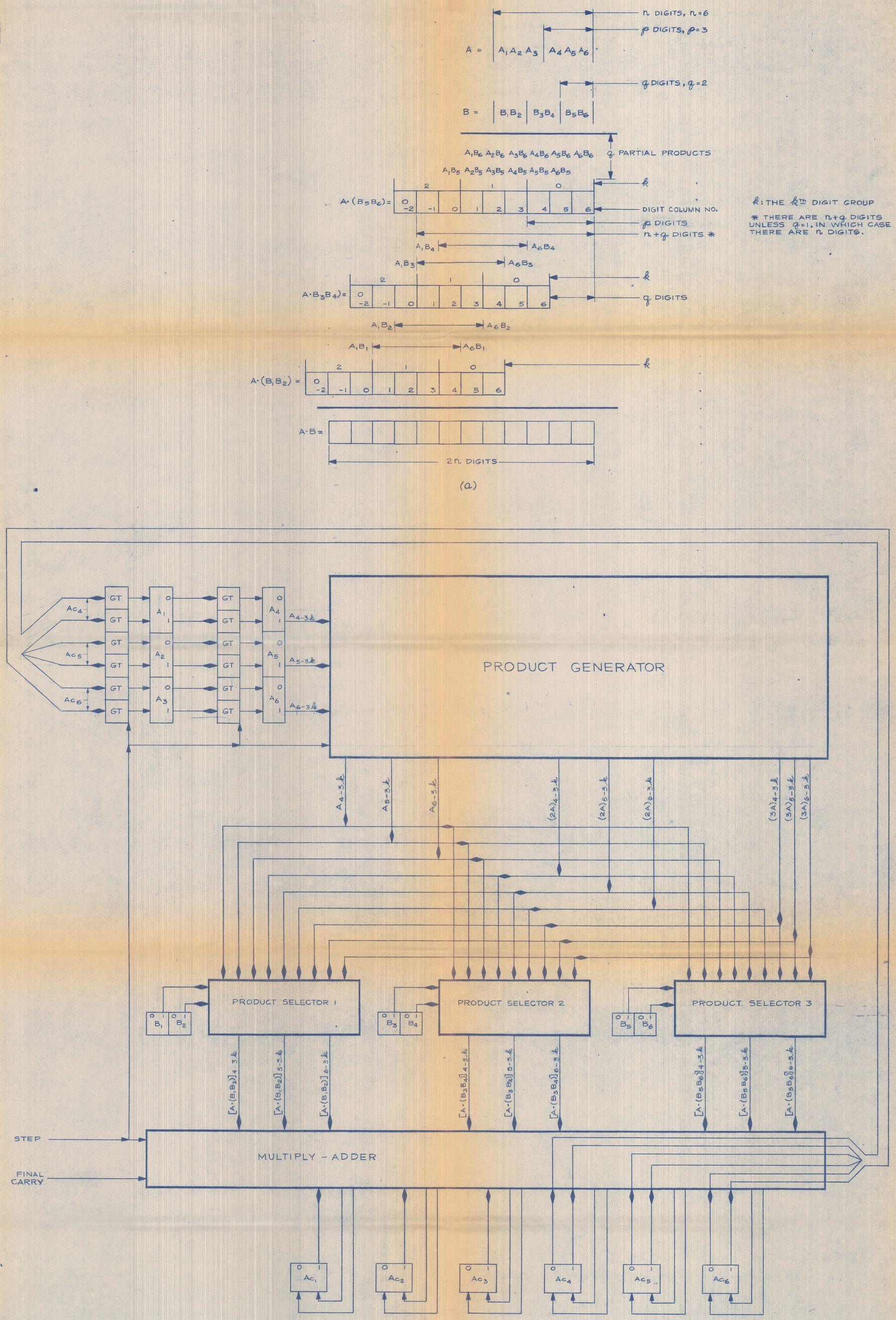


FIG. VI -3

FURTHER DETAIL OF CLASS III MULTIPLICATION

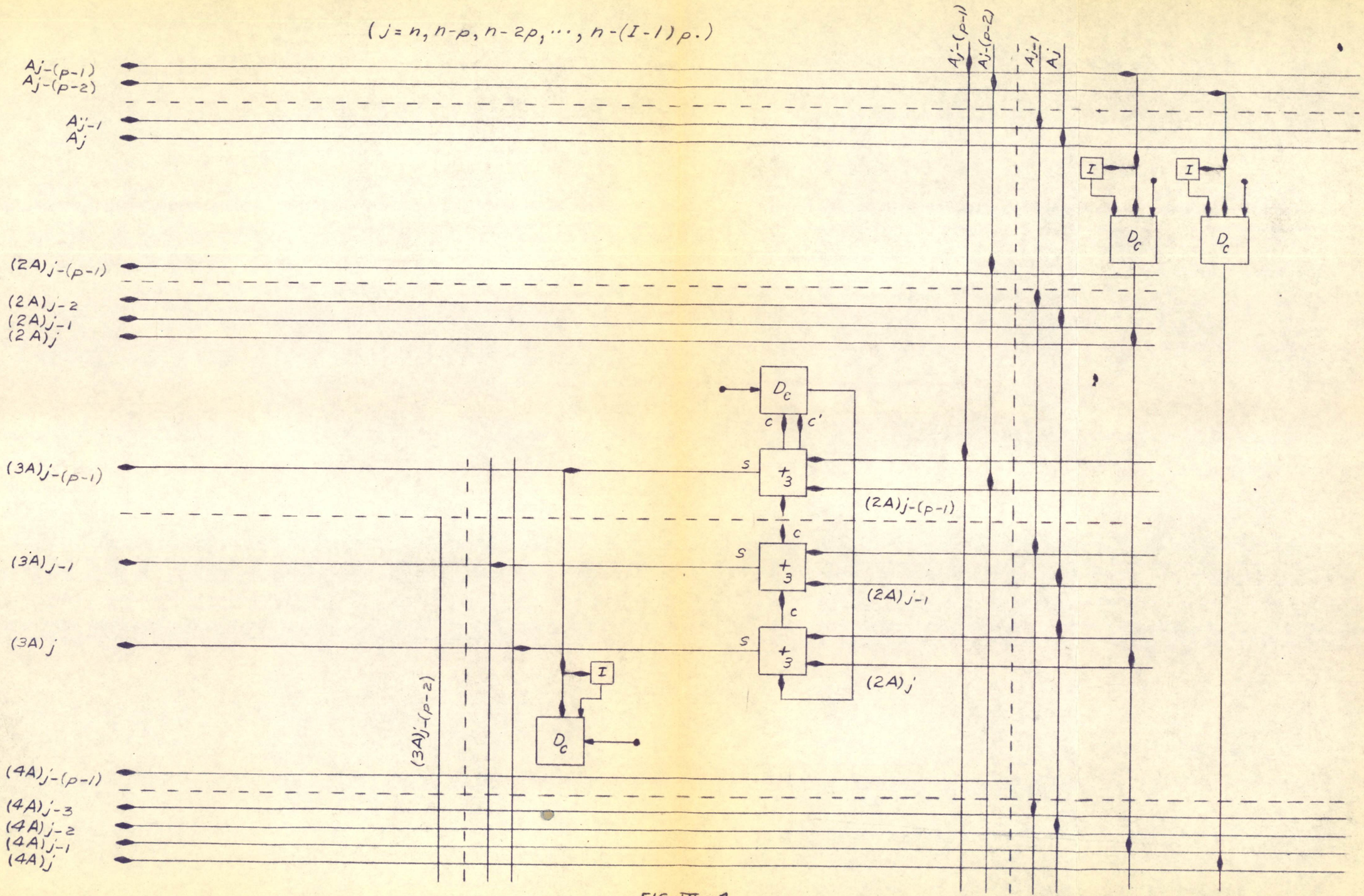


FIG. VI - 4

