"In presenting the dissertation as a partial fulfillment of the requirements for an advanced degree from the Georgia Institute of Technology, I agree that the Library of the Institution shall make it available for inspection and circulation in accordance with its regulations governing materials of this type. I agree that permission to copy from, or to publish from, this dissertation may be granted by the professor under whose direction it was written, or, in his absence, by the dean of the Graduate Division when such copying or publication is solely for scholarly purposes and does not involve potential financial gain. It is understood that any copying from, or publication of, this dissertation which involves potential financial gain will not be allowed without written permission.

Anthony M. Bevan"

A METHOD FOR SENSING THE COMPLETION OF OPERATIONS IN SPEED-INDEPENDENT ASYNCHRONOUS COMPUTER CIRCUITS

A THESIS

Presented to

the Faculty of the Graduate Division

by

Aubrey Marvin Bush

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Electrical Engineering

Georgia Institute of Technology

June 1961

Copy
3 R
12 R

A METHOD FOR SENSING THE COMPLETION OF OPERATIONS IN SPEED-INDEPENDENT ASYNCHRONOUS COMPUTER CIRCUITS

Approved:

Thesis Advisor

Date Approved by Chairman: May 25, 1961

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to Dr. W. B. Jones for his suggestion of the field of study and constant encouragement and guidance during the work, to Mr. S. P. Lenoir for his help in obtaining much of the literature, to Dr. D. C. Fielder and Dr. B. M. Drucker for reading the thesis, and to the Westinghouse Electric Corporation whose fellowship aid made the work possible.

TABLE OF CONTENTS

Page

ACKNOWLEDGEMENTS ii

LIST OF ILLUSTRATIONS v

SUMMARY vi

CHAPTER

I. INTRODUCTION 1

Computer Design Philosophies
Completion Signals in Computers
Special Requirements in Speed-Independent Circuits
The Complementary Function Method

II. THE INVERSE FUNCTION METHOD 9

The Method
Useful Features
Disadvantages

III. A ONE-BIT SUBTRACTOR 15

Design Objectives
System Design
Logical Design

IV. CIRCUIT DESIGN FOR THE ONE-BIT SUBTRACTOR 27

Objectives
Basic Circuits
Input and Output
Operation

V. AN N-BIT SUBTRACTOR 37

VI. RELATION TO GENERAL SEQUENTIAL CIRCUITS 44

VII. CONCLUSIONS AND RECOMMENDATIONS 50

Applications of the Inverse Function Method
Limitations of the Inverse Function Method
Suggestions for Further Study

Page

APPENDICES 54

I. LOGICAL DESIGN OF COMBINATIONAL CIRCUITS FOR THE ONE-BIT SUBTRACTOR 55

II. CURVES USED IN THE CIRCUIT DESIGN OF THE ONE-BIT SUBTRACTOR 59

BIBLIOGRAPHY 67

LIST OF ILLUSTRATIONS

Figure		Page
1.	Block Diagram of a General Operation	4
2.	The Complementary Function Method	8
3.	The Inverse Function Method	10
4.	The Inverse Function Method Utilizing the Feedback Memory Loop	12
5.	Block Diagram of the One-Bit Subtractor	19
6.	Primitive Flow Table for the Selector Control	20
7.	Summary of the Logical Design for the One-Bit Subtractor	23
8.	The Selector without the Checking Circuits	24
9.	Logical Diagram of the One-Bit Subtractor	26
10.	Schematic Diagram of the Single-Level Gate	29
11.	Schematic Diagram of the Double-Level Gate	30
12.	Ideal Base Current Waveform	32
13.	The One-Bit Subtractor Circuit	34
14.	The Auxiliary Amplifier and Output Lamp	35
15.	Subtraction with Occasional Borrow Assimilation	38
16.	Block Diagram of the N-Bit Subtractor	39
17.	General Form for Sequential Circuits	45
18.	One-Bit Subtractor Arranged to Fit the General Form . . .	47

SUMMARY

In asynchronous computer circuits, each operation is allowed to proceed at a rate determined only by the time constants of the circuits involved in that operation. Each operation must provide some signal to indicate to succeeding operations which will depend on its results that it has been satisfactorily completed. The design of asynchronous computer circuits has taken two approaches. In the more common approach, a knowledge of the switching speeds of all elements is assumed, and the necessary interlocking of operations is obtained via this knowledge. In the other approach, the interlocking of operations is done in such a way that the end result is the same regardless of the switching speeds of the circuit elements involved. The latter type circuit is said to be speed-independent. In this type circuit the completion signal for each operation must be generated in such a way that it is not dependent on the switching speeds of the circuit elements involved.

A method for sensing the completion of operations in speed-independent asynchronous circuits is developed in this thesis. It is original in this thesis and is referred to as the "inverse function method." In this method, each operation is paralleled by an operation which is its inverse. If the outputs of the primary operations are fed to the inputs of the inverse operation, then agreement of the outputs of the inverse operation and the inputs of the primary operation will indicate satisfactory completion of the primary operation. Thus if a completion signal generator is provided to indicate 1 when agreement is obtained and an

input completion signal of 1 is present, indicating that the inputs to the primary operation are the desired inputs, and to indicate 0 otherwise, the completion signal generator will give an output completion signal for the operation which may be used to enable the completion of subsequent operations.

The inverse function method offers several very useful features. If the outputs of the inverse operation are taken as inputs to the primary operation following satisfactory completion of the primary operation, a closed feedback loop providing memory of the input and output variables is obtained. This feature permits the use of the inverse function for something other than completion sensing, helping to justify the addition of the circuit elements required by this operation. The inverse function method permits the logic circuits to consider the signals at their inputs before an input completion signal is presented. The primary and inverse operation may thus have already begun, or perhaps even completed, before the input completion signal appears. This feature should make possible very high operating speeds.

To illustrate the inverse function method and demonstrate its feasibility, a one-bit subtractor utilizing the feedback memory loop was designed and constructed. Inexpensive components were used, and very high speeds were not attempted, as demonstration of the method of completion sensing was the principal objective rather than demonstration of high speed circuits.

To illustrate the use of speed-independent circuits using the inverse function method as logical building blocks, an n-bit subtractor was designed around the one-bit subtractor to perform parallel subtraction

with occasional borrow assimilation. The design of the n-bit subtractor was not carried beyond the system design.

The general form for sequential switching circuits given by Huffman (2) is applicable to speed-independent sequential circuits. By means of this general form a comparison of speed-independent asynchronous circuits, asynchronous circuits which are not speed-independent circuits, and synchronous circuits is effected.

The inverse function method does not solve any of the general problems encountered in attempting to design speed-independent sequential circuits, but it does represent a new and better means of sensing the completion of operations. Some suggestions for further study toward solution of general design problems are made with reference to the general form of Huffman as used by Unger (3).

CHAPTER I

INTRODUCTION

Computer design philosophies.--The organization of the functions performed by electronic digital computers so that each of the many operations, such as addition, subtraction, or specialized logical decision functions, takes place in a precise time sequence producing the desired results can be carried out in two broad design philosophies. These philosophies are synchronous design and asynchronous design.

In the synchronous design philosophy all operations take place under the command and control of a central master clock. This clock is an oscillator which generates a regularly recurring signal, dividing all time into discrete clock time intervals. An operation is begun at the signal of the clock, through a control circuit, and is allowed some fixed number of clock time intervals for completion; at the end of this number of clock time intervals the operation is assumed to be complete, and subsequent operations based on the results are begun. Signals throughout this type computer will be meaningful only at specific instants of time, as determined by the central clock oscillator.

In the asynchronous design philoscphy each operation is allowed to proceed at a rate determined only by the time constants of the circuits involved in that operation. All operations are interlocked in such a way that no operation which will depend on the results of some previous operation is allowed to complete until all of its prerequisite operations are complete. Each operation must provide some signal to

indicate to the succeeding operations that it has been satisfactorily completed. Given signals in this type computer can be meaningful only when the associated completion signal indicates that the operation which generated these signals has been satisfactorily completed.

Within these two broad design philosophies there may exist many innovations and combinations. For example, in a synchronous computer a number of different clocks may time operations in different sections of the computer, with all these clocks in turn synchronized with one master clock. Again, one section of an asynchronous computer may have its own isolated clock; then that particular section of the computer is within itself synchronous, resulting in a hybrid combination of the two design philosophies. It is also possible that one section of a primarily synchronous computer might operate asynchronously within itself, to give another form of hybrid combination.

Synthesis methods for the design of computers following the synchronous mode of operation have been systematized to a high degree, and the design of this type computer can be carried out readily, after decisions of an a priori nature as to design objectives and circuit details have been made (1). However, no comparable degree of systematization in general design methods has been achieved for asynchronous computer circuits.

Two approaches have been taken in the design of asynchronous computer circuits. In one approach one assumes a knowledge of the operating times of all elements in the circuit and interlocks the operations in such a way that the necessary inputs for an operation are generated as efficiently as possible and made to appear at the appropriate time, while

in the other approach one attempts to interlock the operations in such a way that the results will be the same regardless of the operating speeds of the elements involved.

The former approach is the more common one; circuits of this type permit easier analysis than those of the latter type. A number of general theorems on the properties of asynchronous circuits of the former type, as well as design methods for restricted classes of circuits of this type, have been published by Huffman (2) and Unger (3).

A theoretical study of the latter type of asynchronous circuits has been made by Muller and Bartky, who have given this class of asynchronous circuits the name "speed-independent circuits," since their operation is not dependent on the speed of operation of the various circuit elements involved (4),(5). Some design methods for this class of circuits have been developed and are summarized in a report published by the University of Illinois (6).

Completion signals in computers.--All approaches to the design of asynchronous computers require the generation of completion signals to indicate that an operation has been completed satisfactorily, and hence that the subsequent operations depending on the results may be allowed to proceed. Consider the box in Figure 1 to represent some logical operation; the outputs of the box are then a logical function of the inputs. Some means must be provided to indicate, first, that the proper inputs have been presented and, second, that the logical function has acted on these inputs to produce the corresponding outputs. This information must be obtained regardless of the limitations placed on the manner in which the inputs and outputs may change.

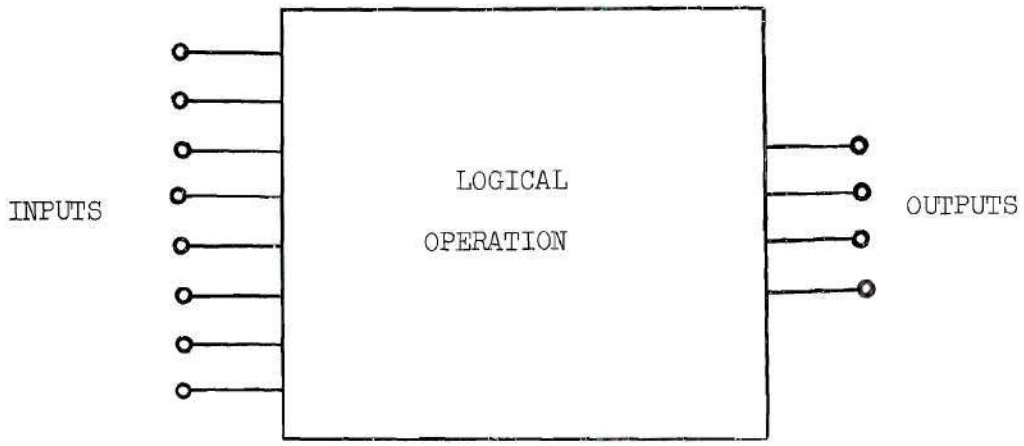


Figure 1. Block Diagram for a General Operation.

The presence of the proper inputs can be ascertained by requiring completion signals of all operations generating these inputs. The completion of the new operation is then indicated by deriving a new completion signal. In synchronous computer circuits the completion of operations is announced by merely waiting a prescribed number of clock time intervals after the inputs are all presented; the clock in this type circuit furnishes all completion signals. For circuits operating in the asynchronous mode, satisfactory completion can be indicated in a number of ways. For a design based on a knowledge of the switching times of the circuit elements involved, a completion signal may be generated by initiating a change on the input to a delay line (or other device having a specific delay between input and output) simultaneously with or after the initiation of the operation in question and considering the operation to be complete when this change appears at the output of the delay line. If the delay time is longer than the maximum possible time required for the actual completion of the operation, then the output of the delay unit will indeed signal satisfactory completion of the operation. This method follows readily from the use of a clock in the synchronous mode and shares most of the disadvantages of the synchronous mode of operation. More sophisticated means can be used, with a corresponding increase in circuit complexity.

Special requirements in speed-independent circuits.--The problem of sensing the completion of an operation and generating a completion signal is not a simple matter in speed-independent circuits. For this class of circuits, the completion of an operation must actually be sensed in some way; that is, merely waiting some prescribed length of time cannot insure that the

operation has occurred, for the switching times of the circuit elements involved must have no effect on the results of the operation. Inevitably, the sensing of the completion of operations in this class of circuits will require the use of many additional circuit elements and a substantial increase in overall circuit complexity.

It should be noted that the term "speed-independent," referring to Figure 1, implies that nothing which happens within the box, that is, in the circuits implementing the logical function and generation of a completion signal for the function, can be dependent on the switching speeds of the individual circuit elements. Thus extra delays could be inserted anywhere within these circuits without causing a malfunction in their operation. However, between such operations or logical functions, on the input or output signal lines, no added delay can be tolerated. To illustrate, suppose an extra delay were permitted in the transmission of the outputs of the previous operations, but not in the transmission of their completion signals. Then obviously a malfunction could be caused, since the proper inputs to this operation would not be present when the completion signal appeared. We restrict the term "speed-independent" to apply only to circuits used to implement a logical function and not to the lines which interconnect operations. The sensing method and generation of the completion signal must be independent of the switching speeds of all circuit elements, both those in the primary circuit and those necessary to indicate completion of the operation of the primary circuit, if the overall circuit is to be speed-independent.

The complementary function method.--One means of sensing the completion of an operation and generating a completion signal in such a way that the

overall circuit operates in a speed-independent manner has been developed by Shelly and is illustrated in block diagram form in Figure 2 (7). In this approach the logical function is paralleled by an entirely dual or complementary function; that is, the logical function is realized in two channels instead of one, with one channel requiring inputs which are the complements of the inputs of the primary channel and delivering outputs which are the complements of the outputs of the primary channel. Before the operation is initiated, the inputs to both channels are set to cause the outputs of both channels to agree. When the inputs for the operation appear, the logical function and its complementary function are allowed to operate independently, limited only by the natural time constants of the circuits involved. When the outputs of the channels change, becoming complementary instead of agreeing, the outputs of the primary channel are taken as the desired output. The complements of these outputs are available at the output of the complementary channel. Satisfactory completion of the operations generating these outputs is thus sensed when the outputs of both channels have changed to become complementary. A completion signal can be generated by comparing the outputs of the two channels, obtaining an "incomplete" signal when the outputs of the two channels agree and a "complete" signal when the outputs are complementary. The circuits used in each channel must be hazard-free circuits; the outputs must not contain momentary false outputs, but must change directly from stable agreement to stable complementation (8). Between every set of inputs, resetting signals must be applied and an "incomplete" signal obtained from the completion signal generator.

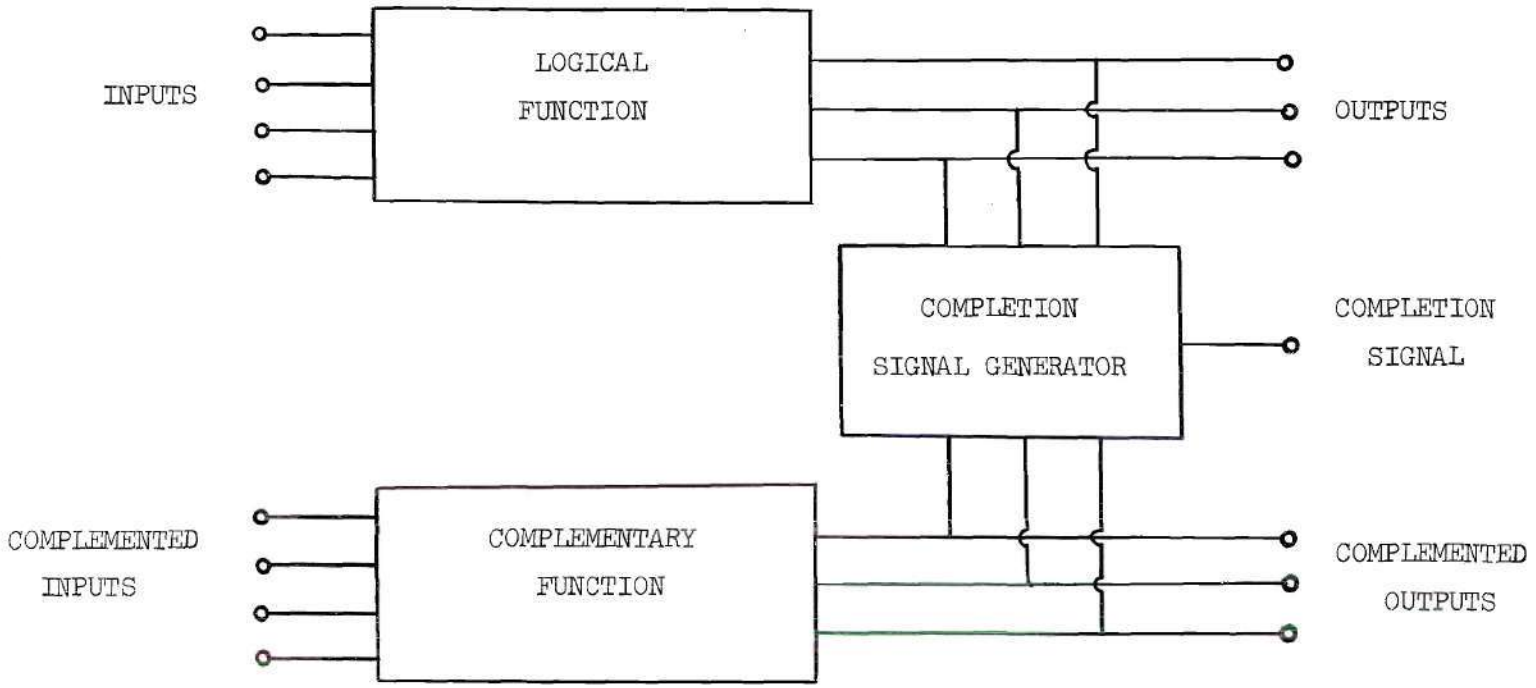


Figure 2. The Complementary Function Method.

