



CONTROL DATA INSTITUTE

160A

TRAINING WORKBOOK

FOR TRAINING PURPOSES ONLY

This manual was compiled and
written by members of the
instructional staff of

CONTROL DATA INSTITUTE
CONTROL DATA CORPORATION

CDI 60273400
AUGUST 1966

HISTORY

This is the same manual that was previously published under publication number 082666.

TABLE OF CONTENTS

| | page |
|------------------------------------|-------------|
| Introduction to Programming | 1 |
| Programming Examples | 14 |
| I/O Instructions..... | 19 |
| OSAS-A | 25 |
| Re-sync Counter | 26 |
| Storage Reference Cycle | 30 |
| Registers and Inverts | 54 |
| F and F' Registers | 71 |
| Input/Output Section | 74 |
| Interrupt | 98 |
| Power Supply | A-1 |
| Control Data Logical Symbols | B-1 |
| Assignment Sheets | C-1 to C-50 |
| Cycle Timing Charts | C-51 |
| Phase Timing Charts | C-52 |

INFORMATION SHEET

INTRODUCTION TO PROGRAMMING

1. Introduction

Any problem to be solved by the computer must first be formulated and analyzed, then outlined or flow charted to display the approach in terms of the general capabilities of the computer. This outline must then be translated or coded, using the instructions available for the computer. The program or list of instructions is placed on a physical input medium so that the relevant data can be read into the computer's internal memory.

The three functions: analysis, programming of the analyzer problems, and coding of the program, may be performed by different individuals or by one person. The distinctions between the functions are apt to overlap, varying from problem to problem. For this discussion, analysis, programming, and coding will be included under the heading of programming to give a sense of unity to preparation of a computer problem.

Any problem must be formulated in detail before a program can be devised. In the case of a straightforward mathematical problem such as a differential equation, the formula is well known (among mathematicians) and a usable approach is usually available in text books or printed literature.

Once the formulas are known, the analyst and programmer must choose a numerical method which meets his problem needs in regard to accuracy, time required, etc. Much has been accomplished in the field of numerical analysis in

in the past decade by means of high speed computers. These machines perform repetitive calculations in so short a time that hundreds of iterative cycles become practicable. The growing amount of information on convergence, accumulation of numbers and general precision of solutions often makes a usable numerical approach available.

As a rule, data handling problems takes longer than mathematical problems. For example, the operation of posting an entry to an inventory account seems very simple, requiring only the subtraction of the number of items used, and replacing the total with the new figure. However, additional information must be kept and pertinent clerical operations are usually wanted in connection with this same account. These extra items can also be handled by the machine if it is to be used efficiently. The questions to be answered might be:

Is there a sufficient amount to cover the request?

If so, make the entry. After the entry has been made, have you reached your stock minimum? If so, set up the name of the vendor and indicate how many items to be ordered. If not, go on to next account.

If there is not adequate stock, indicate how many items can be supplied, and set up for back order --- etc.

A program can include almost any combination of clerical operations. However, the user must outline in great detail what items are to be handled by the computer program, how he wishes exception to be handled, and what his particular organization requires for records.

Many procedures now allow partial computations with desk calculators or electro-mechanical machines, after which someone must examine account records

or reports and use human judgment. Further machine computation may be used again. Human supervision often depends on the size of a number, comparison with other magnitudes, or whether a number lies within a reasonable range. Comparisons can be programmed readily where the full formulation is presented to the programmer. The analyst (or formulator) must determine how the human judgments are made, even where they seem almost intuitive. Eventually, he must set up the numerical expression for each decision.

These comments apply in general to many problems in data handling. Lack of complete formulation and diversity of information desired make data handling problems of this type more difficult to program and analyze. Here, it is imperative that the programmer have a comprehensive understanding of the particular procedure to be put on the machine.

The communication medium between the programmer and the computer can be magnetic tape, paper tape, punched cards, etc. Once the program is completed and the shortest possible path for the solution is found, the input equipment is used to place the instruction or code into the internal memory of the computer. Normally these instructions are located in sequential address locations, each instruction performing a logical step to produce the result of a given problem. This type of programming is called "internal stored program".

Decisions in the program can be made because of conditions found in a register, $A=0$, $A\neq 0$, $A+$, $A-$, etc. When these conditions arise in certain parts of the program, the computer can be made to jump to a secondary program to handle the condition, thereby deviating from the regular or main program.

Computers such as the 160-A or 8090 are capable of communicating with external equipment during the normal program routine. Such output could be available to magnetic tape units, paper tape punches, typewriter, etc. This would give the programmer a visual indication of the operation controlled by his program.

The reader should bear in mind at all times that the computer is not capable of thinking. It is a means to speed up data processing or solve other problems at a high rate of speed. Its only means of intelligence is the written program.

2. Internal Programming Concepts

Once the program has been written and coded in an acceptable form, it is entered into the storage section of the computer. The computer can now, upon proper initiation, execute the instructions of the program. The program instructions are stored in the memory (MCS) section of the computer in a sequential manner, so that the computer will first execute the instruction located at the lowest address of the program and proceed to the highest address. From its address location the instruction is moved to the control section where the computer analyzes the instruction to determine the method of execution. (The original form of the instruction in memory is not altered.)

Normally the next instruction to be executed is located at the address that is one greater in value than the address of the previous instruction.

3. Programming Fundamentals

a. Problem Analysis: The first step to be taken after determining that a problem exists, is to find the shortest and most accurate way to solve the

problem. It is a good idea first of all to sit down, think over the problem, write it down and then try to make the analysis of the best method to follow in solving it. Writing down the problem will usually clarify the situation and will also give you a constant reference to some of the points you are not sure of. It must be remembered at all times when considering the use of any computer for determining a solution to a problem, that you must keep within the capabilities of the computer.

b. Flow Diagrams: After it has been established that a particular problem can be solved best by the computer, it is necessary to formulate the problem in terms of the language of the computer. A program of instructions and the data needed for the solution of the problem must be devised.

A flow diagram is helpful in facilitating the coding or programming of the problem. A flow diagram or flow chart indicates the flow, or steps, in the computation that lead to the solution of the problem.

A basic flow diagram lists the series of simple arithmetic steps which are to be performed by the computer. It is imperative that the coder be thoroughly familiar with the over-all operations and the peculiarities of each computer instruction so that he can construct the outline with regard to the capabilities of the computer.

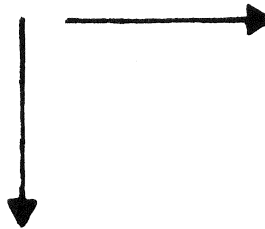
Often more than one flow diagram is necessary for the more complicated problems. The first flow diagram may be nothing more than equations, written in the sequence in which they will be computed, with brief explanations of the steps involved. The second type of diagram usually shows the flow of computation. This diagram can contain the instructions necessary for the data input, the

instructions that operate on the input data to obtain the solutions, and the necessary instructions for the output of the results.

Some problems may require a second type of flow diagram, consisting of many charts and/or diagrams, each a more detailed presentation of the preceding charts.

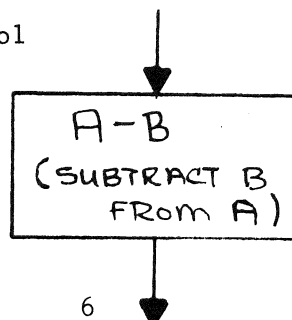
To facilitate the task of forming flow diagrams and to allow other programmers and maintenance men to understand a particular flow diagram, certain symbols have been more or less standardized. The following list of symbols is by no means complete, but is intended to give the coder an example of the basic symbols used in drawing the second type of flow diagram.