A PRODUCT GENERATOR FOR THE CLASS III MULTIPLIERS

NOTE: → REPRESENTS STEP PULSES FROM THE CONTROL SECTION

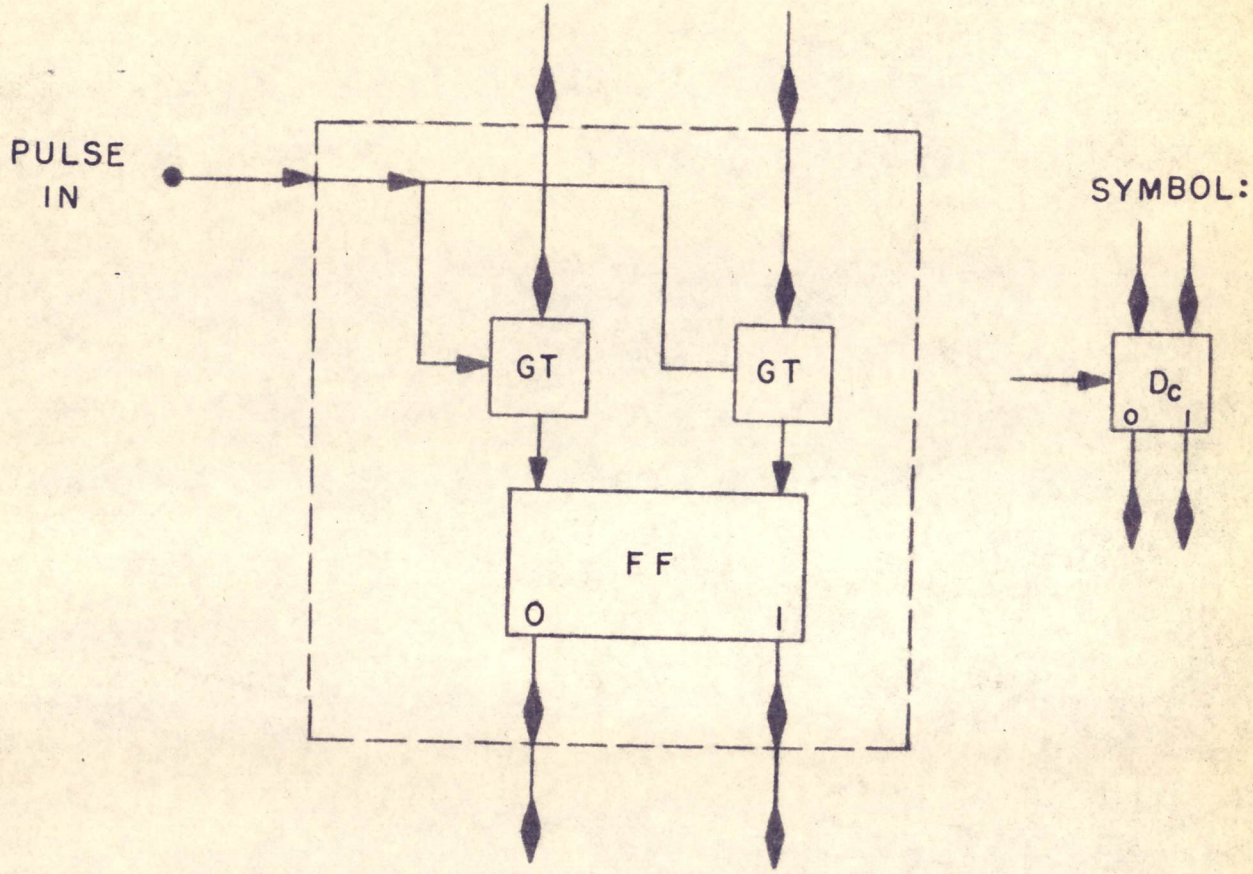


FIG. VI-5