CHAPTER II

THE INVERSE FUNCTION METHOD

The method.--A method for sensing the completion of operations and generating a completion signal in a manner which is applicable to operations having any number of inputs and any number of outputs and which will permit the overall circuit to operate speed-independently is presented here. This method is called the "inverse function method" for the purpose of this discussion. This method is original in this thesis. It has several advantages over the complementary function method presented by Shelly. It has several disadvantages, all of which are shared by the complementary function method. Presentation of no other methods have been found in the literature.

The inverse function method is shown in block diagram form in Figure 3. Even though four inputs and three outputs are shown for convenience, any number might be present. The input completion signal indicates to the completion signal generator that all of the inputs are the desired inputs. The output completion signal indicates, to the succeeding operations, that the outputs are those which should correspond to the inputs. When the associated completion signals do not indicate "complete," the inputs or outputs cannot be considered meaningful.

The outputs of the logical function are taken as the inputs to the inverse logical function. The inverse logical function should then present at its output signals which are the same as the original input signals. With Boolean functions, however, uniqueness is the exception rather

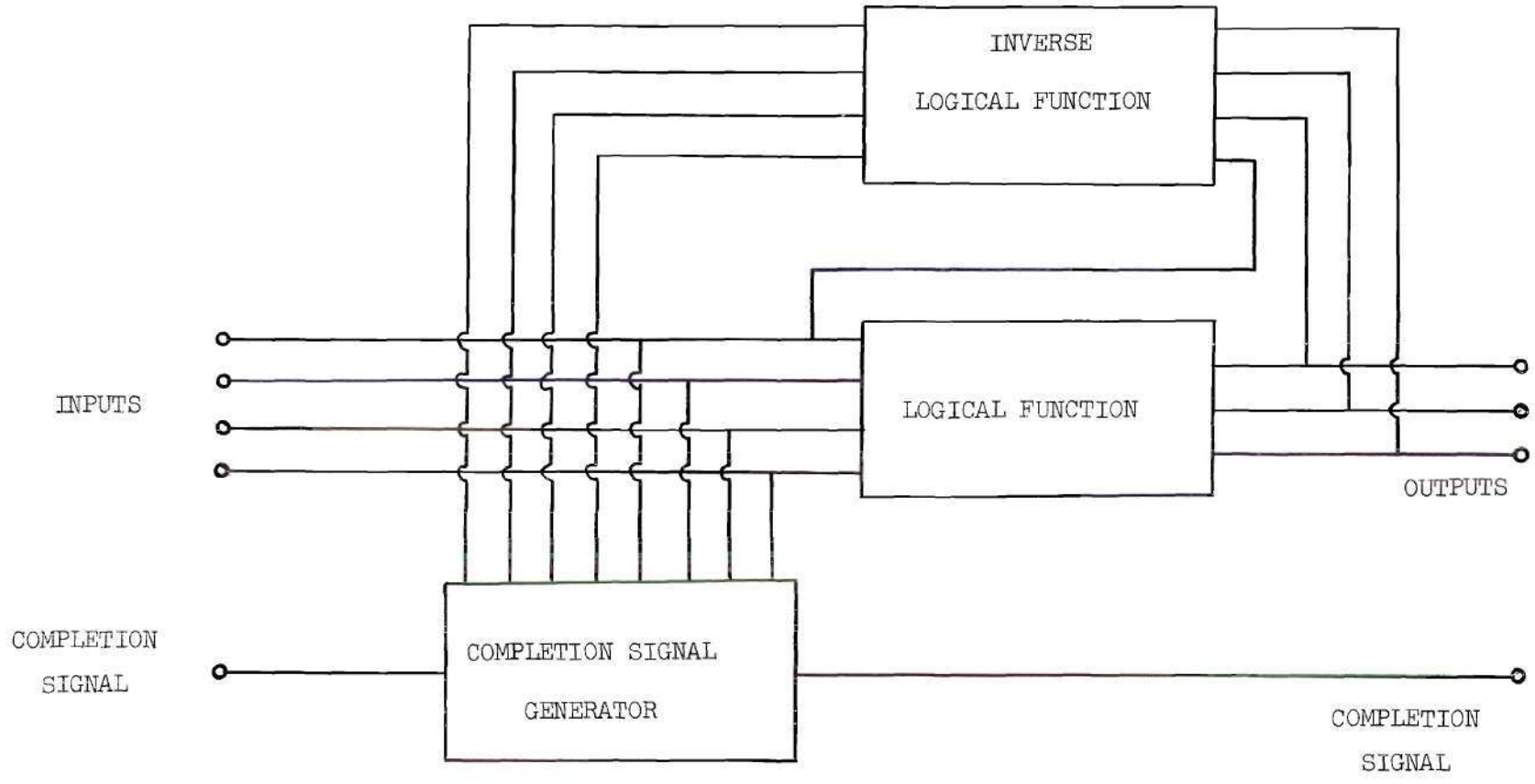


Figure 3. The Inverse Function Method

than the rule, and one, or possibly more, depending on the particular logical function, of the inputs to the logical function must also be presented to the inverse logical function to insure that a unique inverse will be obtained, making the outputs of the inverse function the same as the inputs of the primary function.

If the outputs of the inverse logical function and the inputs of the primary logical function are compared in the completion signal generator, agreement will indicate that the logical function has completed its work on its inputs and that the outputs are those which should correspond to these inputs. If in addition the input completion signal presents a "complete" signal to the completion signal generator, the output completion signal will indicate "complete." Otherwise the output completion signal will indicate "incomplete." The completion of the operation represented by the logical function has thus been sensed. If all the circuits involved are hazard-free circuits, then the overall operation must be independent of the switching speeds of any of its elements and hence is speed-independent.

Useful features.--This method of obtaining a completion signal has some very interesting and useful features. The only resetting necessary with the inverse function method is in the completion signal generator. Between sets of inputs, an "incomplete" signal must be presented to the completion signal generator and an "incomplete" output completion signal obtained. No resetting of the input and output variables is required to prepare the circuit for the next set of inputs.

If a speed independent selector is provided at the input, as shown in Figure 4, which permits the outputs of the inverse logical function to

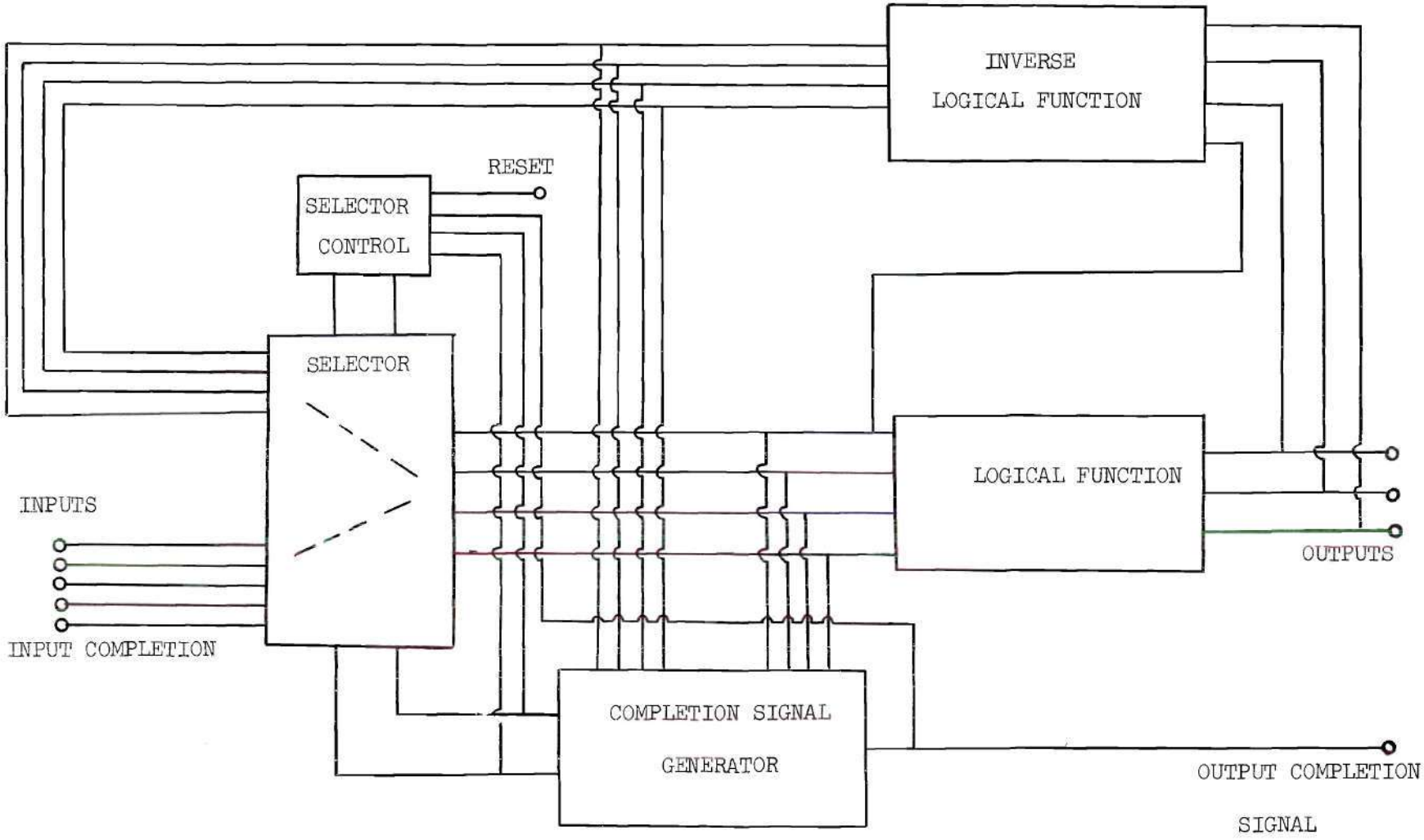


Figure 4. The Inverse Function Utilizing the Feedback Memory Loop.

be used as the inputs of the logical function, a closed feedback loop is formed. If sufficient amplification is provided in this loop, the outputs will be self-sustaining and memory of both the input and output variables is established (3). Thus some of the additional circuits which were added to permit sensing of the completion of the logical function have been used to form a memory unit around the logical function. This feature of the inverse function method could be used to aid in justifying the use of the additional circuit elements made necessary by the generation of the inverse logical function.

Another useful feature, which should give rise to very fast circuits, is that the logical function is permitted to begin its consideration of the inputs before the input completion signal indicates "complete." If a considerable time interval is allowed between the actual presentation of the inputs and the presentation of the "complete" signal, the logical function may have already finished its work, so that the completion signal need only propagate through the circuits in the completion signal generator to give an output "complete" signal. It is to be noted, however, that this feature cannot be used in conjunction with a circuit using the feature of memory discussed above unless the circuit is not in the memory mode when the inputs are presented.

Disadvantages.--The inverse function method has two major disadvantages when compared to computer circuits which are not speed-independent. These are the greatly increased circuit complexity and the necessity of using hazard-free logical function realizations and hazard-free electronic implementation of these logical functions.

The circuits must not give false outputs even when the inputs are allowed to change simultaneously or in an arbitrary order, except for the completion signal, which is restricted to change to "incomplete" before changes in other inputs are allowed, and to change to "complete" only after all other input changes have taken place. This forces, generally, the use of logical realizations which are not simplified beyond the standard sum or standard product form, as this is the only way to positively eliminate possible hazards in general for this type circuit (8). With caution, some simplification can be made, but no generally applicable simplification procedures are available for circuits in which inputs are allowed to change simultaneously.

No major disadvantages have been found in circuits using the inverse function over other speed-independent circuits.

CHAPTER III

A ONE-BIT SUBTRACTOR

Design objectives.--To illustrate the inverse function method of sensing the completion of operations, a simple circuit which uses this method and which forms a possible building block for more complex circuits was designed and constructed. The logical function chosen was subtraction of two binary input variables to obtain two output variables, a difference and a borrow. A speed-independent selector was also designed as a part of the subtractor to illustrate the use of the outputs of the inverse logical function to incorporate the feature of memory into the circuit.

This circuit was found to be simple enough to make construction feasible on the limited budget available. In spite of its simplicity, however, the circuit serves to illustrate the inverse function method and its most useful features. The system design and the logical design of the one-bit subtractor are presented here.

System design.--The one-bit subtractor requires four input signals and three output signals. The inputs are the two variables to be subtracted, an input completion signal, and a reset signal. The minuend will be denoted by X, the subtrahend by Y, the input completion signal by C_{in}, and the reset signal by R. The outputs are the two output variables and the output completion signal. The difference will be denoted by D, the borrow by B, and the output completion signal by C_{out}. All signals switch between two stable states which are represented by the binary variables 0 and 1.

If the variables X and Y are presented and C_{in} indicates "complete" by changing to 1, the circuit will indicate when the borrow B and difference D have been formed by causing the output completion signal C_{out} to change to 1. When C_{out} changes to 1, the selector will switch to cause the circuit to enter the memory mode. With the circuit in the memory mode X, Y and C_{in} may be changed at will without affecting the output. The reset signal is necessary to cause the circuit to leave the memory mode, and it accomplishes this when R is changed from 0 to 1. It is impossible for the circuit to re-enter the memory mode until R is switched back to 0.

When R is switched to 1, the selector switches back to the input signals, opening the feedback loop which provides the memory and introducing the new inputs to the subtraction logic. Opening the feedback loop causes C_{out} to change back to 0. The new inputs to the selector may be in one of three categories: X and Y may be present with C_{in} indicating 1, X and Y may be present with C_{in} indicating 0, or X and Y may be changing with C_{in} indicating 0. The reset signal R may be switched from 1 back to 0 at any time after C_{out} changes to 0.

The circuit may function in one of several ways, depending on the inputs present when R is switched to 1 and the selector switches back to the circuit inputs and on the situation existing when R is switched back to 0. The circuit may also be made to function differently by causing R to switch to 1 at different points in the operating cycle of the subtractor. If R is kept at 1, the circuit will not exhibit a memory but will operate as though the selector were not a part of the circuit at all.

It is evident that the subtractor is a sequential switching circuit, since its outputs are not direct functions of its inputs at all times, but rather depend on the order or sequence in which the inputs are presented. Any attempt to synthesize such a circuit ultimately reaches a point at which the circuit is divided into several sections, each of which is a combinational switching circuit, in which the outputs are direct functions of the inputs.

No sophisticated means of accomplishing the subdivision of the subtractor into combinational circuits was used. It was found instead that the subdivision given in Figure 4, which follows immediately from the development of the inverse function method, could be used. The functions involved are then the logical function, or subtraction logic, the inverse logical function, the completion signal generator, the selector, and the selector control.

The selector, in order to be speed-independent, must not only switch the inputs to the logical function between the outputs of the inverse logical function and the inputs X and Y, but must also indicate when the switching has been completed. That is, it is necessary to do more than instruct the selector to close one path and open the other; it is also necessary to obtain from the selector the information that these instructions have been carried out.

It is necessary then to add to the selector two completion signal generators, in addition to the elements which would be required for a selector which was not required to operate speed-independently. These completion signal generators are used to check each of the two paths in the selector. If a path has been instructed to close and the inputs to

that path agree with the outputs of the selector, then the completion signal generator for that path indicates "complete." Otherwise the completion signal generator indicates "incomplete."