1) Lines of flow



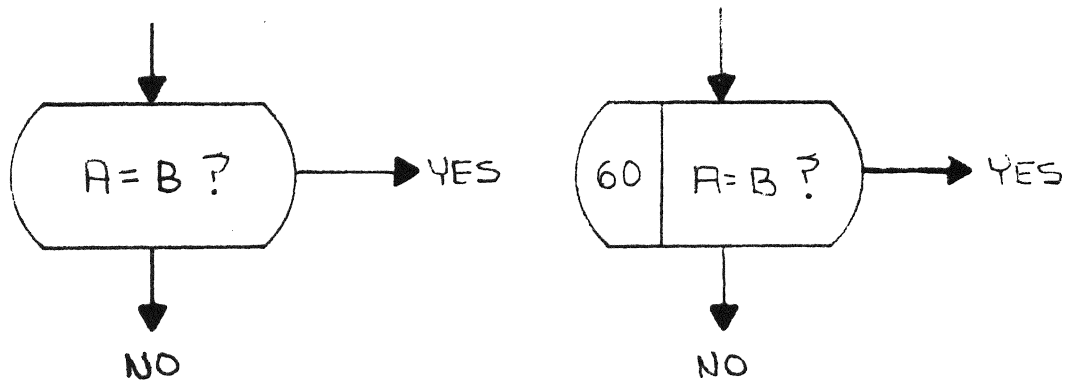
A solid line with an arrow touching the next element of the flow diagram is generally used to indicate the path to be followed by the computer, or, more precisely, the path to be followed by the coder who is formulating the computer instructions from the flow diagrams.

2) Operation symbol



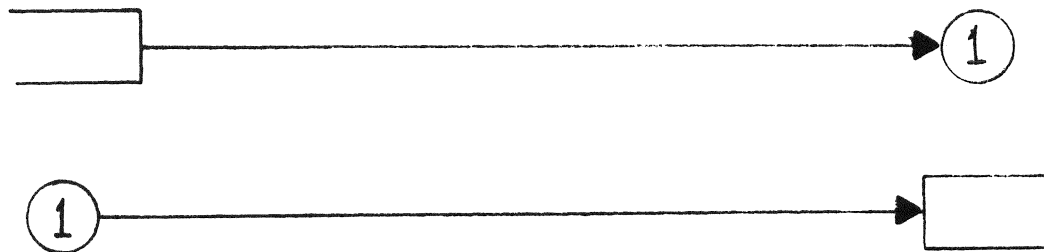
The rectangular box usually contains a statement about a computation or mathematical operation. The content of this box may be a simple statement or a mathematical expression.

3) Decision symbol



The symbol above is used to indicate a branch condition in the program. It indicates that there are two possible progressions from this point in the program. This symbol is sometimes written as (b) above when the programmer has a definite instruction in mind for use at this point.

4) Connectors and remote connectors



To eliminate as much as possible the crossing of lines of flow on a diagram, the above symbols are used to indicate a destination not easily reached in the diagram. Thus, the flow can be broken at a convenient point by terminating it in an arbitrary symbol which will initiate the flow in another part of the flow diagram.

5) Stop symbol

(a)



(b)



To indicate the point(s) in the program where a stop is to be made, whether at the completion of problem or at various points in the program, the stop symbol is used. If a conditional stop is required at this point, the condition of the stop may be included in the symbol as in (b) above.

6) Program entry and exit

(a)



(b)



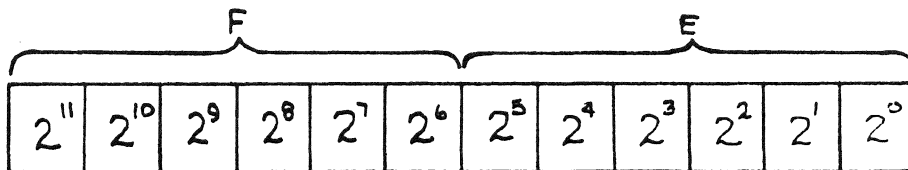
(c)



To enable another programmer to locate the beginning and end of the program in flow diagram format, the entry and exit symbols should be used. In programs of a complex nature there can be more than one program exit. Therefore, appropriate numbers may be placed in the exit symbol as shown in (c) above.

4. Instruction Format

The instruction word as used in this computer consists of 12 bits. These 12 bits are divided into two groups of six each as shown in figure 1.



The "F" portion of the instruction word contains the function code which may be described by a mnemonic code or a unique combination of two octal characters.

The "E" portion of the instruction may be written in either symbolic or numerical form. This portion of the instruction will be either an operand or an address which will lead the instruction to the operand.

The discussion at this time will be limited to the mnemonic codes associated with only a few of the basic instructions. The absolute (numeric) codes and the entire repertoire of instructions will be covered later. The information presented here is intended only as an introduction to programming procedures.

5. Basic Instructions (Mnemonic)

- a. LD - Load the A register with the contents of a memory location specified by the E portion of the instruction.
- b. LC - Load the A register with the COMPLEMENT of the contents of a memory location specified by the E portion of the instruction.
- c. ST - Store the contents of the A register in a memory location specified by the E portion of the instruction. This instruction does not alter the contents of the A register.
- d. AD - Add to the previous contents of the A register the contents of a memory location specified by the E portion of the instruction; the sum appears in the A register.
- e. SB - Subtract from the previous contents of the A register the contents of a memory location specified by the E portion of the instruction. Place the resulting difference in the A register.
- f. LP - Perform a logical product between the previous contents of the A register and the contents of a memory location specified by the E portion of the instruction. The result of the logical product appears in the A register.

- g. LS - Left shift the contents of the A register by the value placed in the E portion of the instruction. This value may be 1 or 2. This shift is open-ended with sign extension, i.e., bits shifted out of bit position 00 are lost and for each shift to the right, the sign bit value is automatically placed in bit position 11.
- i. JP - Jump to an address specified by the E portion of the instruction and execute the instruction located there. This will allow moving to a place in the program that is out of sequence.
- j. HLT - Halt. This will bring the computer to a stop. To begin operation again, a manual operation is required.

6. Symbolic Addressing

When writing a program, the first step is to analyze the problem and the second is to flow chart it. The third step is to convert the flow chart into a program consisting of actual instructions. This program may be first written in mnemonic codes as shown above. At this point, it may not be known exactly how many memory locations will be required for the program and the constants. The programmer usually has not decided exactly where in the memory this program is to go, i.e., he does not know where the first address of the program is to be. To facilitate the writing of the mnemonic program, symbolic addressing may be employed. Instead of using actual numbers for the E portion of the instruction (such as memory location 0100) and for memory locations of individual instructions, a symbolic letter may be substituted. For example: let the first instruction of the program be located at address Q. The second instruction will then be located at Q+1, the third at Q+2 and so on. Several different letters may be used in one program. This is especially true where the program has several separate parts. The E portion of the instruction

will then make reference to a symbolic address. The following example will explain this procedure.

| Address | Instruction |
|---------|-------------|
| T | AD R+2 |
| T+1 | ----- |
| T+2 | ----- |
| R | ----- |
| R+1 | ----- |
| R+2 | 3026 |

When the instruction located at address "T" is executed, the contents of address R+2 (3026) will be added to the previous contents of the A register. The next instruction will be read from address T+1 and so on.

When the program is put into its final form for loading into the computer, the instructions must be written in numeric form which is acceptable to the computer. To convert the symbolic addresses to actual addresses, numerics are substituted for symbols. An example of a complete programming operation will be described in the next topic.

7. Debugging Operations

a. After the coded program has been written out completely and the manuscript carefully reviewed, the program must be put on some input medium for loading into the computer. Once the program has been stored in the computer in binary form, either directly from the input medium or by means of a translation routine, it is ready for execution by the computer. However, if the program is lengthy, it is likely that the coded program as stored has errors. Three common types of errors are listed below:

- 1) Tape errors made in the preparation of the program for input,
- 2) Coding errors, such as listing incorrect or incomplete addresses, transposition of digits in the address portion of the instruction or function code portion, and transcription errors.
- 3) Logical errors, incomplete or erroneous methods used to obtain the solutions.