THE CONTROLLED DELAY UNIT

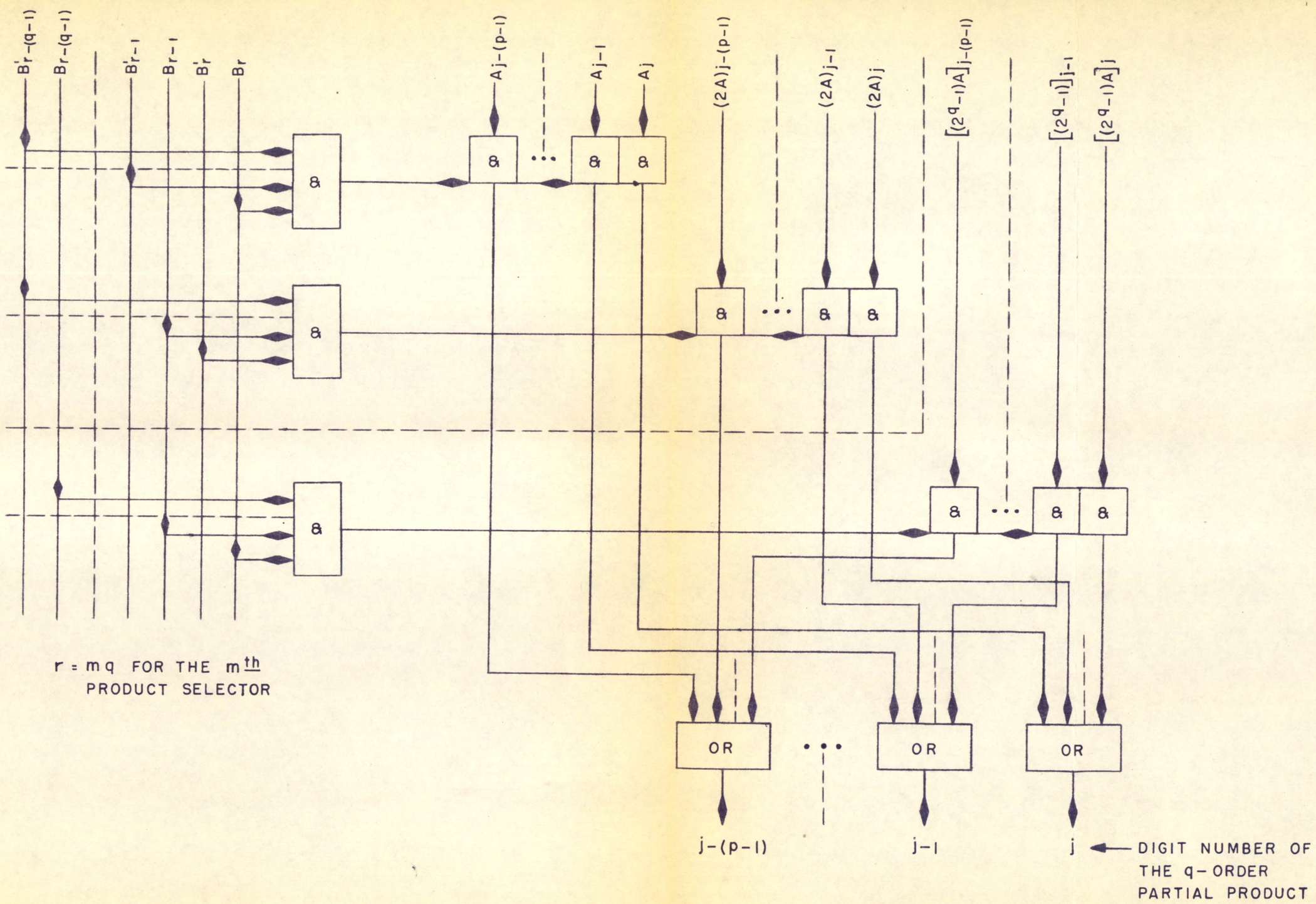
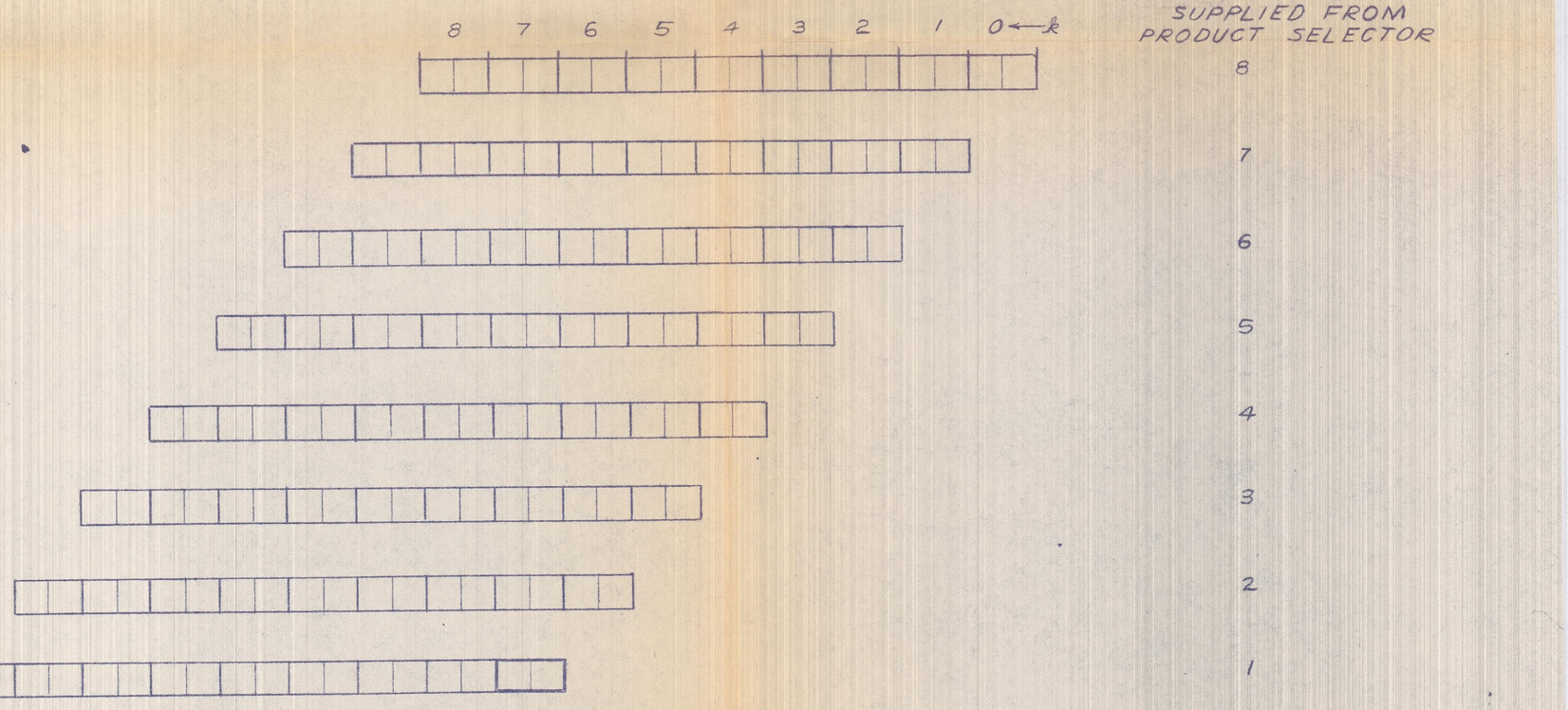


FIG. VI-6

A SELECTOR SWITCH FOR THE CLASS III MULTIPLIERS



(a)