A block diagram of the subtractor is given in Figure 5, with the signals at the inputs and outputs of the blocks labeled. The signals G_i and G_f instruct the selector, while A_i and A_f indicate that the corresponding instructions have been carried out. The signals G_i and A_i refer to the input path between X, Y and L_1, L_2. The signals G_f and A_f refer to the feedback path between I_1, I_2 and L_1, L_2. The paths are instructed to open by a 0 instruction signal and to close by a 1 instruction signal. That an instruction has been carried out is indicated by the corresponding answer signal changing to agree with its instruction signal. It is possible for G_i, G_f, A_i, and A_f to all be 1 or 0 at the same time; that is, both paths may be open or closed at the same time. The feedback path is never instructed to close unless C_{out} is 1; whenever the feedback path is open, as indicated by A_f being 0, the input path is instructed by G_i to close. Thus when R is switched to 1 to reset the circuit, G_f is switched to 0; when A_f indicates 0 and C_{out} indicates 0, G_i is switched to 1 picking up the new inputs.

It is evident that all of the blocks except the selector control represent combinational switching functions. It is not, however, immediately evident that the selector control, with the inputs and outputs shown in Figure 5, can cause the selector to operate properly with only a combinational function. This can be readily discovered and demonstrated by drawing the primitive flow table for the selector control (9). This table is shown in Figure 6.. This flow table not only indicates whether

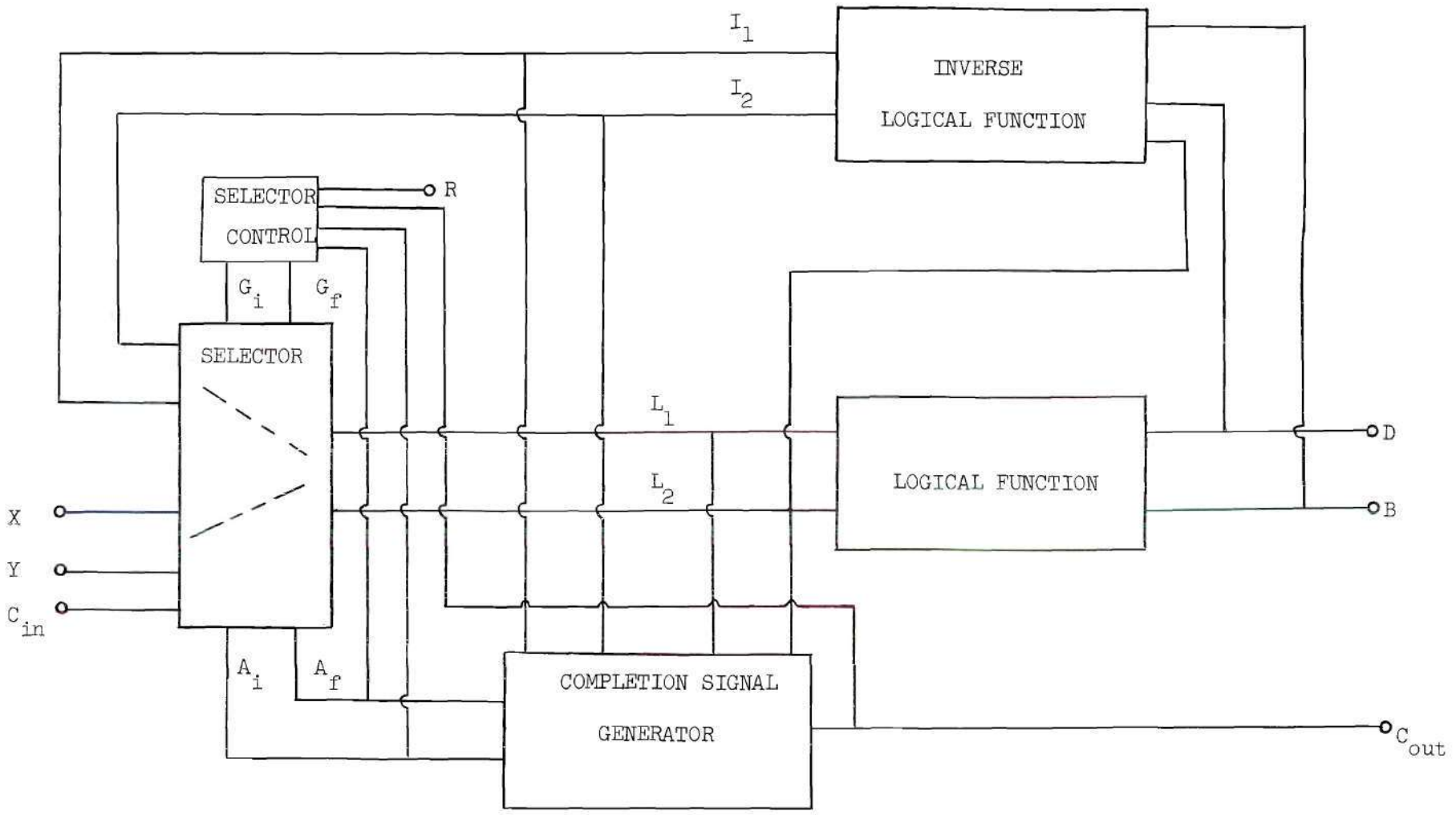


Figure 5. Block Diagram of the One-Bit Subtractor.

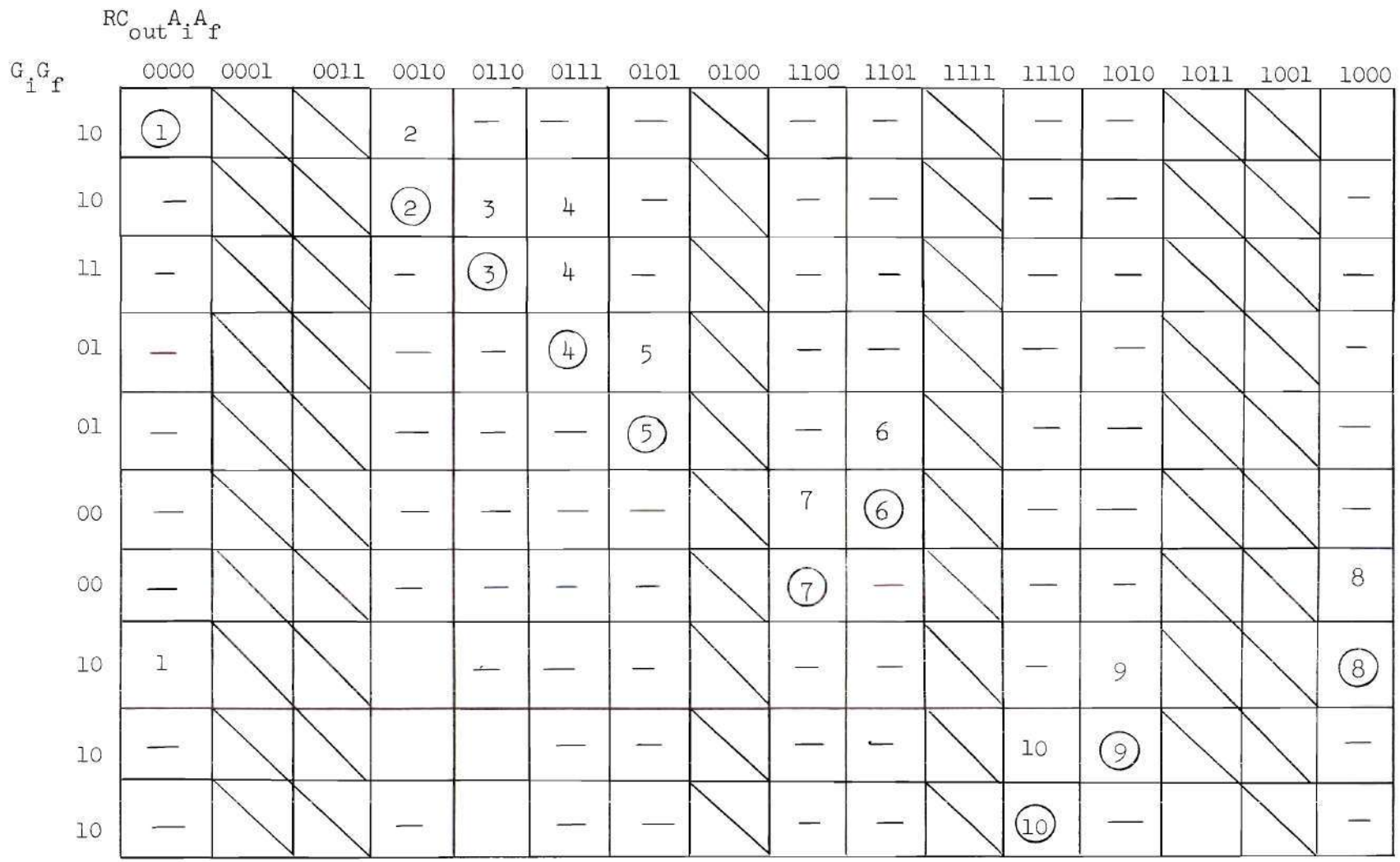


Figure 6. Primitive Flow Table for the Selector Control.

or not the circuit can be designed as a combinational circuit, but it shows in a compact, tabular form all possible sequences of operation for the subtractor. The information contained in many lengthy word descriptions of operating cycles is presented concisely by this table.

Entries across the top of the table represent the inputs to the selector control in the order R, C_{out}, A_i, A_f and entries down the left side of the table represent the outputs in the order G_i, G_f. The circuit can operate only to move between the numbered squares. Those squares with diagonal lines drawn through them correspond to input conditions which can never exist. Input changes cause the circuit to move horizontally in the table to another numbered square. The circuit must then move vertically to the square in which that number is circled.

This flow table may readily be simplified to one row, demonstrating that the selector control is a combinational function. The entire logical design was thus reduced to the design of combinational switching circuits.

Several interesting points should be made in connection with the design of the selector control circuit. If the system subdivision shown in Figure 5 had not resulted in a combinational switching function for the selector control function, but had resulted in a sequential selector control function, then the two courses might have been taken. Under certain conditions, the sequential control function might have been designed directly as a speed-independent circuit. More generally, however, another system subdivision would have been necessary.

Some special conditions under which a direct design for a speed-independent sequential switching circuit may be achieved are stated by Unger (3). These conditions are that inputs are restricted to change one

at a time, that all input and internal changes which an input change calls for take place before the next input change occurs, and that the flow matrix for the function contains no essential hazards. The terms "flow matrix" and "essential hazard" are explained in detail in the reference given. For speed-independence with such a function it is also necessary that the outputs be the last signals to change, and that the outputs influence other sections of the circuit somehow to stimulate the next input change. Otherwise, there can be no assurance that the input changes will not occur more rapidly than the required internal and output changes can take place. Circuits in which the outputs are the last signals to change have been referred to as "last-moving-point circuits"(6).

Logical design.--Except for the selector the logical design of each of the five combinational circuits in Figure 5 can be carried out by straightforward application of well-known design methods. The resulting Boolean functions are given in Figure 7; the details of the design of these circuits is given in Appendix I. The Boolean description of the subtraction logic and the completion signal generator are standard sum forms obtained directly from a table of combinations. Some simplification of the functions for the inversion logic and the selector was made through judicious use of optional entries in the Karnaugh maps for these functions.

The instructions for the selector can be carried out readily by the functions given in Figure 7 for L_1 and L_2. The functions are realized simply by AND gates followed by OR gates as in Figure 8. The completion signal generators which check each path through the selector are identical to the completion signal generator for the logical function. For the path controlled by G_i, a completion signal A_i is obtained by checking X, Y and

Logical Function: $D = L_1'L_2 + L_1L_2'$

$B = L_1'L_2$

Inverse Logical Function: $L_1 = DB' + L_2B'$

$I_2 = L_2$

Selector Control: $G_i = C'_{out} + A_1A_2'$

$G_f = R'C_{out}$

Completion Signal Generator:

$$C_{out} = (A_i'A_f')'\ L_1'L_2'I_1'I_2' + (A_i'A_f')'L_1'L_2I_1'I_2 + (A_i'A_f')'L_1L_2'I_1I_2' + (A_i'A_f')'L_1L_2I_1I_2$$

Selector: $L_1 = G_iX + G_fI_1$

$L_2 = G_iY + G_fI_2$

$$A_i = (G_iC_{in})X'Y'L_1'L_2' + (G_iC_{in})X'YL_1'L_2 + (G_iC_{in})XY'L_1L_2' + (G_iC_{in})XYL_1L_2$$

$$A_f = (G_f)M_1'M_2'L_1'L_2' + (G_f)M_1'M_2L_1'L_2 + (G_f)M_1M_2'L_1L_2' + (G_f)M_1M_2L_1L_2$$

Figure 7. Summary of the Logical Design for the One-But Subtractor.

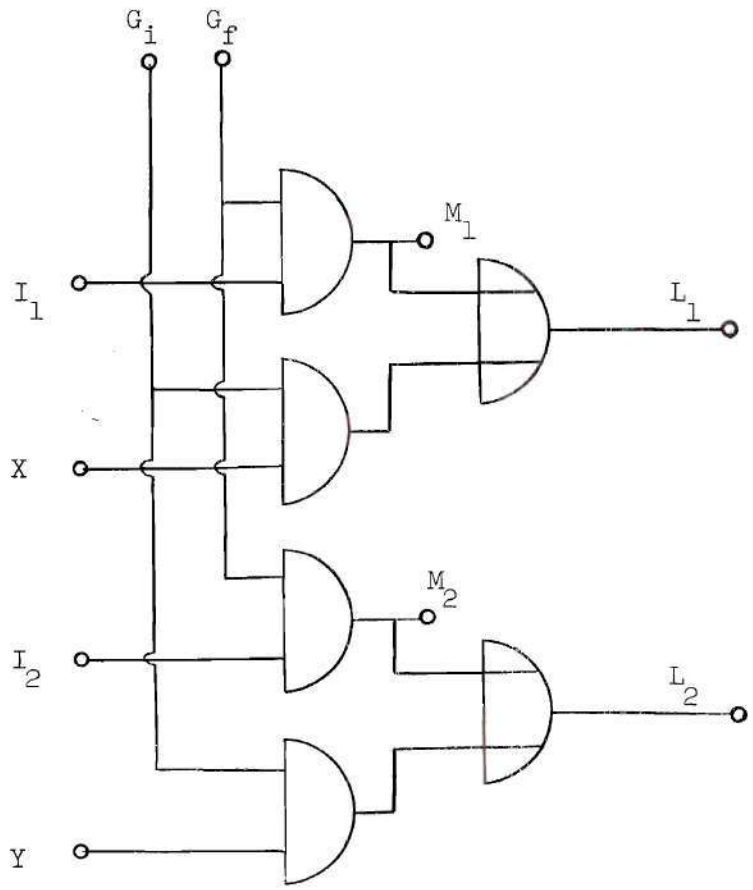


Figure 8. The Selector Without the Checking Circuits.

L_1, L_2 for agreement, with C_{in} and G_i combined in an AND gate to give an indication to this completion signal generator that the path should be closed. The path in the feedback loop is more subtle. This path is never instructed to close unless I_1, I_2 and L_1, L_2 agree, as indicated by C_{out} being 1. Hence checking these signals again would be superfluous. However, it is necessary to check the output of the AND gates at which G_f and I_1 are inputs and at which G_f and I_2 are inputs to be sure that these gates have responded. The outputs of these gates are denoted by M_1 and M_2, as shown in Figure 8. All actions of the selector are checked by the two completion signal generators giving A_i and A_f, and the selector can thus be made to operate speed-independently.

The logic for the entire one-bit subtractor is shown schematically in Figure 9.