Debugging is the term applied to the process of locating errors in a program and correcting them. Debugging a program usually involves a series of trial runs of the program on the computer. Each time the program fails to run properly the failure must be analyzed and the error corrected. Frequently one can discover the error immediately, correct it manually from the control panel, and proceed with the trial runs. At other times the detection of errors is more difficult and requires a thoughtful reconsideration of the program. Error detection may be facilitated by the use of a special program which is coded expressly for this purpose. The methods that can be employed to debug a program will vary depending on the nature of the program, the debugging programs that are available, the availability of computer time, and the personal preference of the individual for one procedure over another. The following paragraphs suggest possible patterns to follow in debugging.

b. Tape preparation errors. This type of error can be minimized by systematic checking of all input tapes before their information is read into the computer. The usual method that is used to prepare the tapes is:

- 1) Use the typewriter to punch tape from the coder's manuscript,
- 2) Discard the resulting typed manuscript from the typewriter,
- 3) Prepare a new typed manuscript from the punched paper tape,

- 4) Compare this typed manuscript with the coder's manuscript to detect errors in punching,
- 5) Correct all detected punching errors and prepare a new tape,
- 6) Obtain from the corrected punched tape a new manuscript which should be retained by the coder to use while debugging.

c. Manual Debugging. In manual debugging, the operations of the program are controlled from the control panel and visually checked on the panel readouts. Although manual debugging does not provide efficient use of computer time, it can, if done correctly, reduce the over-all time required to debug a program because of the versatility afforded by a thinking programmer at the control panel.

8. Summary. We now have a finished program in machine code that can be inserted into the computer. Let us review the steps necessary to resolve our problem.

- 1) Analyze the entire problem,
- 2) Construct the flow diagram which shows the general computational steps to be taken,
- 3) Create the program in terms of computer instructions,
- 4) Place the program on an input medium acceptable to the computer,
- 5) Debug (computer checkout) the program by means of trial runs on the computer.

INFORMATION SHEET

PROGRAMMING EXAMPLE

PROBLEM

Bnk 0 of the computer contains three unknown numbers, stored in memory locations 0050, 0250, and 1000. Flow chart and write a program that will do the following:

1. Add the first number to the second number.
2. Subtract the third number from that sum.
3. Increase the result by $+10_8$
4. Store the result at memory location 2000.
5. If the result is positive, halt with the result displayed in A register.
6. If the result is negative, halt with the complement of the result displayed in A register.

The following restrictions apply.

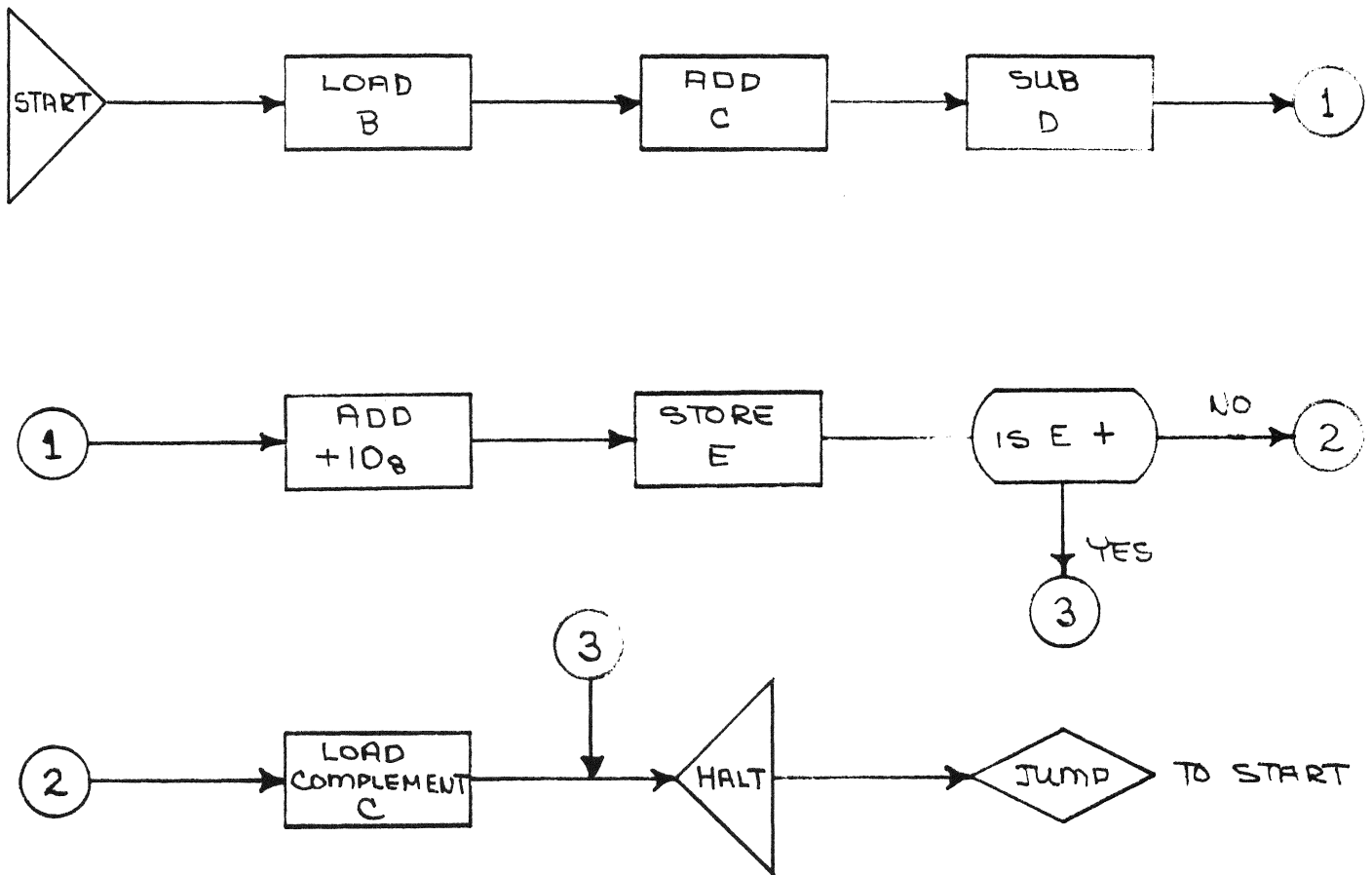
1. The instructions of the program must be stored between memory locations 0100 and 0150, with the starting address = 0100.
2. When the computer halts, you must be able to repeat the program by taking the RUN-STEP switch out of run, and setting it back in the run position.

ANALYSIS

Let the unknowns be located at b, c, d, and e, where b = ML 0050, c = ML 0250, d = ML 1000. The final result will be stored in ML 2000, and symbolized by e. (ML indicates memory location.)

The problem may now be stated: $(b) + (c) - (d) + 10 = (e)$; if (e) is positive stop, if (c) is negative, load the complement into A register and stop.

FLOW CHART



Check the flow chart with the problem. If the chart is satisfactory, write a symbolic code listing.

| ML | Function or Operation | Operand or Operand Address | Comments |
|------|-----------------------|----------------------------|---|
| G | Load | b | (program start) |
| G+1 | Add | c | |
| G+2 | Subtract | d | |
| G+3 | Add | +10 ₈ | |
| G+4 | Store | e | |
| G+5 | Sign check | e | halt if positive, complement if negative. |
| G+6 | Load Complement | e | |
| G+7 | Halt | | |
| G+10 | Jump to Start | | |

Compare listing, flow chart, and problem.

