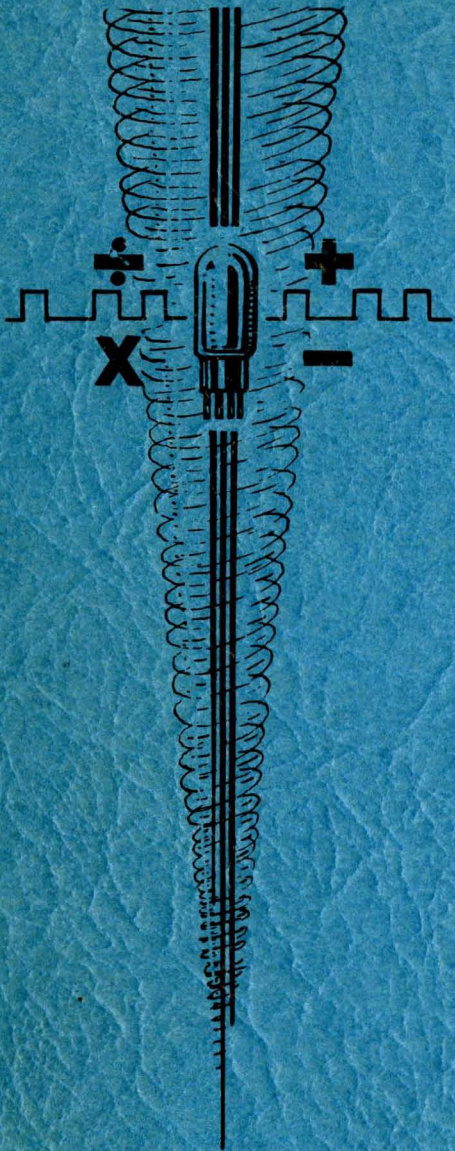


*Released*

# PROJECT WHIRLWIND

Contract N5ori60



R-1777  
A METHOD OF TEST CHECKING  
AN ELECTRONIC DIGITAL COMPUTER  
MARCH 15, 1950

SERVOMECHANISMS LABORATORY  
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Copy 37



PROJECT WHIRLWIND

Report R-177

A METHOD OF TEST CHECKING  
AN ELECTRONIC DIGITAL COMPUTER

Submitted to the  
OFFICE OF NAVAL RESEARCH  
Under Contract N5ori160  
Project NR-048-097

Report by  
Gerald Cooper

SERVOMECHANISMS LABORATORY  
MASSACHUSETTS INSTITUTE OF TECHNOLOGY  
Cambridge 39, Massachusetts  
Project DIC 6345

March 15, 1950  
(Thesis Date: January 14, 1950)

FOREWORD

Because it presents information of lasting value, this thesis report, which has had only very limited distribution, is being issued as a Project Whirlwind R-series report.

Reliability of operation is one of the most important goals of the research being done on the large-scale electronic digital computer. Part of this general problem is the development of methods of error detection and trouble location. A very useful technique adopted by Project Whirlwind is the provision of facilities for the variation of d-c supply voltages to various parts of the computer. By this means, dynamic and incipient failures can be converted into steady-state failures, and deteriorating components removed before they cause errors.

This so-called marginal checking makes use of specially designed check problems. A method of designing such problems for use with the Whirlwind I computer is developed in this thesis.

TABLE OF CONTENTS

FOREWORD . . . . .	2
ABSTRACT . . . . .	7
ACKNOWLEDGMENT . . . . .	8
ORGANIZATION . . . . .	9
1 INTRODUCTION . . . . .	10
1.1 The Automatic Digital Computer . . . . .	10
1.2 Reliability . . . . .	11
1.3 Principles of Checking . . . . .	15
1.31 Built-In Checking . . . . .	16
1.32 Programmed Checking . . . . .	17
1.4 Scope of this Thesis . . . . .	20
2 DESCRIPTION OF WHIRLWIND I . . . . .	21
2.1 Machine Units . . . . .	21
2.11 The Basic Features . . . . .	22
2.111 The Flip-Flop . . . . .	22
2.112 The Gate Tube . . . . .	23
2.113 Restoration . . . . .	24
2.114 The Electronic Switch . . . . .	25
2.12 Simple Combinations of Basic Stages . . . . .	25
2.121 The Binary Flip-Flop Register . . . . .	25
2.122 The Shifting Register . . . . .	26
2.123 The Binary Adder . . . . .	27
2.124 The Binary Counter . . . . .	28
2.125 The Time-Pulse Distributor . . . . .	29

2.13	The System . . . . .	.29
2.131	Central Control . . . . .	.30
2.132	Test Storage. . . . .	.31
2.133	Arithmetic Element. . . . .	.32
2.134	Input-Output Element . . . . .	.33
2.135	Built-In Checking . . . . .	.34
2.136	Test Control . . . . .	.34
2.137	Electrostatic Storage . . . . .	.35
2.2	Machine Cycle . . . . .	.35
2.21	Program Timing . . . . .	.36
2.22	Operation Timing . . . . .	.37
2.3	Programming and Coding . . . . .	.38
2.31	Nomenclature . . . . .	.38
2.32	The Operations . . . . .	.39
3	DISCUSSION OF FAILURES. . . . .	.41
3.1	Classification of Failures . . . . .	.41
3.11	Steady-State Failures . . . . .	.41
3.12	Dynamic Failures . . . . .	.42
3.13	Incipient Failures . . . . .	.43
3.14	Intermittent Failures. . . . .	.43
3.15	Reproducible Failures . . . . .	.43
3.16	Random Failures. . . . .	.44
3.2	Checking Considerations. . . . .	.44
3.21	Influence of Various Types of Failures . . . . .	.44
3.22	Operational Effects of Failures. . . . .	.45

	3.221 Amplifiers . . . . .	.46
	3.222 Gate Tube . . . . .	.47
	3.223 Flip-Flops . . . . .	.48
	3.224 Restorer Pulses . . . . .	.49
	3.3 Marginal Checking . . . . .	.50
	3.4 Summary . . . . .	.53
4	PRINCIPLES OF TEST SEQUENCE DESIGN . . . . .	.54
	4.1 Objectives . . . . .	.54
	4.2 History . . . . .	.55
	4.3 Proposed Procedure . . . . .	.56
	4.31 Subdivision of the Computer . . . . .	.56
	4.32 The Channel Concept . . . . .	.57
	4.33 Use of the Channel Concept . . . . .	.58
	4.34 Multiple Faults . . . . .	.59
	4.35 Trouble Location . . . . .	.61
5	EXAMPLES OF DESIGN PROCEDURE . . . . .	.63
	5.1 Single Fault Sequences . . . . .	.63
	5.11 Arithmetic Control Flip-Flops . . . . .	.63
	5.111 Channel Segment 1 - Sign Control. . . . .	.64
	5.112 Channel Segment 2 - Multiply . . . . .	.65
	5.113 Channel Segments 3A and 3B - Shift. . . . .	.67
	5.114 Channel Segment 4 - Divide . . . . .	.68
	5.115 Channel Segment 5 - Special-Add . . . . .	.71
	5.116 Channel Segment 6 - Point-Off . . . . .	.72
	5.117 The Complete Sequences for Arithmetic Control Flip-Flops . . . . .	.72
	5.12 Input-Output Element . . . . .	.72

5.121	In-Out Register. . . . .	74
5.122	In-Out Control Flip-Flops. . . . .	77
5.13	Step Counter. . . . .	79
5.131	Step Counter Flip-Flops. . . . .	81
5.132	Step Counter Gate-Tubes. . . . .	84
5.2	Multiple Faults - The Program Counter . . . . .	85
5.3	Trouble Location - Control-Pulse Out-ut Units . . . . .	91
6	RESULTS AND CONCLUSIONS . . . . .	95
6.1	Experimental Results . . . . .	95
6.11	Discussion of Low Margin on Line 113. . . . .	97
6.12	Discussion of Low Margin on Line 79 . . . . .	98
6.2	Evaluation of the Design Procedure . . . . .	99
6.3	Suggestions for Further Work. . . . .	101
APPENDIX I - A Short Guide to Coding . . . . .		102
APPENDIX II - Details of Coded Programs . . . . .		105
APPENDIX III - Glossary . . . . .		110
BIBLIOGRAPHY . . . . .		111
LIST OF ILLUSTRATIONS . . . . .		113
ILLUSTRATIONS . . . . .		114

ABSTRACT

This thesis study describes a method of designing check problems to be used in detecting errors and locating trouble in a large-scale digital computer. The computer is systematically subdivided into small sections. A study is made to determine the paths (channel segments) in these sections along which information is routed, as well as the sources, destinations, and methods of transmission of the information.

One of these methods is selected for the routing of each type of information from one of the sources to one of the destinations of each channel segment. The destination is then examined to see whether it received the proper information.

This process, used in conjunction with the Project Whirlwind method of marginal checking, permits a sequence of instructions to the computer to be specified. Limited tests in the Whirlwind I computer indicate that test sequences designed by this method will be quite useful.



ACKNOWLEDGEMENT

The author wishes to express his sincere thanks to Mr. N. H. Taylor for supervising this thesis and making many helpful suggestions for the conduct of the work and the preparation of this report. He is also indebted to Mr. J. W. Forrester for making available the facilities of Project Whirlwind for the preparation of this thesis. Thanks are also due to the personnel of Project Whirlwind with whom the author had several stimulating and helpful discussions, particularly to Mr. R. R. Everett, Mr. C. W. Adams, Mr. J. V. Salzer, and Mr. R. P. Mayer.

ORGANIZATION

This thesis deals with a very specialized topic which requires considerable background material for its complete treatment. The first three chapters provide most of this material while developing some of the bases for the method presented. In detail, the material is arranged as follows:

Chapter 1 presents the problem of checking digital computers and indicates some possible solutions.

Chapter 2 is a fairly detailed description of the operation of WMI.

Chapter 3 discusses the nature of failures and their influence on checking procedures.

Chapter 4 presents a systematic approach to designing special problems to check the operation of the computer.

Chapter 5 gives a number of examples of the applications of the approach to checking WMI.

Chapter 6 discusses and evaluates the results of the thesis.

Appendix I is a short guide to coding prepared by the Electronic Computer Division of the Servomechanisms Laboratory, M.I.T.

Appendix II contains the coded programs which have been developed by the method described in this thesis.

Appendix III contains a glossary of special terms used in this thesis.

1 - INTRODUCTION1.1 The Automatic Digital Computer

The automatic digital computer is a tool for the solution of numerical problems. It is capable of performing entirely automatically any desired sequence of arithmetic operations on given data. This is in distinction to analogue computation where the data is handled as physical quantities in a system obeying physical laws having the same form as the operations involved (see Fig. 1). It also has the ability to choose among several alternate computing routines on the basis of its own calculations. A wide variety of problems can be solved through the use of the automatic digital computer because of the great flexibility implied in these properties.<sup>13\*</sup> The high calculating speed which is characteristic of the automatic digital computer, makes it particularly valuable because of the great number of problems whose solution by more customary means, though possible in principle, is impossible in a practical sense because of the prohibitive amount of time and labor involved.

As one might suspect, a rather complicated device is required to accomplish these results. For example, Whirlwind I (WWI), the electronic digital computer under development at the M.I.T. Servomechanisms Laboratory, consists of approximately 4500 vacuum tubes plus associated components such as resistors, capacitors, inductors, and crystal rectifier diodes. Good computer design requires that each individual component be operated under conditions that ensure as long a life as possible.<sup>10</sup> Nevertheless, each

---

\* Superscripts refer to the numbered bibliography.

component will eventually fail to operate satisfactorily. When such a failure occurs, it is quite likely, though by no means certain, that an error will appear in the computed results. Such errors, if they remain undetected, may cause considerable damage. There is no longer any doubt that computers can be made to work; the tables of functions calculated at the Harvard Computation Laboratory have amply demonstrated this fact. The question is: how consistently will they provide correct results? Reliable results must be obtained if the latent value of the automatic digital computer is to be realized in practice.

## 1.2 Reliability

It has been established that erroneous results are undesirable, but they are also inevitable. This is not just a peculiarity arising from the use of the automatic digital computer; other methods of problem solution are also subject to error. What level of error occurrence can be tolerated? Before this question can be answered quantitatively, a considerable amount of statistical data must be accumulated. It will be difficult to interpret such data on a comparative basis because of the speed differences among various methods. The amount of computation which can be done during the period between errors must be taken into consideration in some suitable manner. Thus far, only qualitative reasoning has been applied to this problem. However, one error per week has been suggested by a leading authority<sup>\*</sup> as a reasonable goal for a computer of the same nature as Whirlwind I.

---

<sup>\*</sup> This statement was made by J. W. Forrester during a discussion of his talk at the Harvard Symposium on Large-Scale Digital Calculating Machinery on September 13, 1949. The proceedings of this symposium have not yet been published.

The fact that errors will occur does not mean that we must be resigned to accept erroneous results. There are many methods for checking on the validity of calculated results (see section 1.3 and references 6 and 9). When it is known that a particular result is in error, it is usually possible to remove the source of the error and then proceed to calculate the correct value. Thus, by using these methods, the effective reliability of the computer can be increased tremendously, although the speed is reduced somewhat. Of course, there are a number of applications in which time is an important factor so that there is no time available for a second calculation under error-free conditions. Such applications will require very special techniques to achieve satisfactory reliability, but error detection is essential even then.

It must be realized that it is quite impossible to detect all errors, for it is certainly conceivable that two failures can occur simultaneously in such a way as to nullify the method being used. This is not to say that there is no method for detecting errors caused by two compensating, simultaneous failures. It is always possible, at least in principle, to devise a scheme which will detect any specified combination of faults, but in the most general case, additional equipment would be required. This additional equipment would give rise to new combinations of faults which could not be detected without further resort to extra equipment. For a given computer, there will always be some combinations of faults which cannot be detected without changing the computer. It becomes necessary to decide upon an adequate degree of error detection. Here the economics of the situation is an important factor. Since any desired level

of checking can be provided at the cost of either money or time, some balance must be achieved between conflicting interests. It seems to be quite generally agreed that the probability of an undetected error occurring in a machine which is checked in all cases of single failures is sufficiently low. However, any method which provides a check in many cases of multiple faults at little extra cost should be carefully considered.

As mentioned previously the time required to detect an error by a given method is frequently of vital importance in applications where time is a factor. These may be exemplified by the problem of directing the flow of air traffic in accordance with the specified schedules and the actual positions and velocities of the aircraft involved.<sup>13</sup> The computer is used to correlate this data and supply the appropriate instructions to the pilots of the aircraft regarding the flight pattern to be maintained. When these instructions are in error, the pilots and the operator of the control tower should be informed immediately so that appropriate emergency measures may be taken. The importance of instantaneous error detection is quite evident in this case. Indeed we may even look askance at the fact that it is possible for some errors to go undetected. However, we must remember that it is quite possible for the aircraft itself to fail. This is a very special case and the accuracy requirements imposed are far greater than would be encountered in most ordinary applications. Even where the danger to human life is not so immediate, speed in error detection is quite important. A considerable amount of erroneous computation might be accumulated before an error is detected. This will be wasteful, but it is again difficult to pass other than a qualitative judgment on the value of speed in error detection. In general, speed is achieved at the cost of equipment

and additional equipment is another source of trouble, aside from the monetary aspects involved. It may be said, with little fear of contradiction, that it is preferable to save equipment at the cost of time, all other factors being equal.

Once a failure is known to exist, the next step is to repair it. Before any repairs can be carried out, the faulty component must be located. Conventional methods of trouble location in small pieces of equipment are completely inadequate for a job of this magnitude. Special automatic methods must be devised to isolate the faulty component. It is not necessary to locate the specific component by these automatic means, but the fault should be isolated to a unit in which conventional techniques may be employed in a reasonably short time. A computer which makes one error a week cannot be said to be very reliable if it takes a week to locate its source. In that time, another error may have occurred and another week of trouble location is in prospect. Under such conditions, all of the operating time of the computer would be taken up in trouble location. The useful output would be practically nil. It does not seem unreasonable to expect that the computer perform useful work during at least half the time it is operating. Indeed, this would be a rather poor computer. On the other hand, it takes a finite amount of time to make the actual repairs. Experience with Whirlwind I indicates that this time is of the order of magnitude of 15 or 20 minutes, on the average. It is pointless to go to great lengths to develop an automatic trouble location scheme which isolates the fault within a few seconds.<sup>20</sup> A perfectly satisfactory method would be one which located the faulty component in less than 40 minutes. This would mean that approximately an hour would be required

to put the machine in operating condition after an error has been detected. Less than 1% of the operating time of the computer (assuming one error per week) would be lost for this reason.

Heretofore the term checking has been applied to methods for the detection of errors in the operation of the entire computer. When an error is detected by such a method, the faulty component is known to be somewhere in the computer. Thus a measure of trouble location has been achieved, even though it is of dubious value. When an error is indicated by a method devised to check a relatively small unit, the faulty component is known to exist in that unit. A large measure of trouble location has been achieved. There is no difference between checking and trouble location. The distinction is one of degree rather than kind.

### 1.3 Principles of Checking

Basically, the problem to be solved by the various checking methods is that of providing a suitable standard to compare with the results obtained from the unit being checked. The choice of the size of the unit to be checked depends on the degree of trouble location that is desired. It is clear that if no fault exists in any of the small units which comprise the computer, then no fault exists in the entire computer. Thus, if a scheme is devised to check every small unit of the computer, including all their interconnections, then this scheme checks the entire computer. The various methods which have been suggested thus far differ in the flexibility with which they may be applied units of varying size.

There are essentially two different approaches to the problem of generating a comparison standard: built-in checking and programmed



checking. Additional equipment generates this standard in built-in checking schemes. In programmed checking, the computer treats the calculation of this standard as another problem which it can solve. Built-in checking permits higher computing speeds but requires more equipment than programmed checking. More than one type of checking is frequently used in a computer. The amount of each type is dictated by economic considerations.

### 1.31 Built-In Checking

Perhaps the most obvious way of checking a unit is to provide an identical one to duplicate its function. The results obtained by the duplicate unit are a standard with which to compare those of the original unit. A special unit is provided to make this comparison. If an error is detected, it may have been caused by a failure in either unit. It is also possible (though improbable) for the same failure to occur in both units simultaneously, giving a correct check in the presence of an error. This has been discussed in section 1.2 and is not a serious objection. It is, of course, possible to provide three or more identical units to guard against such double failures. However, none of the existing or projected computers has incorporated more than two sets of identical equipment because of the cost involved. Checks of this type are known as multiple checks.<sup>9</sup> Multiple checks are generally used for large units; the number of comparison units required when small units are involved is felt to be excessive.

A more efficient use of checking equipment is achieved by a comparison of some identifying characteristic (or tag) of the actual and

expected results rather than the results themselves. Such checks are called tag checks; "casting out nines" is a tag check used in decimal multiplication. In some very special cases, certain characteristics of the result of an operation may be independent of the numbers involved. It is frequently advantageous to provide a check which takes cognizance of the situation. An example of such a predetermined check is given in section 5.12.<sup>6</sup>

A more detailed discussion of built-in checking may be found in reference 6.

### 1.32 Programmed Checking

The process by which a problem is stated in a form suitable for solution by a computer is called programming. Thus, those checking methods which treat the generation of a comparison standard as a problem to be solved by the computer are referred to as programmed checking. There are two approaches possible to programmed checking: mathematical checks and test checking.

Mathematical checks are based on some identity satisfied by the results of a calculation. Checking procedures based on these identities are applicable to all computers for they are essentially another group of problems which are within the problem-solving capabilities of digital computers. There are certain types of identities which are encountered so frequently that names have been assigned to the checking procedures which are based upon their use. However, there is undoubtedly a group of miscellaneous relationships which hold in special cases that cannot be fitted into such a classification. These miscellaneous techniques

depend largely on the alertness and inequity of the programmer and cannot be discussed here.

The first of these schemes is known as repeat checking, wherein the solution of a problem is repeated a specified number of times and both (or more) sets of results are compared.<sup>9</sup> In identity checking, two (or more) equivalent mathematical procedures are used to obtain two (or more) sets of results for comparison.<sup>9</sup> With some problems, a rather good check may be obtained by reversing the process of problem solution and finding the given initial data from the alleged solution, yielding what is known as inverse checking.<sup>6</sup> In sample checking, the solution to a general class of problems is obtained for a special case in which the answer is known; the correct solution indicates that the computer is probably capable of solving this general class of problems.<sup>6</sup> It is possible to examine the solutions of problems which are known to lead to continuous functions for continuity, thus giving rise to smoothness checking.<sup>9</sup>

On the whole, mathematical checks are applicable to large groups of units and hence are not particularly useful in trouble location. They vary in effectiveness with the types of failures that exist, but they are quite useful for an overall check on a specific problem. It is not necessary to know that the whole computer is working to be reasonably sure that the results are correct. Mathematical checks are discussed more fully in reference 6.

In test checking, the results obtained from the solution of specially designed problems are compared with the known answers. Techniques for designing such problems have not yet been fully developed.

Most of the work in this field has been of an exploratory nature.<sup>2,3,4,5,8,21</sup>  
The several approaches which have been suggested are applicable only under very restricted conditions. The most severe shortcoming of some of these is that they make use of single functions in sequences which cannot be performed by the computer. Each operation that the computer performs consists of a predetermined sequence of such functions; the programmer may choose the sequence of operations arbitrarily, but he cannot do so with the functions. Some of the approaches which do not suffer from this shortcoming may be quite useful in specific cases, but they lack generality. This thesis represents an attempt to develop a systematic approach (for use with WVI) which is generally applicable to the design of such problems. The resulting problems will be referred to as test sequences.

To some extent, programming is still an art in which experience and ingenuity are valuable assets. Systematic approaches to programming have been proposed which are quite good, but a completely satisfactory method has not yet been found.<sup>14,16,22,23</sup> In many cases, much more efficient programs may be obtained more easily by methods other than a literal application of the general method. This would seem to imply that a systematic method of test checking is not necessarily the best method in a specific case. Indeed, this thesis contains examples in which departures from the general method have been made in the interests of simplicity and efficiency. In short, the proposed method is only to be used where simpler methods are not immediately obvious.

#### 1.4 Organization of This Thesis

A large amount of detailed information about the computer (KWI) is required to thoroughly explain many of the examples of this thesis. The paper would become quite unwieldy if all this detail were included. Much of the information found in some of the references has therefore been omitted.<sup>12,16,19</sup> However, in order to make this paper reasonably self-contained, a considerable amount of material has been included. The procedure followed in this respect has been to give fairly detailed explanations of the basic elements, followed by examples to illustrate the manner in which more complex details can be explained in terms of the basic elements. The same procedure has been followed in explaining the examples of test sequence design.

Many new terms and concepts which are in standard usage at Project Whirlwind are defined in reference 15. Every effort has been made to define such terms when they are first used. As a matter of convenience, additional special terms have been adopted for some of the concepts arising in checking. These are defined when they are introduced and have been collected for easy reference in Appendix III. Some, but not all, of these terms have come into standard usage at Project Whirlwind and elsewhere.

## 2.1 Machine Units

Whirlwind I (and, indeed, every computer) may be thought of as being divided into four main functional units: the memory (or storage), the control, the arithmetic element, and the input-output device (see Fig. 2). The memory stores the numerical data required for the solution of problems, as well as the instructions (coded into numerical form) which govern the actual steps of problem solution. The control interprets the instructions and provides the machine with the signals needed to perform them. The arithmetic element actually performs certain elementary arithmetic operations (addition, subtraction, shifting, multiplication, and division) whenever the control supplies the appropriate command signals. The input-output device is the link between the machine and the outside world; it provides the machine with raw data and receives the computed results. It is frequently used to supplement the memory by connecting a large-capacity external memory (having a slower operating speed) to the machine.

Two different devices are used for the *internal* memory of WWI. These are the flip-flop and the electrostatic storage tube. The flip-flop is discussed in section 2.111. The electrostatic storage tube is a cathode-ray type tube in which the numbers are stored as charged spots on a dielectric surface. The final model of this tube is expected to have a capacity equivalent to  $10^{24}$  flip-flops, thus saving a considerable amount of space. Its speed of operation is somewhat slower than that of the flip-flop, but it is faster than magnetic tape or drum memories. The

control contains a source of pulses which are routed to the rest of the computer in accordance with the settings of electronic switches. Flip-flops are used to manipulate the numbers in the arithmetic element. A great diversity of external equipment may be connected to the computer by means of the input-output element. Photographic film is expected to be the main external storage medium, but it will be possible to use magnetic tape as well. It is expected that conversion devices will be provided so that analogue information may also be handled.

### 2.11 The Basic Features

With few exceptions, three basic stages are used throughout the computer; the flip-flop, the gate tube, and the electronic switch. Of course, amplifiers are provided where needed to obtain appropriate signal or impedance levels. The flip-flop is essentially a memory device. The gate tube is a sensing device, usually employed to sense the contents of a flip-flop. The electronic switch is what its name implies: a multi-position switch which operates at high speeds.

#### 2.111 The Flip-Flop

The flip-flop is a bi-stable circuit consisting of two vacuum tubes so interconnected that, at any time, one is fully conducting and the other is completely cut off (see Fig. 3). The plate potential of the cut off tube is "high" (nearly equal to the plate supply voltage), while that of the conducting tube is "low" (considerably less than the plate supply voltage, the difference being the drop in the plate load resistor). If tube A (see Fig. 3) is cut off, the potential at the 0 output is high, that at the 1 output is low, and we say that the flip-flop contains 0,

or is cleared. If tube *R* is cut off, the 0 output is low, the 1 output is high, and we say that the flip-flop contains 1, or is set. By applying a positive pulse at an appropriate point in the circuit (the trigger input), the flip-flop may be made to switch its state, i.e., if it formerly contained 0, it will now contain 1, or if it formerly contained 1, it will now contain 0. This action is referred to as triggering or complementing. There are two additional inputs which require negative pulses. One is the clear input, which puts the flip-flop in the clear condition regardless of its condition before the pulse was applied. The other is the set input, which puts the flip-flop in the set condition regardless of its previous content. The flip-flop is often represented symbolically by a rectangle having three input lines and two output lines, as is also shown in Fig. 3, and is abbreviated FF. Each FF in WWI is designated by a system number.

#### 2.112 The Gate Tube

The gate tube is a coincidence device: it provides an output if, and only if, two signals occur simultaneously. Its basic element is usually a pentode, with one signal being applied to the control grid, the other, to the suppressor grid (see Fig. 4). These two grids are biased sufficiently to maintain the tube cut off even when full signal is applied to one (but only one) of them. As mentioned above, its principal use is in sensing the content of a flip-flop. If it is desired to know if the flip-flop contains 1, we merely connect the 1 output to the suppressor of the gate tube through a suitable coupling circuit. This coupling circuit must change the d-c level of the output but must preserve the distinction between the two possible voltage levels (see section 2.113). If the



If the 1 side is high, the suppressor potential will be brought above cut-off, and there will be an output pulse when the control grid is pulsed. The gate is said to be open. If the 1 side is low, the suppressor will remain below cut-off, and no output is obtained when the control grid is pulsed. The gate is said to be closed. Thus, the presence of a pulse on the output line at the time of sensing indicates that the flip-flop contains 1, while the absence of a pulse at this time indicates that it contains 0. The gate tube is represented symbolically by a square having two input lines and one output line, as shown in Fig. 4, and is abbreviated as GT. Each GT in WVI is designated by a system number.

### 2.113 Restoration

Since the flip-flop may remain in one position indefinitely, its output takes on the characteristics of a d-c signal. Rather than use d-c coupling circuits, the scheme of periodically complementing and re-complementing the flip-flops was adopted to permit the use of a-c coupling circuits. This scheme is called restoration and the pulses which are used to complement and re-complement the flip-flops are called restorer-pulses. The restorer-pulses occur in pairs with an interval of 1  $\mu$ sec. between them. These restorer-pulse pairs are repeated at a fixed interval (except during the operation of electrostatic storage) of 16  $\mu$ sec. between the first pulses of successive pairs (the restorer interval). The supply of pulses to the computer is suppressed during the interval occupied by the restorer-pulse pairs, so that operations are not carried out while the flip-flops contain the complements of the numbers they are supposed to hold.

### 2.114 The Electronic Switch

The electronic switch causes only one of  $2^n$  gate tubes to be open, depending on the contents of  $n$  flip-flops (see Fig. 5, drawn for  $n = 2$ ). Plate current for the flip-flop tubes must pass through the diode matrix and all but one of the resistors ( $R_1, R_2, R_3$ , and  $R_4$ ). If, for example, the right hand tubes of both flip-flops should be conducting, current will be drawn through  $R_2, R_3$ , and  $R_4$ . No current will flow through  $R_1$  because it is connected only to the left-hand tubes. Thus the upper line will be at a positive potential with respect to the others and the gate tube connected to it will be open, while all others will be closed. Similarly, a different line will be placed at a positive potential with respect to the others for each different combination of flip-flop contents. An alternate form of electronic switch, using gate tubes rather than a diode matrix, is frequently used.

### 2.12 Simple Combinations of Basic Stages

There are many possible ways of combining the basic stages to obtain devices capable of performing the functions required in a computing machine. To illustrate these possibilities, some of the more commonly encountered combinations will be discussed below. The symbols used in the various diagrams are explained in Fig. 6.

#### 2.121 The Binary Flip-Flop Register

A collection of  $n$  flip-flops is essentially a memory device capable of storing an  $n$ -digit binary number. By permanently assigning each flip-flop to a particular digit column of the number, the so-called binary flip-flop register is obtained. Three digit columns of a typical

register, the A-register (AR), are shown in Fig. 7. Three sets of gate tubes are provided so that the number in the register may be examined, two sets on the 1 side, one set on the 0 side. A pulse applied to one of the sets on the 1 side will cause the number contained in the A-register to appear on the output lines of that set. One set is used to transfer the number in the A-register to the Accumulator (GT05, when the add line is pulsed), while the other set transfers this number to the "bus" (GT02, when the read-out line is pulsed). The bus may be connected to any of several units. The set on the 0 side is used to obtain the "nines-complement" of the number (obtained by replacing each 0 by 1 and each 1 by 0, see Appendix I) for use in subtraction (GT04, when the subtract line is pulsed). The read-in gate tubes (GT01) serve as switches through which the bus may be connected to the input of the A-register when a gating signal is applied. It might be remarked that the flip-flops associated with the electronic switch in reality form a binary register.