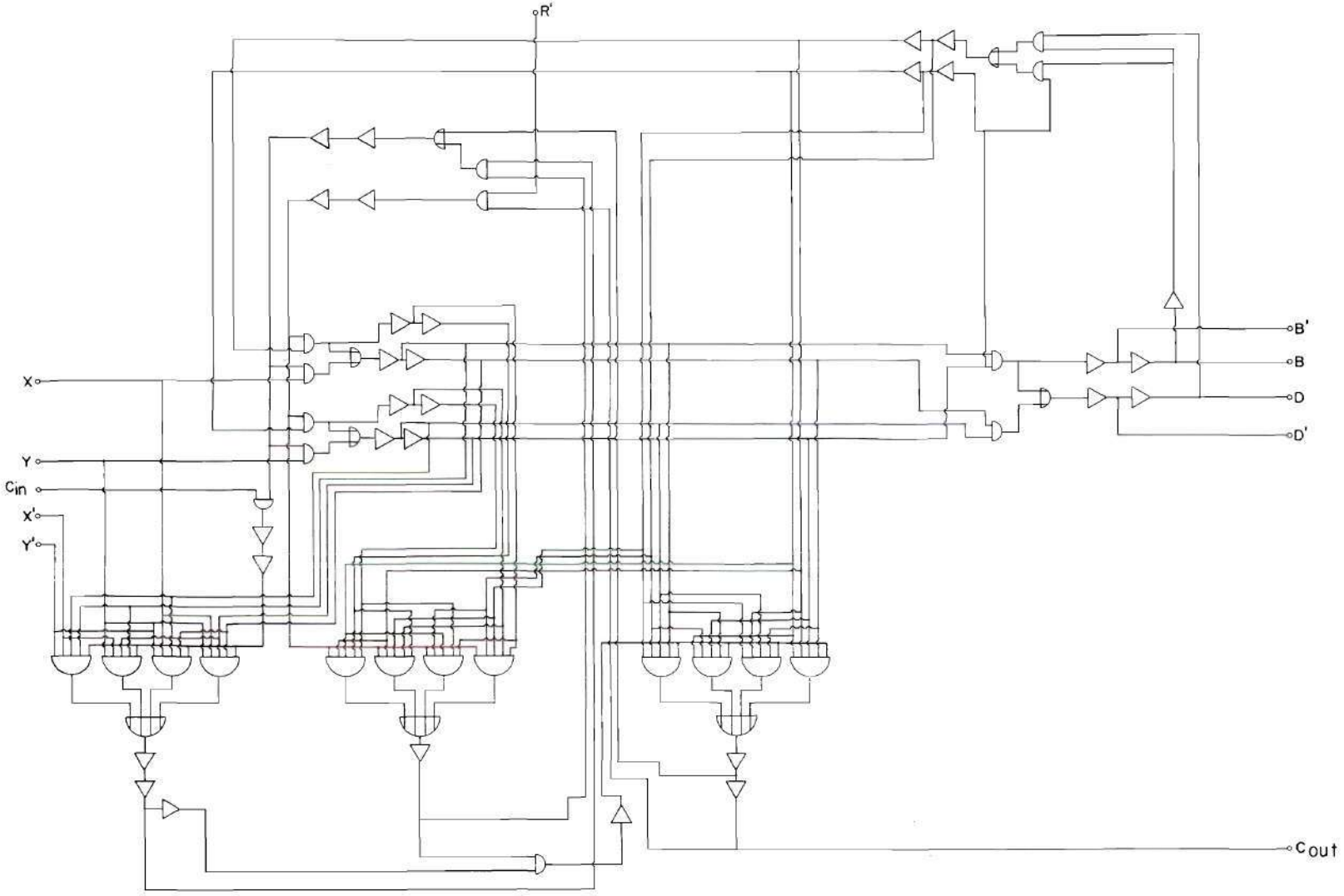


Figure 9. Logical Diagram of the One-Bit Subtractor

CHAPTER IV

CIRCUIT DESIGN FOR THE ONE-BIT SUBTRACTOR

Objectives.--The primary objective of the electronic implementation of the logical diagram in Figure 9 for the one-bit subtractor was the demonstration of a speed-independent circuit. The speed of operation of the circuit was of secondary concern here, even though the study of speed-independent circuits is interesting primarily because of the potential these circuits have for extremely high speed operation with high reliability. The subtractor was designed around inexpensive components to provide an inexpensive speed-independent circuit for laboratory experimentation rather than actual application.

Basic circuits.--The logical diagram of Figure 9 can be broken down into similar groups, all of which can be constructed from single-level AND gates or double-level AND-OR gates followed by inverting amplifiers. The breakdown used required four units consisting of two-input AND gates driving a two-input OR gate, three units consisting of four five-input AND gates driving a four-input OR gate, three units consisting of a single two-input AND gate, and one unit consisting of a two-input AND gate driving a two-input OR gate.

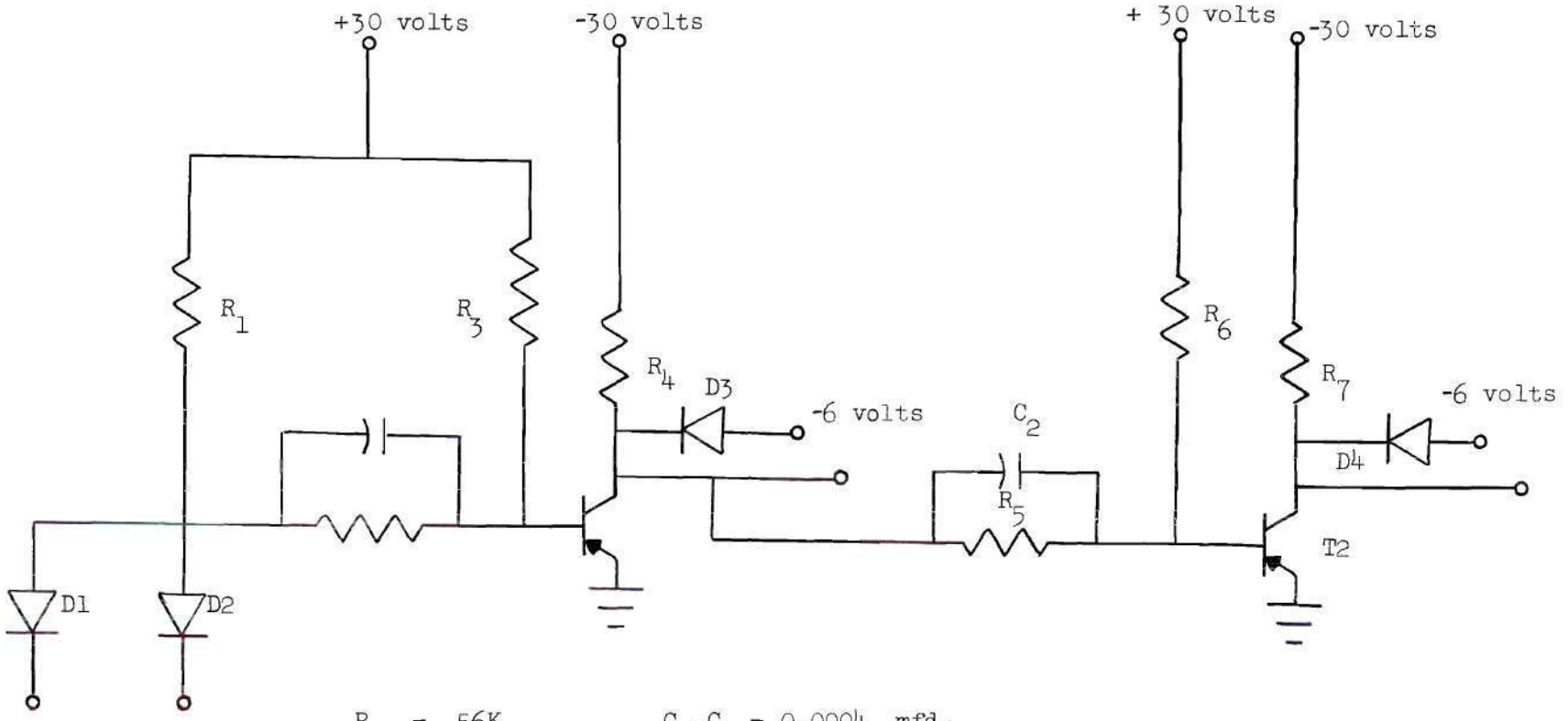
The actual circuit design work was thus the design of a double-level gate, with four five-input AND gates driving a four-input OR gate and the design of a single-level AND gate with two inputs. The smaller double-level gates were then formed by simply omitting some of the structure of the large double-level gate. Amplifiers to provide the required

drive capability were designed to follow the output of each gate. With these circuits as building blocks, the logic for the subtractor was implemented by simply making the appropriate interconnections between gates and amplifiers, following Figure 9.

Voltage-switching diode gates were used, with transistor amplifiers, and the design was carried out avoiding the use of high precision components. Resistors were allowed ± 10% tolerances, and the supply voltages were specified with ±5% tolerances. Transitron type 1N97 diodes and General Electric type 2N394 transistors were chosen; these are inexpensive units designed for general computer application.

The 1N97 is a germanium point contact diode, and the 2N394 is a pnp junction transistor. In order to obtain useful volt-ampere characteristics for the devices, the characteristics of twenty diodes and twenty transistors were checked in the laboratory and representative curves drawn. Collector and base characteristics in the common emitter connection for the transistor and forward and reverse characteristics for the diode are given in Appendix II. Switching times for the transistors were also measured, and representative curves are given in Appendix II. The procedures used to measure the switching times are given in detail by Pressman (11).

The circuit diagram for the single-level gate is given in Figure 10, and the diagram for the double-level gate in Figure 11. Each of the gates was designed with one of the amplifiers shown included as an integral part of the gate; the second amplifier was then made identical to its driver. In both gates the transistor is bottomed to within -0.3 volts of ground when turned on to a collector current of -20 ma.; the collectors are clamped at -6.0 volts on turn-off, and the transistor saturated slightly at turn-on, to provide a signal switch of 6.0 volts with 0



R_1 = 56K

R_2, R_5 = 3.9K

R_3, R_6 = 390K

R_4, R_7 = 1.5K

C_1, C_2 = 0.0004 mfd.

D1, D2, D3, D4: 1N97

T1, T2: 2N394

Figure 10. Schematic Diagram for the Single-Level Gate.

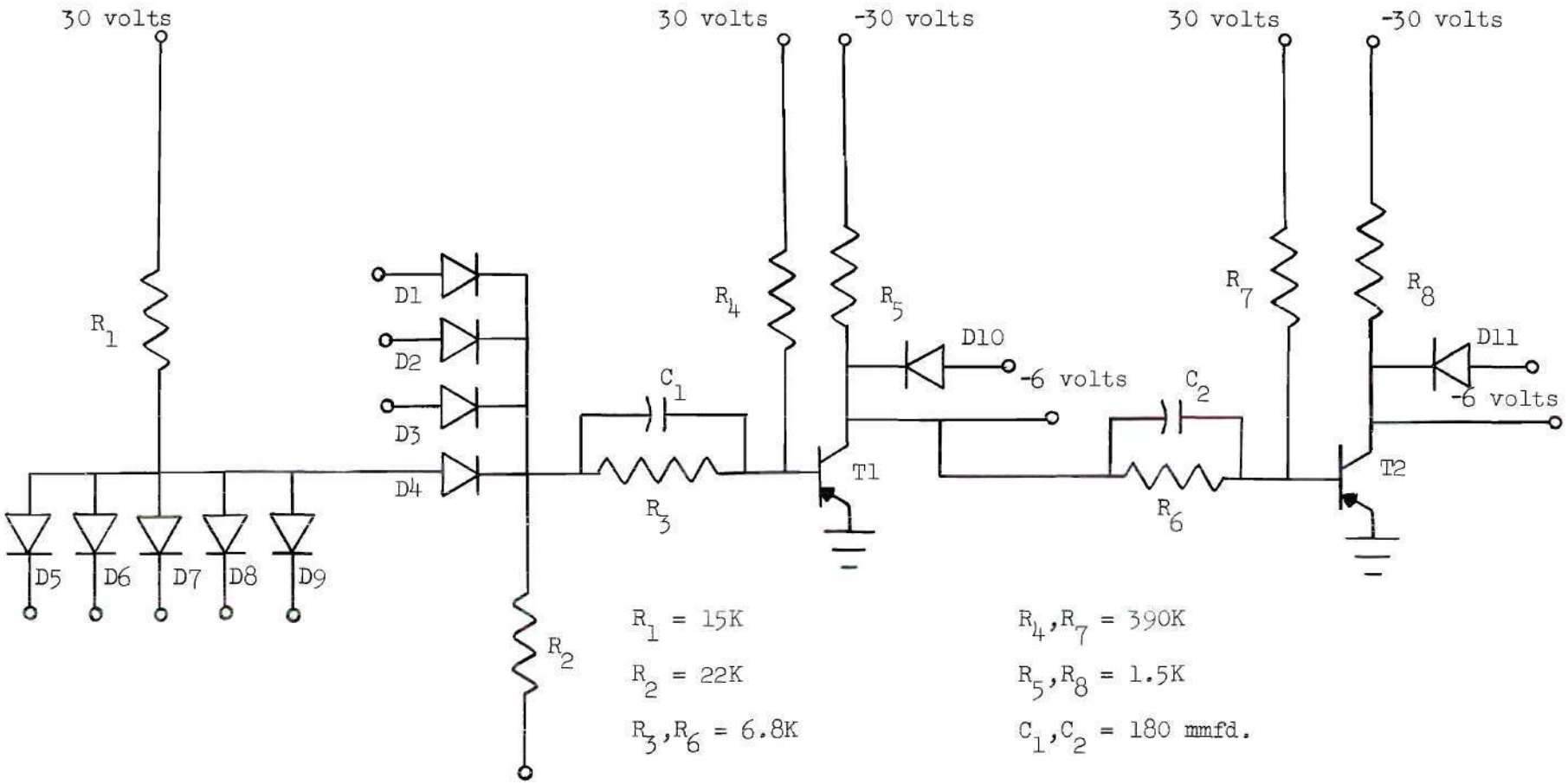


Figure 11. Schematic Diagram of the Double-Level Gate.

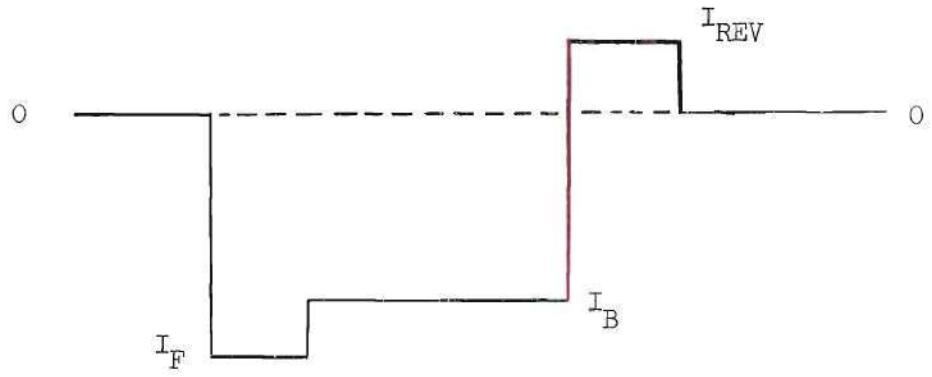
corresponding to -6.0 volts and 1 corresponding to zero volts or ground.

The design method used was the "worst case" approach described by Pressman (11). This method accounts for components and supply voltages being off their nominal values by a specified tolerance, and for specified spreads of transistor gain and switching times. The gates were designed using the step-by-step examples given by Pressman.

The load resistor, for the amplifier following both the single-level and double-level gates, was set at 1500 ohms to provide a nominal collector current of 20 ma. when the transistor is turned on. The 390,000 ohm base resistor used in both amplifiers provides nominally 0.7 ma. reverse base current when the transistor is turned off, keeping the collector current low in the off state even though the common emitter connection is used for the amplifiers.

The RC circuit in the base input lead provides, ideally, the current waveform shown in Figure 12 at the base. This waveform has peaks at both turn-on and turn-off. The peak at turn-on provides an over-drive to give a fast rise time; the low-level current between the peaks, ideally, keeps the transistor just slightly saturated; the peak at turn-off provides the reverse drive required for fast fall time.

In the single-level gate, a 3900 ohm resistor and 400 micromicrofarad capacitor were used, with the 56,000 ohm gate resistor, to provide, nominally, a steady-state drive current of 1.4 ma., and average forward over-drive current of 1.5 ma. during turn-on, and an average reverse drive current of 0.5 ma. during turn-off. This resulted in a calculated rise and fall time of 1.0 microseconds with a fan-out ratio of five. The gate was used satisfactorily to drive six of the double-level gates, as



I_F: Forward Overdrive Current

I_B: Normal Steady-state Base Current

I_{REV}: Reverse Drive Current

Figure 12. Ideal Base Current Waveform.

all gates driven did not change states simultaneously and had other inhibiting inputs.

In the double-level gate, a 6800 ohm resistor and 180 micromicrofarad capacitor were used, with a 22,000 ohm OR gate resistor and 18,000 ohm AND gate resistor, to provide, nominally, a steady-state drive current of 1.0 ma. during turn-on, and an average reverse drive current of 0.2 ma. during the turn-off. This resulted in a calculated rise and fall time of 1.5 microseconds with a fan-out ratio of five. Again, however, all driven gates were not inhibited simultaneously by one gate, and a fan-out of six was used satisfactorily.

The subtractor was constructed on six six-inch by twelve-inch bake-lite boards, and is pictured in Figure 13. Cambridge Thermionic Type 514D terminal posts were used to mount all components except the transistors, which were mounted in miniature transistor sockets.

Input and output.--For steady-state testing of the one-bit subtractor circuit, double-pole-double-throw toggle switches were used to switch in the desired inputs, and the outputs were observed by means of small 2 volt, 60 ma. lamps driven by auxiliary amplifiers as shown in Figure 14. A lamp was provided for the circuit outputs D, B, and C_{out} and for the input R. Transient testing was accomplished by driving an input or inputs with a square-wave generator and observing the outputs on a cathode-ray oscilloscope. During transient testing, the operation of the circuit was slowed by loading the collectors of amplifiers in key positions with capacitors.