The location of the operands may be specified relative to the instructions, since the program start address was given. Start = ML 0100, b = ML0050, The address of b could be written as 0100-50, or from the symbolic listing b = G - 50. Operand locations may be added to the listing, and mnemonic codes substituted for the operation called for.

| ML | F | E | Comment |
|------|-----|-----|--|
| G | LD | b | b = G - 50 |
| G+1 | AD | c | C = G + 150 |
| G+2 | SB | d | d = G + 700 |
| G+3 | AD | 10 | constant = 10 ₈ |
| G+4 | ST | e | e = G + 1700 |
| G+5 | PJ | G+7 | jump to HLT if (e) positive |
| G+6 | LC | e | e = G + 1700. Load complement if (e) was negative. |
| G+7 | HLT | | |
| G+10 | JFI | G | jump to start. |

Now that operand locations have been given relative to instruction locations, address modes may be selected for the instructions, and the third letter of the mnemonic codes added to the listing.

| ML | F | E | Comment |
|------|----------|----------|--|
| G | LDB | b (G-50) | Since the operand is less than 77_8 locations away, Relative mode may be used. |
| G+1 | ADM | 00 | The operand is more than 77_8 locations away, therefore, Memory or indirect modes may be used. Let us use Memory mode. |
| G+2 | G+150 | | Because Memory mode was chosen for the add instruction this location must contain the operand address for the add instruction. |
| G+3 | SBM | 00 | Because of the above Memory mode the subtract instruction has been moved to location G + 3. Memory mode may be used for this instruction also. |
| G+4 | G+700 | | Operand address for the subtract instruction. |
| G+5 | ADN | 10 | Since the constant is less than 77_8 , it may be contained in the E portion of a No address mode instruction. ("A" register will contain the sum upon completion of this instruction.) |
| G+6 | STM | 00 | Memory mode may be used to store the final result. (The content of "A" register is NOT modified by the Store instruction.) |
| G+10 | PJF | G+13 | Check the sign of the result. If it is positive, jump to HALT. If negative, take the next instruction at G + 11. |
| G+11 | LCM | 00 | The result was negative. Load the complement of the result in "A" register. |
| G+12 | G + 1700 | | Operand address. |
| G+13 | HLT | | Program stop. The proper result indication is in "A" register. |
| G+14 | JFI | 01 | Jump to restart program if RUN switch is again operated. |
| G+15 | G | | Jump address. |

Absolute codes may now be substituted for the symbolic codes listed. All memory locations may be found since the starting address is given. (In computing these absolute numbers, remember to perform octal arithmetic.)

Starting Address = 0100

| ML | F | E |
|------|------|---|
| 0100 | 2330 | Subtract the operand address, (0050) from the instruction address to obtain E portion of this instruction. |
| | | $\begin{array}{r} 0100 \\ -0150 \\ \hline 0030 \end{array}$ |
| 0101 | 3100 | |
| 0102 | 0250 | From the first symbolic listing, we find C (operand location) given as G + 150. |
| | | $\begin{array}{r} G = 0100 \\ +0150 \\ \hline 0250 \end{array}$ is absolute operand address. |
| 0103 | 3500 | |
| 0104 | 1000 | |
| 0105 | 0610 | |
| 0107 | 2000 | |
| 0110 | 6203 | From the symbolic listing, the jump instruction was at G + 10, the HLT instruction was at G + 13. Therefore, the jump is 03 places forward. |
| 0111 | 2500 | |
| 0112 | 2000 | |
| 0113 | 7700 | Halt. |
| 0114 | 7101 | |
| 0115 | 0100 | |

The program is now ready to be punched on paper tape, loaded into the computer, and debugged.

INFORMATION SHEETS

I/O Instructions

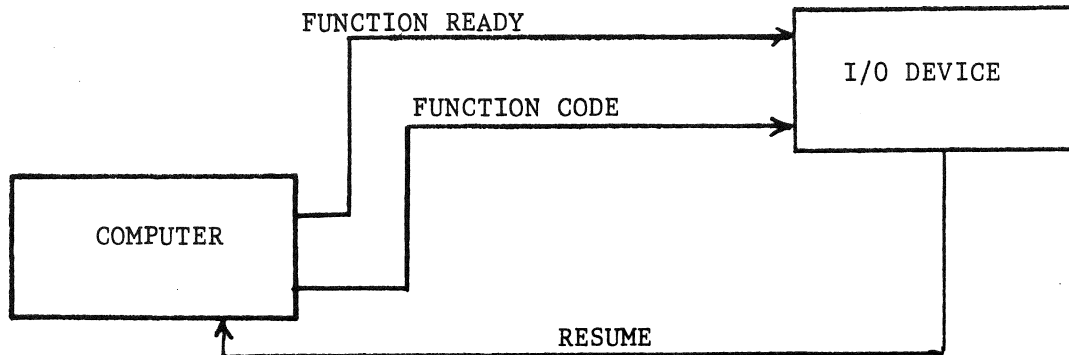
The computer is easily connected to any of the I/O equipment designed to be used with it. There are jacks located at the bottom of, and behind the main logic chassis (10100). All of these jacks are labeled, giving the function they serve. There are two jacks to which normal output cables may be connected to, and two jacks for normal input cables. There are also two jacks provided for buffer output and two for buffer input cables. The cables used to connect the computer to the I/O equipment are made up of 23 twisted pairs of wires. The connecting cables are identical, they may be interchanged with respect to physical connection as well as function they perform. On the cables a binary "1" is zero volts, the binary "0" is -16 volts. All binary digits of a computer word are sent out over the cable at the same time. Each digit is sent over a separate wire.

There may be as many as five different pieces of I/O equipment connected to a channel at the same time. More than five pieces of I/O equipment on a channel at one time will over load the transistors in the output amplifiers. The punch and reader are connected internally and do not count when figuring the number of pieces connected on a cable. Since more than one piece of I/O equipment may be connected on a cable, it is necessary to select the one piece of I/O equipment you want the computer to communicate with. This is done in programming by a external function instruction. This instruction sends out a code that selects the one piece equipment. When executing I/O instructions the speed of the computer and the I/O equipment must be

synchronized. In most cases the computer can execute instructions much faster than the I/O equipment can perform its part of the instruction. This is because most I/O equipment has some mechanical process to perform. Allowing the computer to run at its normal speed would make it impossible for the I/O equipment to keep up. This means that the computer must be stopped and wait for a signal from the I/O equipment that it has completed its part of the operation. This signal starts the computer and allows the computer to do the next instruction. For example, when the P.E.R. is reading tape at the rate of 350 frames per second, the computer could execute 446.4 cycles between each frame. After reading one frame on the paper tape the computer because of its speed will have to be stopped and wait for the tape to move on to the next frame. When the next frame is over the read station, the data from the tape is read and sent to the computer. With the data an input ready signal is sent from the reader that starts the timing chain and allows the computer to store the data. Then the computer asks for more data by sending out a input request signal, stops and waits for the tape to be moved into position for reading the next frame. The following are some examples of I/O instructions.

7500 Instruction: External Function

This instruction selects a piece of I/O equipment to be used in a I/O operation. It requires a D to B cycle sequence.