### 2.122 The Shifting Register

A register in which provision has been made to shift its contents either to the left or to the right is called a shifting register. Three digit columns of a typical shifting register, the B-register (BR), are shown in Fig. 8. The facility to shift the content of each digit column one digit column to the left is provided by GT04 and GT05. The facility for shifting right is provided by GT06 and GT07. The mechanism of shifting is quite simple. Consider shift left. If a particular digit column contains 1, a pulse will appear on the output of GT04 when the shift left line is pulsed. After a slight delay, the flip-flop of the

digit column to the left will be set by this pulse. If the digit column had contained 0, a pulse would have been obtained on the output of G105 (instead of G104) which would clear the flip-flop to the left. Hence, the content of the original digit column would appear in the next digit column on the left. This process is carried on simultaneously for all digit columns, with some special conventions adopted to handle the columns at each end of the register.

### 2.123 The Binary Adder

All arithmetic processes in WWI are performed as a combination of additions and shifts. The device which performs the additions in WWI is called the Accumulator (see Fig.9). The Accumulator (AC) is capable of a number of operations other than addition, but we will only discuss it as a binary adder. Consider only FF01 in the column labeled AC15 (partial sum) and FF02 in the column labeled AC14 (carry). Only G106 of AC15 (partial sum) and G112 of AC14 (carry) are of interest. First let us examine the binary addition table:

0	0	1	1
$\frac{+0}{0}$	$\frac{+1}{1}$	$\frac{+0}{0}$	$\frac{+1}{10}$

Assume that a number is already in AC and that the number contained in AR is to be added to it. Only one digit column will be discussed (AC15); the same process takes place simultaneously in each digit column. When the add gate tubes of AR are pulsed, the number will appear on the lines labeled "from AR". Assume AC15 (FF01) contains 0 initially. If the number being added in is 0, no pulse will appear at the input ("from AR") no change occurs and FF01 continues to contain 0, as it should. If the number being added was 1, a pulse appearing at the input will be applied

to GT06, but it will find GT06 closed. After a slight delay, the pulse will be applied to the trigger input of FF01 causing it to change its content to 1. Thus the correct result is also obtained for this case. Now assume that AC15 (FF01) contained 1 initially. If the number being added in is 0, no change occurs; FF01 continues to contain 1, as it should. If the number being added in is 1, the pulse will find GT06 open and will proceed to set FF02 of AC14. After a slight delay, the pulse will trigger FF01 of AC15 to 0. This, again is the correct result, i.e., the sum is 0, with 1 to carry. The carry is subsequently added in to the next digit column by applying a pulse to the carry line. This uses GT12 to sense the carry flip-flop (FF02). This second addition may also result in a carry, which would require another carry pulse, and so on. However, a shorter method has been devised which requires only one carry pulse. The method utilizes GT05 and is called the high-speed carry. It will not be discussed further.

#### 2.124 The Binary Counter

The binary counter is just what its name implies: it counts the number of pulses applied to its input, indicates this number in the binary system and supplies, if desired, an output pulse when its capacity is exceeded, the counter being returned to 0 when this occurs. A typical binary counter, the Program Counter (PC), having a maximum capacity of 2048, is shown in Fig. 10. Assuming each flip-flop is initially clear, two input pulses must be applied to any flip-flop in order to obtain an output pulse from the associated GT05. Thus, two output pulses are required from PC6 in order to obtain an output pulse from PC5. This, in turn, requires four pulses from PC7, which in turn, requires eight

pulses from PC8, etc. The capacity of the counter is increased by a factor of two for each flip-flop stage. The number of pulses the counter has received is indicated at any time by the contents of the flip-flops, FC15 giving the  $2^0$  digit, FC14, the  $2^1$  digit, etc., with FC5 giving the  $2^{10}$  digit. It is possible to preset this counter so that the overflow pulse is obtained after any predetermined number of pulses (less than 2048, of course).

### 2.125 The Time-Pulse Distributor

It is frequently necessary to perform several different elementary steps in proper time sequence in order to build up a more complex operation. Thus, if we have a supply of timing pulses, we want the first one to go to one part of the computer, the second one to another part, the third one to still another, etc. The time-pulse distributor accomplishes this conversion from a time distribution to a time and space distribution. It is a combination of a binary counter and an electronic switch (see Fig. 11). The output lines are selected in accordance with the contents of the counter. The input pulse, in addition to performing the counting, emerges on the selected line. There is no need for the overflow pulse in this type of operation so that no provision has been made for it.

### 2.13 The System

A somewhat more detailed block diagram of WWI appears in Fig. 12. The interconnections among the various units are shown in detail. A 16-digit binary number will be referred to as a word. A word may represent an instruction or an actual numerical value. Words are transferred from one part of the system to another via a set of 16 cables which are

connected to the units via gate tubes. All digits are transferred simultaneously (parallel digit transmission) with a separate cable being required for each. A word is transmitted to the bus from a particular unit by pulsing its read-out gate tubes. A word which is on the bus will be transmitted to any unit whose read-in gate tubes are open. Another set of 16 cables (the "check bus") is provided as a multiple check on the bus. By a suitable arrangement of the timing, it is possible to check considerably more than the condition of the bus. The other connections, which transmit pulses that control the performance of the various steps of the operations, are made directly. Such pulses are called commands.

### 2.131 Central Control

Central control contains most of the units which are used to interpret the instructions for the solution of a problem (see Fig. 13). It also contains a source of pulses which eventually become commands. The master clock supplies these pulses in various forms. The basic pulse source is the pulse generator which supplies high-frequency clock pulses (HFCP) at a 2 mc. rate and low-frequency clock pulses (LFCF) at a 1 mc. rate. The frequency divider is used in conjunction with the restorer-pulse generator (RPG) to generate the restorer pulses. Clock-pulse control (CPC) determines which pulses will be supplied to the remainder of the computer during such times as restoration and the operation of an auxiliary control. The synchronizer provides single pulses at the control of a rush button. The bulk of the operations are carried out by the time-pulses which appear on successive lines at intervals of 1  $\mu$ sec. (the interval is longer during restoration and the operation of an auxiliary control). These pulses are provided by the time-pulse distributor (TFD)

(see section 2.125). The program counter (PC) is the device which contains the location of the next instruction to be performed. The control switch (CS) determines which operation is to be performed by energizing the appropriate line of the control matrix (CM). Each line of the control matrix opens certain gate tubes (control pulse output units). Each of these gate tubes is sensed on a certain time pulse (in some cases, on two time pulses). The pulses which emerge from these gate tubes are the commands. Thus by setting the control switch to a given position, the sequence of commands required to perform the operation determined by that position is obtained. A different sequence of commands is obtained by changing the contents of the control switch. The program register (P) is a buffer register which serves two different purposes, to be discussed in sections 2.137 and 2.21. Storage selection control will be discussed in section 2.137.

#### 2.132 Test Storage

Part of the memory of WVI consist of 5 flip-flop registers and 27 toggle-switch registers (each capable of holding a word). These registers are known as test storage (TS). The 5 flip-flop registers may receive and transmit words, but the 27 toggle-switch registers can only transmit words. A switch must be manually operated in order to change the contents of a toggle-switch register. The registers of test storage are numbered from 0 to 31; the number assigned to a particular register is called the address of the register. The flip-flop registers may be assigned to any desired address by changing the cables. A 32-position switch called the test storage switch (TSS) determines the register to which the machine has access. If the command storage read-out is given,



the word will be read out of the storage register whose address corresponds to the number contained in the storage switch. The command storage read-in will read a word into the register whose address is in the storage switch (if it happens to be a toggle-switch register, the read-in process cannot take place).

### 2.133 Arithmetic Element

The arithmetic element (AE) is shown in Fig. 14. The numbers are manipulated in the A-register, the B-register, and the accumulator, which have been discussed in sections 2.121, 2.122, and 2.123, respectively. Many of the arithmetic operations require a longer sequence of commands than can be provided with the eight time-pulses available from central control. The remainder of the arithmetic element (with the exception of special-add memory and overflow, sign control, and divide error) is used as an auxiliary control to provide these commands. One of the commands provided by the control matrix on such operations is stop clock, which is sent to clock-pulse control and stops the supply of pulses to the time-pulse distributor. At the same time a command is sent to arithmetic control which permits it to commence operation. Either LFCP or HFCP (depending on the operation) are used to carry out these arithmetic operations. When the operation is complete, an end-carry pulse which permits the time-pulses to resume is sent to clock-pulse control by arithmetic control.

Certain conventions have been adopted regarding number representation. These are explained in Appendix I. The use of the "nines-complement" to represent a negative number means that the negative  $-y$  of a positive number  $y$  is actually represented by  $2-y-2^{-15}$  in this machine.

This is perfectly all right for addition and subtraction as long as end-around carry is provided. Shifting, multiplication, and division are quite complicated for negative numbers if the "nines-complement" is used. The sign control has been provided so that these operations can be carried out for the positive magnitudes of the numbers involved and the proper sign affixed to the result by complementing each digit.

Special-add memory and overflow is necessary if the computer is to have the facility of using two words to represent a double-length number. The addition of the less significant halves of two numbers represented in this fashion may produce a carry into the least significant digit of the other half of the sum. The special-add memory and overflow is used to remember such a carry. An addition of two single-length numbers which results in such a carry means that the capacity of the computer has been exceeded. When such a situation occurs, the special-add memory and overflow will supply an alarm signal which can be used to stop the computation at that point and provide some indication to the operator. A similar situation can occur in the divide operation if the dividend is greater than the divisor. The divide error is used to provide an indication of this situation.

#### 2.134 Input-Output Element

A block diagram of the input-output element is shown in Fig. 15. Words are transferred between the in-out register and the external equipment on commands supplied by the external equipment. When such a transfer is to take place, the command stop clock is sent to clock-pulse control. When the transfer is completed, an in-out control end-carry pulse is sent

to clock-pulse control to permit the time-pulses to resume. A rather complicated system is provided to permit the computer to operate at the same time as the external equipment. The comparison register is used as a built-in check on the operation of the in-out register. Its operation will be described in section 5.12.

### 2.135 Built-In Checking

It was mentioned in section 2.13 that the check bus has been provided to check on the transfer of words via the bus. The check register is used to compare the words transmitted by both buses. When a word is transferred from unit A to unit B via the bus, it is also transferred from unit A to the trigger inputs of the check register via the check bus. If the check register was originally clear, it will receive the word that was transferred. Some time later, the word received by unit B is transferred to the trigger inputs of the check register via the bus. No provision is made for carries. If the two words are the same, the check register will contain 0. The gate tubes on the 1 sides of the flip-flops are sensed. If any of the flip-flops contains 1, an output pulse is obtained. This output pulse is sent to the alarm indicator and is also used to stop the supply of pulses to the computer (with the exception of restorer-pulses and the pulses required for electrostatic storage). The alarm indicator provides visual and aural indication of the discrepancy. Alarm pulses may also be obtained from other parts of the computer, such as the divide error, the in-out element, and the special-add memory and overflow.

### 2.136 Test Control

The operation of the computer is controlled by a number of push buttons located in test control. With these push buttons and some

toggle switches, it is possible to set up any initial conditions that may be required to operate the computer. A single step of the problem can be performed using another push button. Test equipment is provided to set up some special operating conditions that have been found useful for testing purposes.

### 2.137 Electrostatic Storage

The bulk of the memory of WVI will be provided by electrostatic storage tubes (EST). It is ultimately expected to have a capacity of 2048 words. It is convenient to think of electrostatic storage (ES) as being made up of registers having addresses ranging from 0 to 2047. Storage selection control determines whether ES or TS is to be used. If both are used simultaneously, the addresses from 0 to 31 must be assigned to TS while the addresses from 32 to 2047 must be assigned to ES. The program register is used as a buffer between ES and the rest of the computer. When a word is transferred to ES it is actually transferred to the program register. A subsequent command ES write will stop the supply of time-pulses and restorer-pulses and will start electrostatic storage control (ESC). The word in PR will be transferred to the position in ES corresponding to the address contained in the ES decoder (ESD) (it is similar in function to TSS, but it is not a switch) under the control of ESC. After this transfer has been completed, an ESC end-carry is sent to CPC which will permit the supply of time-pulses and restorer-pulses to be resumed.

### 2.2 Machine Cycle

The period of time required to fill the counter of the TFD

when starting from the clear condition will be referred to as the machine cycle. During this period, the computer performs one operation and prepares to perform the next operation.

### 2.21 Program Timing

The series of commands necessary to enable the computer to proceed from one operation to the next is known as program timing (see Fig. 16). For convenience, a separate line of the control matrix is used to gate open the proper commands. Nearly all of the operation lines are connected to the program timing line. Those which are not connected to this line, are nevertheless connected to those gate tubes which provide the commands essential to sequencing. Program timing provides some additional commands which are common to nearly all operations (these additional commands do not interfere with any of the operations which do not require them). We will discuss only that portion of program timing which is essential to sequencing. Some of the commands are concerned with checking the transfers by use of the check register.

On time-pulse 2 (TP2), the commands PC read out and SS read in are given. Both TSS and ESD will receive the address contained in PC, but SSC will determine whether TS or ES is used. In order for the read in to give the correct result, both TSS and ESD must be clear. This will have been accomplished by the command SS clear on TP1. On TP4, the word contained at the address specified by PC is transferred to FR. This transfer is direct if ES is used, but requires the command FR read in if TS is used. TP4 also clears CS after a 1/2  $\mu$ sec. delay. This energizes the line which controls the ri operation. This operation does not have

program timing, but the commands which occur on TP5 of the ri operation are an essential part of program timing. The commands PR read out, CS read in, and SS read in are given on TP5 of ri. This results in CS being set up to perform the operation specified at the address which was contained in PC at TP2. At the same time, SS is set up so that the correct operand will be available from storage. The commands necessary to perform the particular operation are obtained starting with TP6. Inasmuch as CS will not be changed until  $TP4\frac{1}{2}$  of the next machine cycle, it is possible to use TP1 to TP4 for obtaining these commands. On TP7 of program timing, the address contained in PC is increased by 1 by the command add to PC. Thus, on the next machine cycle, the instruction contained at this new address will be performed.

Flexibility in the choice of addresses for successive orders is provided by a special operation, subprogram (sp). The sp operation changes the contents of PC to the number specified in its address section. An additional operation, conditional program (cp) which makes such a change conditional upon the sign of the number contained in AC (if the number is negative, the change is made; if positive, it is not changed), permits the machine to choose its own computing routine on the basis of the results of some computation.

## 2.22 Operation Timing

The series of commands which are concerned with actually carrying out the operation specified by the particular setting of CS is called operation timing. As an example, operation timing for sp will be discussed. The address contained in PC at TP7, indicating the register containing

the instruction which would ordinarily follow, is transferred to AR, where it is available for further use, if desired. Thus, AR is cleared on TP6. On TP8, the commands PC read out and AR read in transfer this address to AR. On TP8 1/2 (TP8 delayed by 1/2 usec.), PC is cleared. The address specified in the sp instruction is still contained in FR (it was there at TP4 of program timing). Hence, on TP1, the commands FR read out and PC read in transfer the desired address to PC. This address will be read out of PC to SS on TP2, thus the next instruction will be obtained from the address specified in the sp instruction.

### 2.3 Programming and Coding

The term programming is used in a general sense to indicate the process by which a problem is prepared for solution by the computer. Coding is a much more specific term referring to the step in which the problem is stated in terms of operations which the computer performs. When a problem has been put in this form, it is referred to as a coded program. We shall be mainly concerned with coding in this thesis. A more detailed discussion of coding for WWI may be found in reference 16. A compact summary of the basic information has been included in this thesis as Appendix I.

#### 2.31 Nomenclature

We have previously defined the term word as a 16-digit binary number. If the word is interpreted as an instruction, the first (left-hand) 5 digits are the coded representation of the operation to be performed. There are 32 operations possible with this convention. In

most cases, the last 11 digits indicate the address of the operand. When TS is used, only the last 5 digits specify the address. When the operation code specifies a shift, the last 5 digit indicates its extent. For convenience in writing codes, two-letter abbreviations have been adopted for each operation (e.g., sp, cp, ri, ca, etc.). Another convenient notation is RC(x), which indicates the address of the register containing x. The symbol CR (y) indicates the contents of the register whose address is y. Thus, the combination of symbols ca RC(x) represents an instruction which will transfer the contents of the storage register containing x to the accumulator. Of course, it is necessary to assign a specific address to the register containing x before the instruction can be used in the computer. If x were put in register 31, this instruction would be written ca 31. In the machine, this would appear as 1000000000011111 (10000 is the code for ca).

The computer distinguishes between a word used as an instruction and a word used as a number on the basis of the time in the machine cycle at which this number is transferred from storage. If it is transferred on TP4 and TP5 (via the program register), it will be treated as an instruction. At any other time, it will be treated as a number. Thus, it is possible to manipulate an instruction as though it were a number. Indeed, it is a very common practice to do just that in programming (see, for example, reference 3).

### 2.32 The Operations

The operations which have been accepted for permanent use in WWI are listed in Appendix I. A brief description of each is included



there. In addition, there are a number of operations which have been temporarily wired into the control matrix. These operations are distinguished by a two-letter code in which the first letter is q. One of these, qc (check), will be used very frequently in this thesis. It compares the contents of the storage register indicated in its address with the contents of the accumulator. The comparison is carried out in the check register in a manner very similar to checking of the bus transfers (see section 2.135). An alarm is obtained if the words do not agree. Though qc is only a temporary operation at present, it seems very likely that it will be made a permanent operation.

3 - DISCUSSION OF FAILURES3.1 Classification of Failures

Each stage in the computer must perform certain clearly defined functions. Its ability to perform these functions depends on many factors. These factors generally vary with time in an irregular fashion. The status of all these factors at a particular instant will determine whether the stage can perform one of its functions at that instant. A failure need not be present at all times. However, if it is not present during a checking process it will not be detected. Thus, the nature of the factors causing failures and the degree to which they can be controlled exert an important influence on the process of error detection.<sup>11</sup> It is helpful to classify failures according to a scheme which indicates this influence.

3.11 Steady-State Failures

An irreversible change in one or more factors which renders a stage permanently incapable of performing a function is called a steady-state failure.<sup>11</sup> The stage will consistently yield the same incorrect response at every application of the signal. A typical example is an open cable; it can never transmit a pulse. However, it must not be concluded that a permanent failure of a component is necessarily a steady-state failure of the stage. If a resistor used to terminate a cable should become open, a distorted pulse waveform would result. Whether this will cause a failure depends on several other factors which need not always be unfavorable.

### 3.12 Dynamic Failures

Closely akin to the steady-state failure, though somewhat more subtle in its effect, is the dynamic failure.<sup>21</sup> The distinguishing factor is that it appears only on certain applications of the signal. It may be that a coupling condenser will charge slightly every time it passes a pulse. If there is not sufficient time for it to discharge between successive pulses, the charge will continue to build up. The amplitude of the output pulses will decrease as the charge on the condenser increases. Eventually the output pulse will be too small to drive the next stage. This effect is referred to as FRF-sensitivity (pulse-repetition frequency) and is an example of a dynamic failure. The open terminating resistor referred to in section 3.11 might be a dynamic failure or it might not be a failure at all. Another interesting example of a dynamic failure found in the operation of WWI (after considerable effort) was the result of "ringing" on a d-c power supply line. The various supply lines have been provided with LC-filters to prevent the coupling of pulses between circuits. The current surges which accompany the switching of vacuum tubes cause the filter circuits to oscillate or "ring" with a fairly small amplitude. It so happened that the timing of a problem was such that ~~X~~ the fourth time a particular pulse occurred coincided with the peak of this oscillation. This was barely sufficient to interfere with correct operation; indeed an error did not always occur on this pulse. This failure was not due to a defective component; it was a design weakness which, in all fairness, the designer could not have been expected to anticipate.

### 3.13 Incipient Failures

Some factors change in a continuous fashion. For example, the cathode emission of a tube tends to decrease gradually as it ages. Eventually, the change will have progressed to a point where correct operation is no longer possible and a steady-state failure will exist. Before this point is reached, the operation of the stage becomes more sensitive to other factors. An incipient failure is said to exist.<sup>11</sup> Slight variations in some factors would not cause any failures in a stage without an incipient failure, but these variations would cause failures when an incipient failure is present. Aging effects are found in all components, but are more pronounced in tubes and crystal rectifiers.

### 3.14 Intermittent Failures

Frequently a stage will be abnormally sensitive to a factor which fluctuates in a random fashion. This sensitivity is not the result of a progressive deterioration as is the sensitivity exhibited by an incipient failure, but is more or less constantly present. The high contact resistance present in a poorly soldered joint might be a cause of such sensitivity. The failure will appear at irregularly spaced intervals. Such a failure is called an 'intermittent failure.'<sup>11</sup> It is quite similar to the incipient failure in appearance, but not in cause.

### 3.15 Reproducible Failures

Another type of failure which is manifested at irregularly spaced intervals is the reproducible failure. Its immediate cause is an external disturbance of some sort: mechanical vibrations, presence

of r-f radiation, and the like. Of course, there must be some other factor which makes the stage sensitive to these disturbances, but it may operate very satisfactorily as long as the disturbance is not present. By artificially causing such disturbances, reproducible failures may be made to occur whenever desired, and hence take on the characteristics of steady-state failures.

### 3.16 Random Failures

Still another failure appearing at irregularly spaced intervals is the random failure.<sup>6</sup> It does not require an abnormally sensitive stage for its occurrence. It generally results from an unusual combination of unfavorable conditions among the factors which are subject to variation. It may be due to an exceptionally large variation of one such factor, e.g., a power line transient far exceeding the normal limits. Although there is always a cause for such a failure, it remains unknown to the operator.

## 3.2 Checking Considerations

### 3.21 Influence of Various Types of Failures

Test checking may be regarded as an experimental procedure for determining the ability of the computer to function properly. If the ability should change during the course of the experiment, it becomes difficult, if not impossible, to interpret the results in terms of the condition of the computer. It is quite possible to obtain the correct solution to a problem even though dynamic, incipient, intermittent, reproducible, and random failures are present. In short, test checking is applicable to only steady-state failures.<sup>11</sup> In designing test sequences,

no consideration need be given to any failures which are not essentially steady-state failures. However, it has been noted that dynamic, incipient, and reproducible failures are very closely related to steady-state failures. If supplementary means can be found to convert these types of failures into steady-state failures during the performance of the test sequences, test checking may be used for their detection. Of course, if an intermittent or random failure should occur at an opportune time, it, too, would be discovered by test checking. However, mathematical checks are much more effective in dealing with irregularly spaced failures.

### 3.22 Operational Effects of Failures

Generally speaking, a failure in a state causes it to lose its ability to perform a certain function. The problem of designing a checking sequence may be stated in terms of these functions; the checking sequence must require each stage to perform all such functions. One of the first tasks, therefore, is to determine these functions for each of the elementary stages. It is not necessary to consider each component failure for this purpose, as will become apparent when it is done in sections 3.221 to 3.224.

When it is desired to use a problem for trouble location, a more detailed knowledge of the operational effects of the failures is required. The actual response of the faulty stage must be known. This is an important practical distinction between checking and trouble location. It would perhaps be well to restate this in the form of the questions which the designer must ask in each case:

Checking: What must the stage do?

Trouble Location: What will the stage do?

Some of the answers in typical cases are given below.

### 3.221 Amplifiers

The function of an amplifier is to provide an output of the proper amplitude to drive the succeeding stages when the appropriate input signal is applied. The output amplitude may lie among a wide range of values when a faulty component is present. The particular amplitude level which is obtained will either be capable of driving a succeeding bi-stable stage or it will not be capable of driving it. Since the output will ultimately be applied to a bi-stable stage, it may be said that the amplifier will not supply any output pulse if a failure exists. This does not mean that the actual amplitude is zero, but that it might just as well be zero as far as its effect on the operation of the computer is concerned. If the output is capable of driving the succeeding stage, no failure exists.

It is, of course, possible to obtain an output when no input signal is applied. However, there is no way of predicting the relation of the times of occurrence of such spurious pulses to the timing of computer operations. Thus, failures of this type are essentially intermittent and need not be considered. As additional justification for disregarding them, it should be noted that such failures are extremely rare in well-designed amplifiers.

To summarize this discussion, the two questions of 3.22 will be answered. An amplifier stage must provide an output whenever the appropriate signal is applied to the input. If a failure exists, no output

will be obtained when a signal is applied to the input. The extreme simplicity of these conclusions tends to belittle their importance. Certainly they are intuitively obvious, but they must be clearly realized in designing test sequences. Note that the failure of the amplifier may be due a large number of causes (open filament, open plate load resistor, grid-to-cathode short, etc.), all of which exhibit the same symptom.

### 3.222 Gate Tubes

The gate tube can fail in three ways:

1. An output pulse is obtained without coincident inputs.
2. No output pulse is obtained for coincident inputs.
3. An output pulse is coupled back to an input line.

Case 3 is extremely rare if care has been given to the layout. Furthermore, it is completely harmless if the input line is driven by an amplifier, as is often true. This case will therefore not be considered in this thesis. Case 2 is quite possible and fortunately, is amenable to easy analysis. Two possibilities must be considered under case 1: the output pulse is obtained without any input or it is obtained with only one input. The former possibility represents an oscillatory condition which is extremely unlikely. If it should exist, it would be extremely difficult to determine the times when a spurious pulse would occur. It would take on the characteristics of an intermittent failure, so that it need not be considered. The latter possibility also may result in spurious pulses at indeterminate times, which, again, need not be considered. However, in the usual application of the gate tube, a signal is applied to one input to determine whether the other input is energized at that time.



It is quite possible that the gate tube will provide an output pulse whenever this sensing signal is applied, even though the other input is not energized. This is a straightforward manifestation and will be considered. To summarize, it will be assumed that to the failure of a gate tube can exhibit only two possible effects:

1. It will never provide an output when a sensing signal is applied.
2. It will always provide an output when a sensing signal is applied.

### 3.223 Flip-Flops

The function of the flip-flop is to receive information from each of three inputs and to retain such information indefinitely. A faulty component may cause it to develop a preferred position, i.e., it will not remain in the other position indefinitely, though it will remain in the preferred position for any desired period of time. Once again an indeterminate time factor appears in the consideration of the effects of failures: the period of time a flip-flop can hold the non-preferred information before reverting to its preferred state. If this period is longer than the time it is required to hold the non-preferred information, this weakness will not have an adverse effect on its operation. The longest interval between restorer-pulse pairs is the maximum amount of time that a flip-flop would be required to hold one state. However, for reasons which are not fully understood, flip-flops have been observed to revert to the preferred position even though restorer pulses are used. Some assumptions must be made regarding the effects of failures of this type

on computer operation when the trouble location is considered. Nevertheless, the checking requirements can be clearly stated:

1. It must receive and retain a 0 via the clear input
2. It must receive and retain a 1 via the set input
3. It must receive and retain a 0 via the trigger input
4. It must receive and retain a 1 via the trigger input.

### 3.224 Restorer Pulses

The loss of restorer pulses in some part of the computer is another type of failure which does not lend itself to a simple analysis. If the contents of the flip-flops are changed at sufficiently frequent intervals, the charge on the coupling condensers will probably be maintained, so that correct operation will continue. However, this is not likely. When a flip-flop remains in the same position long enough, the suppressor of its sensing gate tube will assume the potential of the point to which it is returned. Those gate tubes in which the suppressor are returned to ground will be open, regardless of the condition of the flip-flops. Most gate tubes are designed this way. Those gate tubes in which the suppressor are returned to -15 volts will be closed, regardless of the condition of the flip-flops. Read-in gate tubes are generally of this type. The same symptoms will, of course, appear if a flip-flop is not capable of responding to a signal at the trigger input, for it is equivalent to the loss of the restorer-pulse supply to that flip-flop. Effectively, these failures may be interpreted as flip-flop failures in which:

1. The flip-flop appears to contain 0 and 1 simultaneously.
2. The flip-flop appears to contain neither 0 nor 1.

Because restorer pulses play such a vital role in the operation of the computer, it may be well to include some built-in device to indicate the loss of restorers at some critical point (or points). This subject cannot be discussed any further in this paper.

### 3.3 Marginal Checking

A very useful technique adopted by Project Whirlwind is the provision of facilities for the variation of d-c supply voltages to various parts of the computer. The variation of these voltages is a powerful adjunct to other checking procedures. Dynamic and incipient failures can be converted into steady-state failures through its use. It helps simplify the problem of trouble location. The possibility of removing a deteriorating component before it causes a failure in computer operation is enhanced by this voltage-variation scheme.

A variable-voltage generator is provided which may be placed in series with a number of the d-c supply lines. Each of these lines supplies a rather limited number of stages. A telephone dial may be used to select the line on which the voltage is to be varied. Manual control of the variation is available within the limit imposed by the generator characteristics. In addition, there is provision for automatically switching the generator into each line in sequence and causing the voltage to be varied within predetermined limits before switching to the next line. These limits may, and generally will, be different for each line. It is possible to both increase and decrease the voltage. Since only a limited number of stages is affected by a single voltage-variation line, a considerable degree of trouble location may be affected through its use.

For example, if an incipient failure under normal voltage is converted to a steady-state failure by varying the voltage on a particular line, the trouble is associated with one of the stages supplied by that line.

In general, there is a certain maximum tolerance (both positive and negative) in the supply voltages of each stage, which is determined by its design. A tolerance will be associated with each supply line. This tolerance is some function of the tolerances of the individual stages supplied by it. These tolerances are called margins. When a stage is operated very close to its margin, it becomes much more sensitive to fluctuations in the factors which cause failures. In particular, dynamic and incipient failures show up much more clearly. Decreasing the screen supply voltage on an aging pentode tends to accentuate the decrease in cathode emission. The plate current decrease in a normal tube may be sufficient to cause a steady-state failure with a margin of -30 volts; for an aging tube, the margin at which a steady-state failure appears may be only -15 volts.

These facilities may be used to measure the margin on each line by varying the voltage to the point at which a failure occurs. By measuring the actual margins at regular intervals, considerable information is obtained regarding the effects of aging on the operation of the computer. When margins become unduly low, it is time to replace the offending component. Skilled personnel contend that the trends of this data often indicate the faulty component quite accurately. They have proved their contention on a number of occasions. This procedure of checking the margins has given rise to the term marginal checking to refer to the use of the voltage-variation facilities for any purpose.

It is meaningless to quote the margin available on a given voltage-variation line without specifying the conditions under which it was measured. No failure would be obtained at any voltage if the stages supplied by that line were not in use. One standard set of conditions is the complementing of all flip-flops at a fixed rate, determined by an external pulse source. This is used only for the marginal checking of the flip-flops. By observing the indicator lights, the margin at which each flip-flop fails to complement can be measured. This technique has proven quite effective. No equivalent technique has been developed for other types of stages. It is necessary to design problems which utilize the stage undergoing voltage-variation. Much study must be given to the coordination of marginal checking with test checking.