Operation.--The design objectives were realized properly by the logical design and the circuit design of the one-bit subtractor, as verified by

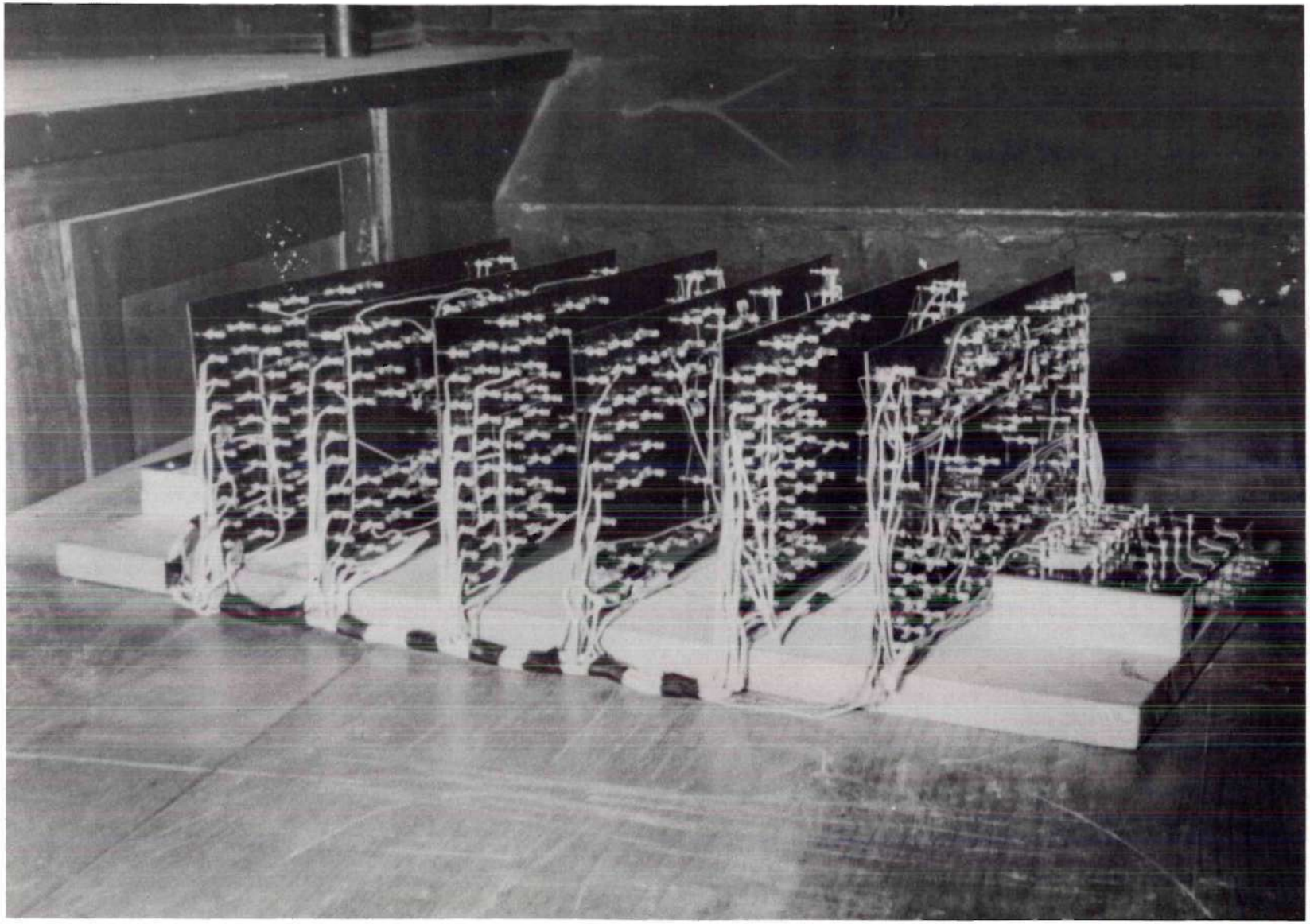
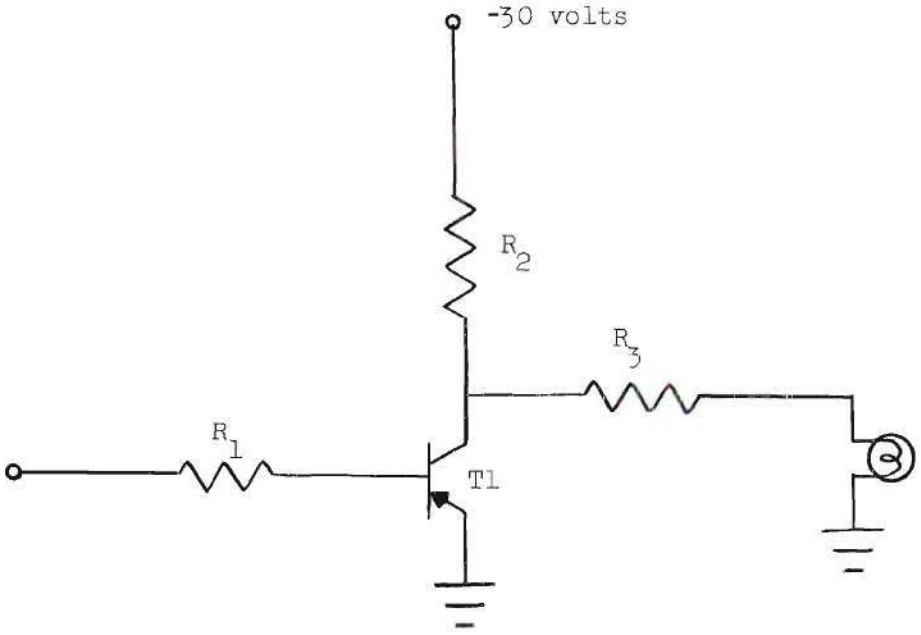


Figure 13. The One-Bit Subtractor Circuit.



R_1 = 1000 ohms

R_2 = 390 ohms

R_3 = 68 ohms

T1: 2N394

Figure 14. The Auxiliary Amplifier and Output Lamp.

the steady-state and transient tests. In the steady-state tests, it was demonstrated that the inputs and outputs of the circuit corresponded as specified in the steady-state condition, provided the circuit had been allowed to warm up. When the circuit was not warm, oscillation of C_{out} between 0 and 1 occurred for the inputs X=1, Y=0, with the oscillation suddenly dying out after the circuit warmed up, going to the proper C_{out}. It was demonstrated that the circuit was speed-independent in the transient tests, as slowing the operation of the circuit by loading key gates did not produce a malfunction of the circuit. A completely exhaustive transient test would have required input equipment not available in the laboratory. The tests made were: (1) square-wave input for R with C_{in}, X, and Y set in all possible combinations, (2) square-wave input for C_{in} with R, X, and Y set in all possible combinations, and (3) square-wave input for either X or Y with R, C_{in}, and one input variable fixed in all possible combinations. Each output was observed on an oscilloscope for each set of inputs.

CHAPTER V

AN N-BIT SUBTRACTOR

The one-bit subtractor which was designed and constructed could very easily be used as a building block in a more complex circuit. With proper design the more complex circuit could also be made speed-independent. As an example, an n-bit subtractor using the one-bit subtractor as a basic building block element is outlined. The complete logical design for the n-bit subtractor was not carried out.

The type of subtraction implemented in the n-bit subtractor might be called iterative parallel subtraction. It is parallel subtraction, with borrow assimilated only as it is needed. The method is shown for three cases with ten bit numbers in Figure 15. All bits are operated on simultaneously to obtain a borrow and a difference for each bit. The borrows are all checked. If any borrow of 1 is found, the subtraction is repeated, subtracting the borrow obtained in the (n-1)st bit from the difference obtained in the nth bit to obtain a new difference and borrow for the nth position. The borrows are then checked again, and the process is repeated if another borrow of 1 is found in any position. When no borrows of 1 are found, the process is terminated, and the last differences are taken as the differences constituting a final result for the subtraction. It should be noted, as illustrated in Figure 15, that several "runs" of the borrow may occur simultaneously.

This method can be implemented by a circuit following the block diagram shown in Figure 16. The diagram is for a typical bit; somewhat

(a)
```
  1111111111
- 1010101010
  0101010101
```

(b)
```
  1000000000
- 0000000001
  ----------
  1000000001
- 0000000010
  ----------
  1000000011
- 0000000100
  ----------
  1000000111
- 0000001000
  ----------
  1000001111
- 0000010000
  ----------
  1000011111
- 0000100000
  ----------
  1000111111
- 0001000000
  ----------
  1001111111
- 0010000000
  ----------
  1011111111
- 0100000000
  ----------
  1111111111
- 1000000000
  ----------
  0111111111
```

(c)
```
  1000100100
- 0001001001
  ----------
  1001101101
- 0010010010
  ----------
  1011111111
- 0100100100
  ----------
  1111011011
- 1001001000
  ----------
  0110010011
```

Figure 15. Subtraction with Occasional Borrow Assimilation.

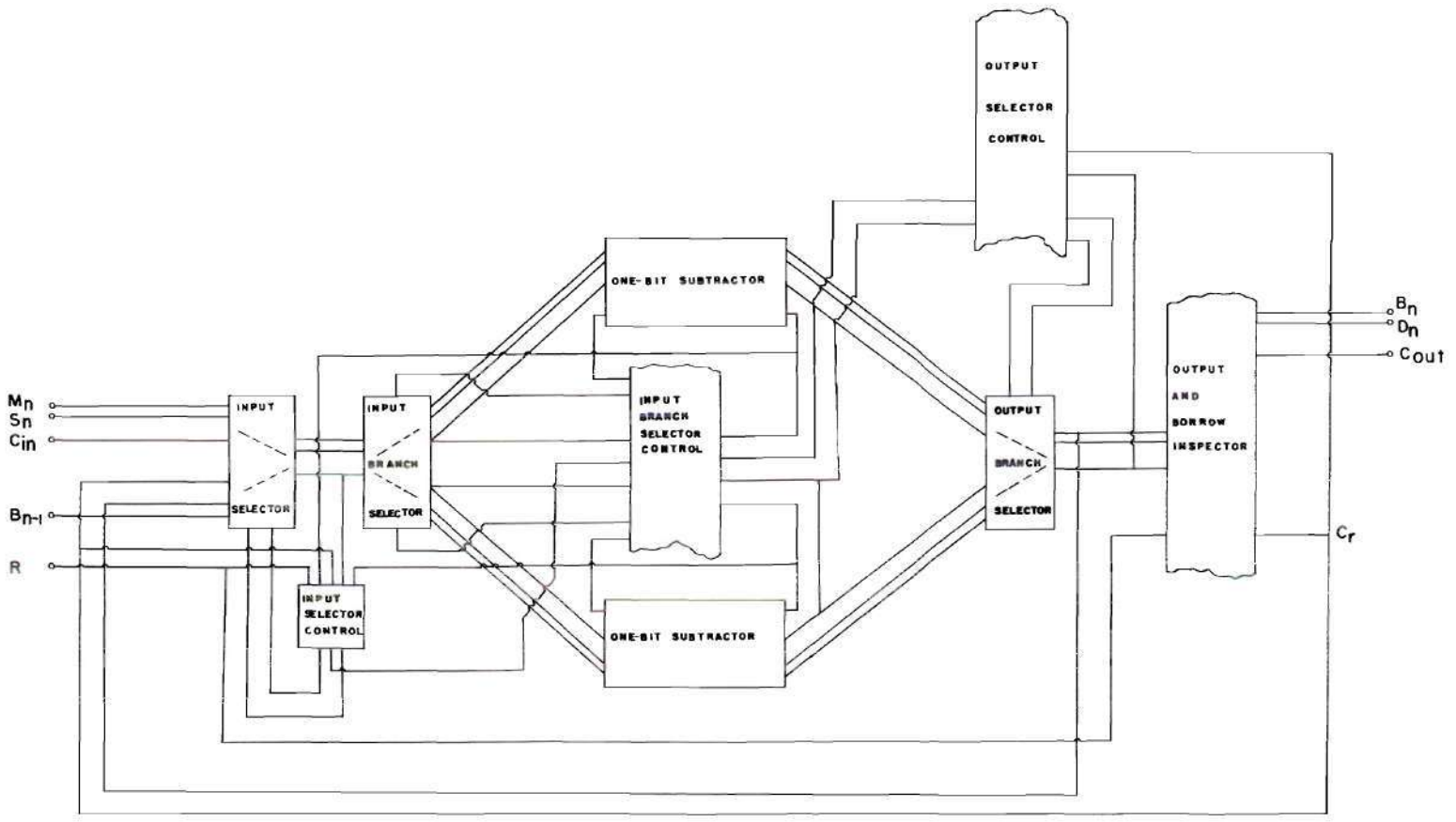


Figure 16. Block Diagram of the N-Bit Subtractor

simpler systems would be appropriate for the first and last bits. Blocks which have broken edges represent circuits which are common to all bits and operate on information obtained from all bits. Other blocks represent circuits which are related only to the typical bit.

The circuit inputs are M_n, S_n, C_{in}, and R. The variables M_n and S_n are the nth bits of the minuend and subtrahend, respectively, C_{in} is the input completion signal, and R is the reset signal. The input completion signal insures that M_n and S_n are the desired input variables. The reset signal is necessary to cause the circuit to consider a new problem after all steps in the present problem have been completed and the information obtained sent to its destination. The outputs for the circuit are B_n, D_n, C_{out}, and C_r. The signals B_n and C_n are the borrow and difference bits obtained at each stage of the subtraction. The signal C_r indicates whether or not another cycle is necessary. When all cycles are complete and the final result is present on the output leads, the output completion signal C_{out} will change to one.

The input selector switches the primary inputs, M_n and S_n, and the inputs for succeeding cycles in the subtraction, D_n and B_{n-1}, into a second selector, the input branch selector. The input branch selector switches to choose between two identical logical branches, each consisting of the one-bit subtractor. The output selector switches the outputs of these branches into the output and borrow inspection unit. Each of the selectors is controlled by a selector control unit, some or all of which may be sequential circuits requiring a further system breakdown before detailed logical design permitting speed-independent operation can be achieved.

The circuit functions in the steps described above for the subtraction. The first step presents to one of the logical branches the primary inputs M_n and S_n. The logical branch which receives these inputs is determined by the number of steps or cycles required for the preceding problem. If all problems required an even number of cycles, the inputs would always go to the same branch for the first step in a problem. However, when a problem involves an odd number of cycles, the branch which receives the inputs for the first step will be changed, since each successive cycle of a problem is carried out in the opposite branch. It would be possible, but unnecessary, to require the circuit to completely reset between problems, so that the same branch was always used in the first step.

As soon as the completion signals from the logical branches which perform the first step announce that the proper outputs have been obtained for all bits, the output selector is instructed to open the paths which are then closed. When the output selectors for all bits indicate by their output completion signals that both paths are open, and when C_r is 0, the new output paths are all instructed to close, and the output and borrow inspection unit begins its work.

An additional output is required from the one-bit subtractors for this application. The signal A_f (see Figure 5) must be taken as an output and presented to the input branch selector control unit and the input selector control unit. When the one-bit subtractor enters the memory mode, as indicated by A_f changing to 1, and C_r has changed to 0, the input selector switches from the primary inputs to the inputs for the second cycle. At the same time the input branch selector is instructed

to open the presently closed path. When the primary input path of the input selector has been opened, the input branch selector is instructed to close the path to the logical branch to be used for the next subtraction cycle, and the one-bit subtractor in that branch is reset. The reset signal for the one-bit subtractor is removed when the output completion signal for that subtractor changes to 0.

The output and borrow inspection unit indicates that another cycle of subtraction is necessary by causing C_r to switch to 1, or that no more cycles of subtraction are needed and that the final result is on the output lines by causing C_{out} to switch to 1. If recycling is indicated, the next cycle will begin as soon as the input selector and the input branch selector have completed their switching and the branch accepting the new inputs is reset. It is to be noted that the input switching action and the action of the borrow inspection unit may be carried out at the same time; the order of completion of these actions is not important. Thus the input switching and resetting operations of all bits may have been completed before the borrows are inspected. On the other hand, it is possible that the borrow inspection will be complete before the input switching is complete for all bits.

The circuit will continue to recycle, each time waiting for the operation of each bit to complete, until the output and borrow inspection unit indicates that the final result has been obtained. The useful features of the inverse function method of completion sensing are used to full advantage in the circuit. One branch is serving as a memory unit to temporarily store the results of one cycle while the other branch is carrying out the next cycle. If the input switching is rapid enough, the

branch accepting the new inputs may have already performed all its work on those inputs by the time the borrow inspection is complete. On completion of all operations, the circuit can serve as a register for the final result for as long as desired. The final result will only be removed when a new problem is accepted.

This is by no means the only circuit in which the one-bit subtractor could be used as a building block to form an n-bit subtractor. It could be used to build up a serial subtractor or a conventional parallel subtractor by serving as a speed-independent half-subtractor building block. These would be simpler circuits than the subtractor presented, but would not have potential operating speeds as high as that of the circuit presented and would not show the useful features of the inverse function method so clearly as the circuit in Figure 16.

CHAPTER VI

RELATION TO GENERAL SEQUENTIAL CIRCUITS

Sequential switching circuits have been analyzed in very general terms, and a general form for the representation of these circuits has been evolved (2). Speed-independent sequential switching circuits using the inverse function method for sensing the completion of operations can be arranged to fit this general form, and some interesting comparisons between speed-independent sequential switching circuits and sequential switching circuits which are not speed-independent may then be made.