THE 2-ORDER PARTIAL PRODUCTS WHEN n=6 AND p=2

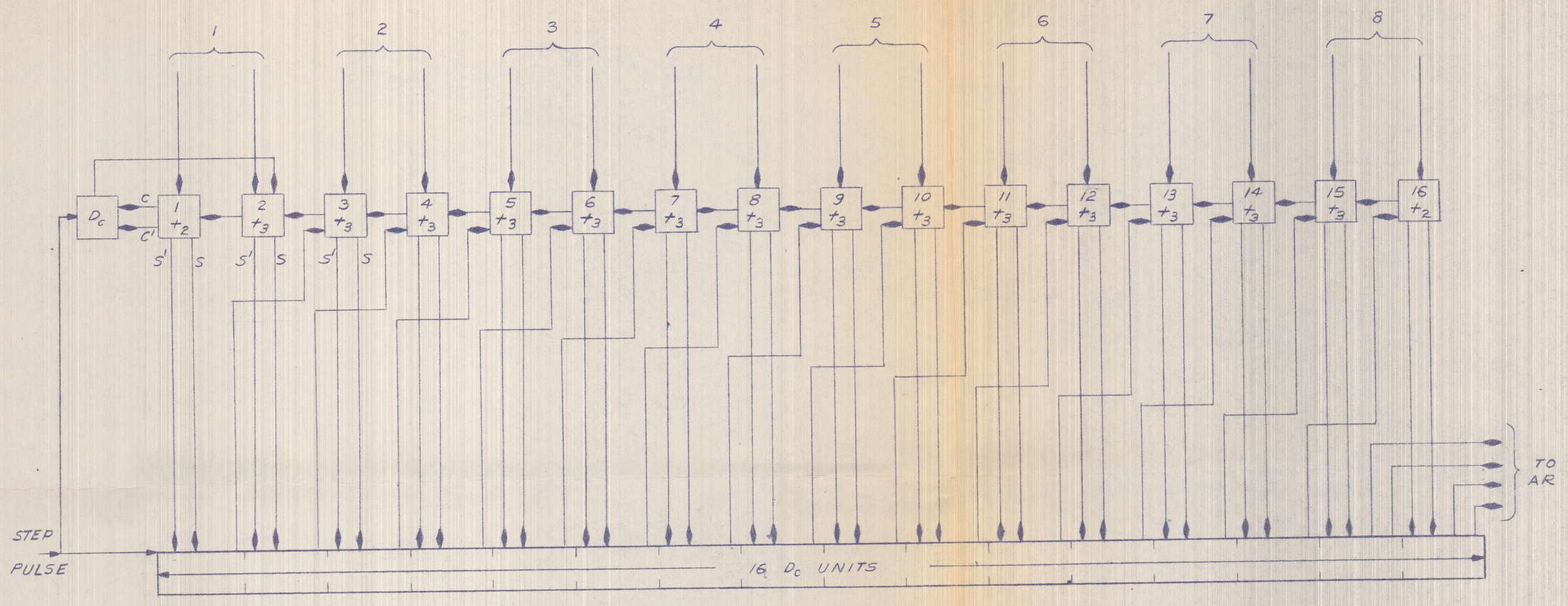


FIG. VI-7

(b)

THE acs B3, 2 x 2 MULTIPLY-ADDER WITH AcR (n=16)

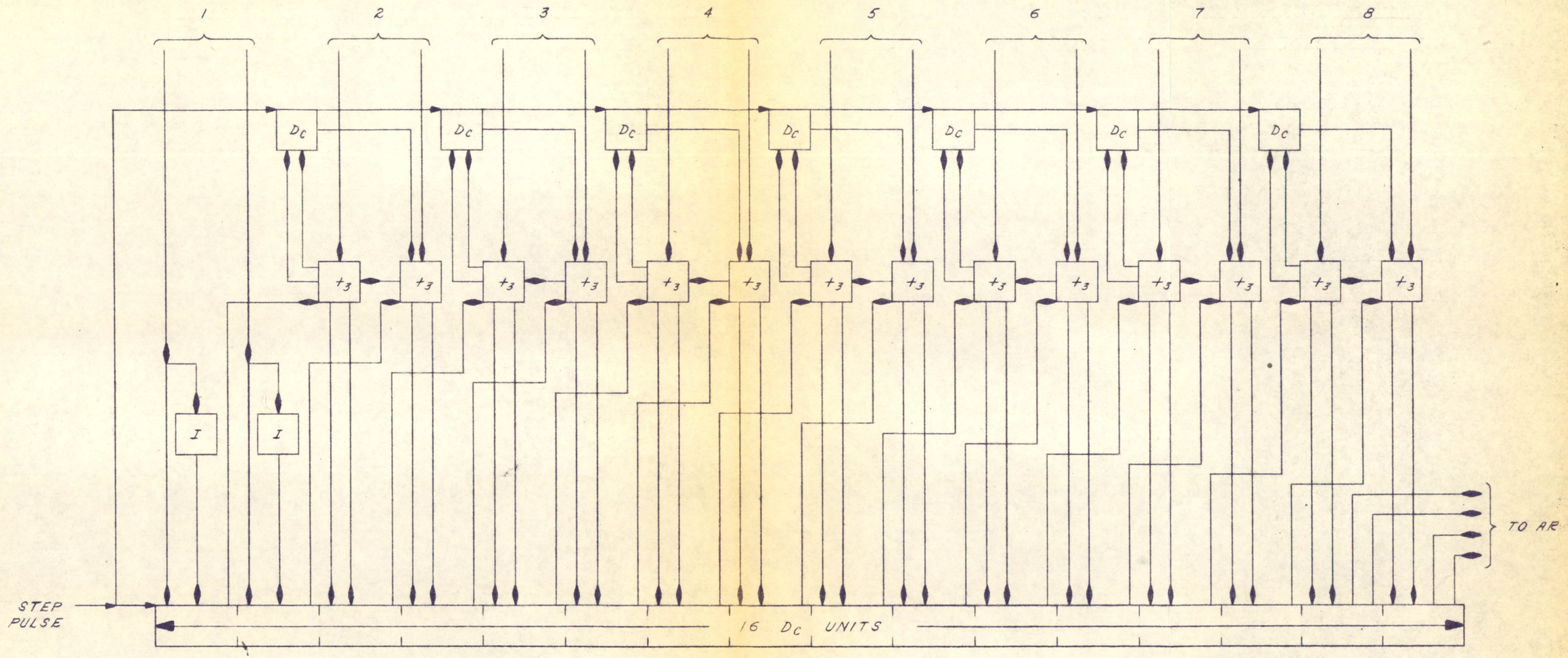


FIG. III-8

THE MAIN PART OF THE acs B4, 2 x 2 MULTIPLY-ADDER WITH AcR (n=16)