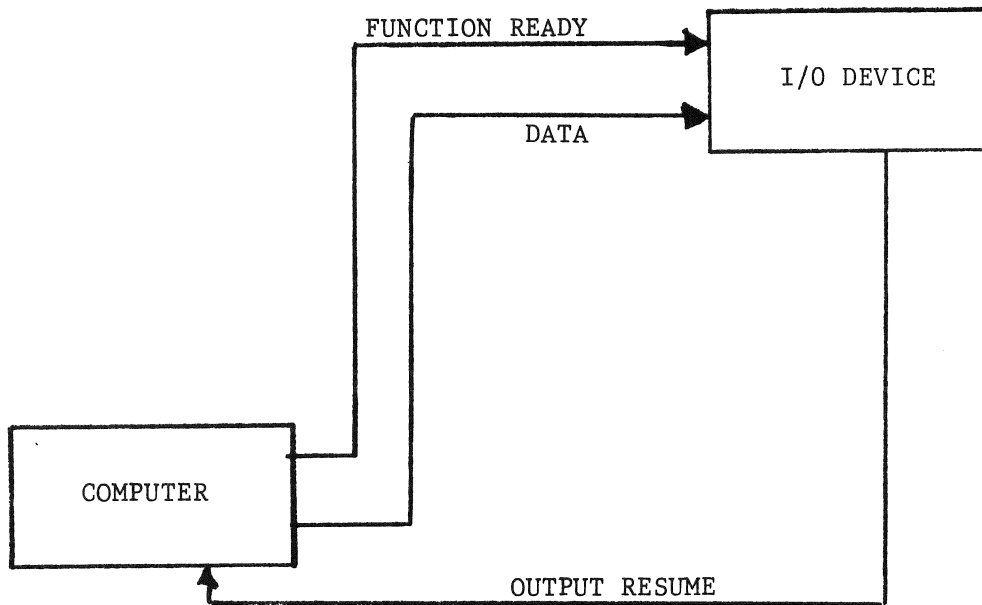
With more than one piece of I/O equipment connected to the input/output cables, the external function instruction sends out a function code that picks one piece of I/O equipment to communicate with the computer. In order to pick a piece of I/O equipment, the computer sends out two signals. These signals are a function ready and the function code. The two signals go out from the computer at the same time during the B cycle. The function ready stops the timing chain and alerts all the I/O equipment connected on the cable that the function code is also on the lines. The function code is a twelve bit code. The upper 6 bits select the piece of equipment. The lower 6 bits tell the equipment if it has been chosen to input to the computer or to receive a output from the computer. Only one of the pieces of equipment connected on the cable accepts the function code. The piece of equipment that accepts the code knows that it has been chosen and whether it will do a input or output operation. This piece of equipment now sends back to the computer a output resume. The output resume signal starts the timing chain and causes the function ready signal to be dropped. With the timing chain started, the computer goes into a D cycle and reads the next instruction.

Summarizing the 7500 Instruction

Most I/O equipment has some mechanical process that slows down its operation. This makes it necessary to stop the computer and give the I/O device time to perform its operation. The 7500 instruction then has two signals that synchronize the computer speed to the speed of the I/O device. Sending the function ready signal stops the timing chain by blocking recirculation. The computer then waits for the I/O equipment to complete its operation. When the I/O equipment finishes its operation, it sends

back to the computer a output resume signal. The output resume starts the timing chain and clears out function ready causing the function ready to be dropped. This removes the block on recirculation and allows the timing chain to run.

73XX Normal Output D-B-CD-C Sequence

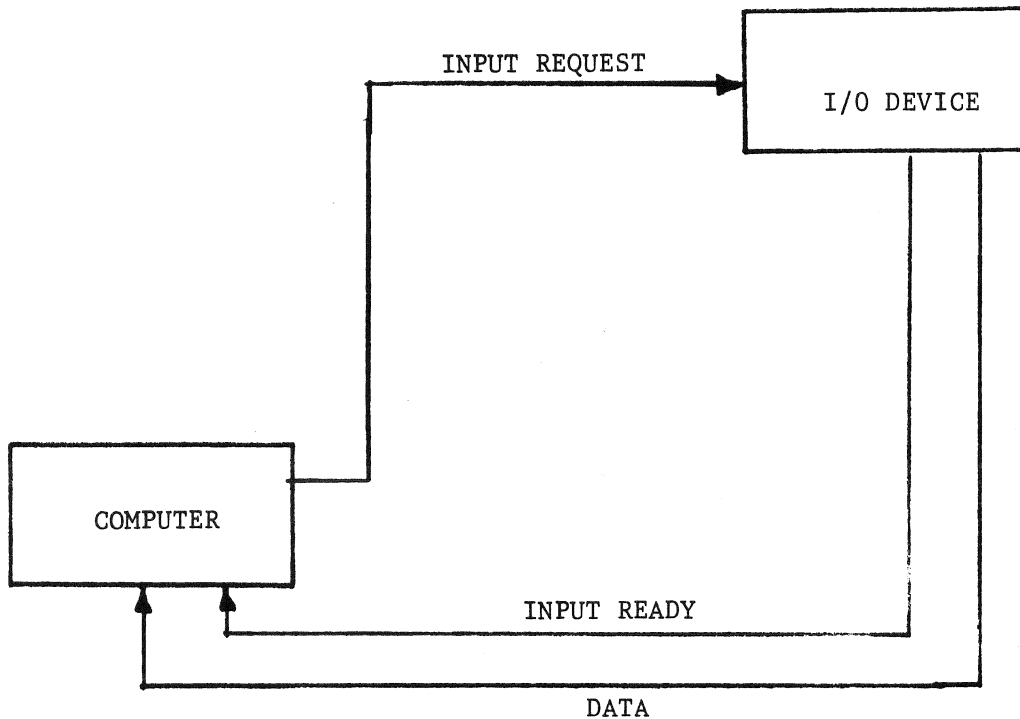


The Normal Output instruction is read up to the Z register and placed in F and F' on a D cycle like all other instructions. The computer then does a B cycle and places the first word address in the A register. Having the first word address, the computer goes into a C cycle. In the C cycle the computer references the first word address and reads the data found at this location up to the Z register. With the data in the Z register the computer sends a information ready signal to the I/O device, and at the same time sends out the data from the Z register. The information ready stops the main timing chain by blocking recirculation. Another important

thing that takes place during a C cycle is that the contents of the A register (the first word address) is increased by one. This makes it possible to read from memory the next character to be sent out from the computer. Because the information ready stopped the timing chain the computer is waiting for the I/O equipment to finish its operation. To tell the computer it has finished its part of the instruction, the I/O equipment sends back to the computer a output resume. The output resume starts the timing chain and the computer does a D cycle. In the D cycle the contents of the A register is compared to the last word address + one which is found at $P + L$ in reference to the location at which the 73XX instruction was read. If the A register and the last word address + 1 are not equal, the computer will do another C cycle and output the next character. The D to C cycle progression is only used on a normal input or normal output. The I/O sequence control F.F. must be set to do a D to C cycle progression. The I/O sequence F.F. gets set during the B cycle of a 72 or 73 instruction and stays set until the contents of the A register is equal to the last word address + one. During the D cycle in which the A register and the last word address + one are equal the I/O sequence F.F. will be cleared out. Now instead of doing a C cycle the computer will do another D cycle and read the next instruction terminating the output. In summary, the 73XX instruction has two signals that synchronize the speed of the computer to the speed of the I/O equipment. The information ready that stops the timing chain and tells the I/O equipment that the data is on the lines. This gives the I/O equipment time to perform its function. The second signal is sent back from the I/O equipment to the computer to let the computer know that the I/O equipment has completed its

part of the instruction and start the timing chain. This allows the computer to go into its next cycle.

72XX Normal Input D-B-C-D-C Sequence



The 72XX instruction is read during a D cycle and the computer goes into a B cycle. During the B cycle the first word address is read into the A register. The first input request is issued during the B cycle. The input request stops the main timing chain and the computer waits for the data to be sent in from the I/O equipment. When the I/O equipment is ready to send the data, a signal must also be sent to start the timing chain. This signal is called the input ready. The data and the input ready signal are sent to the computer at the same time. The data goes into the I' inverters and the input ready signal starts the timing chain. With the timing chain started the computer does a C cycle and stores the data sent to the computer from the I/O equipment. In the C cycle the

first word address is increased by one. This makes it possible to store the next input from the I/O equipment at the next memory location. Following the C cycle the computer goes into a D cycle to compare the first word address (contents of the A register) with the last word address +1. The last word address plus one are equal, the I/O sequence control is cleared out. With I/O sequence clear, the computer does another D cycle and reads the next instruction.