The general form for sequential switching circuits suggested by Huffman is shown in Figure 17. This form can be used for relay circuits, electronic circuits, magnetic circuits, or any other physical realization of a switching circuit, although it is more convenient for some forms than others. The function generator is a combinational switching circuit, and the delay unit is representative of some device, or devices, which repeats its input at its output after some delay in time. The inputs to the function generator are divided into two categories: the primary input variables, which are the actual circuit inputs, and the secondary input variables, which are the outputs from the delay unit. The output variables are also divided into two categories: the primary output variables which are the actual circuit outputs, and the secondary output variables which are the inputs to the delay unit.

When a set of primary inputs is presented, the function generator provides at its output an appropriate set of output variables, both

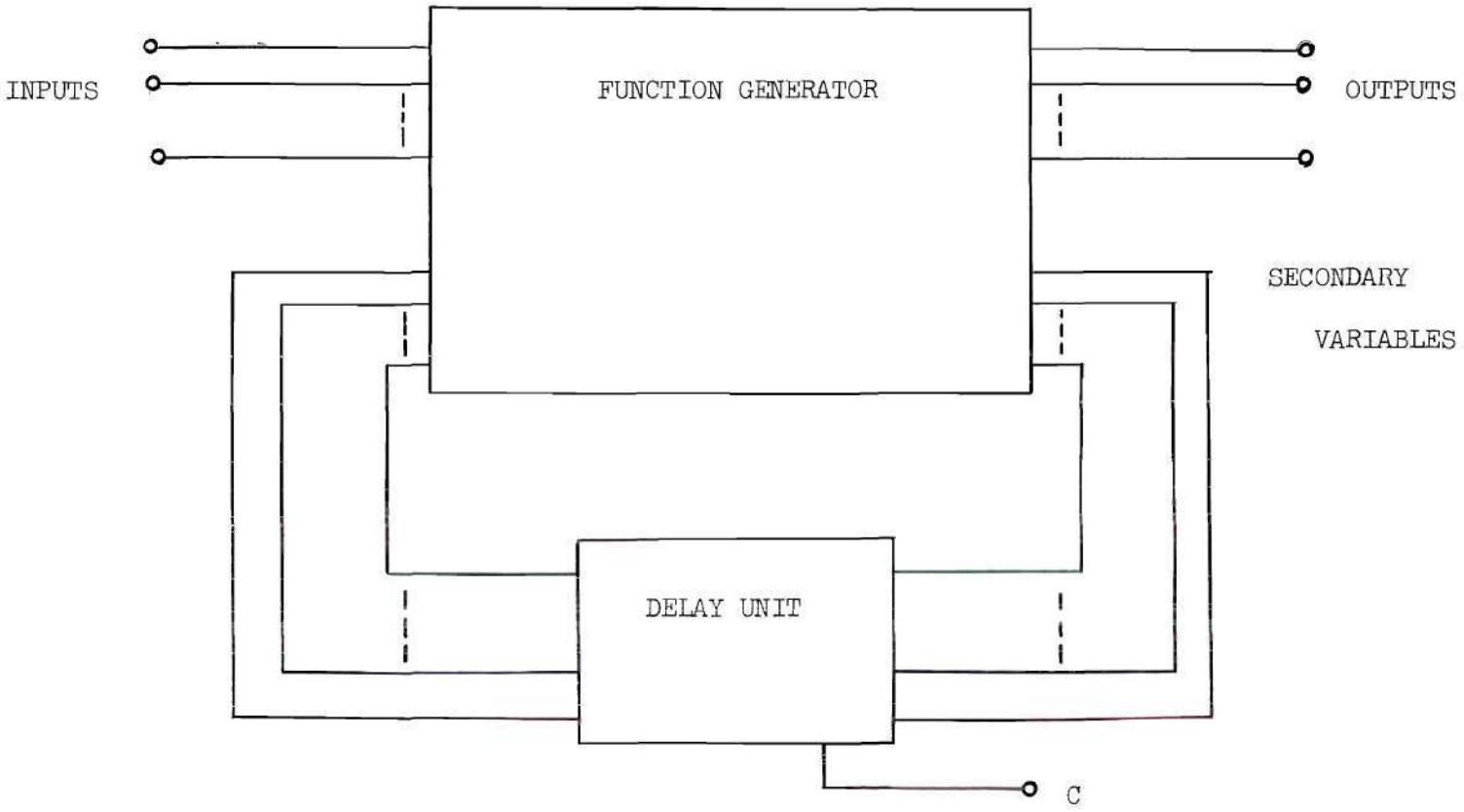


Figure 17. General Form for Sequential Circuits

primary and secondary. The outputs of the delay unit, after a delay, change to agree with the inputs to the delay unit. This results in a new set of secondary input variables appearing at the input to the function generator. The function generator acts on its new input variables to provide a new set of output variables. If any of the secondary variables are changed, the entire process repeats. When the action of the function generator gives secondary outputw chich are the same as the previous secondary outputs, the process ends. The entire circuit is stable after the primary outputs have changed also, and further changes can then occur only as the result of primary input changes.

The one-bit subtractor which was developed as an example of a speed-independent sequential circuit using the inverse function method of completion sensing can be placed in the general form suggested by Huffman as shown in Figure 18. The inputs and outputs are labeled to correspond with Figure 5. The delays in the delay unit are generated by the checking circuits, or completion signal generators, associated with the selector and by the logic circuits and inverse logic circuits themselves. That is, these delays are those between the time that G_i changes to A_i changes to agree, between the time that G_f changes and A_f changes to agree, and between the time that L_1, L_2 change and I_1, I_2 change to agree. The delays present in the delay unit are by no means the only delays present in the circuit; they are, however, the delays that are critical to the sequence of operation of the circuit.

In relay circuits, the delays in the delay unit are generally the action times of the secondary relays. In electronic circuits which are asynchronous, but not speed-independent, the delays might be those occur-

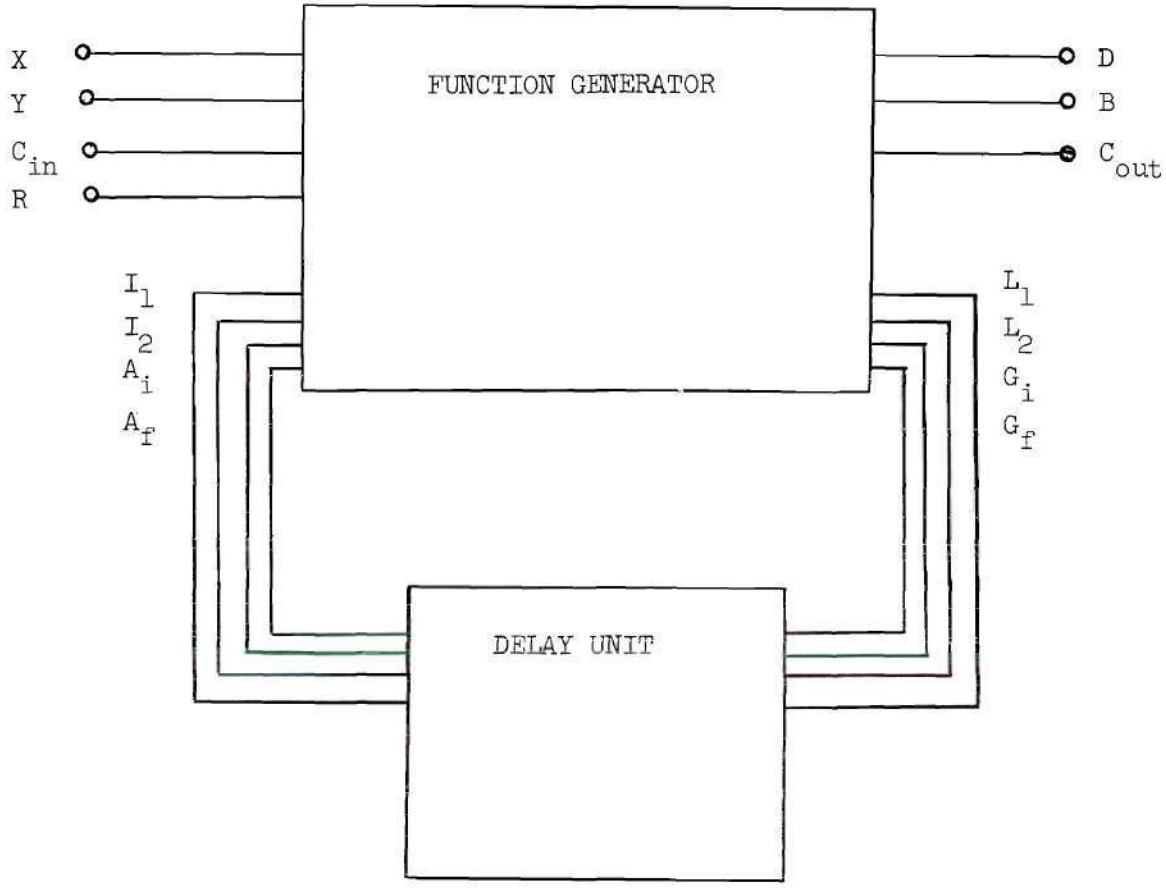


Figure 18. One-Bit Subtractor Arranged to Fit the General Form.

ring in actual delay lines. In synchronous circuits, the delay unit functions as an escapement, with the central clock providing an input to the delay unit, at C in Figure 17 for example, to determine the length of the delay. The delay unit may be a simple circuit or device or it may be some rather complicated circuit.

Generally, the inputs and outputs shown explicitly for the delay unit are the only signals with which the delay is concerned; however, in others, additional signals are involved. In the one-bit subtractor, as arranged in Figure 18, all of the signals required to characterize the delay are not shown as inputs to the delay unit; the delay between G_i and A_i is dependent also on agreement of X, Y with L_1, L_2, and the delay between G_f and A_f on agreement of I_1, I_2 with M_1, M_2. No additional signals are required to characterize the delay between L_1, L_2 and I_1, I_2.

The arrangement of the one-bit subtractor as shown in Figure 18 demonstrates the origin of the circuit delays critical to the operation of the circuit. Other arrangements of the one-bit subtractor to place it in the general form for sequential circuits are also possible. Instead of attempting to show the origin of the essential delays in the circuit, one branch, perhaps only an interconnecting lead, from each feedback path can be required to pass through the delay unit. Such branches correspond to the "state branches" in Unger's analysis of asynchronous circuits (3). In Unger's analysis, each state branch is either merely an interconnecting lead or contains some actual delay. If several state branches require a delay, a sophisticated realization of this delay may be achieved in one of several ways described by Unger. Since no elements

in the one-bit subtractor are actual delay elements, all state branches in an arrangement of this type for the subtractor will be interconnecting leads. This is a general result for all speed-independent circuits, since no circuit which is speed-independent can contain an actual delay element.

CHAPTER VII

CONCLUSIONS AND RECOMMENDATIONS

Applications of the inverse function method.--The inverse function method for sensing the completion of operations in speed-independent circuits is suitable for any operation in which the inputs and outputs are different, regardless of the number of inputs and outputs involved. For operations, such as gating to different parts of the circuit, in which the inputs and outputs are identical, there is of course no need to generate an inverse function; the completion can be sensed by simply comparing the inputs with the outputs. That the inverse function method is feasible has been demonstrated by the design, construction, and testing of the one-bit subtractor, and by the outlined n-bit subtractor.

The inverse function method should be particularly suited to computer circuits where extremely high speeds are desired, since the logic circuits can begin consideration of the input variables before the input completion signal indicates "complete." This is a feature not found in other speed-independent circuits. Any speed-independent circuit, whether it uses the inverse function method or not, is potentially faster than other circuits, since the completion and enabling signals in speed-independent circuits are generated by the completion of the operation concerned, and none of the tolerances necessary in other circuits on the time allowed for operations to complete are necessary. Some time is required for the inverse logic to complete after the primary logic has completed when the

inverse function method for completion sensing is used, but this time should not slow the overall operation of circuits using the inverse function method because of the simpler resetting this method affords over other methods.

The inverse function method should also be suited to circuits in which high reliability is required, if reliability is measured in terms of incorrect results obtained in an operation. An operation using the inverse function method can be made to malfunction only if the inputs to the operation are presented improperly, as for example in allowing the input completion signal to indicate "complete" prematurely, or if a spurious signal of some sort is produced in the completion checking circuits. More components are required for circuits using the inverse function method than in circuits which are not speed-independent, but failure of most of the components cannot produce an incorrect result. Only those components in the completion checking circuits can fail so as to produce an incorrect result, although failure of other components may prevent the circuit from producing any result at all by not allowing the output completion signal to indicate "complete."

Circuits using the inverse function method should permit easy maintenance. If a component fails and causes an incorrect result, then the component which has failed must be somewhere in the completion checking circuits. If a component fails and prevents an operation from completing, then no succeeding operations can complete. The operation whose circuits have failed can then be located easily by tracing the previous operations which have completed until the first incomplete operation is reached.

Limitations of the inverse function method.--In very high speed circuits, the problem of accounting for the transmission times of signals from one part of the computer to another is of interest, as well as accounting for the operating times of the gates, since the transmission times in very high speed circuits may become comparable to the switching speeds of the gates. Transmission time between logical gates involved in an operation does not create problems in speed-independent circuits, although for high speed operation this time must be kept small. Transmission time between operations is still a troublesome problem. If the path over which a completion signal is transmitted contains less delay than the paths over which corresponding signals are transmitted, then malfunctions may result.

Recommendations for further study.--Several areas in which further study should be very interesting and valuable have become apparent during the course of this research.

The feedback loop in the one-bit subtractor which provides the memory function for the subtractor was found to produce oscillation for some inputs. If the circuit was presented the inputs X=1, Y=0, C_{in}=1, the circuit developed the appropriate difference and borrow, D=1, B=0, and entered the memory mode, but the output completion signal sometimes oscillated between 1 and 0 for several seconds before suddenly settling down to the proper value, C_{out}=1. This difficulty was not present when the circuit had warmed up by being operated for several minutes, and hence was not critical to the operation of the circuit. It does indicate that further study of the stability of logical feedback loops might be profitable.

No general minimization methods for complex functions which will always yield hazard-free forms for the functions were found in the liter-

ture. The Karnaugh map and the concept of lift-sets and drop-sets suffices for simple functions, but are not convenient for functions involving more than four variables (8).

It would be extremely desirable to develop some simple means of characterizing speed-independent sequential circuits without resorting to highly abstract modern algebra. Representation of circuits in the form suggested by Huffman (2) and used by Unger (3) might lead to such a characterization, based on the ability to arrange the circuit so that all feedback paths contribute one state branch and so that no state branch contains an actual delay element. In Unger's analysis of asynchronous circuits, it is proved that any circuit for which the input variables are restricted to change one at a time can be realized properly with only one actual delay element. If it were possible to eliminate this delay element by using the inverse function method to sense the completion of a key operation, this delay element might be eliminated. If this could be done, a general method for the systematic design of speed-independent sequential switching circuits might be achieved, perhaps permitting the achievement of optimum design procedures for these circuits.

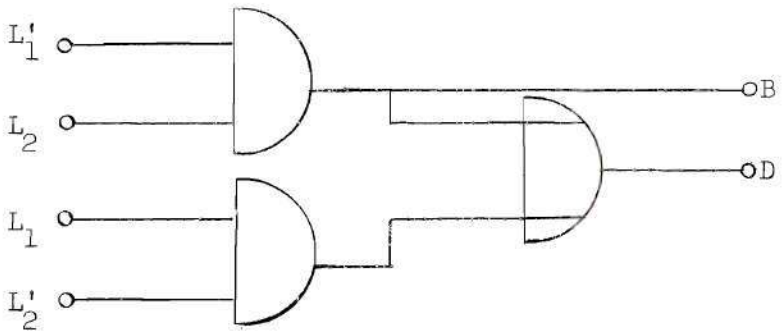
A P P E N D I X

APPENDIX I

LOGICAL DESIGN OF COMBINATIONAL CIRCUITS FOR THE ONE-BIT SUBTRACTOR

Subtractor logic.--The table of combinations for the subtractor logic is given below, and the Boolean functions for the output in terms of the input are derived from it. These functions may be developed with the logical diagram shown.

L_1	L_2	D	B	
0	0	0	0	$D = L_1'L_2 + L_1L_2'$
0	1	1	1	
1	0	1	0	$B = L_1'L_2$
1	1	0	0	



Inversion logic.--The table of combinations for the inversion logic corresponding to the subtraction logic is given below. The signal L_2 is used in addition to B and D to provide a unique inverse. That use of L_2 in this way will result in a unique inverse can be seen from an examination of the table of combinations for the subtraction logic.

D	B	L_2	I_1	I_2
0	0	0	0	0
0	0	1	1	1
0	1	0	Ø	Ø
0	1	1	Ø	Ø
1	0	0	1	0
1	0	1	Ø	Ø
1	1	0	Ø	Ø
1	1	1	0	1

The standard sums for I_1 and I_2 are thus:

$$I_1 = \Sigma(1,4) + \Sigma_{Ø}(2,3,5,6)$$

$$I_2 = \Sigma(1,7) + \Sigma_{Ø}(2,3,5,6)$$

By representing I_1 and I_2 on Karnaugh maps in three variables, it may be found that interpretation of the optional entries with asterisks as 1's gives very simple functions for I_1 and I_2.