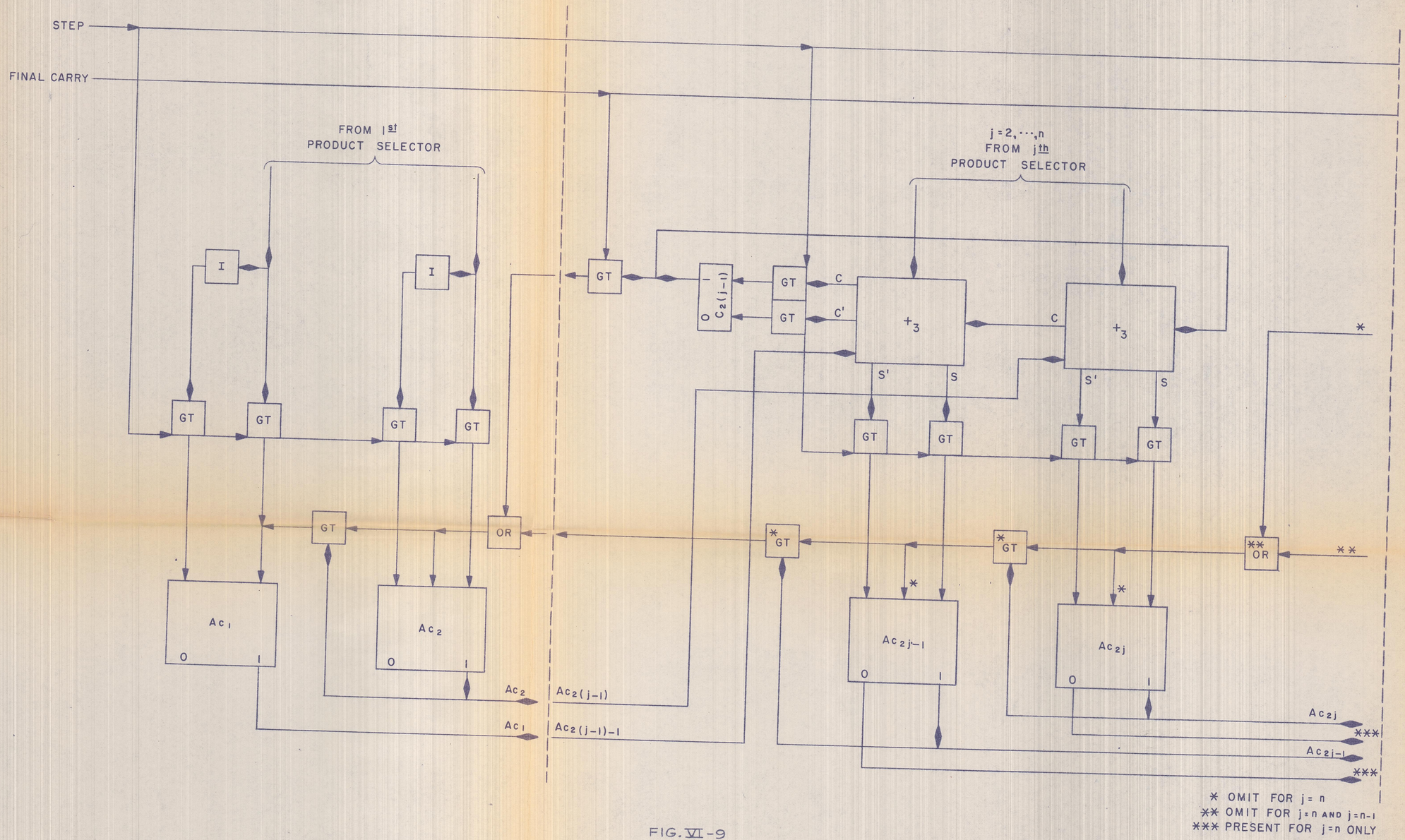


FIG. VI-9

THE acs B4,2x2 MULTIPLY-ADDER WITH AcR

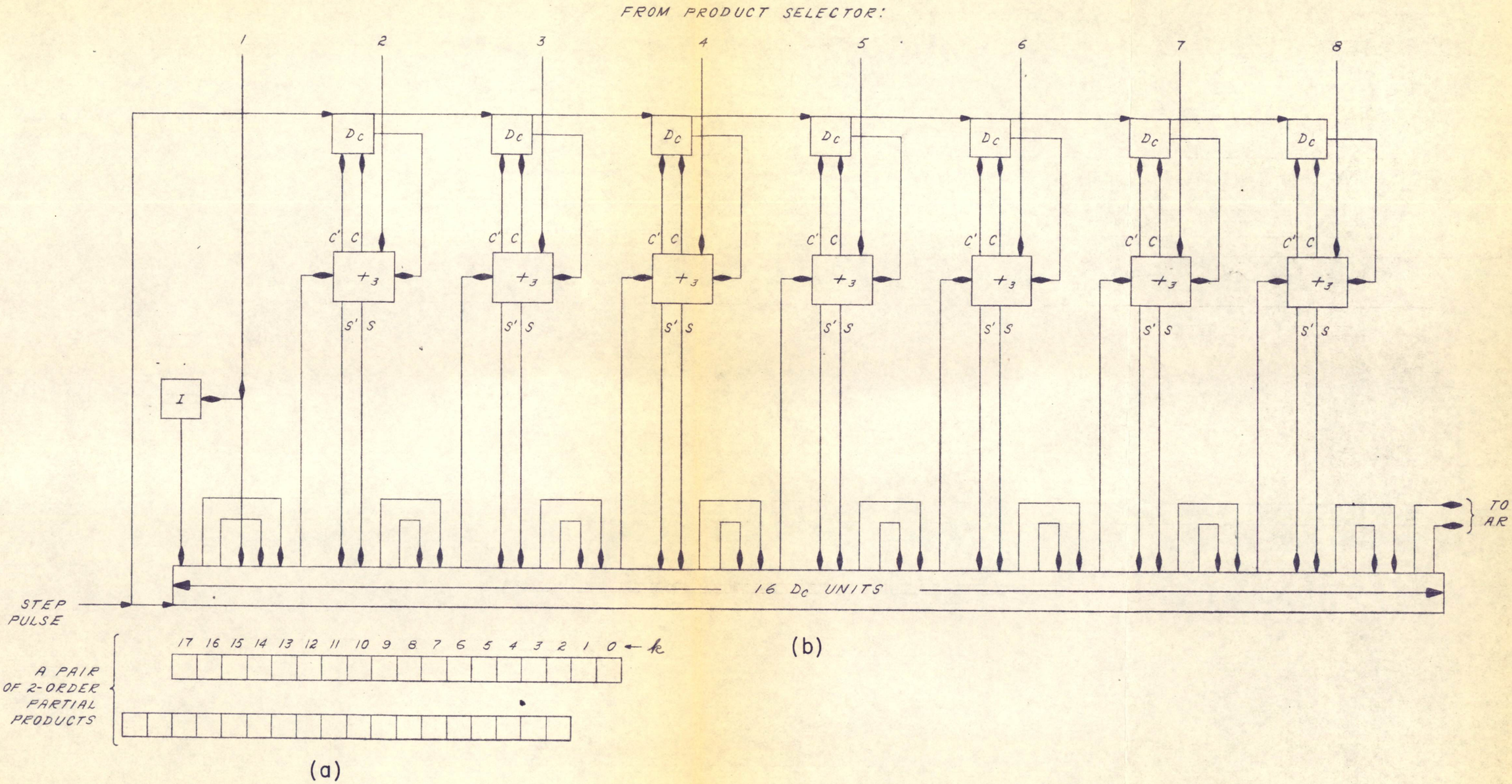


FIG. VI-10

THE MAIN PART OF THE acs B4, 1 x 2 MULTIPLY-ADDER WITH AcR (n=16)