OSAS-A

Machine code "EE, and GGGG" field formations for 1 and 2 word instructions
 Word formats for 2 word instructions FFEE, 2 word instructions EFEE
 GGGG

| Instruction Class | Common Instruction Types in OP field | WORD FORMAT | CONDITIONS | Number of | |
|-------------------|--|--------------|---|--------------|-------------|
| | | | | Coding Lines | Mach. Words |
| CLASS I | BLANK | EEEE | EEEE = Address + Additive, and is evaluated Modulus $2^{12} - 1$ | 1 | 1 |
| CLASS II | No address Direct Indirect & Others | FFEE | EE = Address + Additive, if 00_8 Address + Additive 77_8 otherwise EE = 00_8 with Error Flag | 1 | 1 |
| CLASS III | Relative & Forward & Others | FFEE | EE = (Address + Additive) - Location, if 00_8 (Addr. + Addi) - Loc. 77_8 otherwise EE = 00_8 with Error Flag. NOTE: if the 1st char. of the LOC. field is a "Minus" sign, Instr. becomes a Class II Instr. | 1 | 1 |
| CLASS IV | Relative & Backward | FFEE | EE = Location - (Addr. + Addi), if 00_8 LOC = (Addr. + Addi) 77_8 , otherwise EE = 00_8 with Error Flag NOTE: Same as in Class III above | 1 | 1 |
| CLASS V | Specific Shifts & Others | FFFF | Address and Additive Fields unused, but if present, symbols must be legal | 1 | 1 |
| CLASS VI | BK Instr. & Others | FFFE | E = 0 - 7, E is 4th char in symbolic OP code FFF is determined by 1st 3 chars. of symbolic op code as above Addr. and Addi - field are unused | 1 | 1 |
| CLASS VII | Memory Constant Buffered I/O & Others | FFEE | EE = 00, except ATE, ATX in which EE = 05, 06, respectively GGGG = Address + Additive evaluated Modulus $2^{12} - 1$ | 1 | 2 |
| CLASS VIII | Selective Jump | FFEE | EE = X 0 where X is 3rd char in symbolic OP code GGGG = Address + Additive evaluated Modulus $2^{12} - 1$ | 1 | 2 |
| CLASS IX | INP OUT SJS | FFEE GGGG | EE = 1st line of coding address + additive evaluated as Class III, except SJS is evaluated as Class II GGGG = second line of coding evaluated as Class I | 2 | 2 |

NOTE: ALL INSTRUCTIONS MAY HAVE THE ACTUAL MACHINE LANGUAGE CODE IN THE OP FIELD. In this case the EE and GGGG portions are still evaluated as shown above.

INFORMATION SHEET

RESYNC COUNTER

Various signals entering the computer will be asynchronous with respect to computer timing. One of these signals would be from the manual switches on the computer front panel. The resync counter is used to synchronize these and other signals to the computer timing. Outputs from the resync counter and resync timing network are used throughout the computer.

For convenience in discussing this circuit, flip-flop K810/811 will be called flip-flop #1; K800/801, #2; K812/813, #3; and K802/803, #4. Flip-flops #1 & #2 collectively will be called Rank I and #3 & #4 collectively Rank II.

Inputs to the resync counter are from clock cards C012 & C007 which feed N800 and N801 respectively. N800 will have a "1" output during the even phase of the clock and N801 during the odd phase. Note that during odd time, (N801=1) whatever is contained in Rank I is "force transferred" to Rank II. The counter is actually a straight binary counter within each rank which will count 00, 01, 10, 11 and then back to 00 again.

Refer to table 1. We will assume an initial condition of the resync counter flip-flops. ALL FLIP-FLOPS SET.

From the table it can be seen that the count advances in Rank I and then this count is transferred to Rank II. The count in Rank I increases every .4 micro-seconds. .2 micro-seconds after the count has advanced in Rank I, this same

count is transferred to Rank II. As indicated by the table, the count repeats every 1.6 micro-seconds. This is the same amount of time required for one pass through the Main Timing Chain.

TABLE 1

| TIME | PHASE | K801 | K811 | K803 | K813 |
|----------------|-------|------|------|------|------|
| T ₁ | ODD | 1 | 1 | 1 | 1 |
| T ₂ | EVEN | 0 | 0 | 1 | 1 |
| T ₃ | ODD | 0 | 0 | 0 | 0 |
| T ₄ | EVEN | 1 | 0 | 0 | 0 |
| T ₅ | ODD | 1 | 0 | 1 | 0 |
| T ₆ | EVEN | 0 | 1 | 1 | 0 |
| T ₇ | ODD | 0 | 1 | 0 | 1 |
| T ₈ | EVEN | 1 | 1 | 0 | 1 |
| T ₉ | ODD | 1 | 1 | 1 | 1 |

The Resync Timing circuit (directly below the Resync counter) receives its "anded" inputs from K802, K812 and an even clock pulse from N800. H901 receives a "1" input due to the and circuit being "made" when flip-flops #3 and #4 are cleared, and the clock is on the even phase. .2 micro-second later H902 receives a "1" input and .2 micro-seconds later H903 receives a "1". Therefore, relative to the input to H901, V903 will have a "1" output .6 micro-seconds later. This will occur every 1.6 micro-seconds due to the anded inputs to H901 only being co-incident every 1.6 micro-seconds. Figure 1 shows the relationship of the Counter and Timing circuits.

Referring to the Main Timing Chain, it can be seen that all inputs which could start the timing chain are anded with an output of V903, in the resync timing circuit. This in effect, synchronizes the timing chain with the resync timing circuit. Once the timing chain has been started, it will continue to re-circulate (until recirculation is blocked) from the end of the chain (V007) back to the beginning (H000). As we already know, the time period that exists from the time a "1" input is felt at H000 until a "1" output is received from V007, is 1.6 micro-seconds. This is also the time between pulses from V903. This means then, that a one output from V903 will be co-incident with a one output from V007. As pointed out in the last topic, the period of time during which V007 has a "1" output, is called time X7. We can also say then that the output of V903 is time X7, and could be used as a timing enable. V902, V912, and V922 could then be called time X6 (.2 micro-seconds earlier), and V901, V911 as time X5. Figure 1 shows the timing relationships existing between the resync counter, Resync timing and the Main Timing Chain.