The fact that marginal checking may be used to predict the imminent occurrence of a failure suggests that it may also be used to predict that no failures will occur for some definite period of time (the prediction interval) if the margin is greater than some predetermined value. There is not yet sufficient evidence to justify this conclusion, but it does seem plausible. Of course, this conclusion could only apply to steady-state, dynamic, and incipient failures. By providing some apparatus to produce suitable external disturbances, it might be possible to extend this reasoning to reproducible failures. If the existence of a reasonably long prediction interval can be demonstrated conclusively, test-checking procedures would be considerably more attractive. An overall check problem could be performed at regular intervals (somewhat shorter than the prediction interval) while each supply line was

varied through the range required for the prediction to be valid. If the problem indicated that no failures were present, it may be concluded that problems solved during the remainder of the prediction interval are error-free, provided that some mathematical checks are included to guard against intermittent and random failures. If the prediction interval were long compared to the time required to perform the check problem, the loss of time inherent in test-checking procedures would become negligible.

### 3.4 Summary

Test-checking procedures are only effective against steady-state failures. However, by use of supplementary means, many failures may be converted into steady-state failures. Those which cannot be converted are sufficiently random to be detected by mathematical checks. For purpose of test sequence design, it may be assumed that only steady-state failures need be detected. To use a sequence for trouble location, it is necessary to determine the results which will be obtained in the presence of all possible faults. Each fault is assumed to be manifested in a clearly defined manner, although this is by no means accurate. Those faults which do not exhibit clearly defined effects are generally sufficiently rare to justify their being neglected. Furthermore, if such a peculiar case arises, analysis on the basis of these assumptions should lead to contradictions, thus indicating that a special case is involved. However, these assumptions do not eliminate the need for further study and classification of computer failures. Perhaps such study may lead to a more satisfactory treatment of test checking.

4 - PRINCIPLES OF TEST SEQUENCE DESIGN

4.1 Objectives

Methods of improving the reliability of computing machinery will be in demand as long as there is a need for computers. At present, the demand for such methods is quite pronounced. The main emphasis is towards developing techniques which give useful results as soon as possible, even though they leave something to be desired. More elegant and effective methods will be required in time, but the important thing is to keep the computer operating with a minimum of laborious maintenance. Trouble-location techniques are of more immediate interest than checks on the overall results. The ultimate goal is twofold:

1. to obtain a compact, overall test problem, coordinated with marginal checking, which will indicate that every stage of the computer is working properly (barring the inevitable exceptions) and will continue to work properly for the prediction interval.

2. to obtain a collection of test sequences which may be used (probably in conjunction with marginal checking) to locate the source of any failure which is present with as little ambiguity as possible.

This thesis is mainly concerned with the second of these goals. However, it is felt that some insight into the design of an overall check problem can be gained by studying the checking of small units. In addition, the results of such a study can also be applied to trouble location. Thus, the main objective of this thesis is the development of a systematic method of designing test sequences for checking relatively small units in coordination with marginal checking. A secondary objective is to discover

those limitations of test checking which might require the use of other checking procedures.

#### 4.2 History

E. I. Flumenthal and G. C. Hoberg were the first men to study the problem of test sequence design in detail.<sup>5</sup> They were concerned with locating certain types of steady-state failures in the Five-Digit Multiplier, the prototype of the Arithmetic Element of Whirlwind I. They considered only single failures which are equivalent to the removal of some vacuum tube. The procedure they adopted is quite straightforward. It consists, essentially, of tabulating the functional abilities lost upon the removal of each tube. Generally, this tabulation will suggest a sequence which will detect the presence of the failure. For example, suppose that a particular failure would render a flip-flop register incapable of receiving a 1 in digit column 15. The sequence for checking for this failure is almost self-evident. Read a word which contains 1 in digit column 15 into the register. Read the word received by this register to a comparison unit (such as the check register). Compare the word in the comparison unit with the word originally read into the register. If they agree, then the particular failure does not exist (every other part of the computer is working properly because of the single fault assumption). If they do not agree, then the failure exists. However, these men utilized sequences of commands rather than operations, thus severely limiting the practical applicability of their results.

G. C. Sumner recognized that information is routed along certain clearly defined channels in many typical computers units. He conceived of a method of trouble location analogous to the conventional procedure



of signal tracing. The test sequence would set up the unit so that the information would be routed along a particular channel, supply information to the input of the channel, and check that the appropriate information appeared at the output. He designed a few sequences (using commands) to illustrate the procedure, assuming only single failures.

#### 4.3 Proposed Procedure

The computer will be subdivided into extremely small sections. A method will then be developed for designing a test sequence to check any one section, assuming that all other sections are working properly. Modification of the procedure to include multiple faults will be considered. The use of the sequences for trouble location will be discussed.

##### 4.31 Subdivision of the Computer

A section consists of all stages which are supplied by a single voltage-variation line. This subdivision of the computer automatically coordinates the test sequences with marginal checking. Of course, most stages require more than one d-c voltage supply, and it frequently happens that the same section corresponds to several voltage-variation lines. Thus, there are less sections than there are voltage-variation lines, and one test sequence can be used with several voltage-variation lines. This method of subdivision also helps create a firmer foundation for the single fault assumption. It is more likely that a failure will occur in a section undergoing voltage-variation than in one whose supply voltage is normal, particularly if the periodic checking scheme discussed in section 3.3 (in connection with the prediction interval) is used.

#### 4.32 The Channel Concept

The picture of clearly defined channels of information flow conceived by Sumner (see section 4.2) may be readily generalized to include the entire computer. Information arriving at the input-output element is routed to storage. From storage it may be transmitted to the arithmetic element, where it may be combined with other information to form the desired information. It is then routed to the input-output element via storage and made available to the operator. Another possible channel takes the information from storage to control, where it may be used to manipulate other information in the computer. The paths traced out in this way can become very complex, especially since the information frequently undergoes transformation and may frequently be used to determine its own route. Many channels may coincide for several stages. These complications make Sumner's idea of automatic signal tracing untenable when applied to the whole computer. The practical difficulties involved are very nearly insurmountable.

However, it is very useful to consider a segment of a channel. Essentially, Sumner considered those segments which were within each of the typical units that he treated. Each of those segments could conceivably be shared by many different channels. Blumenthal and Hoberg also considered channel segments, although they may not have been aware of it. Their segments consisted of single vacuum tubes. The channel segments to be considered in this thesis will be those which are within a given section. A channel segment will be defined as a portion of a channel which is included within a given section and includes all stages in that section which are part of the channel. In special cases, this definition may reduce to that of Sumner or that of Blumenthal and Hoberg.

It is not necessary to trace out the entire channel in order to describe each channel segment. A channel segment may be described without any knowledge of the complete channel or channels of which it is a part. It is only necessary to examine a diagram of a section, choose a stage which receives information from sources wholly outside that section, and proceed to trace the flow of information from this stage to all stages until a stage is found which is outside the section. There may be many alternate paths for the flow of information. Each of these is another channel segment. To find all channel segments belonging to a given section, all such paths must be traced out, starting with each stage which receives information from sources wholly outside the section. This process may seem very laborious, but in many cases it is really quite simple. Every alternative route encountered in tracing out these paths should be carefully noted down in order to avoid overlooking one of the paths.

#### 4.33 Use of The Channel Concept

Once a tabulation of all the channel segments of a given section has been made, it is a comparatively simple matter to design a test sequence for each of them. When there are several channel segments sharing the same stage (or stages) it is sufficient to check one of the channel segments in order to check the stage. The choice may be made arbitrarily. However, it is frequently desirable to check all channel segments, if only for the greater assurance it provides. The sources and destinations of the information transmitted by each channel segment should be tabulated too, as should be the command pulses which control

the transmission of this information. It is then only necessary to choose one of the sources (if there is only one source, no choice exists), one of the destinations, and one of the sequences of commands for transcribing the information from this source to this destination. Any convenient method of supplying the source with the information and examining the destination may be used. All choices of this nature may be made quite arbitrarily if there are alternate possibilities. If there is but one possibility, then that is the way it must be done. It is sometimes possible to make choices in such a way as to economize on the amount of storage required. These points should become clear when some examples are considered.

There is one point that requires special attention. Frequently, the information supplied by a channel segment to its destination is used to manipulate some numbers in an arithmetic process. In some cases, it may happen that the correct result will be obtained even though the proper information is not supplied. Whenever such a channel segment is being checked, care must be taken to be sure that the numbers being manipulated do not lead to one of these special cases.

#### 4.34 Multiple Faults

The test sequences which have been obtained on the assumption that all parts of the computer (other than the channel segment being considered) are working properly, may not apply under less restricted conditions. In many cases, it is possible to design other test sequences which are useful when multiple faults are present. However, it is still necessary to make some assumption regarding the combinations of faults

to be considered. It is also convenient to limit consideration to a small unit which may consist of several sections. The concept of a channel segment must be broadened to include all stages of the channel which are in any of these sections. This is very nearly the same type of channel segment that Sumner used.

The procedure requires that careful attention be given to the order in which the channel segments are checked. First, all single-stage channel segments should be checked. Then, design test sequences for all two-stage channel segments which share stages already checked, and so on. Of course, it is quite possible that there are multi-stage channel segments consisting of several stages that cannot be checked independently. If such a situation exists, then the whole channel segment must be treated as though it were a single stage.

If it is desired to expand a test sequence to include more sections than were considered in its original design, it is usually necessary to start all over and design a completely new sequence. The additional stages which come into consideration generally change the composition of the channel segments so that the original sequence is not applicable. In particular, the one-stage channel segments of the smaller unit may become multi-stage channel segments. This feature of the approach is somewhat of a disadvantage because it requires that such analysis must be carried out every time a larger unit is considered. However, in lieu of a better approach, this disadvantage must be tolerated.

#### 4.35 Trouble Location

The test sequences designed for checking the various channel segments may be used in trouble location. It is only necessary to determine the point in the sequence at which an alarm is obtained in order to discover the faulty channel segment. The type of information which was not transmitted is determined by analysing the contents of the various flip-flops. These two steps usually reduce the number of possible locations of the failure to a reasonable value. Manual trouble-location techniques may then be used.

A greater degree of trouble location can be obtained when a number of channel segments share the same stages (see Fig. 17). It is necessary to use a test sequence to check each channel segment. A number of possibilities may be eliminated by considering the results obtained for each sequence. Referring to Fig. 17 and assuming only single failures, the following pattern of results will be obtained:

Defective Stage	Channel Segment #1	Channel Segment #2	Channel Segment #3	Channel Segment #4
A	Alarm	Correct	Alarm	Alarm
B	Alarm	Alarm	Alarm	Correct
C	Alarm	Alarm	Correct	Alarm
D	Alarm	Alarm	Alarm	Alarm

Actually, it is not necessary to perform the test sequence for channel segment #1 because the alarm will always be obtained. This indicates

a real difference between checking and trouble location. For checking, it would be sufficient to perform the test sequence designed for channel segment #1. If no alarm is obtained, then all four stages are operating correctly. If an alarm is obtained, the other three test sequences may be used to isolate the failure.

## 5. - EXAMPLES OF DESIGN PROCEDURE

### 5.1 Single Fault Sequences

Several test sequences have been designed to check typical sections of the computer. It is assumed that every channel segment, other than the one being checked, is operating correctly (i.e., only a single fault exists). Consideration of storage has been omitted because the channel segments consist of a prohibitively large number of stages. Special techniques will probably be more suitable for checking storage. A design for an entire section has also been carried out in the hope that it would point the way to the design of test sequences for larger sections.

#### 5.11 Arithmetic Control Flip-Flows

Test sequence design is probably most simply illustrated in arithmetic control because it consists mainly of disjunct channel segments containing few stages. Arithmetic control has been discussed in section 2.133 where a block diagram (Fig. 14) has been included. The section defined by voltage-variation line 79 will be considered. This section is identical with the one defined by voltage-variation line 80.

The following stages are in this section:

FF304.01	(sign control)
FF306.01	(multiply)
FF307.01; FF307.02	(shift)
FF308.01; FF308.02; FF308.03; FF308.04	(divide)
FF309.01; FF309.02	(special-add)
FF310.01; FF310.02	(point off)

The channel segments, together with the sources and destinations of the information and the sources of the commands, will now be described in



detail. These descriptions will be followed by a discussion of the design of the test sequences.

#### 5.111 Channel Segment 1 - Sign Control

Only one store is involved, FF304.01. It may receive information via its trigger input from itself (assuming a slight delay in transmission), and the sign digits of AR and AC. It supplies information to AC and itself. The command AR sign check (which occurs on mr, mh, and dv, see Fig. 18) is required to obtain the information from AR. The command AC sign check (which occurs on mr, mh, dv, sl, sr, and sf, see Fig. 18) transmits the information from AC. Information for itself is transmitted and received on the command product sign (which occurs on mr, mh, dv, sl, sr, and sf, see Fig. 18) which also transmits the information to AC.

Consider the ability of this channel segment to handle the information 1. Assume that it has been decided to supply this information from AR. The more easily accessible destination is AC. Either mr, mh, or dv may be used to transmit the information. Suppose mr is chosen (dv is a somewhat more complicated operation). The operand must be a negative number, say  $-1/2$ . Thus we must order mr PC( $-1/2$ ). Some number must have previously been in AC, and, in order for the information 1 to be transferred on the command product sign, it should have been positive, say  $+1/2$ . This can be put in AC by the instruction ca (PC( $1/2$ )). The resulting product is, of course,  $-1/4$ . Thus the test sequence has been developed:

```
ca RC(1/2)
mr PC(-1/2)
qc RC(-1/4)
```

Care must be taken to be sure that the correct answer is not obtained if the channel segment fails to supply the proper information. In this problem it is quite apparent that the results of the multiplication would be  $+1/4$  if the proper information were not supplied, so that the check is indeed valid. Note that only the signs of the operands are instrumental in producing the desired information; their magnitudes may be chosen arbitrarily. The particular numbers given were chosen because of their simple representation in binary notation.

To check its ability to transmit the information 0, the sequence of instructions:

```
ca PC(1/2)
mr RC(1/2)
qc PC(1/4)
```

may be used. If the information 0 were not transmitted, then the information 1 would be transmitted. The result would be  $-1/4$  so that the check is valid.

These six instructions completely check the ability of this channel segment to perform its function, assuming that any failure symptom which may exist will exert its effect throughout the time required to carry out these sequences. Note that it was not necessary to use all of the information sources or destinations. Indeed it would have been quite difficult to base a check on the use of FF304.01 as its own source and destination.

#### 5.11? Channel Segment 2 - Multiply

Again, only one stage is involved, FF306.01. It receives the information 1 from central control via the set input on the command multiply (which occurs on mr and mh, see Fig. 18). It receives the information 0

via the clear input from the step counter end-carry pulses, which may occur on mr, mh, dv, sl, sr, and sf. It may also receive the information 0 on AC1 to point off (on sf only). The information is transmitted to DR15 only whenever a high-frequency clock pulse occurs.

To check the ability of the channel segment to transmit the information 1, all that is necessary is to use either the mr or mh operation. The sequence:

ca PC(1/2)  
mr PC(1/2)  
qc PC(1/4)

does just this. It is, of course, one of the sequences used for channel segment 1 and has been used again in the interest of economy. If the information 0 were transmitted, none of the steps of the multiplication would be performed. No step counter end-carry would be available to permit the flow of time pulses to resume. The computer would remain inactive indefinitely ("prolonged stop clock"). At present, no alarm is given when this situation exists, although the operator can easily detect it by observing the indicator lights. It has been proposed that an alarm indication for this condition be included in WWJ. It is likely that this proposal will meet with approval.

If the channel segment cannot transmit the information 0, spurious shifts and additions in the arithmetic element would occur at all times (though such actions are proper on TPP of mr or mh). Thus, the instruction ca PC(1/2) will place 1/2 in AC and the instruction qc PC(1/2) can be used to check that it remained there. While no definite prediction can be made regarding the content of AC in the presence of the spurious

actions engendered by the failure of FF306.01 to transmit the information 0, it is safe to assume that it would not be 1/2 (though there are times when it would be 1/2). A more definite check cannot be obtained.

It is now possible to indicate the explicit role of the single fault assumption in the design of these sequences. The sequence required to check FF304.01 depends on the correct operation of FF306.01, while the sequence for checking FF306.01 depends on the current operation of FF304.01. Assuming only single faults, both flip-flops cannot fail simultaneously. If there is a failure in FF304.01, FF306.01 will operate properly to effect the check and vice versa. As it happens, even if both should fail simultaneously, some alarm indication would be obtained, but this is just a fortunate coincidence. No effort was made to obtain alarm indication in the presence of multiple faults.

#### 5.113 Channel Segments 3A and 3B - Shift

Because channel segment 3A, consisting of FF307.01, is quite similar to channel segment 3B consisting of FF307.02, they have been grouped together. With minor changes (e.g., replacing rr and mh by sl for 3A and sr for 3B), the discussion in section 5.112 is applicable.

To test that segment 3A can transmit the information 1, the sequence:

```
ca RC(1/4)
sl 1
qc "C(1/2)
```

may be used. Similarly, the sequence:

```
ca RC(1/2)
sr 1
qc PC(1/4)
```

may be used for segment 3B. To check that both segments can transmit it information O the sequences designed for channel segment 2:

```
ca PC(1/2)
qc PC(1/2)
```

is satisfactory.

Since the choice of the operands is arbitrary, the desire to use storage economically dictates that same set of numbers be used whenever possible. This criterion has guided the selection of the operands in this sequence. Further reduction of storage requirements can be obtained by making use of the contents of AC after the qc operation. This is a standard device used in coding. The sequences for channel segments 3A and 3B may thus be reduced to:

```
ca PC(1/2)
qc RC(1/2)
sr 1
qc PC(1/4)
sl 1
qc RC(1/2)
```

This saves two orders.

#### 5.114 Channel Segment 4 - Divide

The first multi-stage channel segment to be discussed consists of FF308.01, FF308.02, FF308.03, and FF308.04. Several apparent difficulties may become evident from a study of the block diagram. Attempts to apply the channel concept to the two flip-flops of the divide-pulse distributor may meet with failure because of the complicated manner in which they are interconnected. However, if they are treated as a generalized flip-flop stage having three stable states, the channel segment becomes clearly defined. No real difficulty is introduced by

the need for information to pass through an external section (AC) in being transmitted from the divide-pulse distributor to FF308.03. It is only necessary to choose the operands so that the appropriate information is supplied to FF308.03. Because FF308.03 is intended to provide an alarm when an improper division is performed, to check this function it is necessary to actually order a division which results in an alarm. The analysis of this channel segment becomes quite straightforward once these few difficulties have been resolved. In fact, all information is transmitted along this channel by a sequence of commands which automatically follows the command divide (in normal operation).

The division of  $1/4$  by  $1/2$  would, of course, cause the information to be transmitted to the FF308.01. This would initiate the transmission of all types of information from the divide-pulse distributor. The particular operands require that both divide zero and divide one be transmitted to FF308.03. However, divide zero is not transmitted at a time when FF308.03 contains 1 - only an improper division (more fully discussed below) can provide this condition. A sequence for checking this channel segment is:

```
ca RC(1/4)
dv RC(1/2)
qc RC(+0)
sl 15
qc RC(1/2)
```

That this is not a special case in which the correct result is obtained though the proper information is not transmitted can only be verified by the tedious process of manually carrying out the steps of the division in the same manner as the computer, taking due note of the possible failures.

Such verification has been made for this sequence, but the details are not presented here. Some failures will result in a "prolonged stop clock" which has been discussed in section 5.112. These particular operands, chosen for their convenience, are therefore satisfactory. Had it proven otherwise, a trial and error procedure would have been necessary. The same sequence that is developed in section 5.112 for checking that FF306.01 can transmit the information 0 may be used to check that FF305.01 can transmit the information 0.

At this point, probably the most natural choice for the required improper division is  $1/2$  divided by  $1/4$ . If FF308.03 is working properly, an alarm should be obtained on the first step of the process. If no such alarm is obtained, the computer will continue to operate on the numbers in the arithmetic element as though it were performing a valid division. When the step counter indicates that the process is complete, AC will contain 0 in all digit columns, while FF will contain 1 in all digit columns. If the operation is followed by sl 15, an overflow will occur during the round-off, resulting in an arithmetic check alarm. An alarm will be obtained in both correct and incorrect operation. In correct operation, it will be a divide error alarm, while in incorrect operation, it will be an arithmetic check alarm. The sequence,

```
ca RC(1/2)
dv RC(1/4)
sl 15
```

must be performed separately from other sequences because the computer must be restarted quite frequently. This situation always arises when it is necessary to check a stage designed to give an alarm.

### 5.115 Channel Segment 5 - Special-Add

Another multi-stage channel segment consists of FF309.01 and FF309.02 (see Fig. 19). While it is possible to analyze it using the channel concept, a sequence may be designed more directly. Although FF309.01 receives information on many operations, including sa, FF309.02 receives information only on sa. Furthermore, if the sum of the two operands involved in a special-add is greater than or equal to +1, FF309.01 receives the information 1 and FF309.02 receives the information 0, representing a carry of  $+2^{-15}$ . If the sum is less than or equal to -1, FF309.01 receives the information 0 and FF309.02 receives the information 1, representing a carry of  $-2^{-15}$ . These carries are transmitted to AC on the command special carry of a subsequent ca, cs, or cm.<sup>\*</sup> Thus the sequence:

```
ca PC(1/2)
sa PC(1/2)
cc PC(+0)
ca PC(1/4)
cc PC(1/4+2-15)
```

may be used to check that a carry of  $+2^{-15}$  can be transmitted.

The sequence:

```
ca PC(-1/2)
sa PC(-1/2)
cc PC(-0)
ca PC(1/4 + 2-15)
cc PC(1/4)
```

may be used to check the carry of  $-2^{-15}$ . As usual, the numbers were chosen for convenience.

---

\* If both flip-flops contain 0, a carry of +0 is transmitted, while if both flip-flops contain 1, a carry of -0 is transmitted. The correct result cannot be obtained unless the correct information is supplied by these flip-flops.



### 5.116 Channel Segment 6 - Point-Off

Two stages, FF10.01 and FF10.02, compose this channel segment. The design details may be carried out in a perfectly straightforward manner. Only the sequences will be given here:

```
ca RC(1/2)
qc RC(1/2)
ca RC(1/4)
sf x
qc RC(1/2)
```

(x represents the address of any convenient register).

### 5.117 The Complete Sequence for Arithmetic Control Flip-Flops

The complete sequence for checking voltage-variation lines 79 and 80 is to be found in Appendix II as Test Sequence Number II. The short sequences have been combined in such a way as to minimize the amount of storage required. Addresses have been assigned to the various registers. The resulting coded program can be used with test storage. The sequence for checking that a divide error alarm is obtained has not been included.

### 5.12 Input-Output Element

The input-output element (IOE) (see Fig. 15) was designed with a considerable amount of built-in checking. The comparison register (COR) is used in a multiple checking scheme to check the in-out register (IOR). These two registers are operated in such a way as to give a fairly thorough check of the external film equipment, assuming single faults.<sup>17</sup> A special predetermined check is used to guard COR against single faults. Consider Fig. 5. After the word has been transferred from the film to

IOB (the reading process), both IOR and COR should contain the word which has been read. After the recording process, both IOR and COR should contain the complement of the word which has been recorded. In any event, both registers should have the same contents when the film equipment provides a completion pulse. The pulse will command add IOR to COR, which will supply the word in IOR to the trigger inputs of COR. In a manner entirely analogous to the checking of transfers discussed in section 2.135, this should leave COR containing 0 (with the exception of COR check which contains 1). The completion pulse is delayed  $1/2$   $\mu$ sec. and used to sense GTO<sup>4</sup> of each digit column. If one of these should provide an output it will be applied to GTO<sup>7</sup> which will be open (only single fault possible), so that an in-out alarm will be given. Then, to make cert in that each digit of COR can contain 1 (and that COR check can contain 0), this delayed completion pulse is further delayed by 0.1  $\mu$ sec. and used to complement all flip-flops. It is delayed another 0.3  $\mu$ sec and applied to GTO<sup>7</sup> and to all GT's O<sup>4</sup> in series. It is applied to GTO<sup>7</sup> so that an in-out alarm is obtained if COR check contains 1. Since the remaining FF's should contain 1, the pulse applied to GTO<sup>4</sup> should eventually be applied to GTO<sup>2</sup> and emerge as a continue operation pulse. The continue operation pulse performs a number of functions; the one which is directly concerned with the checking process is setting COR check to 1. If the continue operation pulse does not emerge, COR check will remain in 0 (if it were in 1, an alarm would have already been obtained). Thus, after an additional delay of 1.3  $\mu$ sec., the completion pulse is used to sense GTO<sup>6</sup>. If COR contains 0, GTO<sup>6</sup> will be open and an in-out

alarm is obtained. This same type of predetermined check is also used in checking the check register (check-register check).

Test sequences will be designed for the flip-flops of ICR (making use of the facilities provided by COR) and the flip-flops of in-out control (IOC). There are eight voltage-variation lines supplying the 0 sides of the flip-flops of ICR (each line supplies two digit columns:  $n$  and  $n+8$ ). Eight more supply the 1 sides of these flip-flops. These same lines also supply the flip-flops of all the repetitive units (e.g. AR, BR, AC, BC, PR, etc.). One voltage-variation line (151) supplies the 0 sides of all the flip-flops of IOC (except FF410.06 and FF410.07, which are not standard flip-flops). The 1 sides of these flip-flops are supplied by voltage-variation line 152. The two special flip-flops are supplied by voltage-variation lines 160 (0 side) and 161 (1 side).

#### 5.121 In-Out Register

Although the digit columns are supplied by several voltage-variation lines, it is convenient to treat the entire register as a single section. Moreover, it may be treated as a channel segment. The test sequence design makes use of COR to obtain an alarm when a failure exists. This technique requires that at least one of the digit columns of COR contain 1 after the completion pulse commands add ICR to COR when there is a failure in ICR. An alarm pulse would then be obtained when the delayed completion pulse senses (TO) of that digit column.

When a failure occurs in one of the flip-flops, it is not possible to predict the behavior of those flip-flops to its left with any degree of certainty. This is largely because the shifting gate tubes

(GT06 and GT07) are a-c coupled to their flip-flops. Should a flip-flop remain in one position permanently, both gate tubes would be open. Thus, the next flip-flop to the left will be pulsed on both the clear and set inputs almost simultaneously. In all probability, this will cause it to trigger, but there definitely is doubt as to its actual behavior. It is also important to note that the command read shift applies a pulse to GT10 on the 0 side of FF410.04. The output of the GT is used to shift COR. If IOR 0 should transmit a pulse on the IOR one out line, FF410.04 would be set to 1 and the contents of COR would no longer be shifted. Thus, a failure of one of the flip-flops of IOR may also affect the shifting of COR. The maximum possible number (and, indeed, the correct number) of times that the contents of COR will be shifted is 16, the number of shift pulses provided by the film unit. It should also be noted that COR will be shifted at least once (a shift pulse is required to produce a pulse on the IOR one out line).

To check IOR, it is only necessary to specify words to be read from the film. These words must be chosen so as to avoid the possibility of coincidental agreement of the contents of IOR and COR. We will discuss the choice of these words when COR is not shifted the correct number of times separately from the choice when COR is shifted correctly. If COR is shifted correctly, it will contain the correct word. If one of the flip-flops of IOR always transmits a 1, the word should contain a 0 in that digit in order that COR be provided with 0 in that digit before the command add IOR to COR. To check every digit column of IOR,

every digit of the word should be 0. Thus, one of the words to be read is +0. If, on the other hand, one of the flip-flops of IOR always transmits a 0, the word should contain a 1 in that digit column. Hence, the word -0 should also be read.

If COR is not shifted the correct number of times, COR 15 will be used to obtain the alarm. Note that COR 15 will contain one of the first fifteen digits of the word read, depending on the number of shifts. IOR 15, on the other hand, will always contain the sixteenth digit of the word read (unless it is the faulty digit). Thus, if we read a word in which the sixteenth digit is different from the first fifteen, we will get the desired alarm (assuming IOR 15 is not at fault). This gives us the choice of using either  $+2^{-15}$  or  $-2^{-15}$  for this purpose. Now, suppose that IOR 15 always transmits 1. If  $+2^{-15}$  is used, it will provide COR 15 with 0 and the alarm is obtained. If, on the other hand, IOR 15 always transmits 0,  $-2^{-15}$  will provide COR with 1, yielding the alarm. Thus, both words should be used. A little thought will show that these two words would also be satisfactory when the correct number of shifts is performed, so that +0 and -0 are unnecessary. In fact, any two words which are the complements of each other will provide the desired check when the correct number of shifts is performed.

Thus far, IOF auxiliary has not been considered. It is a rather special case, for, if it has failed, the content of every digit of IOF is indeterminate. However, COR would be shifted the requisite 16 times and would therefore contain the word being read. There is some possibility that IOF would also contain this word for every read operation, but

its probability is quite negligible. For practical purposes, it may be assumed that IOR auxiliary is checked.

#### 5.122 In-Out Control Flip-Flops

In-out control is also largely self-checking, regardless of the words being transferred between storage and the remote equipment. It is considered as a single section, although it is composed of two sections. The major exception to this statement is provided by a failure in FF410.06, which controls the supply of restorer pulses to the in-out element. It is necessary to stop the flow of restorer pulses to IOE during the operation of the remote equipment because it is not synchronized with the rest of the computer. This is the function of FF410.06. If it fails to stop this flow, it is not possible to predict the behavior of IOE. In all probability, this situation will be detected, but the uncertain time relations make it impossible to carry out any analysis. If FF410.06 never permits the restorer pulses to flow to IOE, a similar difficulty of analysis obtains. The loss of restorer pulses has been discussed in section 3.224.

The analysis of the remaining portion of IOC is quite straightforward. However, it involves a tremendous amount of detail which would not add very much to the understanding of the method being used. It requires a thorough knowledge of the interaction of the remote equipment and the in-out element proper. These details have been omitted here. It turns out that only a very simple sequence is required to check IOC. This sequence has been combined with that required for checking IOR and is presented below. One point worth noting is that sufficient

time must be allotted for the completion of an re operation in order to check the ability of FF<sup>10.04</sup> to transmit 1. The instructions in A5 and A6 have been used to obtain a delay of approximately 1280 uses. (with TS) for this purpose.