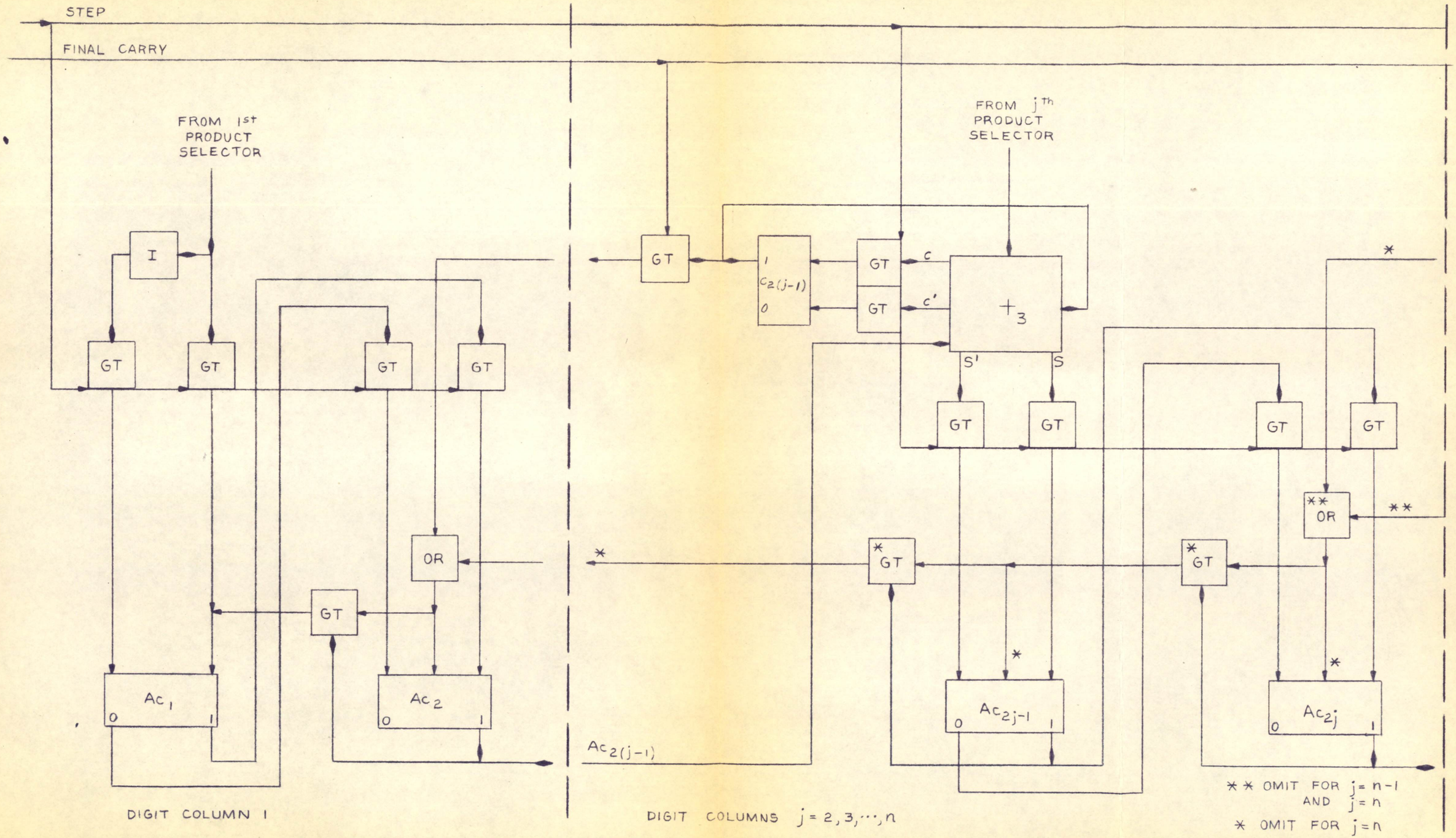
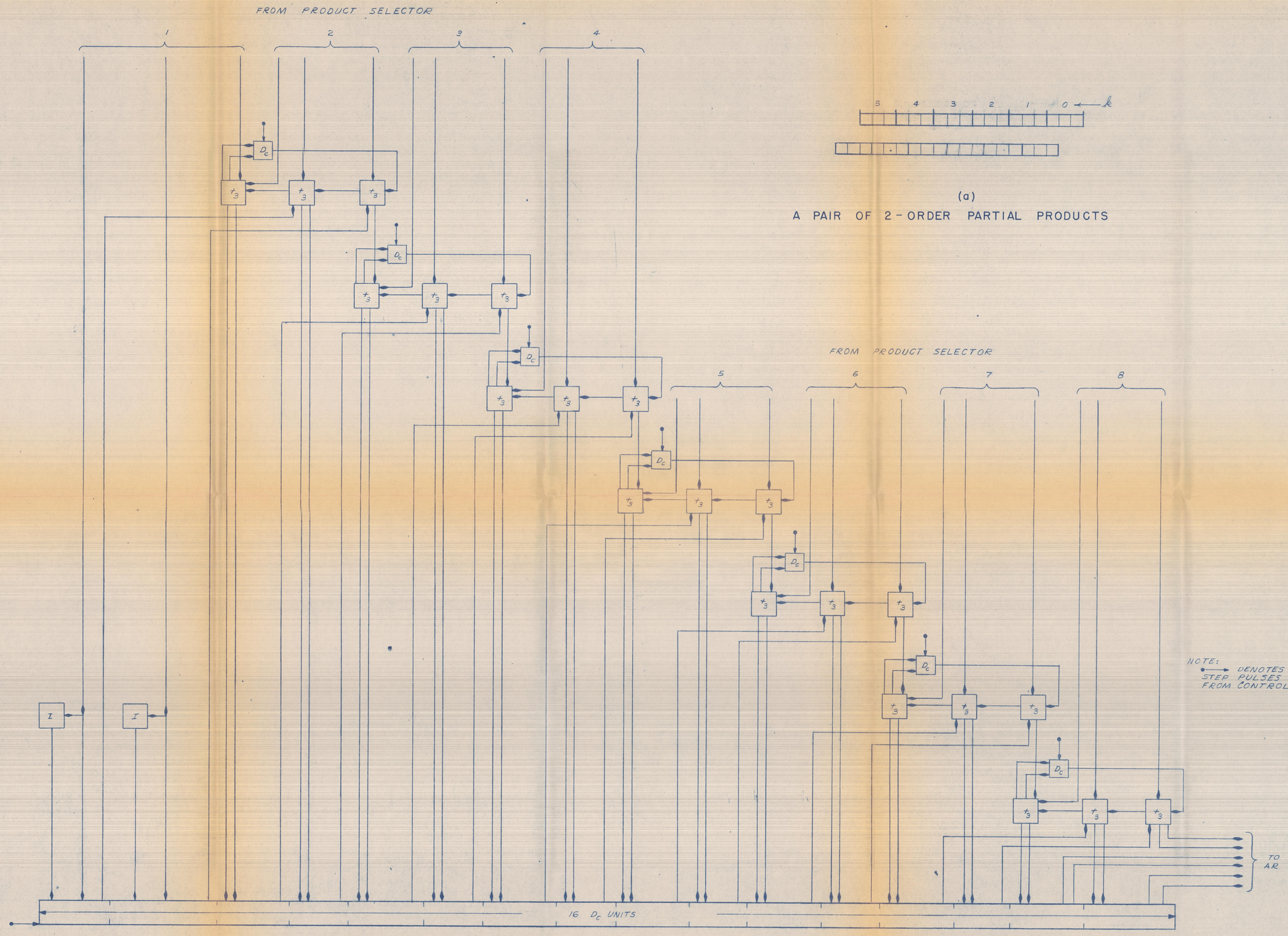