I_1:

L_2 \ DB	00	01	11	10
0	0	Ø	Ø	1
1	1	Ø	0	0*

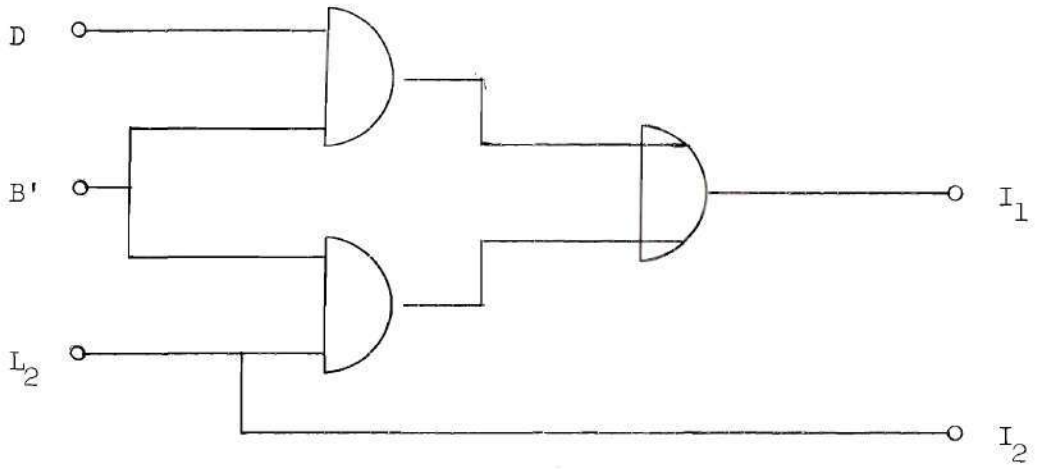
$I_1 = DB' + L_2B'$

I_2:

L_2 \ DB	00	01	11	10
0	0	Ø	Ø	0
1	1	Ø*	1	0*

$I_2 = L_2$

Careful examination of these maps shows that the functions above will give hazard-free circuits if realized in the form stated, as shown below.



Completion signal logic.--Without considering all entries in a table of combinations for the completion signal logic, conditions which give rise to a 1 for C_{out} are readily found from a word description of the desired behavior. A 1 should be obtained only if either A_i or A_f is 1 and L_1, L_2 agree with I_1, I_2. The conditions for C_{out} to be 1 are given below and an appropriate function obtained. Rather than use A_i and A_f directly, A_i' and A_f' were used. When either A_i or A_f is 1 $(A_i'A_f')'$ will also be 1. This permits the elimination of two amplifiers.

$(A_i'A_f')'$	L_1	L_2	I_1	I_2
1	0	0	0	0
1	1	0	1	0
1	0	1	0	1
1	1	1	1	1

$$C_{out} = (A_i'A_f')'L_1'L_2'I_1'I_2' + (A_i'A_f')'L_1'L_2I_1'I_2 + (A_i'A_f')'L_1L_2'I_1I_2' + (A_i'A_f')'L_1L_2I_1I_2$$

This function is the standard sum which would be obtained from the complete table of combinations for the function.

Selector control logic.--The functions for G_1 and G_2 in the selector control are derived below with the aid of Karmaugh maps in the four input variables.

RC_{out}

A_iA_f	00	01	11	10
00	1	①	0	1
01	①*	0	0	①*
11	①*	0	①	①*
10	1	1	1	1

G_i

$$G_i = C_{out}' + A_iA_f'$$

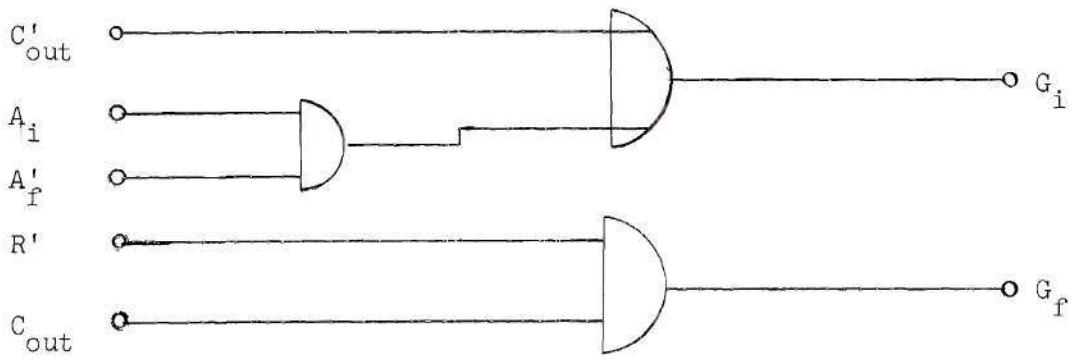
RC_{out}

A_iA_f	00	01	11	10
00	0	①*	0	0
01	①	1	0	①
11	①	1	①	①
10	0	1	0	0

G_f

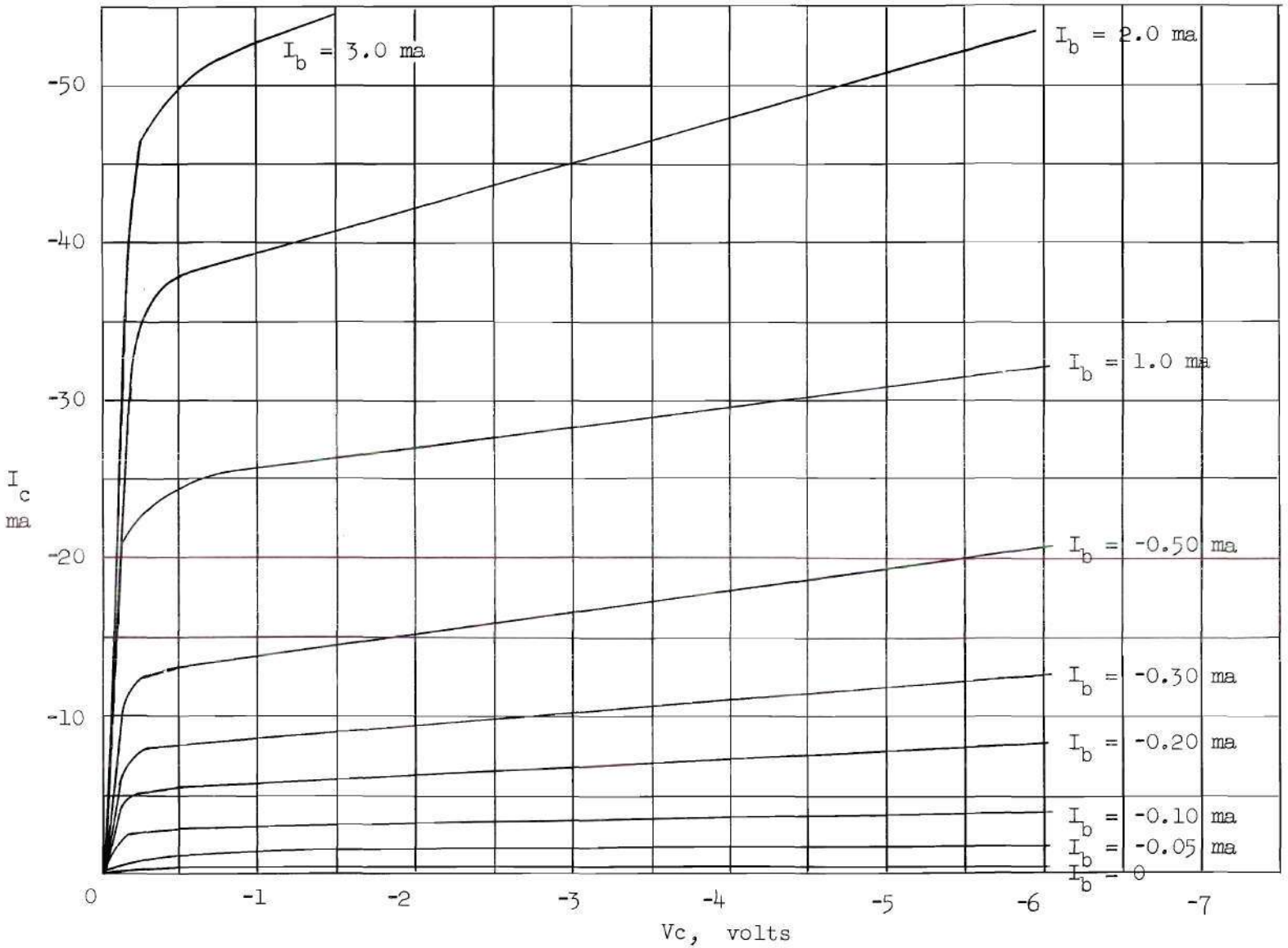
$$G_f = R'C_{out}$$

These functions may be developed by the logical diagrams shown below.