Instructions

A1 rf K1

A2 re B2

A3 ca B5

A4 ts C1

A5 ac C1

A6 op A5

A7 re --

A8 rf K2

A9 rd C2

A10 rd C3

A11 rd C4

A12 rd C5

A13 ca C2

A14 qc B1

A15 ca C3

A16 qc B2

A17 ca C4

A18 qc B3

A19 ca C5

A20 qc B4

A21 re --

A22 op A1

Constant Storage

B1 + 0

B2 - 0

B3 +  $2^{-15}$ B4 -  $2^{-15}$ B5 -  $2^{-6}$ Variable Storage

(FF) C1 (counts delay - see A5 and A6)

(FF) C2

(FF) C3 store numbers read

(FF) C4

(FF) C5

Remote Equipment

K1 - recorder

K2 - reader - film for use with

this reader should contain

blocks of four words separated

by a distance which permits the

reader to stop before it reaches

the next block. Each block should

contain +0, -0, + $2^{-15}$ , and- $2^{-15}$ , in that order.

### 5.13 Step Counter

The step counter (SC) is that part of arithmetic control which counts the number of shifts which are performed in the arithmetic processes which require the use of arithmetic control (mr, mh, dv, sl, sr, and sf). In all these operations (with sf nearly always an exception), SC is the device which provides the end-carry pulse which returns control to central control. SC must be reset (or preset) in all operations (except sf) so that the end-carry is obtained after the proper number of shifts. A block diagram of SC is shown in Fig. 20. Block schematics, which show the individual stages, are given in Fig. 21 and Fig. 22. We will design test sequences which may be used for the entire step counter.

There are 11 different voltage-variation lines which supply SC. They are (stage numbers refer to the block schematics):

<u>Line No.</u>	<u>Application</u>	<u>Stages Affected</u>
77	FF screens (0)	FF 10, 11, 12, 13, 14, 15
78	FF screens (1)	FF 10, 11, 12, 13, 14, 15
113	GT screens	GT01 (5 digits), 02 (6 digits) 04 (5 digits), 05, 307.06, 307.07
199	GT control grids	GT01 (5 digits), 02 (6 digits) 04 (5 digits), 05, 307.06, 307.07
221	GT suppressors	GT02 (6 digits), 04 (5 digits) 05, 307.06, 307.07
200	GT suppressors	GT01 (5 digits)
	RA control	RA V1 (step counter output, Fig. 22)
97	TT Plates	TT V205, 305, 405, 505, 605, 705
	IA screens	IA V101, 201, 301, 401, 501, 601 (Fig. 22)
	HD screens	HD V102, 202, 302, 402, 502, 602 (Fig. 22)



96	GC plates and screens	GC V102 (V102 cathode)
17	RA plates and screens	RA V103, 104, 105, 106, 704, 801, 804, 805
	GC plates and screens	GC V101
	GT plates	GT 04 (6 digits)
	RD plates	RD V102, 202, 302, 402, 502, 602 (Fig. 22)
		RA V1 (Fig. 22)
201	RA control	RA V104, 105, 106, 704, 801, 804, 805
		RA V101, 201, 301, 401, 501, 601 (Fig. 22)
141	RA control	RA V103
	RD control	RD V102, 202, 302, 402, 502, 602 (Fig. 22)

It is clear that lines 77 and 78 define the same section. Line 113 defines a different section. It may be shown that the sections defined by all the other lines (with some few exceptions) may be considered as part of the section defined by line 113. This is clear for lines 199 and 221. In addition to GT01 (which is supplied by line 113), line 200 supplies buffer amplifier RA V1. However, a failure of RA V1 will affect GT02 (6 digits), so that a test sequence designed to check GT02 will discover the presence of this failure. Similarly, for line 97; failure of the trigger tubes (TT) will affect the read-in process and is checked with GT01, while failures of the buffer amplifiers and bus drivers (RD) are checked with GT02. The failure of the gate generator (GG) supplied by line 96 will be checked with GT01 (see section 6.11). With the exception of RA V103, 104, 105, 106, 704, 801, 804, and 805, the stages found on lines 17, 201, and 141 have been discussed. The failures of

RA V10<sup>4</sup>, 105, 106, and 70<sup>4</sup> will be checked with GT05. The failure of RA V103 will be checked with GT01. The failure of RA V801 will affect the restorer-pulse supply and will not be considered (see section 3.224). Thus every stage except RA V80<sup>4</sup> and 805 will be checked by the sequence designed for line 113. The failure of these buffer amplifiers will affect the information supplied to FF306.01 and FF308.01, and will therefore be checked by Test Sequence Number II. Nevertheless, they will also be considered in connection with lines 77 and 78.

#### 5.131 Step Counter Flip-Flops

The most convenient channel segment uses all of the flip-flops, with FF15 transmitting information to FF14, FF14 transmitting it to FF13, etc. Information is supplied to this channel segment by the command add to SC. The way to check that all flip-flops can transmit both 0 and 1 is to start it from the clear position and have it count until its maximum capacity is exceeded. This can only be accomplished by ordering either sl 31 or sr 31. Either of these operations would leave 0 in AC, the sign depending on the sign of the number that was in AC before the shift operation. Since this is a case in which the check depends on the manipulation of numbers, we must observe the effect on a failure on these operations. If one of the flip-flops cannot transmit 1, the SC end-carry pulse would never be obtained, resulting in a "prolonged stop clock". This condition is therefore checked. If one of the flip-flops cannot transmit 0, it is as though it were by-passed. The capacity of the counter is reduced by a factor of 2, so that the end-carry would be obtained after 15 shifts. If BR were clear before the

start of the shift operation and if sl 31 were ordered, then AC would contain 0 (with appropriate sign) after the 15 shifts regardless of its original contents. Thus, sl 31 does not provide the desired check. However, if AC originally contained  $1/2$  and sr 31 were used, BRO would contain 1 after 15 shifts. The subsequent round-off would put  $2^{-15}$  in AC so that this would provide the desired check. The sequence:

ca RC(1/2)  
sr 31  
qc RC(+0)

is sufficient to check all the flip-flops of SC.

The ability of SC to be reset (and preset) deserves some consideration. The amplifiers which drive the two reset lines (BA V804 and 805) are checked with arithmetic control while the preset line is driven directly by a control-pulse output unit and should be considered with that section. Nevertheless, the crystal rectifier diodes which connect these lines to the flip-flops might fail, so that a test sequence should be designed to check the ability of SC to be reset and preset. It was felt that it would be suitable to include this sequence with the other sequence designed for lines 77 and 78, partially because of the shortness of the other sequence and partially because the ability of the flip-flops to receive information from these sources is affected by voltage variation on this line. It would be well to use this sequence with the other voltage-variation lines affecting the SC resets and preset.

The test sequence is quite easily designed. First the preset was considered. The failure of any digit to preset properly would result in the wrong number of shifts, regardless of the number specified in the

instruction (the single fault assumption is still used.) Since SC is preset to 011111 under normal conditions, it would be desirable to have it contain 100000 before the preset pulse in order to check all digits at once. However, it is not possible to perform an operation that would leave this quantity in SC. The best we can do is to use two sequences, one testing the ability of FF's 11, 12, 13, 14, and 15 preset properly, the other testing this ability of FF10. For the first case, SC is made to contain 000000 before presetting by performing the af operation with 1/2 originally in AC. For the second case, SC is made to contain 100001 before presetting by performing the af operation with +0 originally in AC. The preset takes place on the ca operation, but this does not affect validity of the check on FF10. The particular numbers were chosen for economical use of storage. The sequence follows:

```

ca PC(1/2)
af x
sr 6
qc PC(2-7)
ca PC(+0)
af x
ca PC(1/2)
sr 14
qc PC(2-15)

```

(x is the address of any convenient register)

To check the multiply and divide resets, it is only necessary to perform a multiplication and a division. The operands were chosen for economy of the required storage, and the tedious process of determining the results when an incorrect number of shifts is obtained was carried out. It was found that these operands are satisfactory. The sequences are:

Multiply Reset

ca RC(2<sup>-7</sup>)  
 mr RC(2<sup>-7</sup>)  
 qc RC(2<sup>-14</sup>)

Divide Reset

ca RC(2<sup>-15</sup>)  
 dv RC(2<sup>-1</sup>)  
 qc RC(+0)  
 sl 15  
 qc RC(2<sup>-14</sup>)

The complete sequence, written in a form suitable for use with test storage, is given in Appendix II as Test Sequence Number I. An additional check on the performance of the af operations is included, though it is not needed.

5.133 Step Counter Gate Tubes

The same general procedure may also be applied to the gate tubes. As discussed in section 3.222, the only checks that are included determine whether the gate tube is capable of passing or rejecting a pulse at the times when it is sensed. This procedure will also check the various amplifiers discussed in section 5.13. There are no particular difficulties involved, but one example will be given because this is the first time that gate tubes have been discussed.

Consider GT01, the read-in gate tube. It is tested on TP5 of every operation, because the control switch is always clear at that time. However, the information is only used for the sl and sr operations. If we want to check all of these gate tubes at once, we could try to order sl 31 or sr 31. This would always leave 0 (with appropriate sign) in AC. If GT01 of TP5 failed to pass a pulse, the

effect would be to order sl 30 or sr 30, which would also leave 0 (with appropriate sign) in AC. Two steps must therefore be used, sr 16 to check GT01 of FF1 and sr 15 to check all other GT's 01 (sr was chosen in order that the initial contents of AC be significant). Thus, the sequence:

```
ca RC(1/2)
sr 16
qc RC(+0)
ca RC(1/4)
sr 15
qc RC(+0)
```

It should be clear that the alarm will be obtained from one or the other of the two halves of the sequence in case a failure exists. To check that all of these gate tubes do not always transmit a pulse, the obvious expedients of ordering either sr 0 or sl 0 will do. Hence:

```
ca RC( $2^{-15}$ )
sl 0
qc RC( $2^{-15}$ )
```

Note that the use of sr 0 with AC containing  $2^{-15}$  is not satisfactory.

This same technique may be applied to the other gate tubes. The complete sequence is given in Appendix II as Test Sequence Number III. A considerable amount of tailoring was required in coding this sequence to fit into test storage. The sequences developed here appear in somewhat altered form in Test Sequence Number III.

## 5.2 Multiple Faults - The Program Counter

A test sequence to check the program counter has been designed in order to illustrate the application of the procedure for multiple faults. A failure in the program counter affects the order

in which the instructions are performed, so that a good deal of attention must be given to this phase of the problem. Actually, there exists an excellent method for checking the program counter (as well as TS and TSS) based on a special order which has been temporarily wired into the control matrix.<sup>4</sup> However, this should not adversely affect the value of this test sequence in illustrating the procedure.

A block diagram of PC appears in Fig. 10 and its operation is briefly discussed in section 2.124. The voltage-variation lines involved will not be listed. They are the same lines that supply all repetitive units (e.g., AC, IOR, etc). Only 5 digits of PC (PC11 to PC15) will be discussed since operation with TS requires only these 5 digits. We will assume that the only GT failure possible is the inability to pass a pulse when it is sensed. Failures in the read-in gate tubes or the resets will not be considered. With this limitation, there is but one way of supplying information to PC: the command aid to PC on TP7 of program timing. Let us assume that PC has been cleared before the computer is started. It is also assumed that AC has been cleared. These conditions can be checked by observing the indicator lights before pressing the restart push button.

The computer will necessarily perform the instruction that is stored in register 0. The reason for the assumption that the gate tubes do not pass a spurious pulse when sensed should now be clear. If this possibility were admitted, the first instruction could conceivably be obtained from any of the 32 registers. The design would become unduly complicated. We will now discuss the design in detail.

Consider the first add to PC pulse. It will require PC15 to trigger from 0 to 1, thus providing a means of checking both the flip-flop itself and the trigger tube (not shown in the diagram) which amplifies the pulse. There are two possibilities:

1. This pulse will succeed in triggering the FF to 1.
2. The FF will continue to contain 0.

It is not possible to determine which event has taken place until the subsequent command, PC read out. If the read-out gate tube (GT02) has failed, the second instruction would be obtained from register 0 regardless of the behavior of the flip-flop. However, a multiple check on GT02 is provided by GT03, so that an alarm would be obtained on the transfer check. If both GT02 and GT03 should fail simultaneously, the transfer check would indicate agreement, and the instruction contained in register 0 would indeed be performed twice in succession. The same sequence of instructions would result from case 2. In order to set an alarm in these two cases, it is necessary to store an appropriate instruction in register 0. This instruction should yield an alarm when performed twice in succession. Such an instruction is ad RC(1/2).

The addition of 1/2 to 1/2 will result in an overflow and give an arithmetic check alarm. If no alarm is given, the second instruction will be taken from register 1 and it may be concluded that the flip-flop is capable of triggering and that both gate tubes can transmit a sensing pulse.

The second add to PC pulse (which will not occur unless the previous functions were performed correctly) will require that three



functions be performed:

1. PC15 trigger to 0
2. GT05 of PC15 transmit a sensing pulse
3. PC14 trigger to 1.

There is a slight possibility that a flip-flop which can trigger to 1 cannot trigger to 0, but consideration of this possibility greatly complicates the design of the test sequence. The consideration of this possibility will be postponed until the end of this section. If GT05 does not transmit the pulse, PC 14 cannot possibly be triggered. Hence, the contents of PC after this second add to PC pulse will be either 0 or 2. Again, it is not possible to determine which event occurred until the subsequent command, PC read out. The failure of the read-out gate tube (and also GT03) in this digit column will cause the instruction in register 0 to follow the instruction in register 1, regardless of the behavior of the flip-flop. If the flip-flop failed to trigger, this same situation would be obtained. Since register 0 contains ad (RC1/2), and arithmetic check alarm will be obtained if register 1 contains ca RC(1/2). If no alarm is given, the next instruction is taken from register 2.

The next add to PC pulse will change the contents of PC to 3, for it has already been demonstrated that PC15 is capable of triggering. Furthermore, the command PC read out will yield the proper read-out. Thus, the instructions will be obtained from register 0, 1, 2, and 3 in succession. The instruction contained in register 2 may therefore be fairly arbitrary, subject to the condition that it will not cause an alarm when performed in the proper order.

The fourth add to PC pulse will change the contents of PC14 and PC15 to 0 and attempt to change the content of PC13 to 1. If it fails, the next instruction is taken from register 0, as would be true if the read-out gate tube of PC13 fails. To obtain an alarm when this failure occurs, it is necessary to use an instruction in register 3 which leaves leaves 1/2 in AC. A suitable sequence thus far is:

```

0 ad RC(1/2)
1 ca RC(1/2)
2 qc RC(1/2)
3 qc RC(1/2)

```

If this failure does not exist, the next instruction is correctly taken from register 4.

Continuing to reason in the same way, it is found that the instructions in registers 4, 5, 6, 7 will necessarily be performed in the proper order. Register 7 should contain an instruction which will leave 1/2 in AC so that an alarm will be obtained if the next instruction is taken from register 0 instead of register 5. Similarly, register 15 should contain an instruction which leave 1/2 in AC. The contents of other registers may be chosen more or less arbitrarily, subject to the condition that they do not cause an alarm when performed in the normal order.

Now let us discuss the modifications which are necessary if the possibility of a PF failing to trigger from 1 to 0 is considered. We return to the second add to PC pulse. If this failure exists, PC will contain either 1 or 3 after this pulse. The subsequent PC read out command will cause the next instruction to be taken from register 1 in the first case and from either register 1 or 3 in the second case, depending

on whether GTO2 and GTO3 have failed or not. If it is taken from register 1, the instructions in registers 0, 1, and 1 will be performed in that order. Register 1 should contain an instruction which yields an alarm when performed twice after the instruction in register 0, but not when it is performed once after the instruction in register 0. Furthermore, it should be such that an alarm is obtained when the instructions are performed in the order 0, 1, 0. The instruction ad PC(1/4) satisfies these requirements. The other possibility is that the next instruction is taken from register 3; the instructions are then performed in the order 0, 1, 3. Suppose that register 3 also contains ad PC(1/4). With this order of instructions, an alarm is obtained. If none of these faults is present, then the instructions will be performed in the correct order, i.e., 0, 1, 2, 3. In order to avoid an alarm in correct operation, register 2 should contain su PC(z), where z is greater than 0.

By continuing this reasoning to the other digits of PC, it is possible to obtain a complete sequence on the basis of the assumptions. The details become highly involved, though, and will not be given here. The process of determining appropriate operands (e.g. the quantity z mentioned above) to be used with the various instructions eventually requires the solution of simultaneous inequalities. A set of solutions, using numbers which are expressed simply in binary notation, was obtained by trial and error. The sequence which is based on this set of solutions is given in Appendix II as Test Sequence Number IV. Again, only the contents of certain registers need be specified to obtain this check.

The others may be used for any purposes desired. The choice of the registers to hold the operands was made quite arbitrarily. The two sp instructions were used to provide continuity in the coded program. Their use requires that the PC read-in gate tubes be working properly.

### 5.3 Trouble Location - Control-Pulse Output Units

The test sequences designed in this thesis may be used in trouble-location procedures. When an alarm is obtained during the performance of a test sequence, a failure is known to exist. The type of alarm indication, together with the contents of the various registers, should provide information which simplifies the task of locating the trouble. In addition, if this alarm indication is not obtained with the normal supply voltages, but only under marginal checking conditions, it is reasonable to assume that the defective component is supplied by the particular voltage-variation line involved. If the margin thus obtained is abnormally low, the defective component should be found and repaired (or replaced).

There are two closely related approaches to the use of information provided by test sequences in trouble-location procedures. One is to compare this data with a list of the symptoms manifested by the various possible failures. Such a tabulation may be obtained by determining the computed results in the presence of each possible failure. If the information obtained from the performance of a test sequence matches one of the tabulated sets of symptoms, the fault is fairly well located. However, it is quite possible that the alarm was caused by a

combination of failures not considered in the compilation. This leads us to the second approach to trouble location, deduction based on the available data. Deduction may be used to supplement a list of symptoms, or it may be used as a completely independent method of trouble location. It requires highly skilled personnel and, on the average, is more time consuming than the use of a list. While the tabulation of symptoms is a tedious and unattractive task, it makes it possible for relatively unskilled personnel to handle routine problems in trouble location. Highly skilled personnel may be called in when the list proves inadequate.

The use of both of these approaches will be illustrated in connection with Test Sequence Number V (see Appendix II), which has been designed to check the control-pulse output units (see Fig. 18) under the single fault assumption. The design details have been omitted, but they are quite straightforward. Test Sequence Number V does not apply to all these units because the 32 registers of test storage are inadequate for all the short sequences which are required.

To illustrate the first approach, assume that unit number 1 will not pass a pulse when sensed. The pulse which it normally provides is used to read out the contents of the step counter on TP7 of the sf operation (see Fig. 18). This failure would thus result in 0 being stored as the supposed scale factor. The only sf operation is stored in register 12. The scale factor which is stored as a result of this operation is checked by the gc instruction in register 14. The check is not performed until  $TT\frac{1}{2}$ , so that PC will contain 15 at that time.

Digit columns 15, 14, and 12 of the check register will have received 1 from register 3 but not from AC. The check register will be complemented as part of check register check, but will not be recomplemented because the series path of gate tubes will not transmit a pulse. Thus, CR will contain 0 in digit columns 15, 14 and 12 and 1 in all others. An alarm pulse will be obtained which will go to the alarm indicator and CPC, but it may be that TP1 will have occurred before the alarm pulse arrived at CPC. Thus TPD may contain either 2 or 1. AC will contain +0 as will register 8. The contents of other registers are not immediately relevant.

Now suppose that the symptoms described above are observed during voltage variation. Using the second approach, we can deduce that the failure is caused by a control-pulse output unit. Further, it is apparent that AC and register 8 contain +0 when they should contain  $11 \times 2^{-15}$ . This means that register 8 did not receive the scale factor on the instruction sf 8. This could be due to the failure of 50 to read out or of register 8 to read in (or both). Either unit number 1 failed to provide the command SC read out or unit number 67 failed to provide the command storage read in (rt.11). Now, unit number 67 is also used on the ta and td operations, and, as it happens, there is an instruction using ta stored in register 11. If 67 were the defective unit, register 9 would contain +0. If 1 were the defective unit, register 9 would contain its correct contents,  $31 \times 2^{-15}$ . This reasoning assumes only single faults.

If multiple faults were present, the problem becomes considerably more complicated. If the multiple fault results in symptoms which can be explained by a single fault (e.g., the failure of units 1 and 10 to provide a pulse would have the same effect as the failure of unit 67 to provide a pulse), the single fault should be investigated, either by replacement of the supposedly defective unit or by observation of appropriate waveforms. If the unit appears to be operating satisfactorily, then the next step is to investigate these combinations of faults which explain the symptoms. This is also the procedure to be followed when a multiple fault yields symptoms which cannot be explained by a single fault.

Further examples of the use of test sequences in trouble location are given in sections 6.11 and 6.12.

6 - RESULTS AND CONCLUSIONS6.1 Experimental Results

Now that the proposed method of designing test sequences has been discussed in some detail, we may well question their usefulness. It is quite conceivable that the assumptions involved in their design severely limit their applicability in all practical situations. The final answer can only be obtained from the actual use of the test sequences. In order to get some indication of their value, some time was spent in performing them on WWI. Extensive tests could not be carried out because only a limited time was available. The results are therefore only suggestive, not conclusive.

The most logical way of determining the effectiveness of a test sequence is to cause a failure in one of the stages it is designed to check. The voltage-variation facilities provide a convenient means for causing such a failure. Measurements were made of some of the margins available with some sample problems. Several problems used for testing and demonstration purposes were included to provide a basis for a comparison. Test Program Number I causes each flip-flop (except some in control) to receive both 0 and 1.<sup>2</sup> Test Program Number II performs each of the operations which have been accepted for permanent use in WWI.<sup>3</sup> Display Program Number I consists of two separate problems: one displays the curves  $y = x$ ,  $y = x^2$ , and  $y = x^3$  on an oscilloscope, the other displays a family of parabolas.<sup>7</sup> The measurements have been tabulated in Fig. 23.

The margins obtained with Test Program Numbers I and II did not differ materially from those obtained with the test sequences.



This indicates that the test programs are about as effective as the test sequences for the types of failures that occurred. On the other hand, the margins obtained with Display Program Number I differed significantly from the margins obtained with the test sequences. In general, Display Program Number I had larger margins than the test sequences, indicating that the test sequences are more effective. However, the negative margin on line 79 with Display Program Number I was about half the margin with Test Sequence Number II. The low margin was traced to a defective flip-flop in the divide-pulse distributor (see section 6.12), but Display Program Number I does not use the divide operation at all! This points up the need for further study of the factors which cause failures, particularly with regard to the influence of voltage variation. The most plausible explanation for the difference in the values of the margin is that a dynamic failure existed which was accentuated by the frequency with which the multiply operation is performed in Display Program Number I. Perhaps the fact that the margin was measured by noting the voltage at which the pattern on the display oscilloscope became distorted (no alarm was obtained) had some influence on this difference. In other words, mistakes in computation might appear before an alarm is obtained. Further investigation to obtain a more satisfactory explanation could not be carried out.

A clear-cut example of the inadequacy of Display Program Number I for checking purposes (of course, it was not intended as a check problem) is provided by the negative margin on line 113. The margin obtained with Display Program Number I was about twice that obtained

with Test Sequence Number III. This low margin was traced to a defective read-in gate generator in the step counter (see section 6.11). Display Program Number I does not use the shift operation and hence is not affected by this failure.

Of some interest is the difference between negative and positive margins on line 113. It indicates that the failure of a gate tube to pass a pulse is far more likely than the generation of a spurious pulse. Also of interest are the differences between positive and negative margins on lines 79 and 80. It has been more or less standard practice to use the positive margins as an indication of the condition of a flip-flop when the complementing technique (see section 3.3) is employed. The positive margins on these lines were more than adequate, but some of the negative margins were quite low. This suggests that negative margins might well be taken into account in the complementing procedures.

The most significant result of the use of the test sequences is their application to the location of the causes of the low margins. This is described in detail below.

#### 6.11 Discussion of Low Margin on Line 113

In measuring the margin on line 113, with Test Sequence Number III, it was noted that the alarm was consistently obtained on the gc instruction contained in register 10. The check register showed that digit column 1 was at fault. AC contained +1/2 instead of +0, indicating that no shift was performed. Since other shift operations were performed successfully, the fault was probably due to the failure of the read in

gate tube of digit column 11 to pass a pulse. An oscilloscope probe placed on the suppressor of this gate tube showed the gating signal had an unusually low amplitude. Furthermore, the amplitude of this gate varied with the voltage applied to line 113. This was a rather surprising result because the gate generator was not supplied by line 113. However, further research showed that a wiring mistake had been made. The screen of 6C V102 (see Fig. 21), which should have been supplied by line 96, was actually supplied by line 113. This substantiates the reasoning of section 5.13 in which it is shown that a failure on line 96 will be detected by this sequence. It is interesting to note that this mistake did not affect operation at normal voltage, even though the screen was supplied with +90 volts instead of +120. The entire investigation, from the time the results were analyzed to the time the correction in wiring was made, took approximately 45 minutes. This is less than the arbitrary standard of 1 hour discussed in section 1.2 and is considered fairly noteworthy.

#### 6.12 Discussion of Low Margin on Line 79

In measuring the margin on line 79 with Test Sequence Number II, it was noted that the alarm was consistently obtained on the gc instruction contained in register 29. The check register indicated that the discrepancy was in digit column 1. This was confirmed by AC which contained 0 instead of +1/2. This could most easily be accounted for by a failure to shift left. However, since only the flip-flops of arithmetic control were undergoing voltage variation, the failure to shift would mean that FF307.01 cannot transmit 1. This would result in

a "prolonged stop clock", not a check register alarm. To make this hypothesis even more untenable, there was another sl instruction in register 15 which apparently gave no trouble. The only other conclusion was that there was something wrong with the divide operation. It could not be the failure of FF308.01 to transmit 1, for that would result in a "prolonged stop clock". The only other possible source of error was the divide-pulse distributor, but the results obtained were not any of those which had been calculated for steady-state failures of the divide-pulse distributor. The complementing check was the performed on these flip-flops and FF308.04 was found to have a low margin. The tubes were replaced and the margin (with Test Sequence Master II) was increased from -17 volts to -31 volts. Furthermore, the alarm was obtained at a different point in the sequence. The only possible explanation of the inconsistent results is that the fault was not steady-state, even at the reduced voltage. The time required to correct this situation was about 75 minutes. Some time could undoubtedly have been saved if the complementing procedure had been used immediately.

## 6.2 Evaluation of the Design Procedure

The previous sections of this chapter have discussed the results obtained from the use of some of the test sequences. Even though these investigations were very limited in scope, some very useful information was obtained, particularly regarding trouble location. The test sequences apparently do accomplish their avowed aim, and indeed, even exceed expectations. The location of a non-steady-state failure in the divide-pulse distributor, despite the fact that an alternate method

were used for confirmation, was a notable achievement. A complete set of test sequences would be a powerful tool in achieving computer reliability. Can the design procedure outline in this thesis be used to obtain such a set? In theory, at least, the answer is apparently yes. The limitations seem to be of a practical nature and concern the amount of work required for some of the more complicated channel segments. The effort required in some of these cases may not justify the results. It would probably be more efficient to devise alternate schemes. A certain amount of study should precede the design of a test sequence in order to determine whether the effort might better be expended in other directions.

The efficiency problem is most likely to be serious for those channel segments which control the manipulation of information in other parts of the computer. The time-pulse distributor is perhaps the greatest offender in this respect, since it controls most of the operation of the computer. A good deal of analysis is required to determine whether an alarm is actually obtained in the presence of a failure. An indication of the magnitude of this task may be had from the fact that about 25 hours were required to determine whether an alarm would be obtained with program timing only (assuming all registers clear and operation 0 giving program timing) for an open circuit in each of the crystals of the matrix. Incidentally, it is interesting to note that an alarm would be obtained for only 6 of the 24 crystals. In cases of this nature, it is probably most efficient to circumvent the detailed analysis by the installation of a built-in checking device (such a device has been proposed for the time-pulse distributor). However, it seems likely that the procedure is not unduly complicated in most cases. Of course, the

complexity increases rapidly when multiple faults are considered.

Test sequences designed for complicated channel segments are of dubious value in trouble-location procedures. In many cases the test sequence will only specify the channel segment in which the failure exists. The knowledge that a failure exists somewhere in a large number of stages is not particularly helpful. Such information could probably be obtained much more simply. The effort of designing a test sequence is hardly justifiable.


### 6.3 Suggestions for Further Work

The possibilities of test checking have not been fully explored in this thesis by any stretch of the imagination. A much simpler approach to the problem might very well exist. If such an approach could be found, it would certainly be a great boon to the people who are concerned with designing test problems.

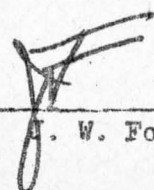
Regardless of whether a new approach can be found, much work remains to be done. Quantitative investigations of the factors influencing failures are extremely important. The design of test sequences should take into account as many of these factors as possible. Of course, there are the tasks of actually designing test sequences for the entire computer and analyzing the results for trouble location purposes.

In conclusion, it should be emphasized that the method presented in this thesis can undoubtedly be used for designing test sequences. The results obtained are encouraging and serve as a starting point for future study which should be directed towards simpler and more efficient methods.

Signed

  
Gerald Cooper

Approved

  
J. W. Forrester

APPENDIX I

A Short Guide to Coding.

**A SHORT GUIDE TO CODING**

**Using the Whirlwind I Code of October 1949**

**Electronic Computer Division  
Servomechanisms Laboratory  
Massachusetts Institute of Technology  
Cambridge, Massachusetts**



A SHORT GUIDE TO CODING  
(using the Whirlwind I code of October 1949)

## COMPUTER PROGRAMS

**Program.** A program is a sequence of actions by which a computer handles a problem. The process of determining the sequence of actions is known as programming.

**Flow diagrams.** A flow diagram is a series of statements of what the computer has to do at various stages in a program. Lines of flow indicate how the computer passes from one stage of the program to another.

**Coded program.** Programs and flow diagrams are largely independent of computer characteristics, but instructions for a computer must be expressed in terms of a code. A set of instructions that will enable a computer to execute a program is called a coded program, and the process of preparing a coded program is known as coding.

**Orders and operations.** Individual coded instructions are known as orders and call for specific operations such as multiply, add, shift, etc.

**The computer code.** The computer code described here is that of Whirlwind I, an experimental computer using binary digits, single-address order code, parallel operation, and electrostatic storage. It is expected that computers of this type will ultimately achieve an average speed of 50,000 operations per second.