FIG. VI-11

THE acs B4, 1X2 MULTIPLY-ADDER WITH AcR



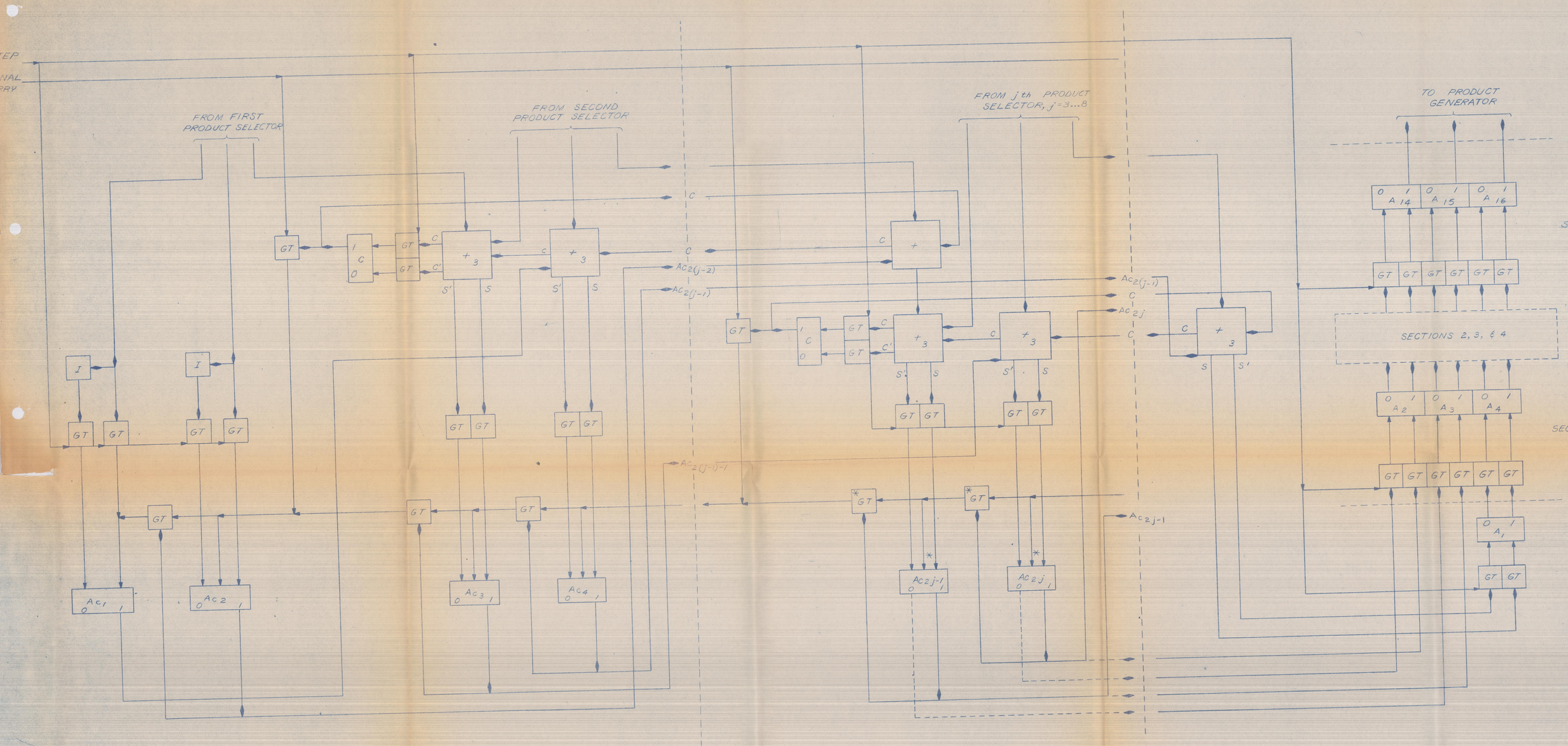
(a)

A PAIR OF 2-ORDER PARTIAL PRODUCTS

FIG. VI-12

(b)

THE MAIN PART OF THE acs B4, 3x2 MULTIPLY-ADDER WITH AcR (n=16)



NOTE:
- - - : THESE LINES ARE PRESENT WHEN j=8 ONLY
* ABSENT WHEN j = 8

FIG. II-13

THE acs B4, 3 x 2 MULTIPLY-ADDER WITH AcR AND AR
(n = 16)

Chapter VII

Comparison of the Multipliers

7.1 An optimum multiplier

The equipment counts and average multiplication times of the multipliers which have been discussed are presented in Table VII-1, and a comparison of the multipliers on the basis of these two measures is greatly facilitated by the scatter plot of Fig. VII-1 (the points of which are numbered to correspond with the entries in Table VII-1). The aid to a comparison furnished by the scatter plot is increased by circling the points corresponding to those multipliers for which $t_{max} = t_{min} = t_{ave}$. A dotted line is drawn on the figure at the estimated value of the maximum multiplication time which gives an instruction overshoot time of zero when a magnetic core memory is used, thus pointing up the optimum multiplier as defined in section 6.5. It is seen, the, that the optimum Class III multiplier is also the optimum multiplier of all those studied.

It is worthwhile to emphasize here the rather obvious fact that the multiplication time and the equipment count depend partly upon the time delays and the equipment weights for the circuits which realize the logical blocks. This study aimed at minimizing these two measures thru rearrangement of the logical blocks only.