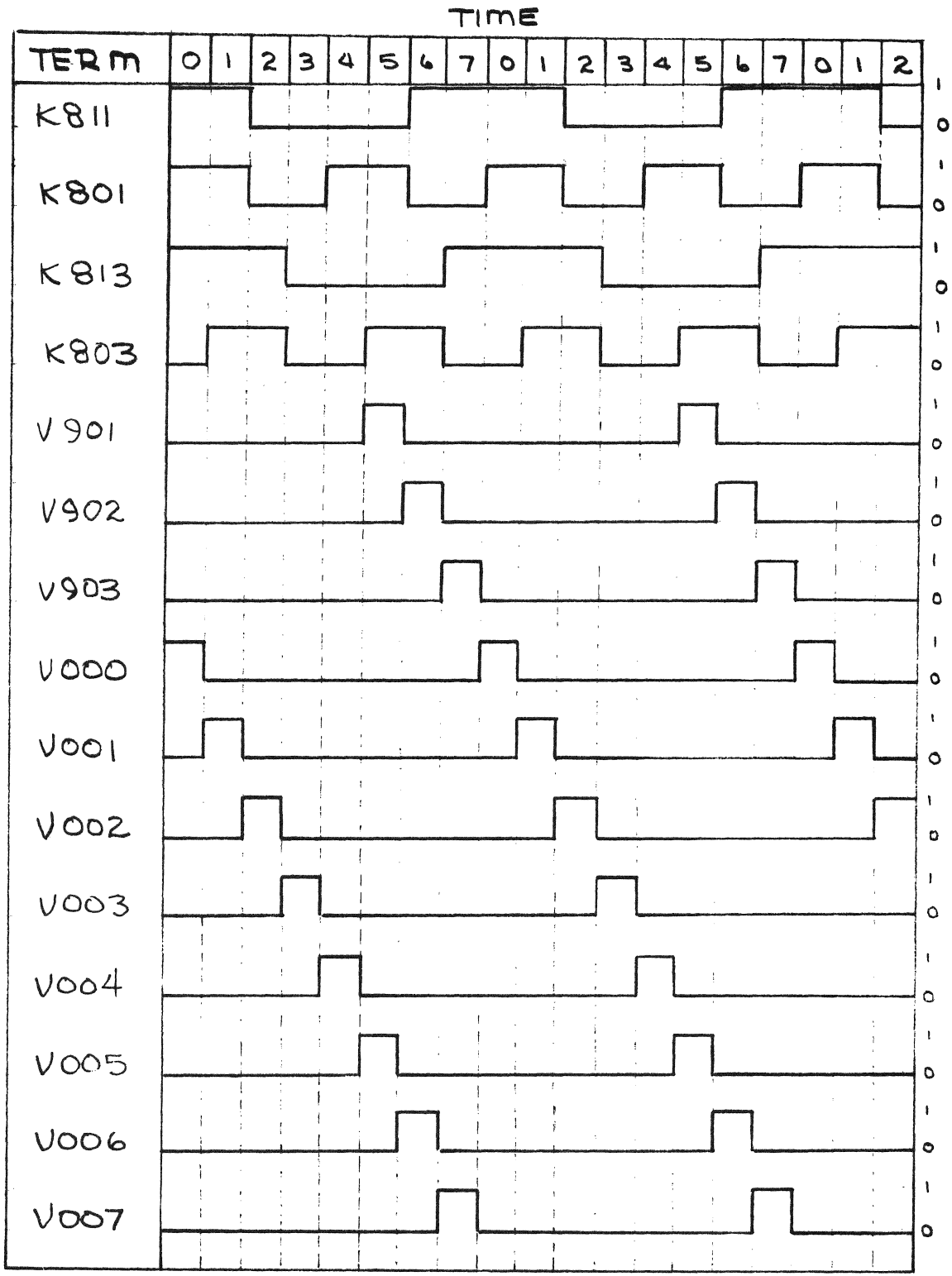


FIGURE 1

INFORMATION SHEET

STORAGE REFERENCE CYCLE

The storage reference (SR) cycle is the basic timing unit for the computer; all operations depend upon the execution of at least one such cycle. There are five such cycles (A, B, C, D, & Buffer), each used for a different operation effecting memory core storage. The fundamental purpose of the SR cycle is to select a word from memory and deliver it to the logic circuits.

Each of the five SR cycles is divided into four quarters, 1.6 micro-seconds each, to make a total time required for each cycle of 6.4 micro-seconds. Four phases occur during every cycle; read, write, inhibit, and divert. These phases overlap each other through out the 6.4 micro-second cycle as explained below.

Read Phase-

Two drive lines are energized in such a direction as to shift the selected toroid cores, making a 12 bit word, to the "0" state. The output of the cores are fed into the Z or BFR via the read amplifiers.

Write Phase-

The current is reversed in the write phase and now attempts to set each of the 12 cores to a "1" state, and do so, except where prevented by the inhibit line for that plane. Therefore, the inhibit phase must occur at the same time as the Write Phase.

Inhibit Phase-

An inhibit line is placed through each core of a plane thereby allowing the same bit of every location in memory to be inhibited at the same time. There are twelve planes, containing 4096 cores, in each memory bank and each plane has an inhibit line. The input to the inhibit lines comes from the Z or BFR register via an inhibit amplifier. If a bit is zero the inhibit line to the corresponding plane will prevent the core of the selected location from being turned to one. If the bit is a one the corresponding core will be turned to a "1" since the inhibit line for that plane will not prevent the action.

Divert Phase--

This phase is enabled from the last of the first quarter to the end of the cycle therefore encompassing both the read and write phases. It is used to make the final selection of the two proper drive lines to turn the desired 12 bits of a memory location (address), thereby accomplishing a read or write operation.

In every cycle a location in memory core is referenced and the contents of that location read by turning all twelve cores to zero. All the cycles, except in some cases the C cycle, place the contents in either the Z or BFR registers. Before the cycle is completed, the contents read out of the location or a new word will be written back into that same location to insure the contents are not destroyed.

A description of each cycle and its variations is given below.

D CYCLE - Read the Instruction

This cycle is used for the following functions.

1. Read up from memory core a twelve bit binary word and interpret to determine what operation to perform.

NOTE - ONLY in a "D" CYCLE will a 12-bit word be used as an instruction.

2. Perform some logical operation using the lower 6 bits of the 12 bit word read from memory.
3. Check for the completion of an input/output operation.

If the instruction is using only a D cycle for its execution, functions one and two will be accomplished in the 6.4 micro-second period allotted. When the instruction requires more than a D cycle for its execution, only function one will be used in the 6.4 micro-second period allotted for the D cycle. During input/output operations the D cycles used after the interpretation of the instruction will use function three only.

First Quarter - Prepare the address to reference in memory core and place in the S register, which sends it to the MCS Control circuitry.

Second Quarter - Read the instruction from MCS at the address prepared in the first quarter.

Third Quarter - Translate what function to perform using either the upper 6 bits of the word or the entire 12 bits.

If a one cycle instruction, begin to execute the instruction that was translated.

Fourth Quarter - Execute the instruction or continue to execute.

Write the instruction back into MCS at the address that was prepared in the first quarter.

A CYCLE - Read Address

This cycle is used in indirect and memory addressing mode instructions only. The twelve bit word read up from MCS is used as an address that will be reference in the next cycle for an operand.

First Quarter - Prepare the address to be referenced in MCS and place in the S register.

Second Quarter - Read up the 12 bit address from the address prepared in the first quarter.

Third Quarter - No operation performed.

Fourth Quarter - Write (Restore) the 12 bit address back into MCS at the address prepared in the first quarter.

B CYCLE - Read Operand or Address

The functions of the B cycle are to:

1. Perform an operation using the operand read up during the beginning of the cycle.
2. Read up a 12 bit word that will be used as an address in the following cycle.

The B cycle is used in all instructions requiring 12.8 micro-seconds (two cycles) or greater, except the 7600 instruction that uses a D-C cycle progression.

First Quarter - Prepare the address to be referenced for the operand or address (12 bit word) and place in the S register.

Second Quarter - Read up the 12 bit word to be used as the operand or address from MCS at the address prepared in the first quarter.

Third Quarter - 1. Execute the desired operation using the operand.
2. If the word is used as an address no operation is performed.

Fourth Quarter - 1. Restore (Write) the word back into MCS at the address prepared in the first quarter.
2. Continue execution if necessary.
3. Issue an external control signal if the cycle is being used during an input operation and stop computer operation until an input control signal is received from external equipment.

C CYCLE - Write/Restore Operand.

The C cycle is used only with instructions that modify the contents of memory and input/output instructions. The functions of the C cycle are as follows:

1. Store a new word into memory from either internal registers or external equipment.
2. Read a word from memory and send to the external equipment.

First Quarter - 1. Detect the Input Control Signal and start computer operation if in an input operation.

2. Place the word from external equipment in the A register if in a 7600 instruction.
3. Prepare the address to be referenced in MCS and place in the S register.
4. Transfer new word from internal registers to the A' register to prepare for storage in MCS.

Second Quarter - 1. Read the contents of the address prepared in the first quarter.

- a. If the instruction is a 73XX the word is allowed into the Z register.
- b. In any other instruction, except 73XX the address will be read (all cores turned to zero to clear the address for the new word) but not placed in the Z register.