## COMPUTER COMPONENTS

**Registers and words.** A register has 16 digit positions each able to store a one or a zero. A word is a set of 16 digits that may be stored in a register. A word can represent an order or a number.

**Arithmetic element.** Arithmetic operations take place in the arithmetic element, whose main components are three flip-flop registers, the A-register, the accumulator, and the B-register (AR, AC, BR). The 16 digit positions of AR starting from the left are denoted by AR 0, AR 1, . . . , AR 15. Similarly for AC, BR. Words enter AC through AR; BR is an extension of AC.

**Storage.** The term "register" by itself refers to the main electrostatic storage, which consists of  $2^{11}$  or 2048 registers, each of which is identified by an address. These addresses are 11-digit binary numbers from 0 to 2047. The computer identifies a register by its address.

**Input-output.** All information entering or leaving the computer is temporarily stored in the input-output register (IOR). The computer regulates the flow of information between the internal storage and IOR, and also calls for any necessary manipulation of external units. The descriptive names of the input-output orders were chosen for photographic film reader-recorder units, but the orders are applicable to other types of external equipment.

**Control element.** The control element controls the sequence of computer operations and their execution. Instructions are obtained from storage in the form of individual orders, each of which is represented by a single word.

**Inter-connections.** The four main elements (storage, control, arithmetic, and input-output) are connected by a parallel communications system, known as the bus.

## REPRESENTATION OF ORDERS

**Operation section.** When a word is used to represent an order the first (left-hand) 5 digits, or operation section, specify a particular operation in accordance with the order code.

**Address section.** The remaining 11 digits, or address section, are interpreted as a number with the binary point at the right-hand end. In the majority of orders this number is the address of the register whose contents will be used in the operation. In orders sl, sr, the number specifies the extent of a shift; in rf, rb, the number specifies an external unit; in ri, rs, the address section is not used.

**Example.** The order ca x has the effect of clearing AC (making all the digits zero) and then putting into AC the word that is in the register whose address is x. If q is a quantity in some register, the order needed to put q in AC is not ca q but ca x, where x is the address of the register that contains q.

## REPRESENTATION OF NUMBERS

**Single-word representations.** When a word is used to represent a number the first digit indicates the sign and the remaining 15 are numerical digits. For a positive number the sign digit is zero, and the 15 numerical digits with a binary point at their left specify the magnitude of the number. The negative -y of a positive number y is represented by complementing all the digits, including the sign digit, that would represent y. (The complement is formed by replacing every zero by a one and every one by a zero.) In this way a word can represent any multiple of  $2^{-15}$  from  $2^{-15} - 1$  to  $1 - 2^{-15}$ . Neither +1 nor -1 can be represented by a single word. Zero has two representations, either 16 zeros or 16 ones, which are called +0 and -0 respectively.

**Overflow — increase of range and accuracy.** With single-word representation the range is limited to numbers between  $2^{-15} - 1$  and  $1 - 2^{-15}$ . Programs must be so planned that arithmetic operations will not cause an overflow beyond this range. The range may be extended by using a scale factor, which must be separately stored. Accuracy can be increased by using two words to represent a 30-digit number.

## COMPUTER PROCEDURE

**Sequence of operations.** After the execution of an order the program counter in the control element holds the address of the register from which the next order is to be taken. Control calls for this order and carries out the specified operation. If the order is not sp or cp(-) the address in the program counter then increases by one so that the next order is taken from the next consecutive register. The sp and cp(-) orders permit a change in this sequential procedure.

**Transfers.** A transfer of a digit from one digit position to another affects only the latter digit position, whose previous content is lost.

**Negative zero.** The subtraction of equal numbers produces a negative zero in AC, except when AC contains +0, and -0 is subtracted from it.

**Manipulation of orders.** Words representing orders may be handled in the arithmetic element as numbers.

**Procedure in the arithmetic element.** The execution of an addition includes the process of adding in carries; this process treats all 16 digits as if they were numerical digits, a carry from AC 0 being added into AC 15. A subtraction is executed by adding the complement. Multiplication, division, shifting and round-off are all executed with positive numbers, complementing being performed before and after the process when necessary. For round-off the digit in BR 0 is added into AC 15.

## NOTATION FOR CODING

**Addresses.** A coded program requires certain registers to be used for specified purposes. The addresses of these registers must be chosen before the program can be put into a computer, but for study purposes this final choice is unnecessary, and the addresses can be indicated by a system of symbols or index numbers.

**Writing a coded program.** Registers from which control obtains orders may be called action registers, and should be listed separately from registers containing other information, which may be called data registers. A coded program is written out in two columns; the first contains the index number of each action or data register, and the second column indicates the word that is initially stored in that register. In many cases part or all of a word may be immaterial because the contents of the register in question will be changed during the course of the program. This state of affairs is indicated by two dashes, for example, ca --.

**The abbreviations RC, CR.** Abbreviations used in referring to the register that contains a certain word or to the word in a certain register are

RC . . . = (Address of) Register Containing . . .  
CR . . . = Contents of Register (whose address is) . . .

**The symbol ri x.** When an address forms part of an order it is represented by the last 11 digits of a word whose first 5 digits specify an operation. An address x that is not part of an order is represented by the last 11 digits of a word whose first 5 digits are zero, which is equivalent to specifying the operation ri. Thus the word for an unattached address x may be written ri x. It could also be written  $x \times 2^{-15}$ .

THE ORDER CODE

AC = Accumulator AR = A-Register BR = B-Register

x is the address of a storage register; n is a positive integer; k designates an external unit

Order	Operation			Function
	Name	Code		
		Decimal	Binary	
ri --	read initially	0	00000	Take words from external unit until internal storage is full.
rs --	remote unit stop	1	00001	Stop external unit.
rf k	run forward	2	00010	Prepare to use external unit k in forward direction.
rb k	run backward	3	00011	Prepare to use external unit k in backward direction.
rd x	read	4	00100	Transfer to register x a word supplied by external unit.
rc x	record	5	00101	Arrange for transfer of contents of register x to external unit.
ts x	transfer to storage	8	01000	Transfer contents of AC to register x.
td x	transfer digits	9	01001	Transfer last 11 digits from AC to last 11 digit positions of register x.
ta x	transfer address	10	01010	Transfer last 11 digits from AR to last 11 digit positions of register x.
cp(-)x	conditional program	14	01110	If number in AC is negative, proceed as in sp; if number is positive disregard the cp(-) order, but clear the AR.
sp x	subprogram	15	01111	Take next order from register x. If the sp order was at address y, store y + 1 in last 11 digit positions of AR.
ca x	clear and add	16	10000	Clear AC and BR, then put contents of register x into AC. If necessary, add in carry from previous sa addition.
cs x	clear and subtract	17	10001	Clear AC and BR, then put complement of contents of register x into AC. If necessary, add in carry from previous sa addition.
ad x	add	18	10010	Add contents of register x to contents of AC, storing result in AC.
su x	subtract	19	10011	Subtract contents of register x from contents of AC, storing result in AC.
cm x	clear and add magnitude	20	10100	Clear AC and BR, then put positive magnitude of contents of register x into AC. If necessary add in carry from previous sa addition.
sa x	special add	21	10101	Add contents of register x to contents of AC, storing result in AC and retaining any overflow for next ca, cs, or cm order. Only orders 1 through 15 may be used between the sa order and ca, cs, or cm orders for which the sa is a preparation.
ao x	add one	22	10110	Add the number $1 \times 2^{-15}$ to the contents of register x. Store result in AC and in register x.
mr x	multiply and round off	24	11000	Multiply contents of register x by contents of AC; round off result to 15 numerical digits and store in AC. Clear BR.
mh x	multiply and hold	25	11001	Multiply contents of register x by contents of AC and retain the full product in AC and the first 15 digit positions of BR, the last digit position of BR being cleared.
dv x	divide	26	11010	Divide contents of AC by contents of register x, leaving 16 numerical digits of the quotient in BR and $\pm 0$ in AC according to sign of the quotient. (The order sl 15 following the dv order will round off the quotient to 15 numerical digits and store it in AC.)
sl n	shift left	27	11011	Multiply the number represented by the contents of AC and BR by $2^n$ . Round off the result to 15 numerical digits and store it in AC. Disregard overflow caused by the multiplication, but not that caused by round-off. Clear BR.
sr n	shift right	28	11100	Multiply the number represented by the contents of AC and BR by $2^{-n}$ . Round off the result to 15 numerical digits and store it in AC. Clear BR.
sf x	scale factor	29	11101	Multiply the number represented by the contents of AC and BR by 2 sufficiently often to make the positive magnitude of the product equal to or greater than 1/2. Leave the final product in AC and BR. Store the number of multiplications as last 11 digits of register x, the first 5 digits being undisturbed.

NOTES ON THE ORDER CODE

**Effect of operations.** The functions of the various orders are described above. It is to be assumed that AR, AC, BR, and the register whose address is x are undisturbed unless the contrary is stated.

**AR.** AR is primarily a buffer register for passing words into AC. After orders ca x, cs x, ad x, su x, sa x, and ao x it contains the number originally contained in register x. After orders cm x, mr x, mh x, and dv x it contains the magnitude of the contents of x. The effect of sp x and cp(-)x is stated above. No other order changes the contents of AR.

**BR.** A number stored in BR always appears as a positive magnitude, the sign of the number being assumed to be that indicated by the sign digit in AC. This convention has no effect on the logical result of the operations involving BR except that when BR contains a number that will be used later it is necessary to retain the appropriate sign digit.

**Alarms.** If the result of an arithmetic operation exceeds the register

capacity (i.e., if overflow occurs), a suitable alarm is given except as mentioned in connection with orders sa x and sl n.

**Shift orders.** A multiplication overflow in sl is lost without giving an alarm, but an overflow from round-off gives an alarm. Orders sr 0 and sl 0 only cause round-off, an alarm being given if an overflow occurs. The integer n is treated modulo 32, i.e., sl 32 = sl 0, sl 33 = sl 1, etc.

**Scale factors.** If all the digits in BR are zero and AC contains  $\pm 0$ , the order sf x leaves AC and BR undisturbed and stores the number 33 in the last 11 digit positions of register x.

**Division.** Let u and v be the numbers in AC and register x when the order dv x is used. If  $|u| < |v|$  the correct quotient is obtained and no overflow can arise. If  $|u| > |v|$  overflow occurs and gives an alarm. If  $u = v \neq 0$  the dv order leaves 16 ones in BR and round-off in a subsequent sl 15 would cause overflow and give an alarm. If  $u = v = 0$  a zero quotient is obtained.

APPENDIX II - Details of Coded Programs

## TEST SEQUENCE NUMBER I

(To be used with voltage-variation lines 77 and 78)

<u>Register #</u>	<u>Contents</u>	<u>Order #</u>
0	+0	
1	$2^{-7}$	
2	$2^{-14}$	
3	$33 \times 2^{-15}$	
(FF)4	(receives scale factor)	
5	1/2	
6	$2^{-15}$	
7	ca 5	1 24
8	sr 31	2 25
9	qc 0	3 etc.
10	sf 4	4
11	ca 5	5
12	sr 13	6
13	qc 2	7
14	ca 4	8
15	qc 3	9
16	ca 5	10
17	sf 4	11
18	sr 6	12
19	qc 1	13
20	nr 1	14
21	qc 2	15
22	ca 4	16
23	qc 0	17
24	ca 6	18
26	dv 5	19
26	qc 0	20
27	el 15	21
28	qc 2	22
29	ep 7	23

PC reset to 7; PC end-carry may, but need not, reset PC.

## TEST SEQUENCE NUMBER II

(To be used with voltage-variation lines 79 and 80)

<u>Register #</u>	<u>Contents</u>	<u>Order #</u>
0	1/2	
1	1/4	
2	-1/2	
3	-1/4	
(FF)4	(receives scale factor)	
5	$1/4 + 2^{-15}$	
6	+0	
7	-0	
8	ca 0	1 26
9	mr 0	2 27
10	qc 1	3 etc.
11	sf 4	4
12	cc 0	5
13	sr 1	6
14	qc 1	7
15	sl 1	8
16	qc 0	10
17	sa 0	11
18	qc 6	12
19	ca 1	13
20	qc 5	14
21	ca 2	15
22	sa 2	16
23	cc 7	17
24	ca 5	18
25	cc 1	19
26	dv 0	20
27	qc 6	21
28	sl 15	22
29	qc 0	23
30	mr 2	24
31	cc 3	25

PC reset to 8; PC end-carry should reset FC.

## TEST SEQUENCE NUMBER III

(For use with voltage-variation lines 113, 199, 221, 200, 97, 96 17,  
201, and 141)

<u>Register #</u>	<u>Contents</u>	<u>Order #</u>
0	+0	
1	$14 \times 2^{-15}$	
(P)2	ts 0 (=1/2) (changed by order #8)	
3	$33 \times 2^{-15}$	
(P)4	(receives scale factor)	
5	1/2	
6	$2^{-9}$	
7	$2^{-15}$	
8	ca 5	1 25
9	sr 16	2 26
10	qc 0	3 etc.
11	sf 4	4
12	ad 4	5
13	oc 3	6
14	er 1	7
15	td 2	8
16	ca 6	9
17	rt 6	10
18	ef 4	11
19	ad 4	12
20	qc 2	13
21	ca 7	14
22	el 14	15
23	qc 5	16
24	sr 15	17
25	qc 7	18
26	el 0	19
27	oc 7	20
28	sf 4	21
29	qc 5	22
30	ca 4	23
31	qc 1	24

PC reset to 8; PC end-carry must reset PC and Register #2.

## TEST SEQUENCE NUMBER IV

(For checking the program counter)

<u>Register #</u>	<u>Contents</u>	<u>Order # (in correct operation)</u>
0	ad 20	1 15
1	ad 21	2 16
2	su 27	3 etc.
3	ad 21	4
4	ad 28	5
5	su 20	6
6	ad 21	7
7	ad 21	8
8	qc 29	9
9	sp 15	10
10	----	
11	----	
12	ad 21	
13	----	
14	----	
15	ca 20	11
16	qc 20	12
17	sp 31	13
18	----	
19	+0	
20	+1/2	
21	+1/4	
22	----	
23	----	
24	ad 20	
25	----	
26	----	
27	+1/8	
28	+1/16	
29	+15/16	
30	----	
31	ca 19	14

PC reset to 0; PC end-carry must not reset PC

TEST SEQUENCE NUMBER V(To be used with voltage-variation lines 1<sup>h</sup>, 110, 190, 191, and 231.)

<u>Register #</u>	<u>Content</u>	<u>Order #</u> (in normal operation)
0	dv 1	
1	+0	
2	-2 <sup>-15</sup>	
3	+11x2 <sup>-15</sup>	
4	+31x2 <sup>-15</sup>	
5	-1/4	
6	+1/2	
7	-1/2	
(F) 8	(reserves scale factor)	(reset to +0)
(FF) 9	(receives address on <u>ta</u> )	(reset to +0)
(F) 10	(is added to on <u>ao</u> )	(reset immaterial)
11	ta 0	3 22
12	af 8	4 23
13	ca 8	5 etc.
14	cc 3	6
15	sl 14	7
16	cc 6	8
17	mr 7	9
18	cc 5	10
19	cp 21	11
20	dv 1	
21	sr 14	12
22	cc 2	13
23	ta 10	14
24	ca 9	15
25	cc 4	16
26	ao 10	17
27	cm 10	18
28	cc 1	19
29	ad 3	1 20
30	ep 11	2 21
31	dv 1	

FC reset to 29; FC end-carry may, but need not, reset FC.  
AC should be clear at start.

APPENDIX III - Glossary

- Channel:** Any of the routes which may be taken by information supplied to the input of the computer.
- Channel Segment:** The portion of a channel which is contained in a section.
- Component:** The most elementary parts of which the computer is constructed, e.g., resistors, condensers, vacuum tubes, etc.
- Section:** All stages which are supplied by a given voltage-variation line.
- Stage:** An elementary circuit containing at least one vacuum tube (or its equivalent) which performs a single function (e.g., a flip-flop, a gate tube, or an electronic switch, but not a register, a counter, nor an adder).
- Test Sequence:** A group of instructions to the computer which cause information to be routed along certain channel segments. As used in this thesis, test sequence refers to a group which may be performed with the 32 registers of test storage.
- Unit:** Any portion of the computer which consists of at least one stage.



BIBLIOGRAPHY

1. Adams, C. W., "Temporary Operations for Whirlwind I", Project Whirlwind, Servomechanisms Lab., M.I.T., Engineering Note E-250, 1949.
2. Adams, C. W., "Test Program Number I. Computer Complementing", Project Whirlwind, Servomechanisms Lab., M.I.T., Engineering Note E-295, 1949.
3. Adams, C. W., "Test Program Number II, Operation Matrix Check", Project Whirlwind, Servomechanisms Lab., M.I.T., Engineering Note E-296, 1949.
4. Adams, C. W., "Temporary Operation qs - Switch Check; and Test Program Number VI - Switch Check", Project Whirlwind, Servomechanisms Lab., M.I.T., Engineering Note E-313, 1949.
5. Blumenthal, E. I. and C. G. Hoberg, "A Trouble-Location Scheme for a Digital Electronic Computer", S.M. Thesis, M.I.T., 1948.
6. Cooper, G., "Checking Methods for Digital Computers", E.E. Seminar, M.I.T., 1949.
7. Cooper, G., "Display Program Number I: Family of Parabolas and Powers of X", Project Whirlwind, Servomechanisms Lab., M.I.T., Engineering Note E-300, 1949.
8. Cooper, G., "Proposed Method for automatic Test Checking of Arithmetic Control Flip-Flops", Project Whirlwind, Servomechanisms Lab., M.I.T., Memorandum E-877, 1949.
9. Eckert, J. P., J. W. Mauchly, E. H. Goldstine, and J. G. Prainerd, "Description of ENIAC and Comments on Electronic Digital Computing Machines", APT Report 171.2P, University of Pennsylvania, 1945.
10. Eckert, J. P., "Reliability and Checking", Theory and Techniques for Design of Electronic Digital Computers, University of Pennsylvania, 1947.
11. Everett, R. P., "Checking by Check Problems", Project Whirlwind, Servomechanisms Lab., M.I.T., Memorandum E-161, 1947.
12. Everett, R. P., "Whirlwind I Computer Block Diagrams", Project Whirlwind, Servomechanisms Lab., M.I.T., Report E-127, 1947. This report is to be superseded by Report E-180, (reference 19) when it is printed.

13. Forrester, J. W., "Digital Computers as Information-Processing Systems", Project Whirlwind, Servomechanisms Lab., M.I.T., Report R-166, 1949.
14. Goldstine, H. H. and J. von Neumann, "Planning and Coding of Problems for an Electronic Computing Instrument", Institute for Advanced Study, 1947.
15. Hoberg, C. G. and J. M. Ulman, Jr., "Glossary of Computer Terms", Project Whirlwind, Servomechanisms Lab., M.I.T., Report R-138, 1948.
16. Israel, D. R., "Introduction to Coding", Air Traffic Control Project, Servomechanisms Lab., M.I.T., Engineering Note E-2000, 1949.
17. Salzer, J. M., "Study of Input-Output Checking", Project Whirlwind, Servomechanisms Lab., M.I.T., Engineering Note E-225, 1949.
18. Salzer, J. M., C. W. Adams, and R. F. Mayer, "Description of Whirlwind I Codes", Project Whirlwind, Servomechanisms Lab., M.I.T., Engineering Note E-235, 1949.
19. Salzer, J. M., A still untitled report to be published shortly, Project Whirlwind, Servomechanisms Lab., M.I.T., Report R-180.
20. Stibitz, C. F., "The Organization of Large-Scale Calculating Machinery", Proceedings of a Symposium on Large-Scale Digital Calculating Machinery, Harvard University, 1948.
21. Sumner, G. C., "Trouble Location in a Large-Scale Electronic Digital Computer", S.M.Thesis, M.I.T., 1948.
22. Welchman, W. C., "Notations for Coding", Project Whirlwind, Servomechanisms Lab., M.I.T., Conference Note C-93, 1949.
23. Welchman, W. C., "Example of Coding Procedure", Project Whirlwind, Servomechanisms Lab., M.I.T., Conference Note C-94, 1949.

LIST OF ILLUSTRATIONS

<u>Fig. No.</u>	<u>Drawing No.</u>	<u>Title</u>
1	A-35132	Comparison between Digital and Analogue Methods of Computation
2	A-34353-1	Digital Computer--Main Units
3	A-35129	Flip-Flop
4	A-35130	Gate Circuit
5	A-35131	The Electronic Switch
6	B-37001-5	Block Diagram Symbols, WWI
7	C-37056	A-Register, WWI
8	C-37097-6	B-Register, WWI
9	D-37173-2	Accumulator, WWI
10	B-37062-6	Program Counter, WWI
11	C-37068-6	Time-pulse Distributor, WWI
12	D-37071-6	System Block Diagram, WWI
13	B-37098-6	Central Control, WWI
14	D-37072-10	Arithmetic Element, WWI
15	D-37178-2	Input-Output Element, WWI
16	B-37195-1	Program Timing, WWI
17	A-35133	Multi-Channel Trouble Location
18	D-35146	Control Matrix Output Connections, WWI
19	B-37174-2	Special Add Memory and Overflow, WWI
20	D-37071-6	Step Counter, WWI (Block Diagram)
21	D-31828-2	Step Counter, WWI (Block Schematic)
22	A-32723-1	Step-Counter Output, WWI (Block Schematic)
23	A-35134	Margins of Correct Operation for Sample Problems.

## DIGITAL APPROACH

ADDITION:

$$\begin{array}{r} 5 \\ + 3 \\ \hline 8 \end{array}$$

( DECIMAL )  
( NOTATION )

$$\begin{array}{r} 101 \\ + 11 \\ \hline 1000 \end{array}$$

( BINARY )  
( NOTATION )

MULTIPLICATION:

$$\begin{array}{r} 5 \\ \times 3 \\ \hline 15 \end{array}$$

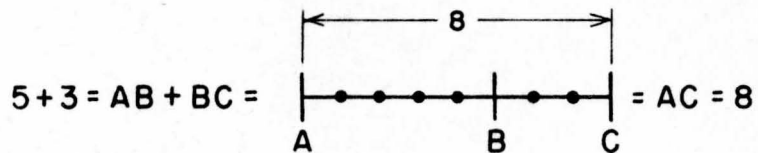
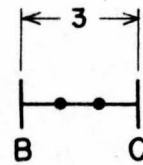
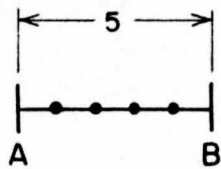
( DECIMAL )  
( NOTATION )

$$\begin{array}{r} 101 \\ \times 11 \\ \hline 101 \\ 101 \\ \hline 1111 \end{array}$$

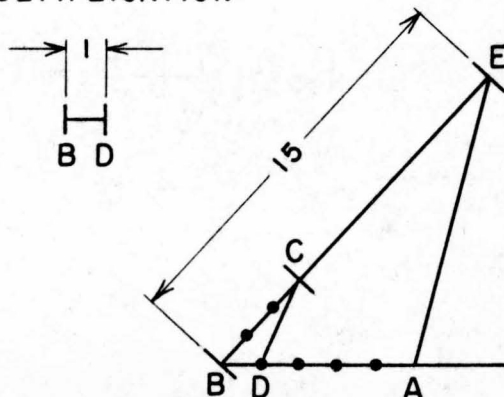
( BINARY )  
( NOTATION )

## ANALOGUE APPROACH

ADDITION:



MULTIPLICATION:



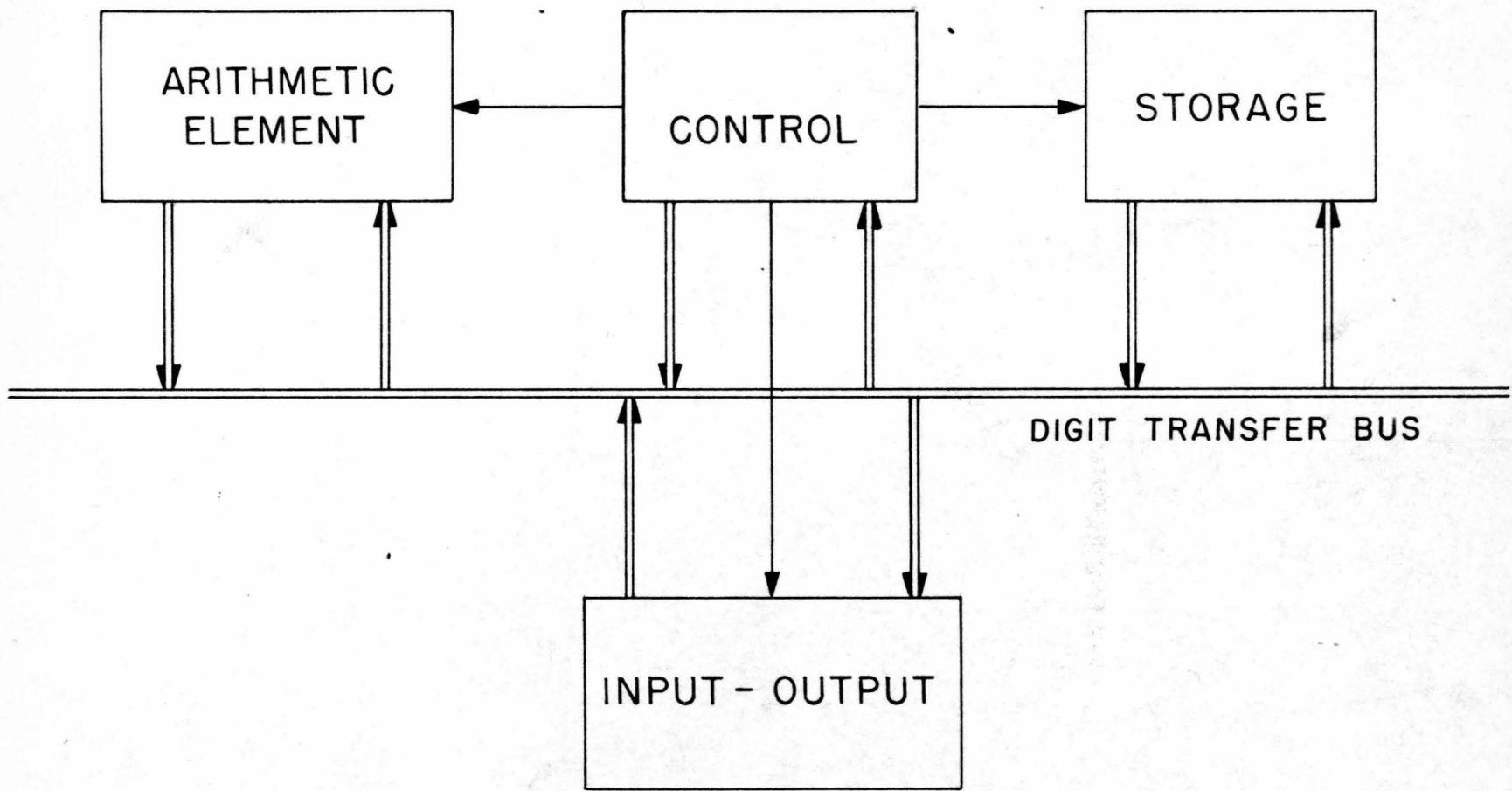
IF  $AE \parallel DC$ , THEN  
 $\Delta ABE \sim \Delta DBC$   
 HENCE,

$$\frac{BE}{AB} = \frac{BC}{DB} = \frac{BC}{1}$$

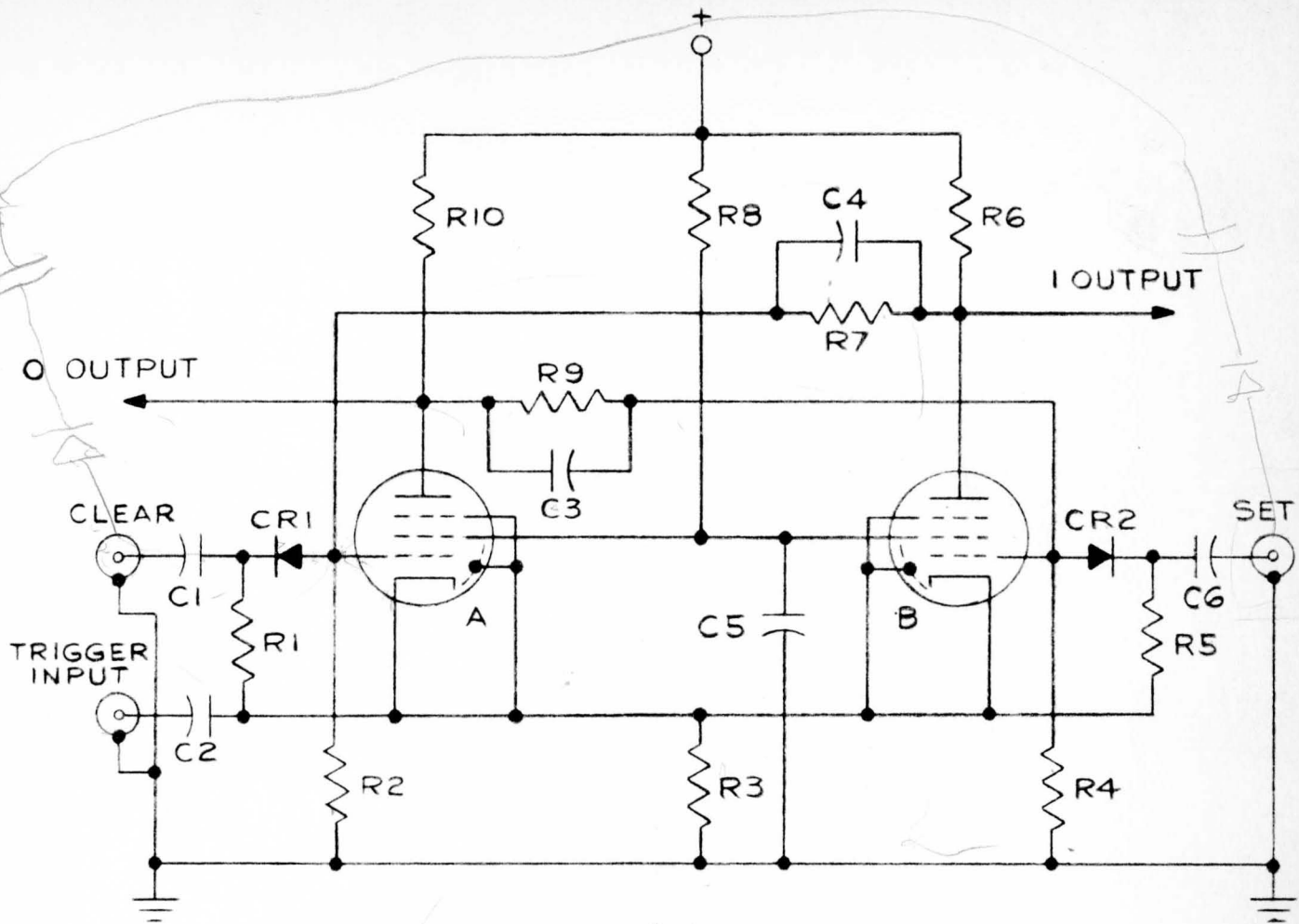
$$BE = AB \times BC = 5 \times 3 = 15$$

COMPARISON BETWEEN DIGITAL AND ANALOGUE  
 METHODS OF COMPUTATION

FIG. 1

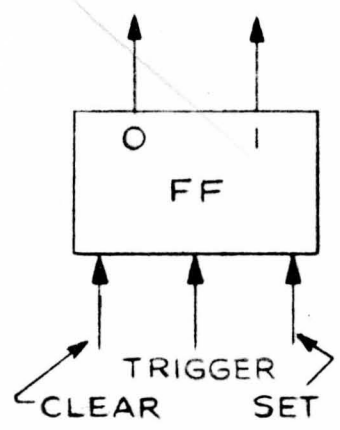


DIGITAL COMPUTER - MAIN UNITS



(a)