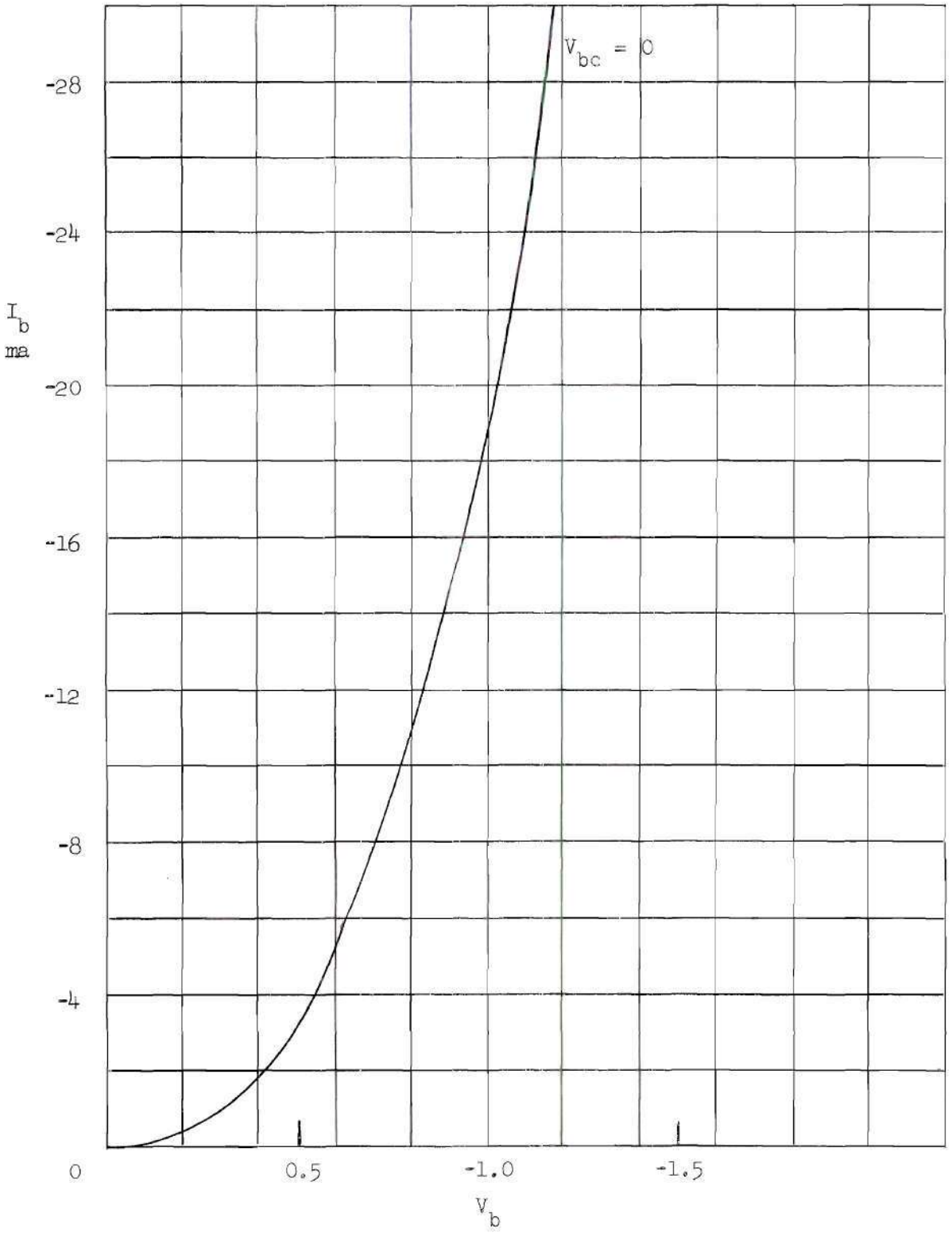


APPENDIX II

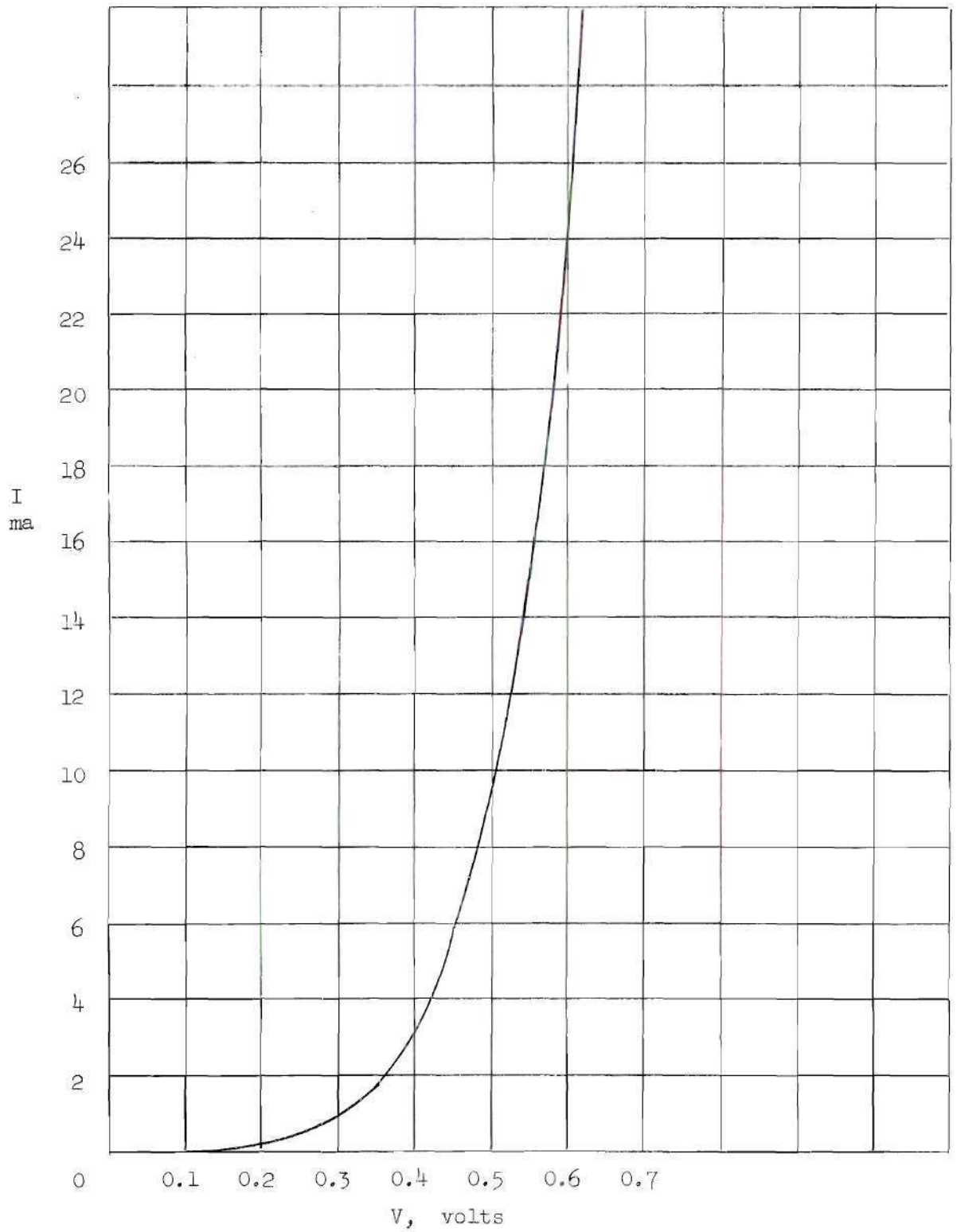
CURVES USED IN THE CIRCUIT DESIGN OF THE ONE-BIT SUBTRACTOR



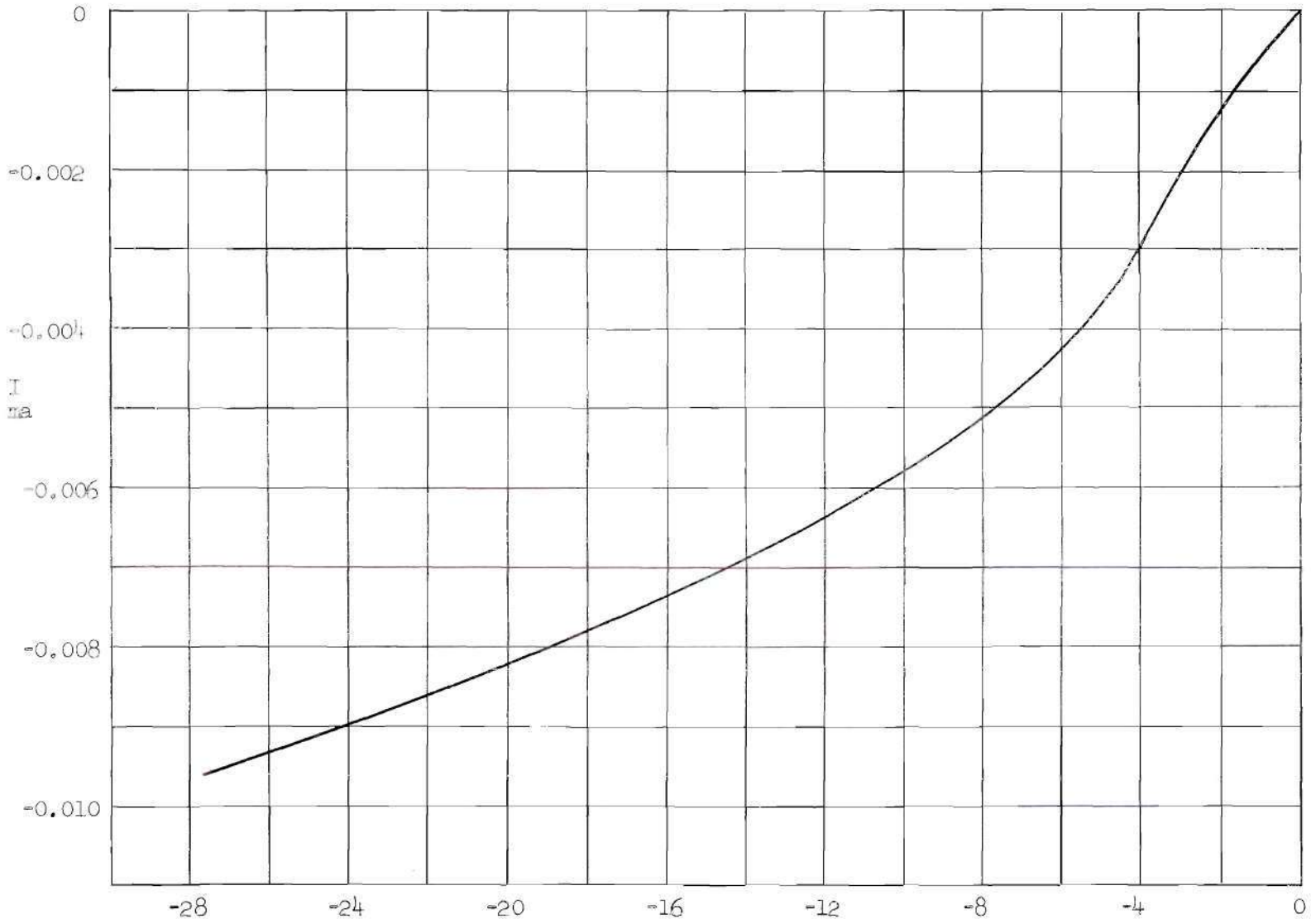
2N394 Collector Characteristics, Common Emitter.



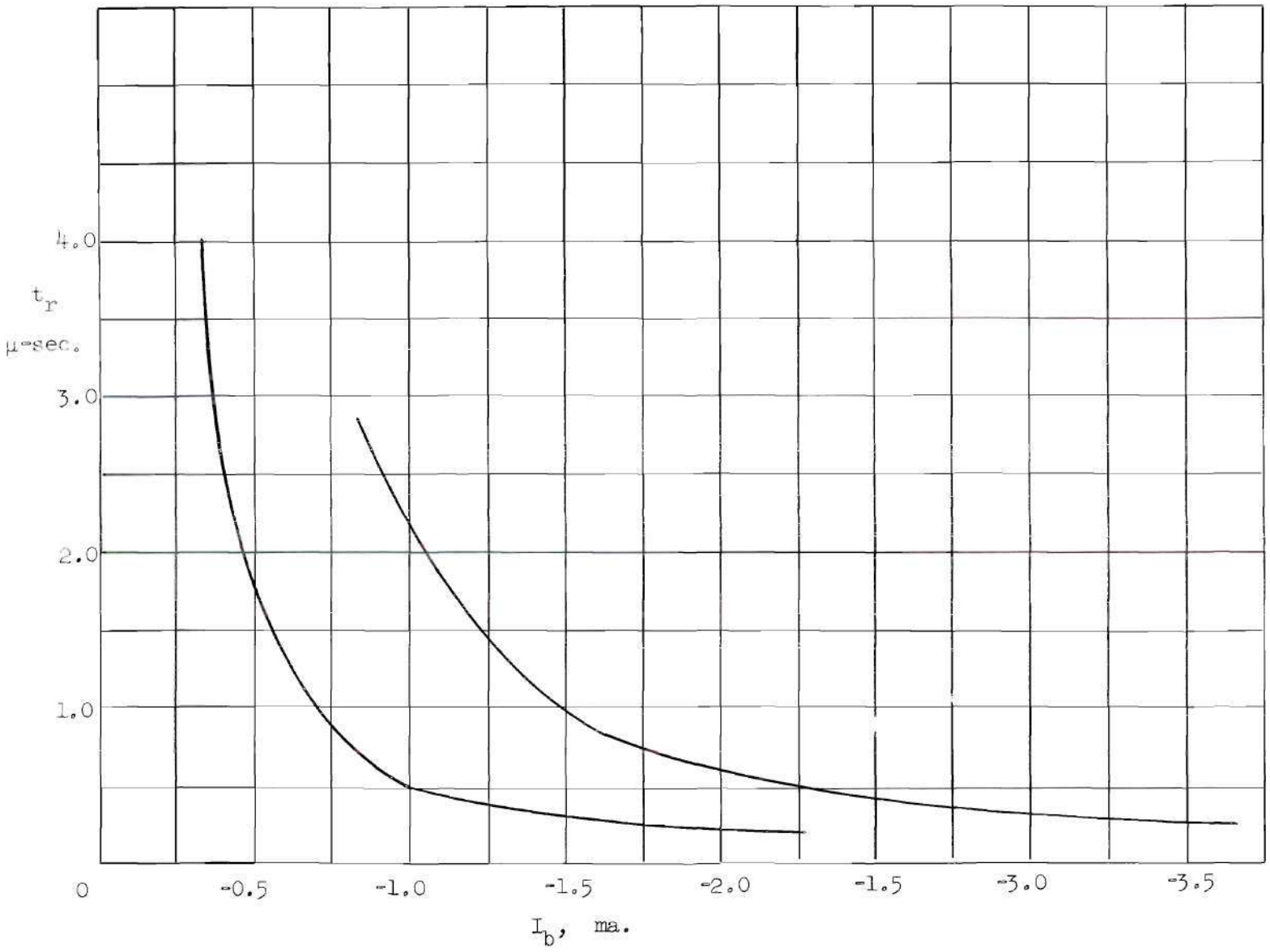
2N394 Base Characteristics, Common Emitter.



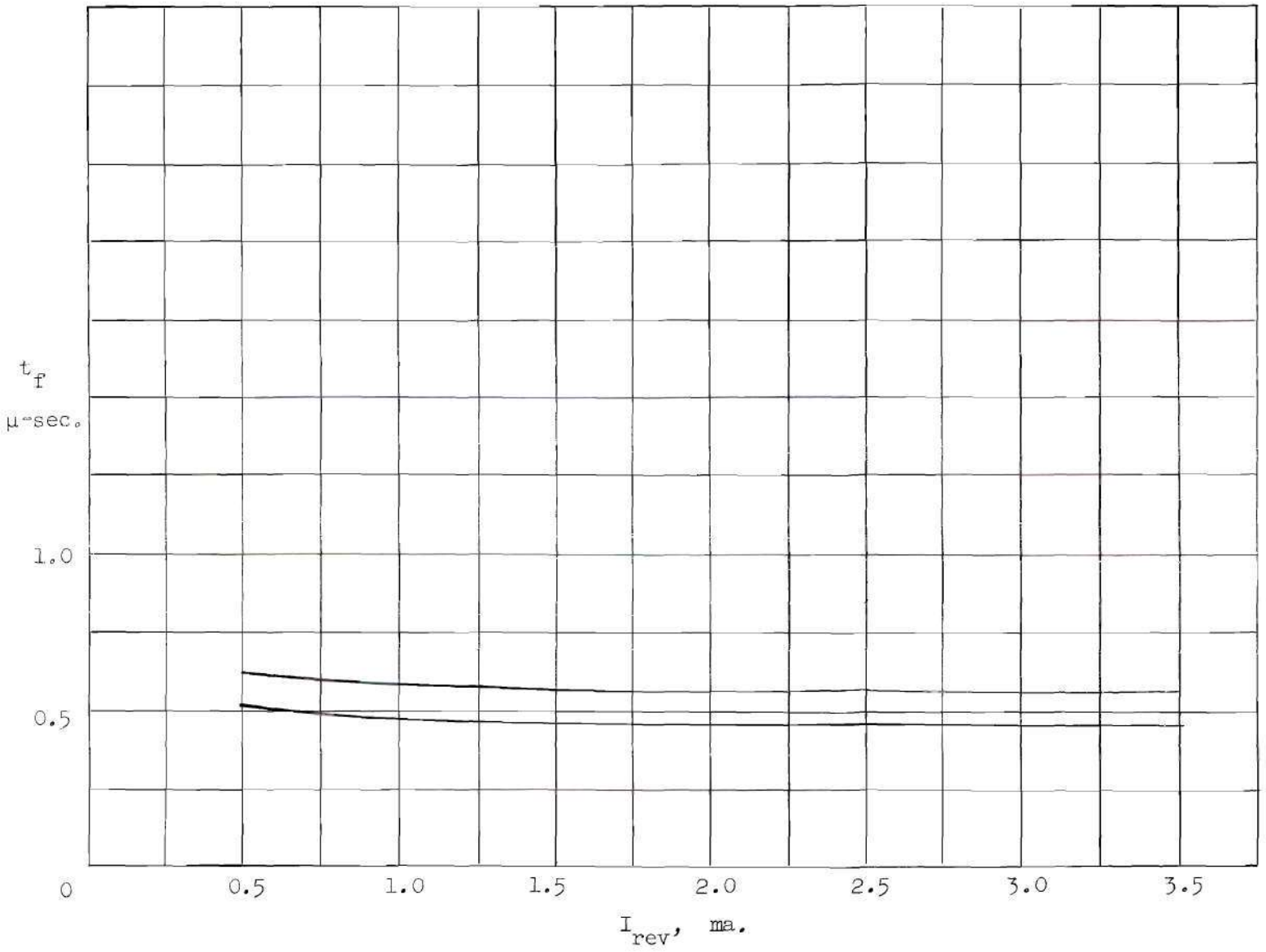
Forward Characteristics, 1N97



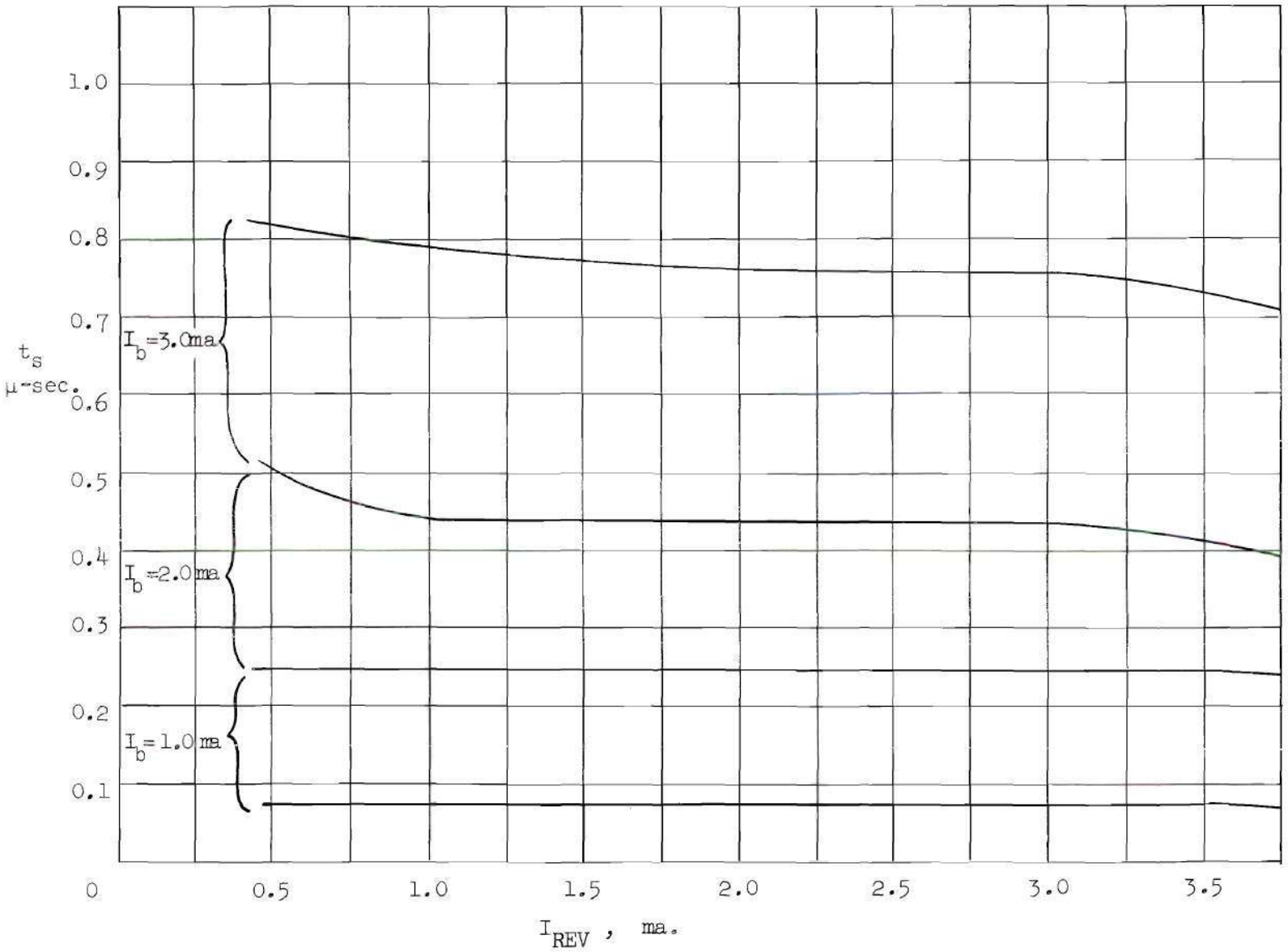
Reverse Characteristics, 1N97.



Rise time to I_c = -20.0 ma., V_c = -6.0 volts.



Active Region Fall Time from I_c = -20.0 ma., V_c = -6.0 volts.



Storage Time for Turn-off From I_c = -20.0 ma., V_c = -6.0 volts.

BIBLIOGRAPHY

References Cited in Text

1. Phister, Montgomery, Logical Design of Digital Computers, John Wiley and Sons, Inc., New York, 1958.

2. Huffman, D. A., "A Study of the Memory Requirements of Sequential Switching Circuits," Technical Report 293, Research Laboratory of Electronics, Massachusetts Institute of Technology, April 1955.

3. Unger, S. H., "A Study of Asynchronous Logical Feedback Networks," Technical Report No. 303, Research Laboratory of Electronics, Massachusetts Institute of Technology, June, 1957.

4. Muller, D. E., and Bartky, W. S., "A Theory of Asynchronous Circuits I," Report No. 75, University of Illinois, Graduate College, Digital Computer Laboratory, November, 1956.

5. Muller, D. E., and Bartky, W. S., "A Theory of Asynchronous Circuits II," Report No. 78, University of Illinois, Graduate College, Digital Computer Laboratory, March, 1957.

6. "On the Design of a Very High Speed Computer," Report No. 80, Second Edition, University of Illinois, Graduate College, Digital Computer Laboratory, October, 1957.

7. Shelly, James H., "Design of Speed-Independent Circuits," File No. 226, University of Illinois, Graduate College, Digital Computer Laboratory July, 1957.

8. Huffman, D. A., "The Design and Use of Hazard-Free Switching Networks," Journal of the Association for Computing Machinery, Vol. 4, No. 1, pp. 47-62, January, 1957.

9. Huffman, D. A., "The Synthesis of Sequential Switching Circuits," Technical Report No. 274, Research Laboratory of Electronics, Massachusetts Institute of Technology, January, 1954.

10. Caldwell, Samuel H., Switching Circuits and Logical Design, John Wiley and Sons, Inc., New York, 1958.

11. Pressman, Abraham I., Design of Transistorized Circuits for Digital Computers, John F. Rider Publisher, Inc., New York, 1959.

Other References

Grabbe, Ramo, and Woolridge, Handbook of Automation, Computation, and Control, Volume 2, John Wiley and Sons, Inc., New York, 1959.

Ledley, Robert S., Digital Computer and Control Engineering, McGraw-Hill Book Co., Inc., New York, 1960.

Poppelbaum, W. S. and Wiseman, N. E., "Circuit Design for the New Illinois Computer," Report No. 90, University of Illinois, Graduate College, Digital Computer Laboratory, August, 1959.