It is interesting to consider other possibilities for the definition of how to specify an optimum multiplier. Clearly the definition of optimum for a multiplier could be formally broadened so that time is variable, and with each value of time a specific, minimal equipment count multiplier (of a closed class) would be associated as optimal. Then a specification of a time, significant for some reason, would automatically designate the associated optimal multiplier as the optimum. For instance for a value of t = 10 μsec., Fig. VII-1

Table VII-1

Summary of the Multipliers

Number	Multiplier	Equipment Count (n = 16)	Average Multiplication Time (n = 16) (μsec.)
1	C-a.cs B1	4772	17.3
2	C-a.cs B2	4578	17.1
3	K-acs B1	5110	15.2*
4	K-acs B2	4752	11.3*
5	C-a.c.s B	4172	24.8
6	C-as.c B	4522	20
7	C-ac.s B1	3868	18
8	C-ac.s B2	3724	23.7
9	K-ac.s D1	4082	18.5
10	C-ac.s D2	3856	18.8
11	K-ac.s D3	4180	18.1
12	C-asc B	4036	20
13	C-acs B1	3924	16.2
14	C-acs B2	4172	14.9
15	K-acs D	4347	9.8*
16	C-$\bar{c}$.a.s B	3684	22
17	K-ac.s D3, q = 2	5730	10.9*
18	K-acs B2, q = 2	7310	8.4*
19	K-acs B4, 2 x 1	5794	6.3*
20	K-acs B4, 3 x 2	5330	8.9*
21	K-acs B4, 1 x 2	4314	19.8*

* $t_{max} = t_{min}\ t_{ave}$ for these multipliers.

shows that multiplier 15, namely K-acs D (formerly Aas), would be an optimal multiplier. Or this scheme could be applied with equipment count as the variable if there was some maximum value of equipment count[1] to use in find-ind the optimum multiplier. If there is no significant value of either of the measures to fasten upon in defining an optimum, then perhaps it would be best to minimize time-equipment count product, time-equipment count sum (probably with weighting coefficients), or some other combination of the two measures. The combination to be used might be chosen, for instance, simply because it is the easiest to work with in finding an optimum multiplier from Class III as there seems to be no other criterion.

7.2 On a general background to the multipliers studied

The different schemes for the mechanization of multiplication are innumerable; this report attempted to present in organized fashion only a few of them. This organization can be visualized as shown in Fig. VII-2. The development of all the multipliers discussed here was based upon the fact that one way to find the product of two numbers is to sum the digits of the partial products. There are undoubtedly many other ways of finding products; for example, a halving and doubling procedure can be used.[2]

The particular multipliers which could be developed upon the above mentioned basis of summing the digits of the partial products of course were dependent upon the logical blocks available and also upon the method used for discovering how to interconnect these blocks to achieve certain logically defined aims. For instance the multipliers could have been designed using the method presented by D. A. Huffman in his MIT doctoral thesis "The Design of Sequential Switching Circuits". An unusual feature of this method is that

[1] Such a situation might result if the device were to occupy a limited space and if equipment count could at least roughly be related to space requirements.

logical blocks for storage, such as flip-flops or controlled delay units, are not necessary but can be derived from more elementary units.

7.3 A multiplier as the nucleus of an arithmetic element

The measures of multiplication time and equipment count are useful in comparing the multipliers only if the one finally chosen is to be used only to multiply. However, if it is to operate as part of a general purpose computer then, since equipment must be conserved, it is important that part of the equipment used to multiply also be used to add, subtract and divide. Varying amounts of additional equipment will be needed to convert the different multipliers introduced into such arithmetic elements, A count of this equipment plus the original equipment count, and the SAIT introduced in section 1.1 could be used for comparing the arithmetic elements resulting from the various multipliers.

Although beyond the scope of this report, a brief consideration of the steps which must be taken to convert the various multipliers into arithmetic elements and of the concomitant equipment needs is given in the next three sections. No consideration is given to the times required for the added operations however.

7.3.1 Adaptation of the multipliers for addition

If one of the numbers to be summed is placed in the accumulator and the other in the A register, then many of the Class I multipliers need only the addition of a control to supply the proper commands to form the sum. The Class I multipliers which combine a shift right with a command necessary to form the sum required, in addition to a control, gate tubes so that a shift left can be per-

[2] See Table 13.4 and the accompanying text on page 204 of Synthesis of Electronic Computing and Control Circuits by the staff of the Harvard Computation Laboratory, Harvard University Press, 1951.

formed as the last command of the summation sequence. Since a separate shift left command is often required in an arithmetic element these gate tubes may not be "excess" equipment, i.e., equipment used only for summations. The foregoing remarks apply also th the Class II multipliers with the one alteration that the shift left must be for q digits instead of one digit. In addition, the rightmost q digit columns of the B register must be initially set to 0 ···01.

If for the Class III multipliers for which $p \geq q$ the rightmost p digits of the A register are set to 0···01: then one of the numbers to be summed can be placed in the B register., and the digits of this number will appear on appropriate lines from the product selectors. However, if $p < q$ considerable additional equipment in the form of flip-flops, three input adders, etc. will ne needed, since one of the numbers to be summed can no longer be placed in the B register. This is so because the product selectors no longer provide a sufficient number of lines to furnish the entire contents of B to the (insufficient number of) adders. The extra flip-flops will be used to fill in the "gaps" in the carry register; the number to be summed can then be stored in this register. The extra adders will be used to make up a "complete string" with those already present. Level "and" gates and "or" gates will also be necessary in order to allow the effective interchange of connections which will be necessary within the device whenever it is to be converted to an adder. Furthermore, either the A register or the B register must be cleared initially.

For either of the above cases of the relative values of p and q the other number to be summed can be placed in the accumulator register and then shifted p digits to the left.

(An extra set of shift gates is therefore required as well as an extension of the accumulator to the left by a few flip-flops and the addition of a few three input adders.) The sum will then be formed by pulsing the step line.

7.3.2 Sign handling and adaptation for subtraction

No mention has been made of designing the multipliers to handle negative numbers; but since the magnitude of the product of two numbers is independent of their signs, any multiplier that will handle positive numbers can be converted to handle both positive and negative numbers by the addition of: (1) flip-flops for storage of the signs, and (2) a very simple control circuit using the outputs of the sign storage flip-flops to determine the sign of the product.

A somewhat more complex control must be added if the multiplier is generalized to a device able to sum both positive and negative numbers by making use of 1's (or "9's") complements.[3] Such a device can be made to subtract simply by inverting the one's and zero's and the sign of the number to be subtracted and then proceeding as in additions.

7.3.3 Adaptation for division and other operations

Once the arithmetic element is able to subtract, the main alteration necessary in order that it be able to divide is the addition of a divide control unit. The ability to perform other operations may also be required of the arithmetic element such as the shift left already mentioned above and the "mask" or "sum-modulo-two" operation (a partial add without carry storage).

---

[3] See Digital Computer Laboratory Report R-127, Vol. I, section 4.12.

A conclusion can be drawn from sections 7.3.1 thru 7.3.3 that all the multipliers with the exception of those of Class III for which $p < q$ are roughly comparable from the standpoint of the equipment to be added when used as the nucleus of an arithmetic element.

7.4 Logical complexity and its effect upon serviceability

It has been suggested that the realizations of the Class III multipliers would be more difficult to service than those of the Class I and Class II multipliers because the former are logically more complex than the latter. The authors feel that the logical complexity of the Class III multipliers is not sufficiently greater than that of the other two classes to cause more than a negligible increase in the difficulty of servicing provided the groups of three input adders and controlled delay units in the multiply-adder of such a multiplier are treated as units, e.g., each group or each member of each group could constitute a single plug-in unit.