FLIP-FLOP CIRCUIT DIAGRAM



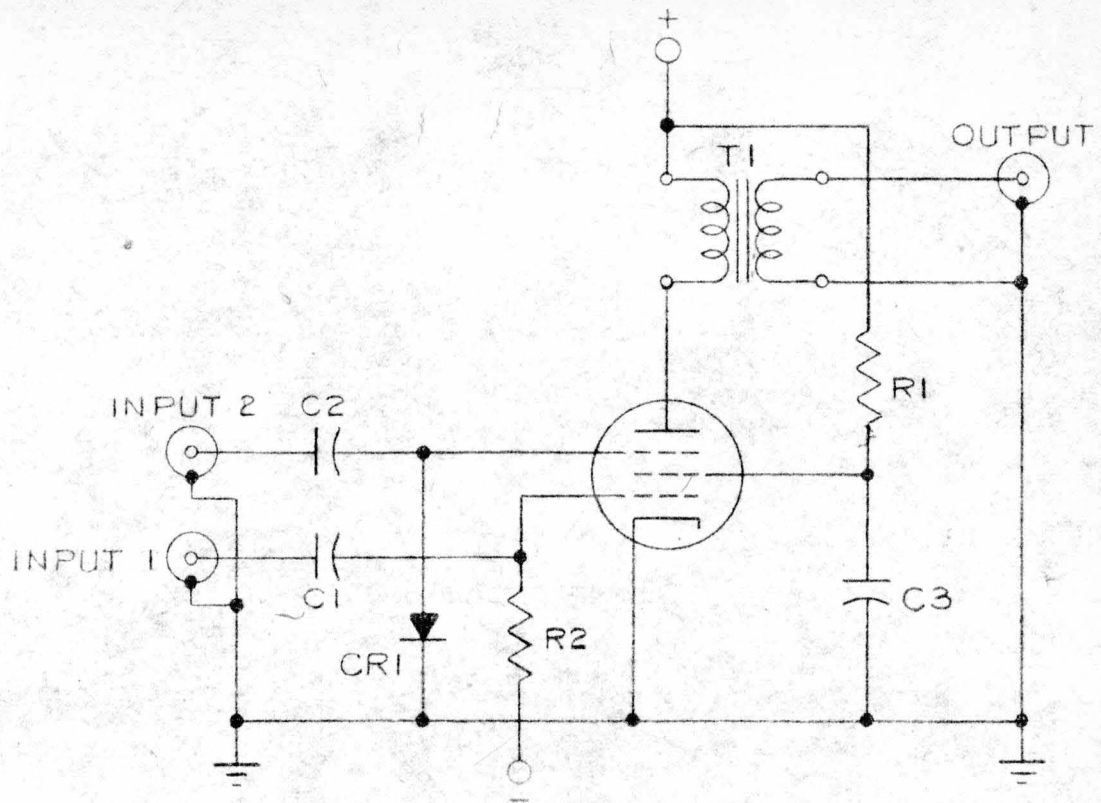
(b)

BLOCK REPRESENTATION OF FLIP-FLOP  
FIG. 13

FLIP-FLOP CIRCUIT DIAGRAM

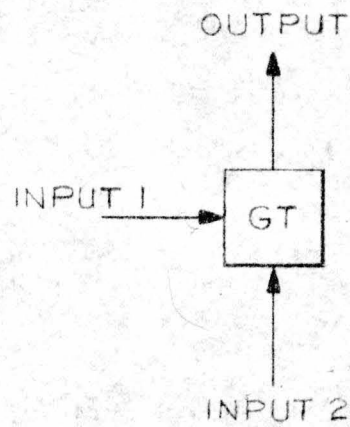
FIG. 3

A-351-9



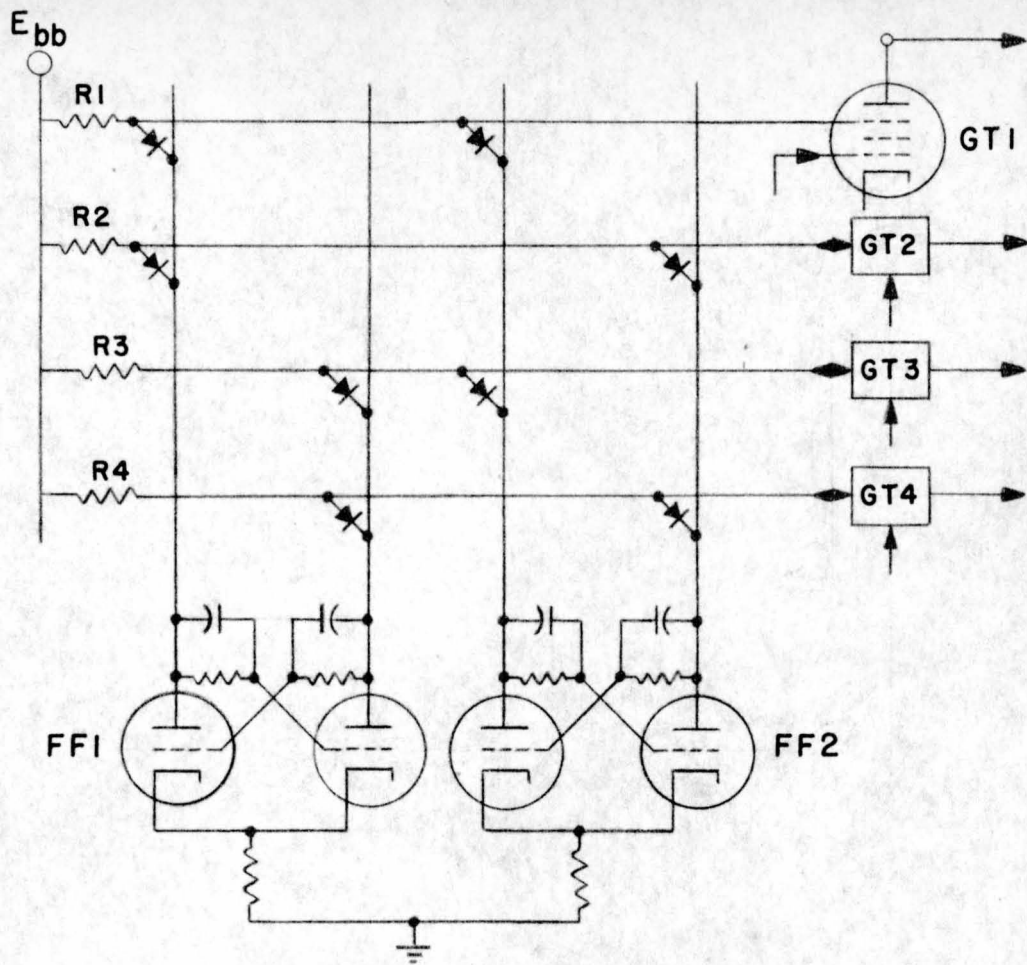
(a)

GATE CIRCUIT  
(INPUT 2 FROM FLIP-FLOP)



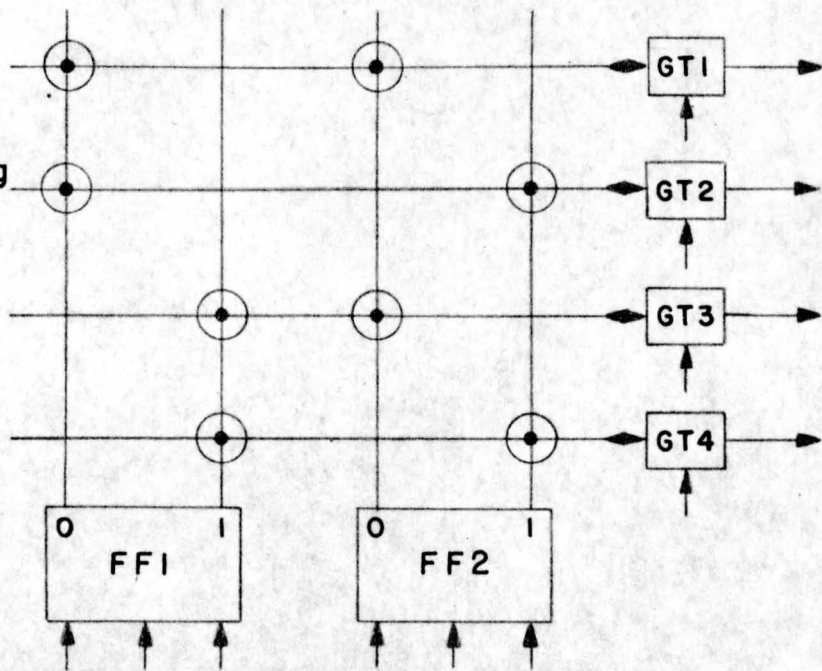
(b)

BLOCK REPRESENTATION OF GATE CIRCUIT



(a) SCHEMATIC REPRESENTATION

NOTE:  
Any "ON" crystal  
in matrix turns  
off corresponding  
gate tube

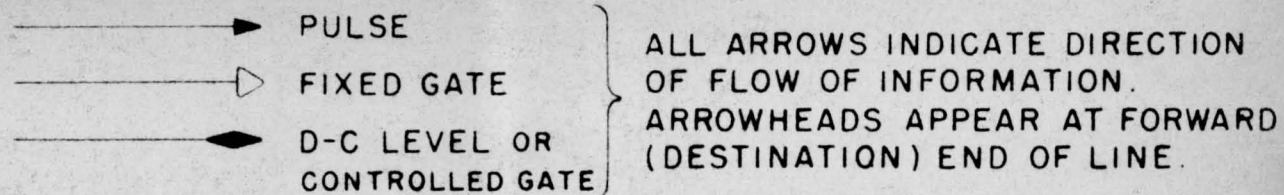


(b) BLOCK DIAGRAM REPRESENTATION

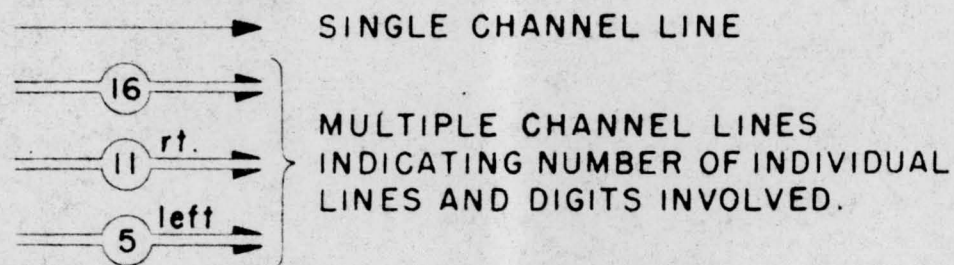
THE ELECTRONIC SWITCH



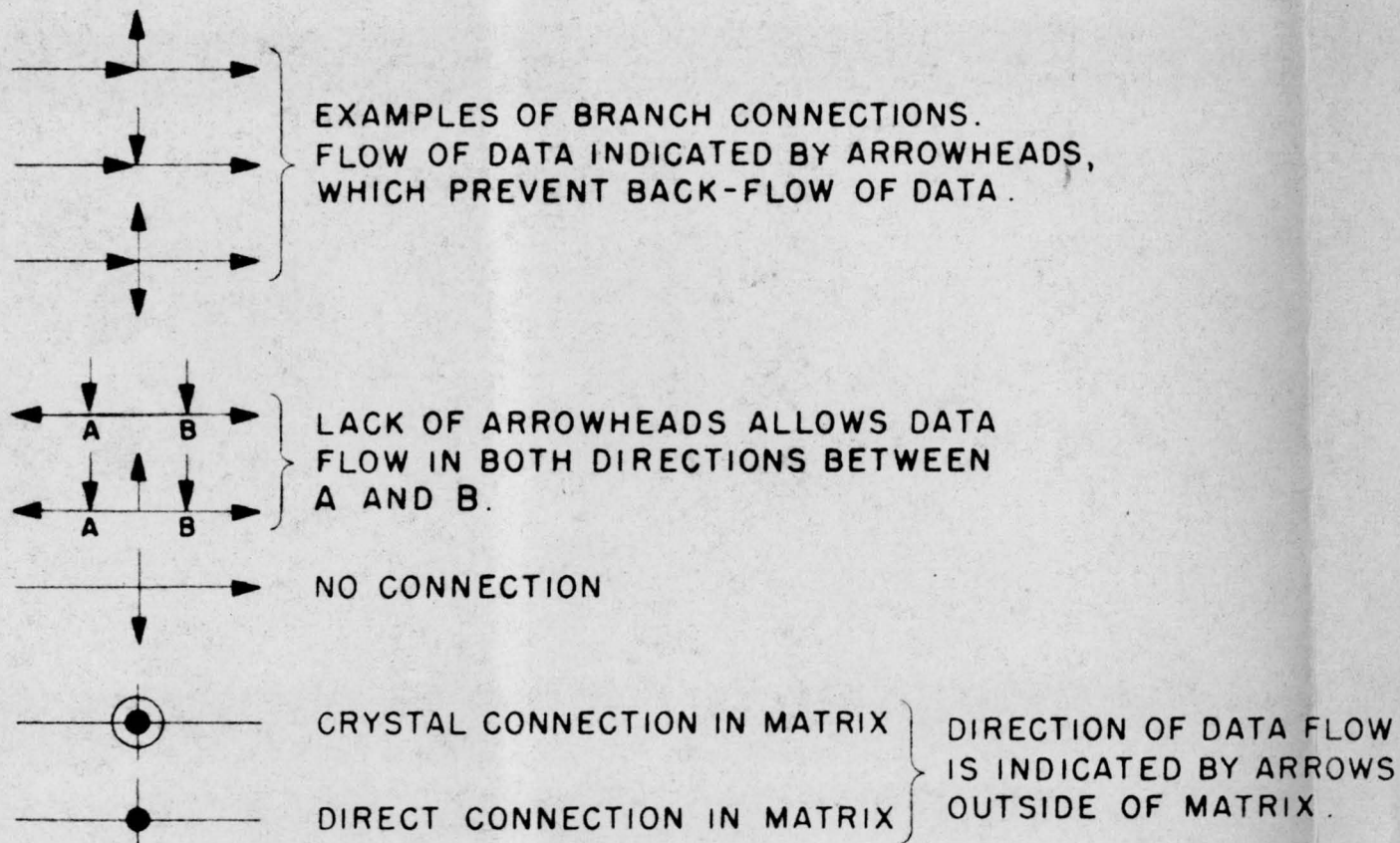
**TYPES OF DATA**



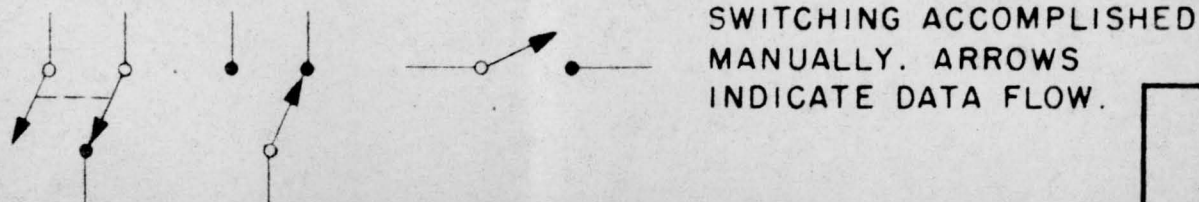
**DATA CHANNELS**



**CONNECTION OF LINES**



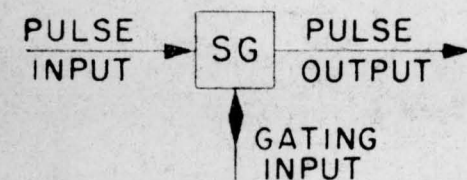
**SWITCHES**



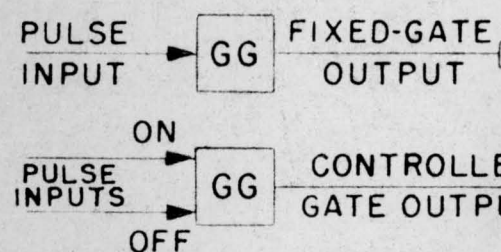
**COMPONENTS**



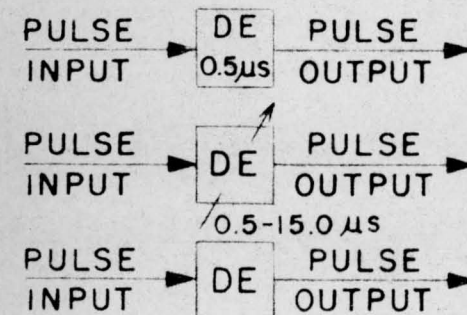
**GATE TUBE**  
PASSES ANY NUMBER OF PULSES DURING GATING



**SYNCHRONIZING GATE**  
PASSES ONLY ONE PULSE FOR EACH GATING.



**GATE GENERATORS**  
OUTPUT IS GATE OF FIXED LENGTH (USUALLY ABOUT 0.4 μ SEC)  
OUTPUT IS GATE OF VARIABLE LENGTH AS DETERMINED BY TIME BETWEEN INPUT PULSES BUT LIMITED TO A FIXED MAXIMUM



**DELAY ELEMENT**  
DELAYS DATA BY THE FIXED INDICATED AMOUNT  
DELAYS DATA BY AMOUNT MANUALLY VARIABLE WITHIN INDICATED RANGE  
SLIGHT DELAY WHICH IS REQUIRED LOGICALLY

**FIG. 6**

SERVOMECHANISMS LABORATORY OF THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY  
DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345

**BLOCK DIAGRAM SYMBOLS, WWI**

GRADED BY: DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW:  
GRADE I FOR REFERENCE ONLY  
GRADE II PRELIMINARY DESIGN  
GRADE III FINAL DESIGN

SCALE: DR. F. B. [Signature] July 21 '49  
ENG. [Signature] 7/22/49 CK APP.

**B-37001-5**

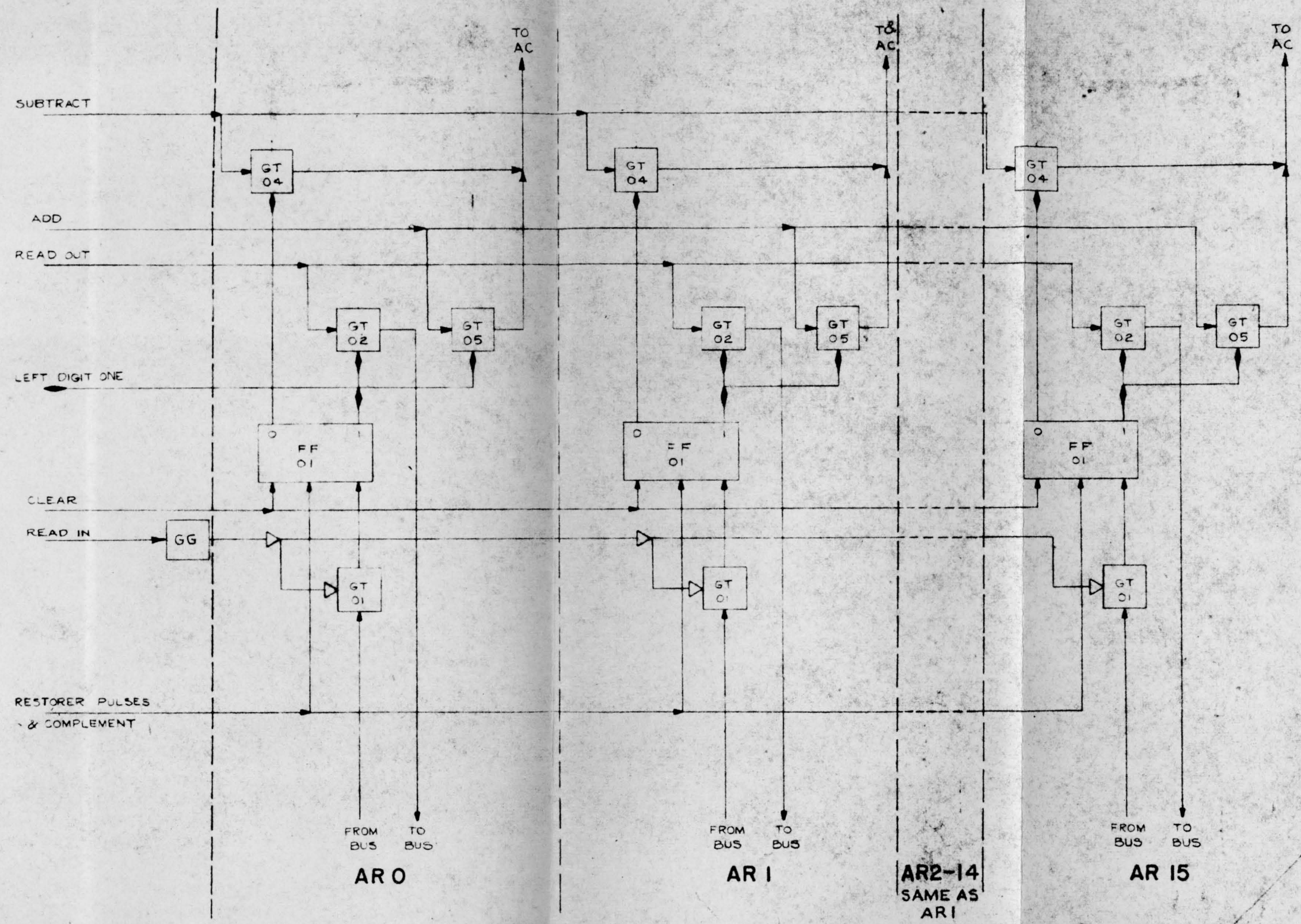


FIG. 7

GRADED BY: DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW:  
 GRADE I FOR REFERENCE ONLY  
 GRADE II PRELIMINARY DESIGN  
 GRADE III FINAL DESIGN

*DATE 7/8/48*

SERVO-MECHANICALS LABORATORY OF THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345	
BLOCK DIAGRAM FOR A-REGISTER, WWI	
SCALE: <i>1/2" = 1"</i>	DR: <i>McN</i>
DATE: <i>1/29/48</i>	APP: <i>23-48</i>
C-37056-4 8 REDUCTION	

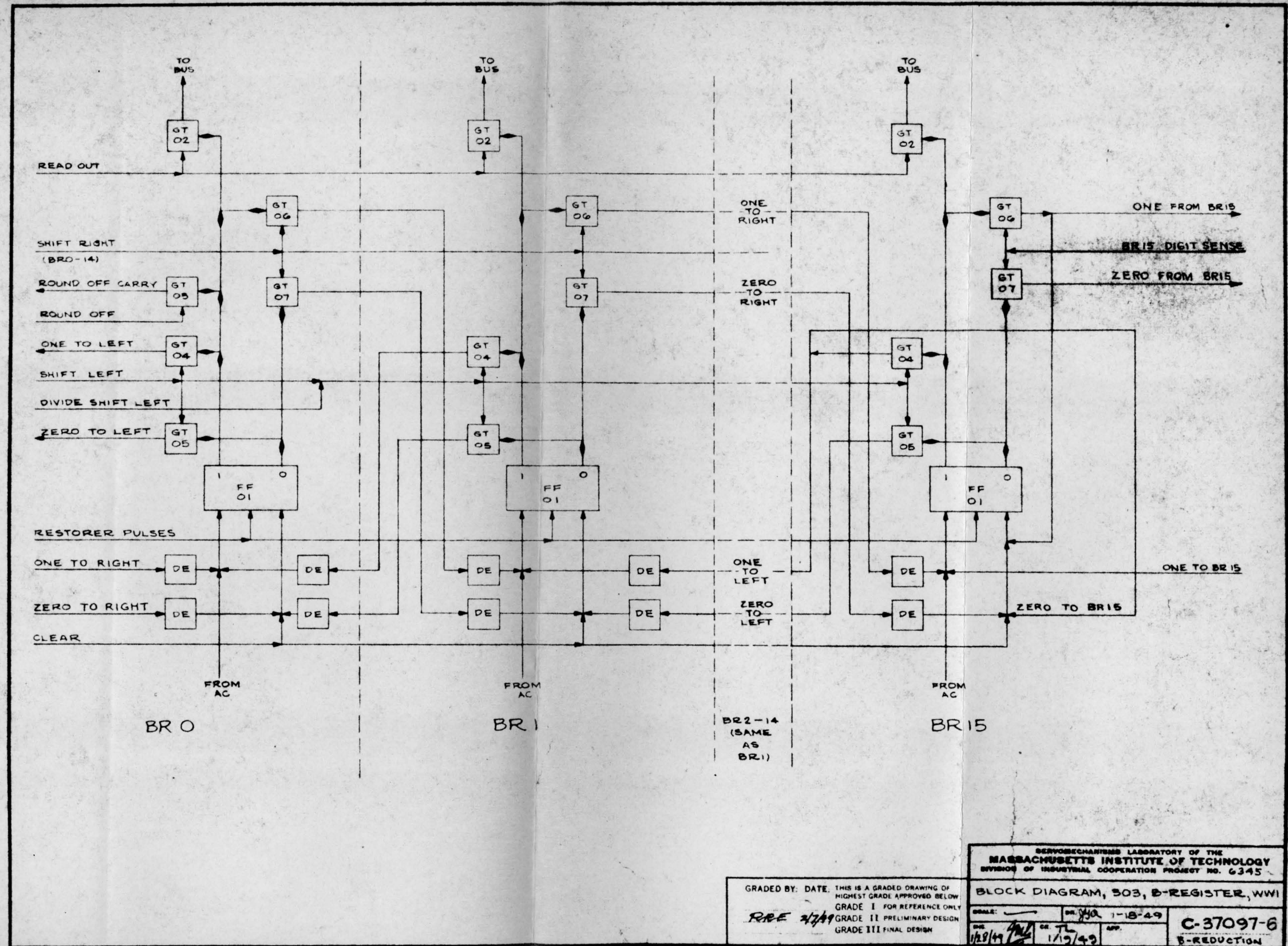
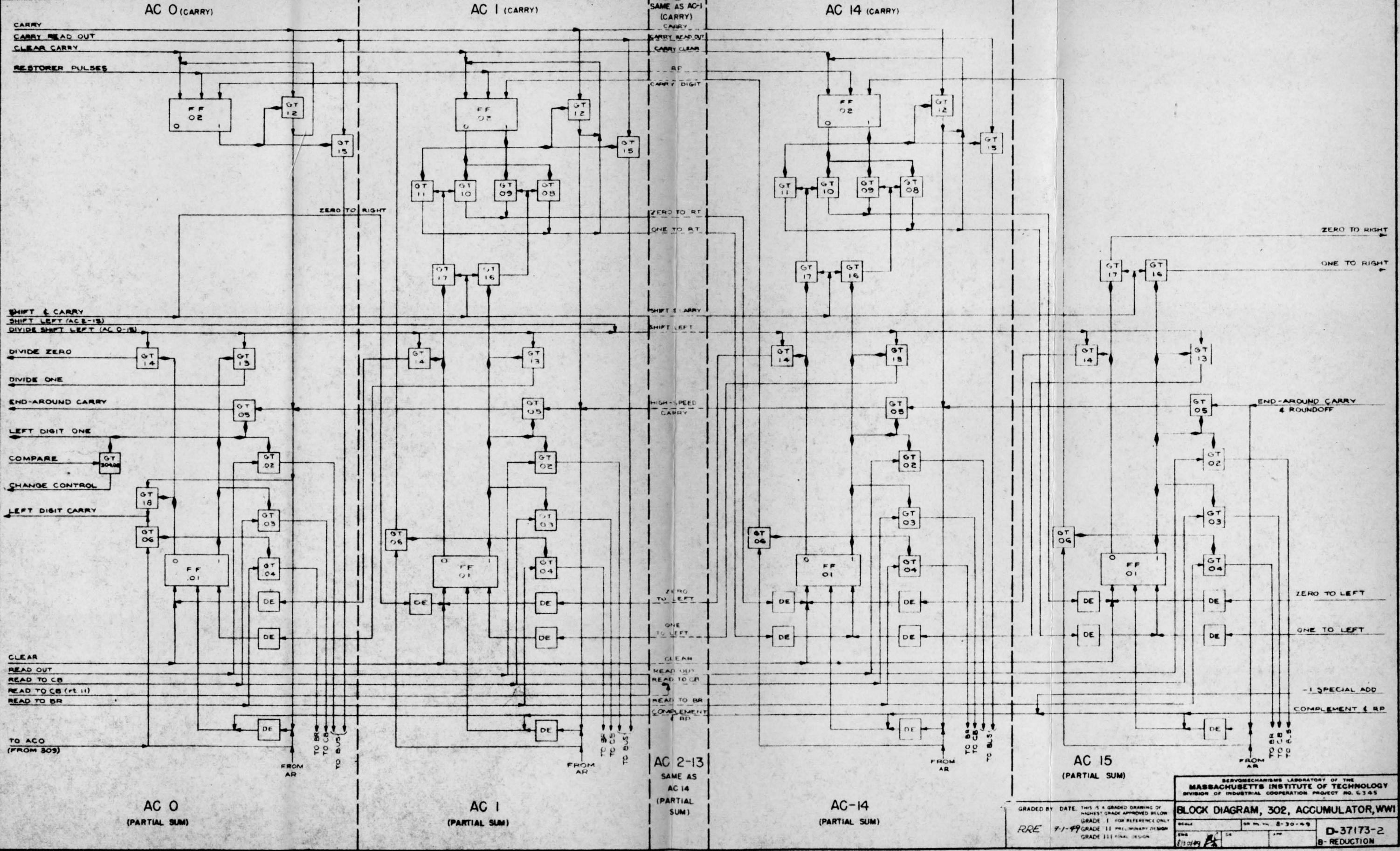


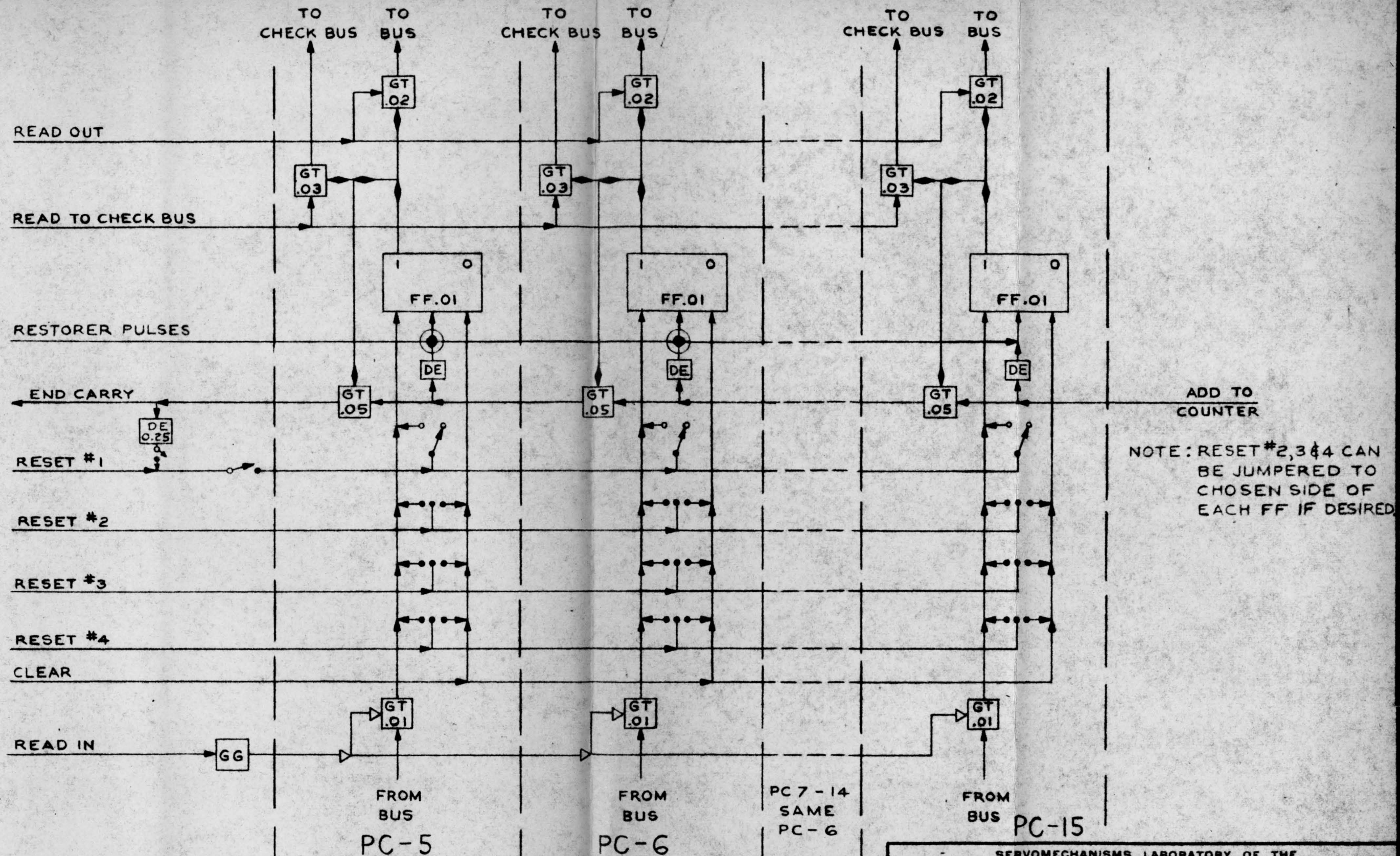
FIG. 8

D-37173-2



SERVO-MECHANISMS LABORATORY OF THE  
 MASSACHUSETTS INSTITUTE OF TECHNOLOGY  
 DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345  
**BLOCK DIAGRAM, 302, ACCUMULATOR, WWI**  
 GRADED BY DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW  
 GRADE I FOR REFERENCE ONLY  
 RRE 7-1-49 GRADE II PRELIMINARY DESIGN  
 GRADE III FINAL DESIGN  
 SCALE 1" = 1" DR. NO. 8-30-49  
 D-37173-2  
 B-REDUCTION

FIG. 9



SERVOMECHANISMS LABORATORY OF THE  
**MASSACHUSETTS INSTITUTE OF TECHNOLOGY**  
 DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345

**BLOCK DIAGRAM, 102,  
 PROGRAM COUNTER, WWI.**

SCALE: DR. A.M.G. 6-22-49

ENG. 6/27/49 APP. B-37062-6

GRADED BY: DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW:

GRADE I FOR REFERENCE ONLY

GRADE II PRELIMINARY DESIGN

GRADE III FINAL DESIGN

*TOPA 6/27/49*

NOTE: ANY "ON" CRYSTAL IN UPPER MATRIX  
 TURNS OFF CORRESPONDING GATE TUBE

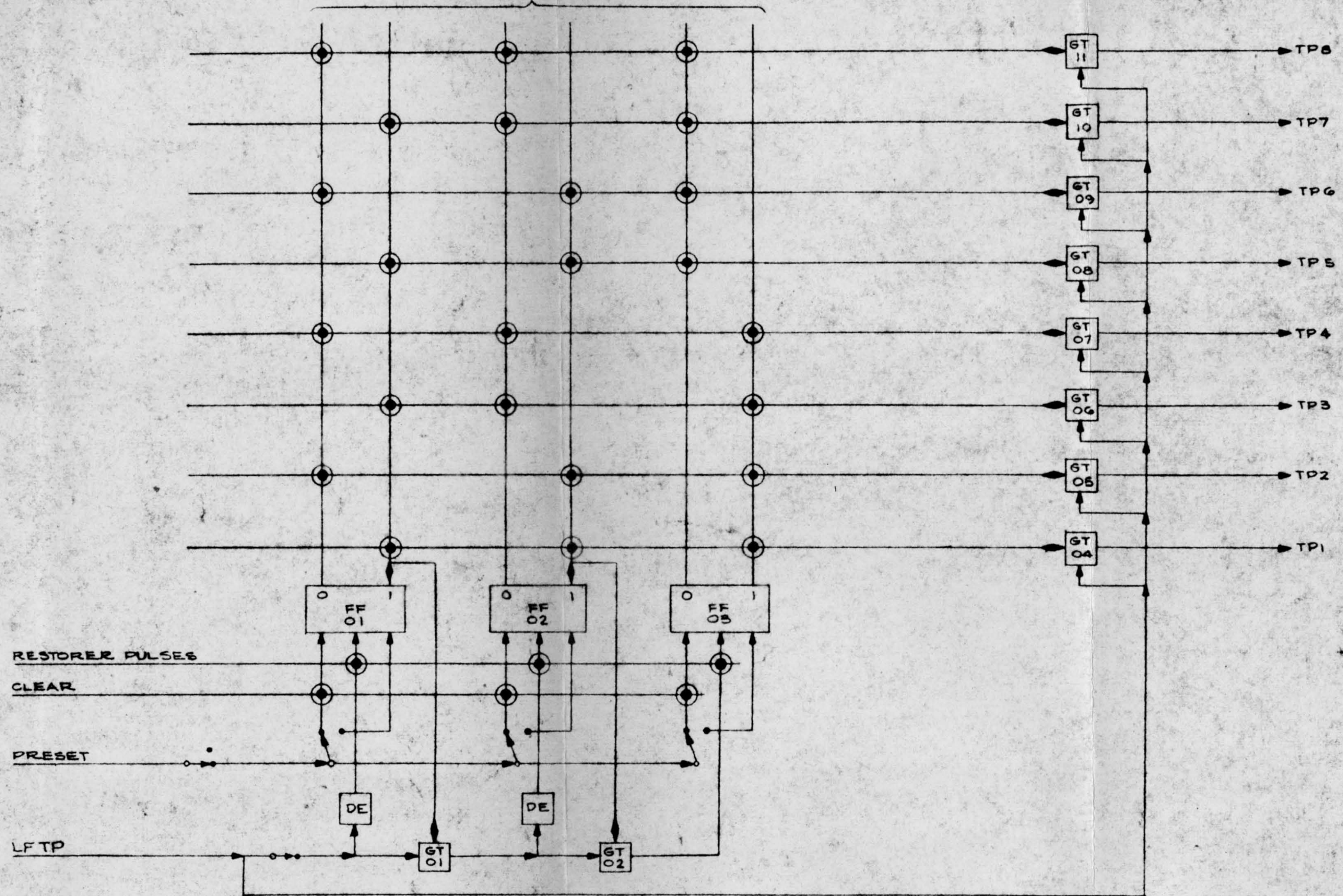
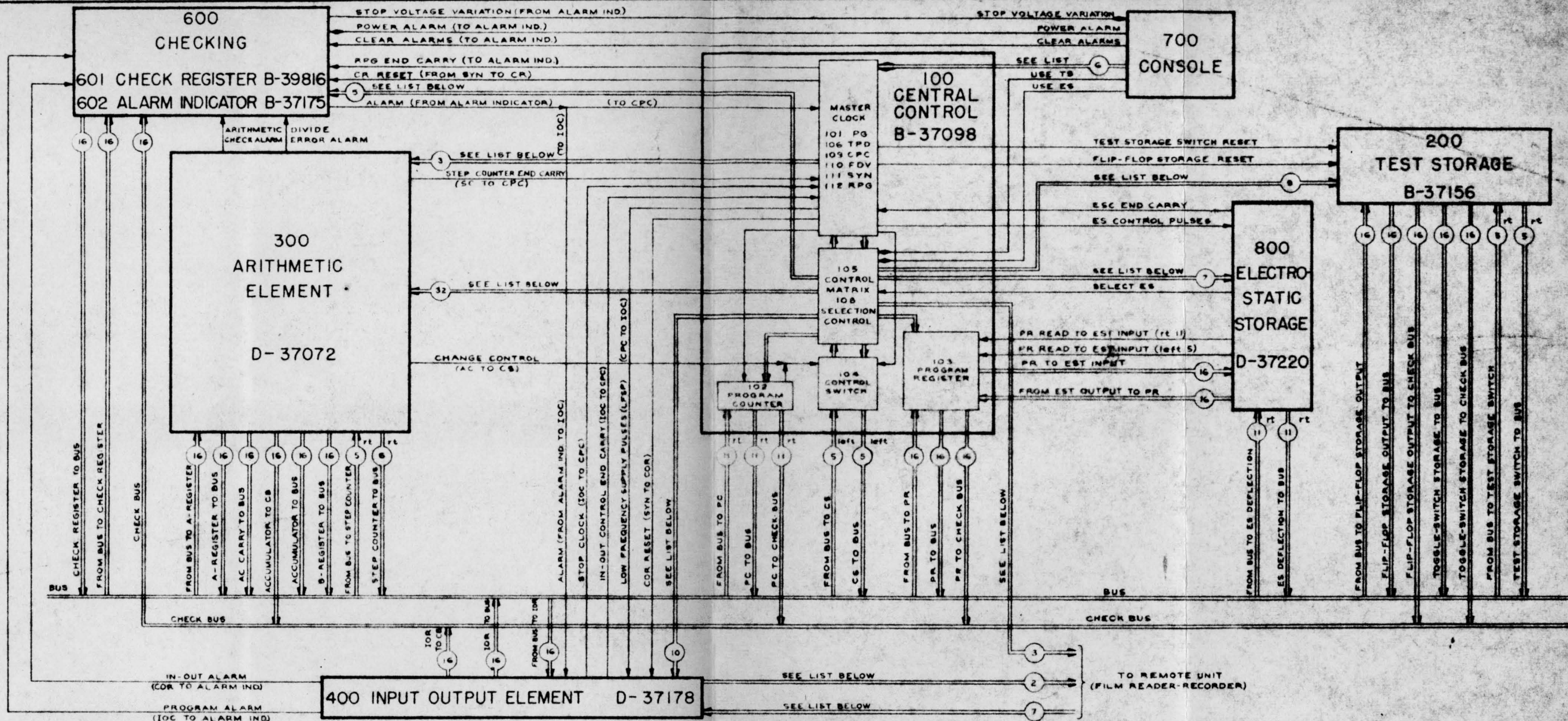


FIG. 11

GRADED BY: DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW  
 GRADE I FOR REFERENCE ONLY  
 GRADE II PRELIMINARY DESIGN  
 GRADE III FINAL DESIGN  
 DATE 4/11/49

SERVOMECHANISMS LABORATORY OF THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345			
BLOCK DIAGRAM, 106 TIME PULSE DISTRIBUTOR WWI			
SCALE:	DR. YJA 2-23-49	C-37068-6	
ENG. [Signature]	CR.	APP.	



100 TO 200 (9 LINES)

- 108 TO 201 TEST STORAGE SWITCH CLEAR
- TSS READ IN
- TSS READ OUT
- 108 TO 202/3 TEST STORAGE READ OUT
- TS READ TO CHECK BUS
- TEST STORAGE CLEAR
- TS CLEAR (RT II)
- TS READ IN
- TS READ IN (RT. II)

100 TO 300 (32 + 3 LINES)

- 105 TO 301 A-REGISTER CLEAR
- AR READ IN
- AR READ OUT
- ADD
- SUBTRACT
- 105 TO 302 ACCUMULATOR CLEAR
- AC CARRY CLEAR
- AC READ OUT
- AC READ TO CB
- AC READ TO CB (RT. II)
- AC READ TO BR
- END AROUND CARRY
- COMPARE
- 105 TO 302/9 CARRY

100 TO 300 (CONTINUED)

- 105 TO 303 B-REGISTER CLEAR
- ROUND OFF
- 105 TO 304 MAGNITUDE
- AR SIGN CHECK
- PRODUCT SIGN
- 105 TO 304/8 AC SIGN CHECK
- 105 TO 305 STEP COUNTER PRESET
- SC READ IN
- SC READ OUT
- ADD TO SC
- 105 TO 305/6 MULTIPLY
- 105 TO 307 SHIFT LEFT
- SHIFT RIGHT
- 105 TO 305/8 DIVIDE
- 105 TO 309 ARITHMETIC CHECK
- SPECIAL ADD
- SPECIAL CARRY
- 105 TO 305/10 POINT OFF
- 109 TO 308 LOW-FREQUENCY CLOCK PULSES (LFCP)
- 109 TO 308/7/10 HIGH-FREQUENCY CLOCK PULSES (HFCP)
- 111 TO 305 STEP COUNTER TEST RESET

100 TO 400 (10 LINES)

- 105 TO 403 IN OUT REGISTER CLEAR
- IOR READ IN
- IOR READ OUT
- IOR READ TO CB
- ADD IOR TO COR
- COMPARISON REGISTER SET
- 105 TO 404 IN-OUT READ
- 105 TO 403/4/10 IN-OUT RECORD
- rd SENSE
- rc SENSE

100 TO 600 (5 LINES)

- 105 TO 601 CHECK REGISTER CLEAR
- CR READ IN
- CR READ IN (RT. II)
- TRANSFER CHECK
- CR CHECK

100 TO 800 (7 LINES)

- 108 TO 811/12 ES CONTROL READ
- ESC WRITE
- 108 TO 813 SELECT I6
- SELECT RIGHT II
- 108 TO 820 ES DEFLECTION CLEAR
- ESD READ IN
- ESD READ OUT

100 TO REMOTE UNIT

- 105 TO RU RUN FORWARD
- RUN BACKWARD
- REMOTE UNIT STOP

400 TO REMOTE UNIT

- 403 TO RU ONE OUT (TO RECORDER)
- ZERO OUT (TO RECORDER)

REMOTE UNIT TO 400

- RU TO 403 ZERO IN (FROM RECORDER)
- ONE IN (FROM RECORDER)
- RU TO 404 ZERO IN (FROM READER)
- RU TO 403/4 RECORD SHIFT (FROM RECORDER)
- RU TO 403/4 READ SHIFT (FROM READER)
- RU TO 410 INITIATION
- RU TO 403/4 COMPLETION

700 TO 100

- 700 TO 111 RESTART
- CLEAR
- SINGLE PULSES
- COMPUTER COMPLEMENT
- RESET
- CHANGE TO PUSHBUTTON

THE FOLLOWING LINES ARE NOT INDICATED IN THIS DRAWING :

1. CLEAR FROM SYNCHRONIZER (111) TO ALL FLIP-FLOPS OF WWI EXCEPT THOSE IN THE PULSE GENERATOR (101) AND FREQUENCY DIVIDER (110). CLEARING IS DONE IN STARTING AND TESTING WWI.
2. RESTORE PULSES FROM RESTORER PULSE GENERATOR (112) TO ALL FLIP-FLOPS OF WWI EXCEPT THOSE IN THE PULSE GENERATOR (101) AND FREQUENCY DIVIDER (110). THE FLIP-FLOPS IN THE IN-OUT ELEMENT (400) ARE NOT RESTORED DURING ITS OPERATION. TWO D-C COUPLED FLIP-FLOPS IN IN-OUT CONTROL (410) ARE NOT RESTORED AT ANY TIME. - NO RESTORER PULSES GO TO THE "START DELAY" FLIP-FLOP FF 08 IN THE CLOCK PULSE CONTROL (109).

GRADED BY: DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW

GRADE I FOR REFERENCE ONLY

REBE 8-15-49 GRADE II PRELIMINARY DESIGN

GRADE III FINAL DESIGN

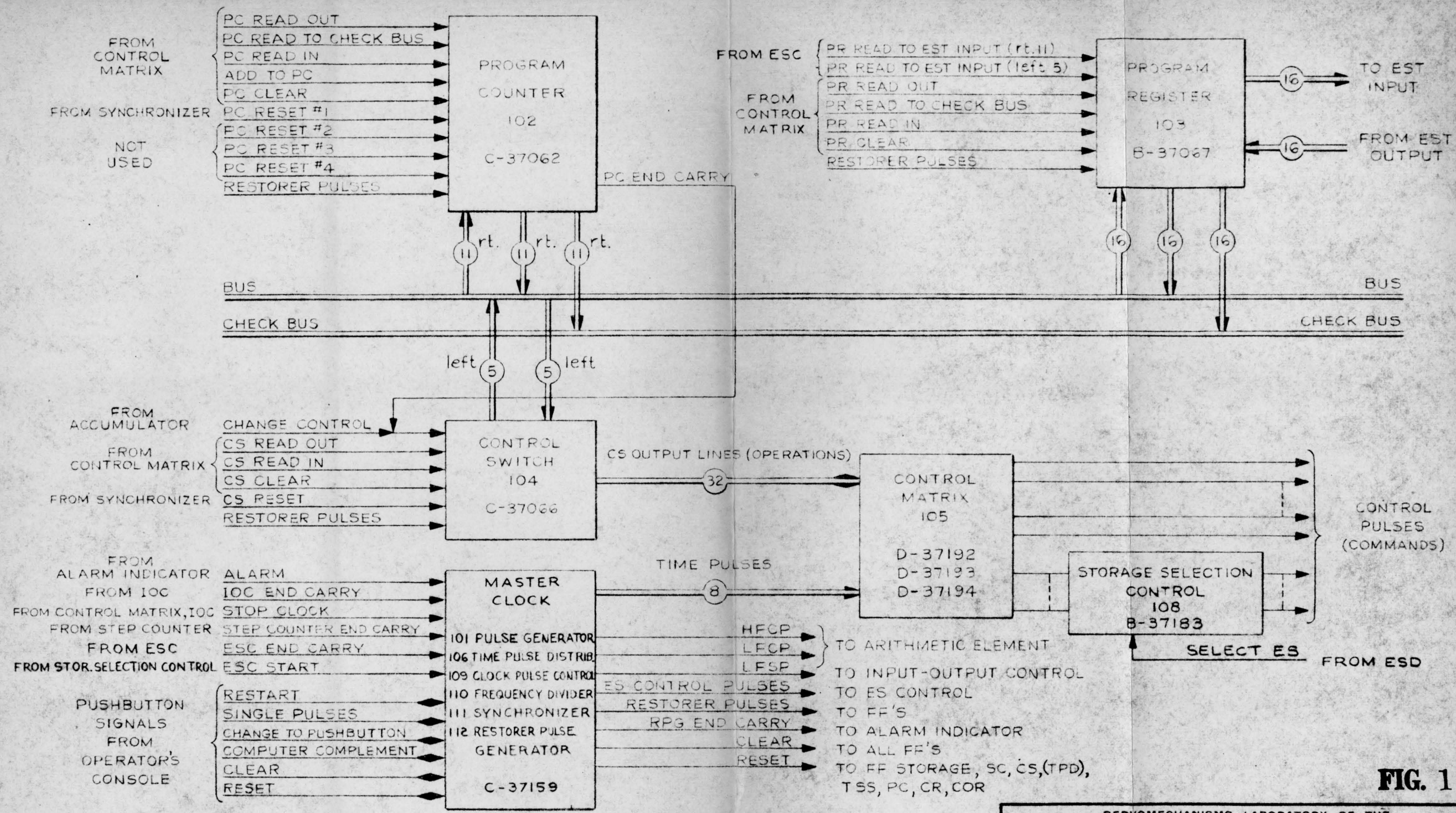
RESEARCH LABORATORY OF THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 63-45

SYSTEM BLOCK DIAGRAM, WWI

DATE: 8/11/49

D-37071-6 8-REDUCTION

FIG. 12



**FIG. 13**

**SERVOMECHANISMS LABORATORY OF THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY**  
DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345

BLOCK DIAGRAM, 100, CENTRAL CONTROL, WWI

SCALE:	DR. F. Brunswick June 22 '49	
ENG. 6/28/49 <i>[Signature]</i>	CK.	APP.

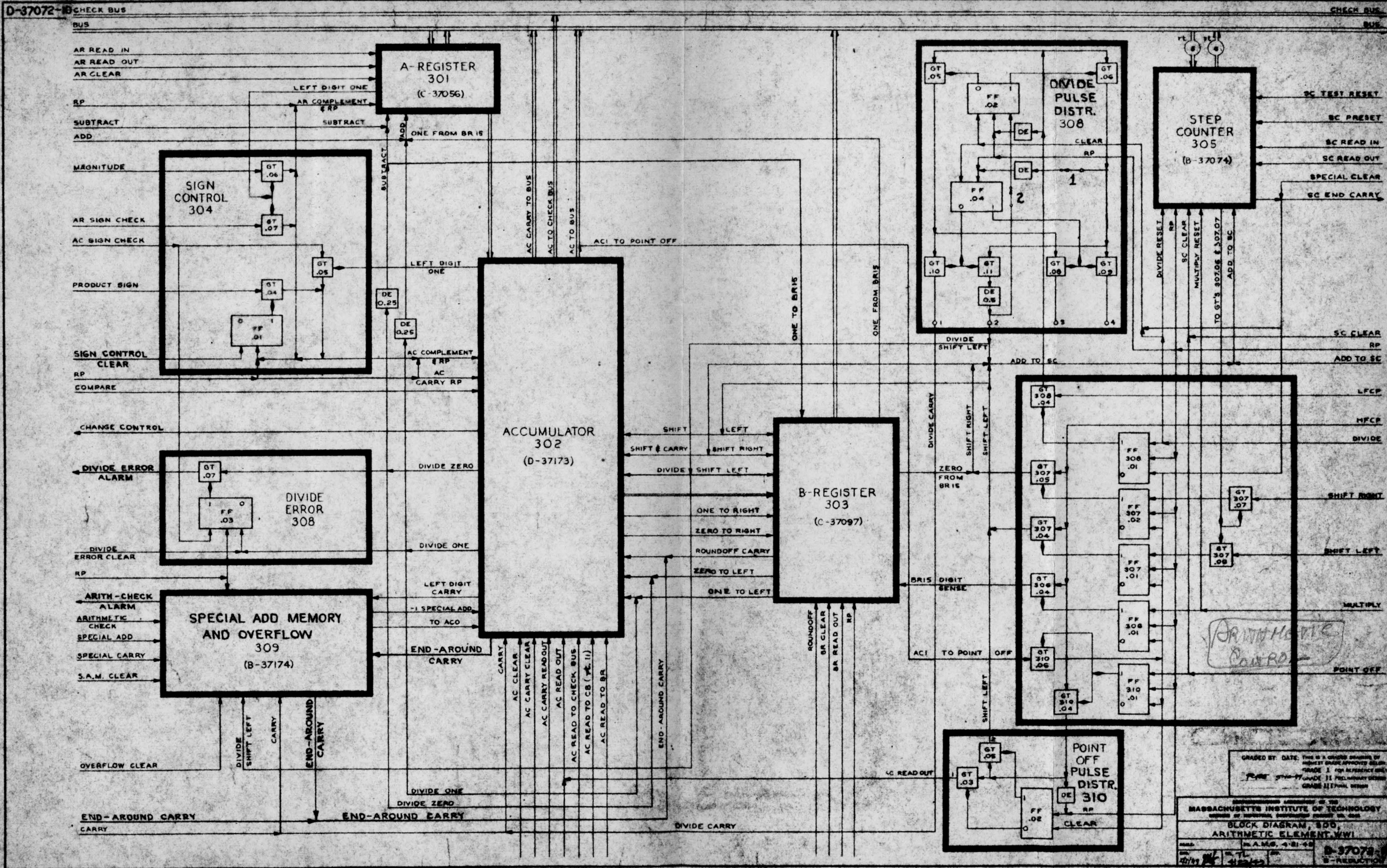
**B-37098-6**

GRADED BY: DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW:

*RRE 6/29/49*

GRADE I FOR REFERENCE ONLY  
GRADE II PRELIMINARY DESIGN  
GRADE III FINAL DESIGN

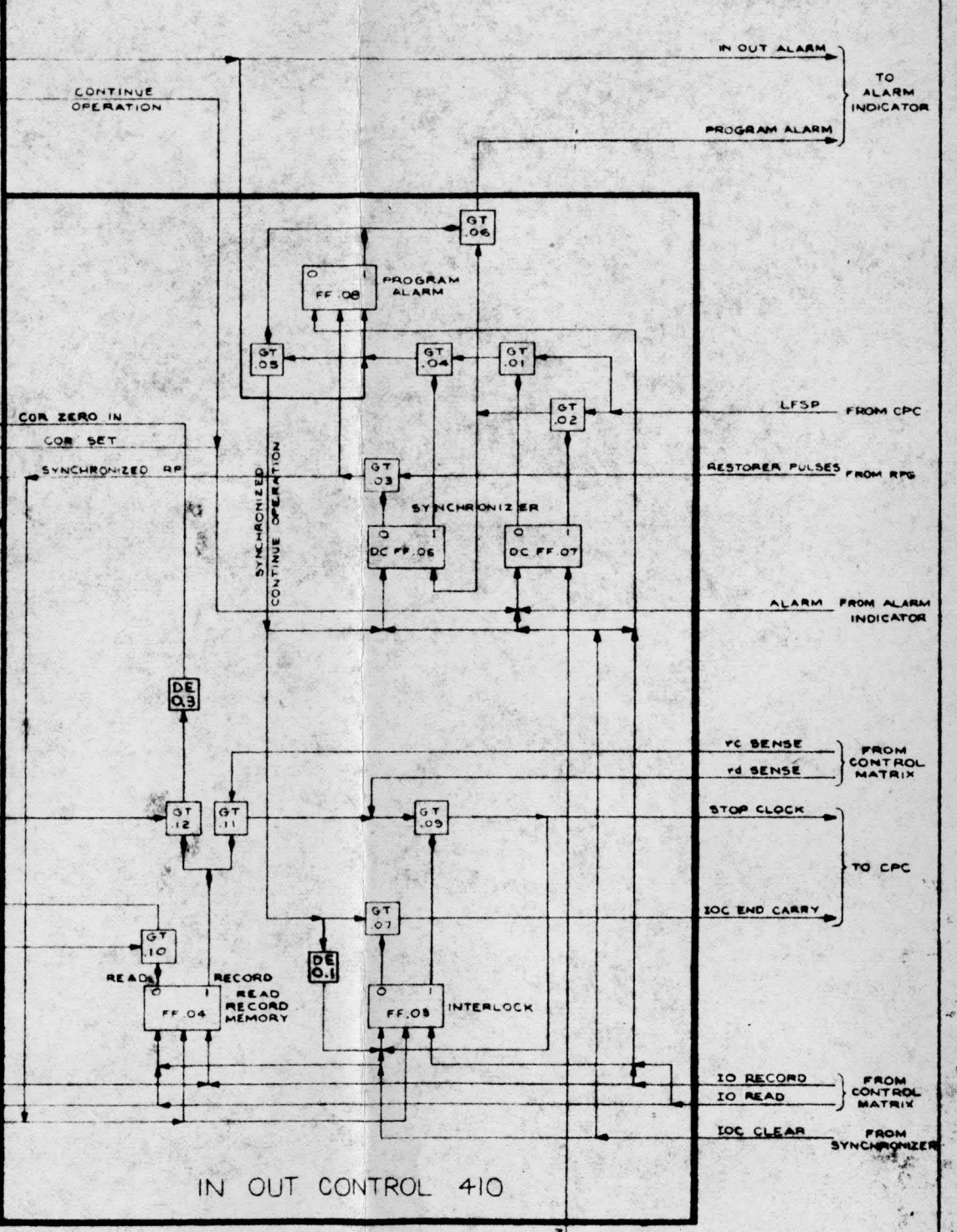
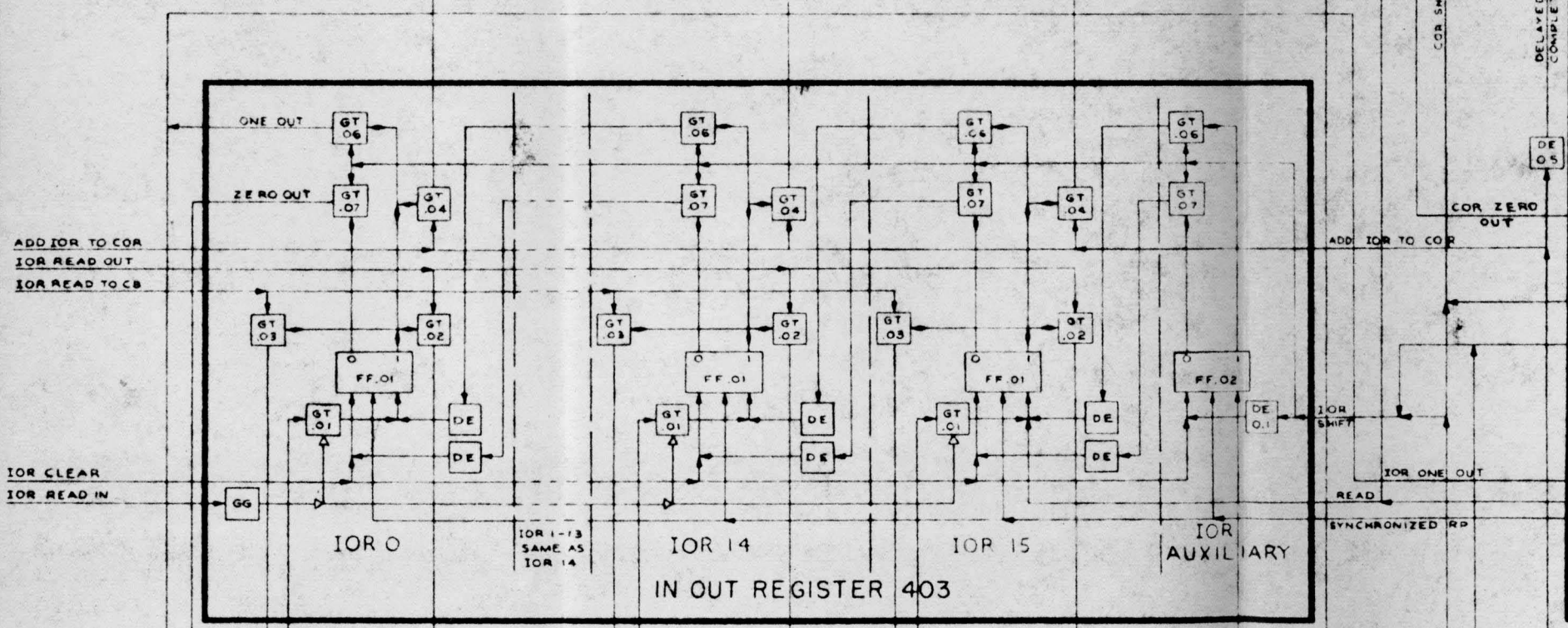
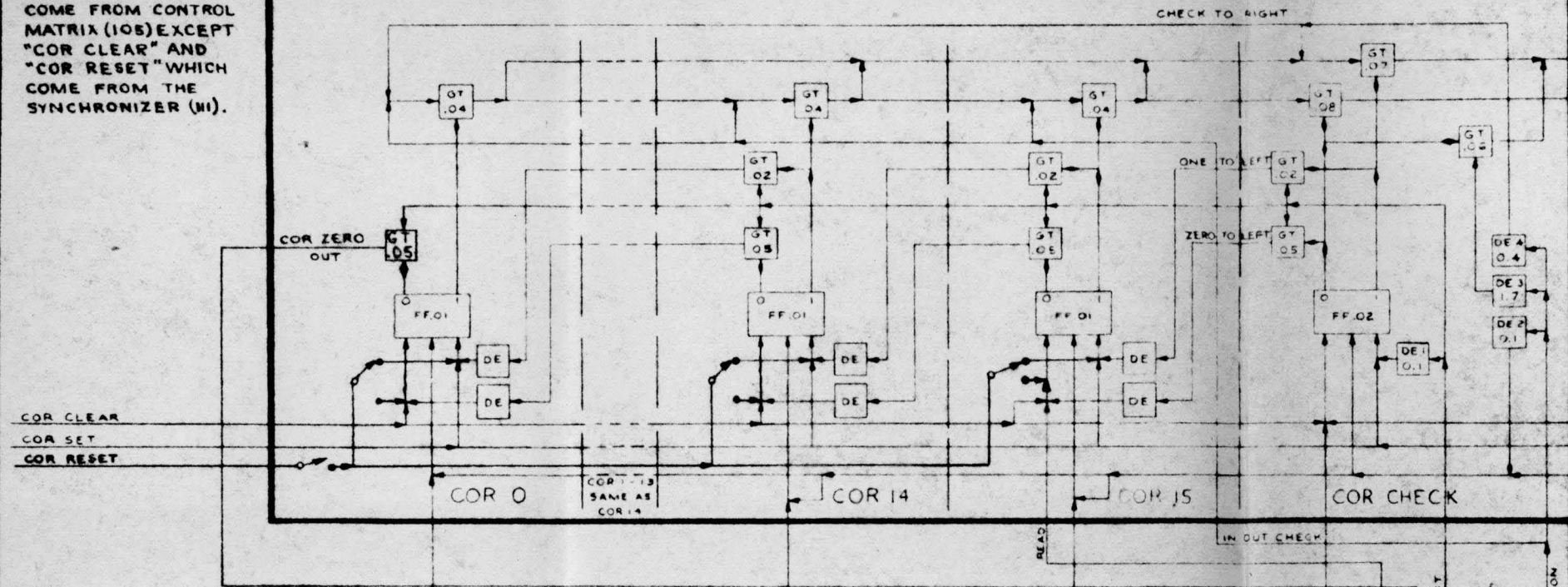




**FIG. 14**

NOTE:  
INPUT LINES BELOW  
COME FROM CONTROL  
MATRIX (105) EXCEPT  
"COR CLEAR" AND  
"COR RESET" WHICH  
COME FROM THE  
SYNCHRONIZER (M).

### COMPARISON REGISTER 404



GRADED BY: DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW  
 GRADE I FOR REFERENCE ONLY  
 GRADE II PRELIMINARY DESIGN  
 GRADE III FINAL DESIGN

REVISION 7-10-43

REPRODUCED LABORATORY OF THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 554-B

BLOCK DIAGRAM, 400, INPUT OUTPUT ELEMENT, W

SCALE: DE 7-10-43 7-8-43  
 8-REDUCTION

D-37178-2

FIG. 15

PROGRAM TIMING

CONTROL PULSES (COMMANDS)	TIME PULSES								CONTROL MATRIX OUTPUT		
	1	2	3	4	5	6	7	8			
102 PROGRAM COUNTER READ OUT		2							2	12	
PROGRAM COUNTER READ TO CHECK BUS		2½							2½	9	
ADD TO PROGRAM COUNTER						7				14	
103 PROGRAM REGISTER CLEAR		2							2	38	
(SEE 200) PROGRAM REGISTER READ IN				4						39	
PROGRAM REGISTER READ OUT					‡					28‡	
104 CONTROL SWITCH CLEAR				4						68	
CONTROL SWITCH READ IN					‡					66‡	
CONTROL SWITCH READ OUT						6				77	
201 STORAGE SWITCH CLEAR { TSS ½ μs LATER	*			4					*	4	*20
820 STORAGE SWITCH READ IN { SELECTS TS OR ES AND LEAVES UNSELECTED SWITCH CLEAR		2			‡				2		17
STORAGE SWITCH READ OUT			3			6			3		37,18,37
200 STORAGE READ OUT { PR ALWAYS, TS ONLY IF SELECTED				4						4	76
103 STORAGE READ TO CHECK BUS				4						4	74
305 STEP COUNTER PRESET				4						4	3
STEP COUNTER READ IN					‡						2‡
601 CHECK REGISTER READ IN		3				6			3		49
TRANSFER CHECK		3½				6½			3½		56
CHECK REGISTER CHECK							8½				47
810 ES READ		3							3		81

\* STORAGE SWITCH CLEAR ALWAYS OCCURS ON TIME PULSE 1, BUT OUTPUT NUMBER 82 OR 19 IS USED, DEPENDING ON WHETHER OR NOT ES WRITE OCCURS. SEE EACH INDIVIDUAL OPERATION.

‡ THESE OUTPUT UNITS, WHICH ARE PULSED ON TIME PULSE 5, ARE GATED BY THE ri OPERATION LINE AND NOT BY THE PROGRAM TIMING LINE. (OUTPUT NUMBER 17 IS GATED BY BOTH). THIS IS BECAUSE "CONTROL SWITCH CLEAR" ON TIME PULSE 4 SELECTS ri, DE-ENERGIZING THE PROGRAM TIMING LINE.

GRADED BY: DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW:  
 \_\_\_\_\_ GRADE I FOR REFERENCE ONLY  
RRE 11-21-99 GRADE II PRELIMINARY DESIGN  
 \_\_\_\_\_ GRADE III FINAL DESIGN

SERVO MECHANISMS LABORATORY OF THE  
 MASSACHUSETTS INSTITUTE OF TECHNOLOGY  
 DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345

TIMING DIAGRAM, WW1  
 PROGRAM TIMING  
**FIG. 16**

DR. F. B. NOV. 15 '49

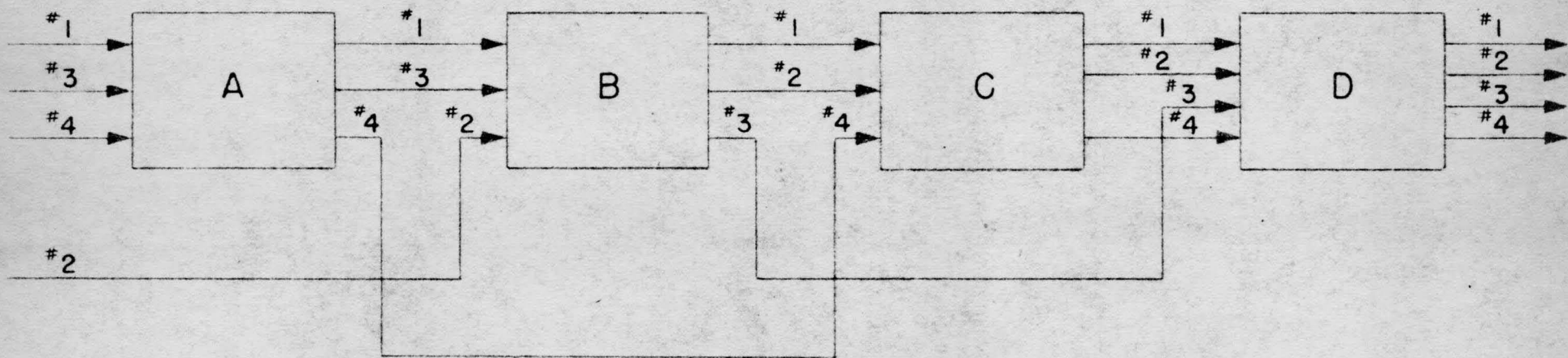
SCALE: 4/15/49 R.P.M.

CK.

APP.

B-37195-1

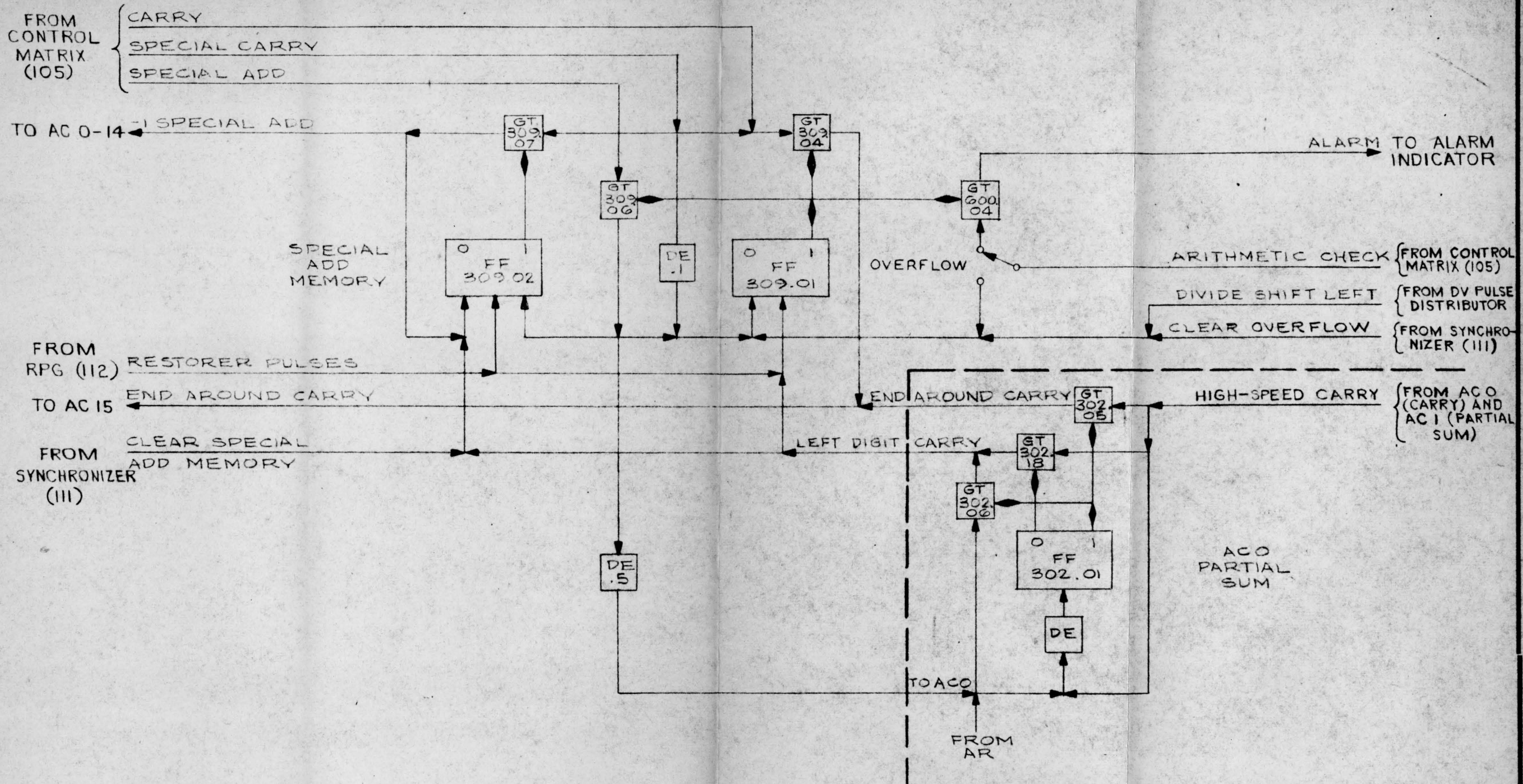
ALABAMA... NO. 100... K&S CO. N.Y.



MULTI-CHANNEL TROUBLE LOCATION



B-37174-2



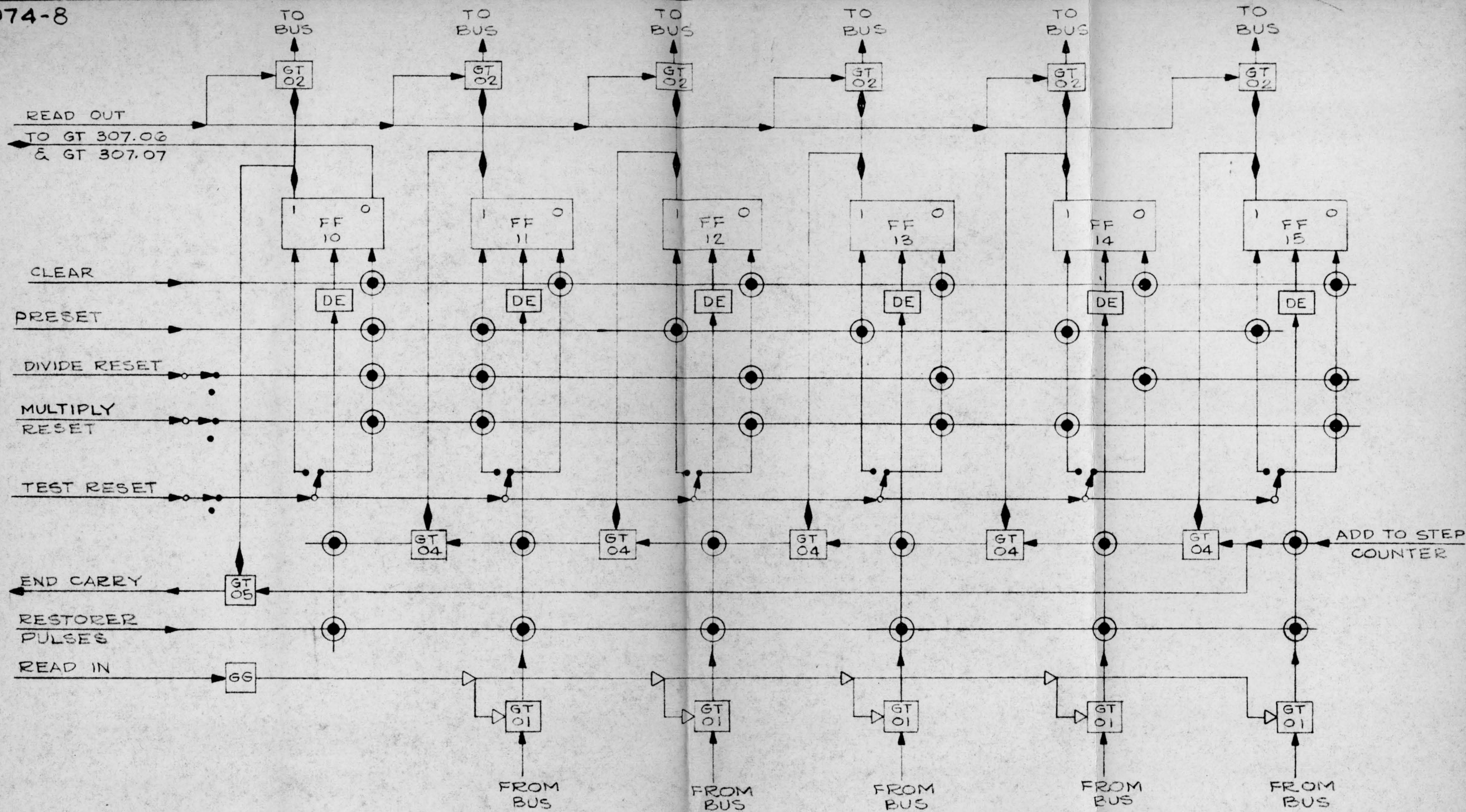
SERVOMECHANISMS LABORATORY OF THE  
**MASSACHUSETTS INSTITUTE OF TECHNOLOGY**  
 DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345

GRADED BY: DATE: THIS IS A GRADED DRAWING OF  
 HIGHEST GRADE APPROVED BELOW:  
 GRADE I FOR REFERENCE ONLY  
*PCE* *3/24/49* GRADE II PRELIMINARY DESIGN  
 GRADE III FINAL DESIGN

**BLOCK DIAGRAM, 309**  
**SPECIAL ADD MEMORY AND OVERFLOW, WWI**

SCALE: \_\_\_\_\_ DR. *gja* 2-23-49  
 ENG. *gml* 3/9/49 CK. \_\_\_\_\_ APP. \_\_\_\_\_

**B-37174-2**  
**FIG 10**



NOTE: FF NUMBERS CORRESPOND TO BUS NUMBERS.

GRADED BY: DATE: THIS IS A GRADED DRAWING OF HIGHEST GRADE APPROVED BELOW:  
 PRE 3-30-49 GRADE I FOR REFERENCE ONLY  
 GRADE II PRELIMINARY DESIGN  
 GRADE III FINAL DESIGN

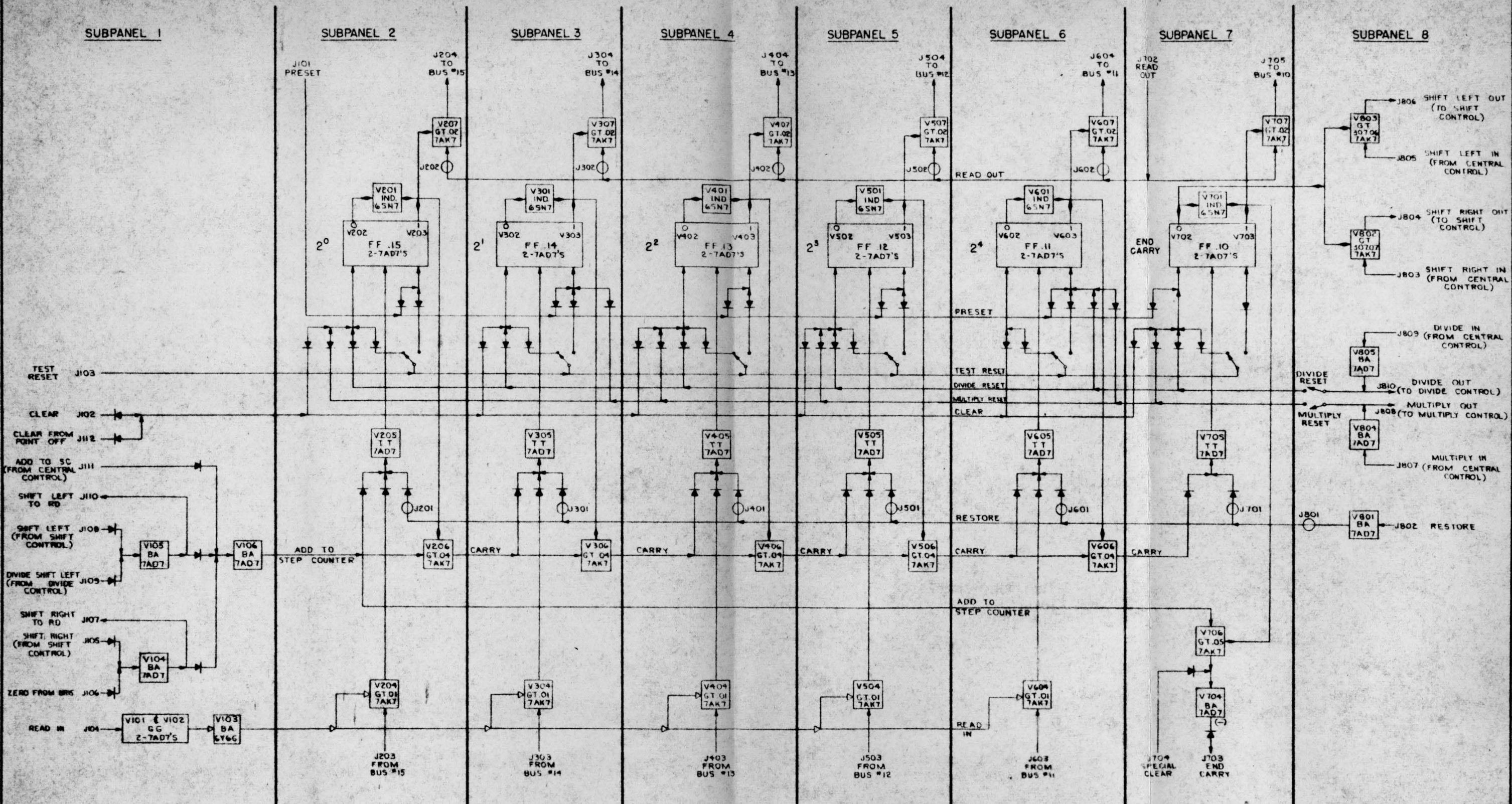
SERVOMECHANISMS LABORATORY OF THE  
**MASSACHUSETTS INSTITUTE OF TECHNOLOGY**  
 DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345

BLOCK DIAGRAM, 305, STEP COUNTER, WWI

SCALE: DR. *gja* 2-25-49

ENG. *JMS* 3/16/49 CK. APP.

**B-37074-8**  
**FIG. 20**



DRAWING REFERENCES:  
 CIRCUIT SCHEMATIC: D-39764  
 BLOCK DIAGRAM: B-37074  
 ASSEMBLY: D-32308

*Handwritten signature*

RESEARCH LABORATORY OF THE  
 MASSACHUSETTS INSTITUTE OF TECHNOLOGY  
 DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6346

BLOCK SCHEMATIC, 305,  
 STEP COUNTER, WWI

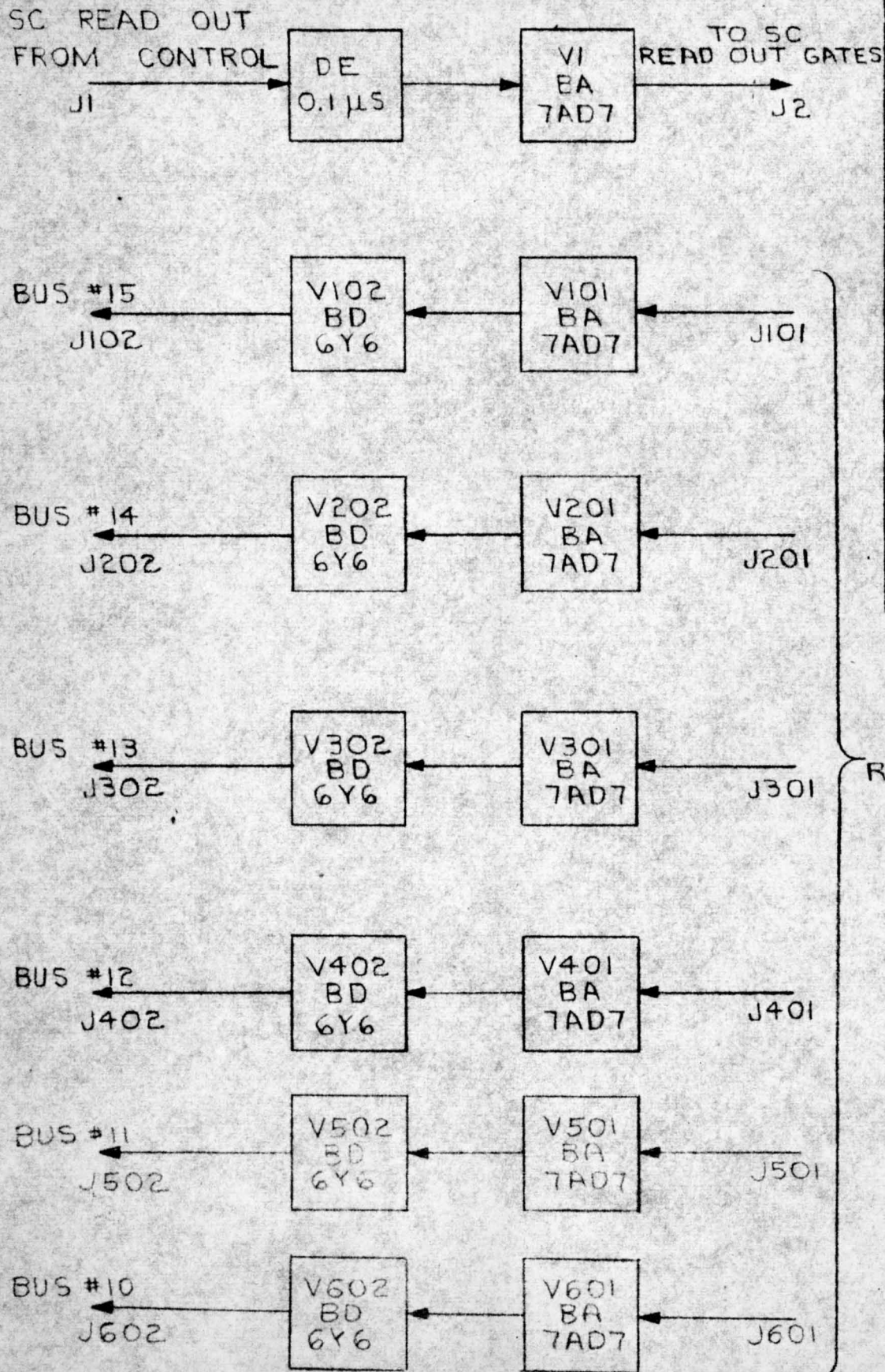
DATE: 5-18-49  
 DRAWN BY: 5126/48  
 CHECKED BY: [Signature]

D-31828-2  
 8-REDUCTION

TRADED BY DATE: [Signature]  
 [Handwritten notes and stamps]



A-32723-1



SERVOMECHANISMS LABORATORY OF THE  
**MASSACHUSETTS INSTITUTE OF TECHNOLOGY**  
 DIVISION OF INDUSTRIAL COOPERATION PROJECT NO. 6345

BLOCK SCHEMATIC, STEP-COUNTER  
 OUTPUT, 305, WW1

SCALE: DR. D.L.S. 8-11-48  
 ENG. *M.M.* 8-18-48 APP. *S*

**A-32723-1**

FROM SC READ OUT GATES

GRAPHIC DATE: THIS SCHEMATIC DRAWING IS  
 HIGH PRIORITY PROJECT  
 -GRADE 1 FOR SECURITY CLASS  
 8/23/48  
 GRADE 1 FOR SECURITY CLASS

**FIG. 22**  
 DRAWING REFERENCE:  
 CIRCUIT SCHEMATIC: D-32735

PROBLEM BEING PERFORMED	VOLTAGE-VARIATION LINE NUMBER									
	77		78		79		80		113	
DISPLAY PROGRAM #1 (POWERS OF X)	(1) -32	(1) +34	(1) -30	(1) +31	(1) -10	(1) +49	(1) -13	(1) +30	(1) -17	(1) +48
DISPLAY PROGRAM #1 (PARABOLAS)	(1) -32	(1) +34	(1) -30	(1) +31	(1) -10	(1) +49	(1) -16	(1) +30	(1) -18	(1) +48
TEST PROGRAM #1	-28	+39	-29	+27	(2) -20	+41	-24	(2) +33	-9.5	+46
TEST PROGRAM #2	-25	+30	-26	+25	-18	+40	-22	(2) +23	-10	+38
TEST SEQUENCE #1	-26	+33	-25	+26						
TEST SEQUENCE #2					-17	+42	-22	+31		
TEST SEQUENCE #3									-11	+39

NOTE 1: NO ACTUAL ALARM OBTAINED. MARGIN MEASURED BY NOTING VOLTAGE AT WHICH THE PATTERN ON THE DISPLAY OSCILLOSCOPE BECOMES DISTORTED.

NOTE 2: NO ACTUAL ALARM OBTAINED. THE "PROLONGED STOP-CLOCK" CONDITION OCCURS AT THE INDICATED VOLTAGES.

MARGINS OF CORRECT OPERATION  
FOR SAMPLE PROBLEMS

(DATA TAKEN ON DECEMBER 9, '49)

**USN**

---

---