Third Quarter - Issue an external control signal and send the output word if in an output instruction.

Fourth Quarter - Store the new word or restore the original word back into MCS at the address prepared in the first quarter.

BUFFER CYCLE - Buffer Channel Input/Output

The Buffer cycle is used to input a word into MCS or to output a word from MCS to or from Buffer Channel Equipment. To accomplish this, normal computer operation must be stopped for the 6.4 micro-second period required for the Buffer cycle to take place. During this 6.4 micro-second period, the word is sent to or from MCS via the Buffer Data Register (BFR).

The Buffer cycle is also used to enter the contents of the A register into certain memory locations during the 0100 instruction, but normal computer operation is stopped until all locations are changed which could take from 6.4 micro-seconds up to .26 seconds.

First Quarter - Transfer BER to the S register to prepare the location to be referenced in MCS.

Second Quarter - Read word from MCS to address prepared in first quarter.

- a. During Buffer Input this word is not allowed into the BFR, therefore the contents are cleared out of the location ready for the new word.
- b. During Buffer output, place this word into BFR so it can be sent to Buffer equipment.

Third Quarter - Add one to BER contents and place in BER.

Fourth Quarter - 1. Store new word from Buffer equipment or restore the original word, read up from MCS and send to Buffer equipment, back into MCS at the address prepared in the first quarter.

2. Check if Buffer input/output operation is completed.
3. If in an input operation and not completed, issue an external control signal requesting another word.
4. If in an output operation, issue an external control signal and send the new word to external equipment. If this is the last word, terminate the Buffer operation.
5. In any case return to normal operation.

At the end of all five cycles the proper Storage Bank Control and cycle will be set so that the next cycle can be executed immediately. (Refer to section one, page 11 for the discussion on SBC)

If each of the 130 instructions in the computer repertoire is interpreted in terms of the different SR cycles necessary to complete the execution of that instruction, it is found that all instructions may be accounted for by arranging the five cycles in the patterns below, called Storage Reference Sequences or Progressions.

(Refer to Section 1, pages 15-19 for Addressing Mode discussion)

- | | |
|---------|---|
| D-D | No address mode instructions |
| D-B-D | Direct, Relative (Forward and Backward), Constant, and Specific Addressing modes for non-memory modifying instructions. |
| D-C | Relative Addressing mode (Used only with instruction 7600). |
| D-A-B | Indirect and Memory Addressing modes for non-memory modifying instructions. |
| D-B-C | Direct, Relative (Forward and Backward), Constant, and Specific Addressing mode instructions that are memory modifying. |
| D-A-B-C | Indirect and Memory Addressing mode instructions that are memory modifying. |
| D-BUF | Instructions 0100, 7200, and 7300 where a Buffer Channel operation will take place (Buffer not busy) |

Listed below are samples of the five Storage Reference cycles and comments on each step. Refer to Figure 1-1 for Block Diagram.

SAMPLE D CYCLE - 6.4 micro-seconds

STEP D 1 P & +1---ADDER

Used in the first part of every D cycle (except second D cycle in Input/Output instructions) to update the address where the new instruction will be found, unless the preceding instruction executed was one that requires a jump of more than one location for the next instruction.

1a P & +2---ADDER

Used to update the address (where the new instruction will be found) in the first part of a D cycle following an instruction that checks for a buffer busy-not busy condition. This is done to jump the next sequential location when the buffer is not busy. Also used after all memory and constant addressing instructions to obtain the next instruction.

1b P & \pm Z_L---ADDER

Used to update the address, where the new instruction will be found, by adding or subtracting the E portion of the previous content of the Z register (instruction or operand) to or from the contents of the P register. This is done in a D cycle following an instruction that requires a jump forward or backward, not more than 77₈ location, for the next instruction.

1c +Z---ADDER

Used to update the address, where the new instruction will be found, in the first part of the D cycle following an instruction that checks for a jump-no jump condition. This is done to jump to the new address, anywhere in the same bank, for the new instruction.

2 A'---S---P & MCS Cont.

The contents of the Adder, obtained in either steps 1, 1a, 1b, or 1c, are placed in the S register since this register will always contain the address that is being read in MCS for the operand or instruction. In each D cycle the address is placed in P to update it and in every cycle placed in MCS Cont. to cause the desired drive lines to be activated to read the contents of the address prepared in either steps 1, 1a, 1b, or 1c.

3 CLEAR Z & F

Prepares these registers for the instruction.

4 MCS---Z---F---

Reference the address prepared and placed in the S register and read the contents of this location from MCS into Z, then into F. Only in the D cycle will the contents read from MCS be placed in F and used as an instruction. Note that the location read, at this moment, is all zeros and in order to restore the instruction a write operation at the end of the

cycle will be used (Z---MCS). The F register and translators interpret the instruction and send the correct enables to Main Control and Storage Bank Control to indicate the desired instruction.

5 Main Cont. & SBC

Main Control makes the necessary enables to allow the proper transfers to execute the instruction. The SBC receives enables from the F register and translator so that it can set the required SBC for each cycle of the instruction and change the Bank number for a specific SBC, if desired.

6 This step is used when an instruction requires only a D cycle for its execution. A check could be made for the following; is Buffer Control (Channel) busy, are the contents of the A register negative, positive, zero, or not zero, are the Jump or Stop switches properly set for a jump or stop condition, is the instruction an error or halt instruction, or is the instruction a no-operation instruction. If the operation is not a check procedure then some arithmetic operation could take place by taking the E! portion of the instruction and combining it with the contents of the A register.

Should the D cycle be a two cycle instruction or greater, this step will be passed through without any operation occurring.

7 Z---MCS

The instruction read from MCS in Step 4 is stored back into the same address it was read from in order to preserve the instruction. This step will occur in every cycle executed.

8 Go to next cycle & set SBC

The next cycle required (A, B, C, or D) and the SBC that will be used during the cycle, are set at the end of each cycle to prepare for the next cycle that will be progressed into immediately, if a stop condition has not been sensed.

SAMPLE A CYCLE - 6.4 micro-seconds

A1 P & +1---ADDER

Prepare the address that will be read for the address of the operand, Used in all memory addressing mode instructions.

1a +Z_L---ADDER

Used in the Indirect Addressing mode instructions to prepare the address that will read for the address of the operand.

2 A'---S---MCS Cont.

The output of the pyramid, obtained in either steps 1 or 1a, is passed through the A' register in the S register. The S register output is sent to MCS Cont. to reference the prepared address for its contents.

3 (Clear Z) MCS---Z

Read the contents of the address prepared in either 1 or 1A and place in the Z register, after that register has been cleared of its previous contents. This address now contains all zeros.

4 Z---MCS

Write the contents (operand address) of Z back into the address specified by the S register. (Z retains the number) this prevents the destruction of the address from MCS.

5 Go to B cycle & Set SBC to IND

All A cycles prepare an address to be used in the next B cycle which refers to the Indirect Bank for the operand.

SAMPLE B CYCLE - 6.4 micro-seconds

B1 +Z_L---ADDER

Used in all Direct Addressing mode instructions to prepare the address (contained in the Z register from the A cycle) where the operand will be found. This address will be one of the first 100₈ locations; therefore the lower six bits of Z are used.

1a P & \pm Z_L---ADDER

Used in all Relative Forward and Backward Addressing mode instructions to prepare the address where the operand will be found. The E portion of Z is added or subtracted to the contents of the P register.

1b P & +1---ADDER

Used in all constant addressing mode instructions to prepare the address where the operand will be found.

1c +Z---ADDER

Used in memory and Indirect Addressing mode instructions to prepare the address where the operand will be obtained. Since the A cycle preceding this B cycle read up the address and placed it in Z, the contents of Z needs only to be placed in the S register via the Adder and the A' register.

1d 0000---ADDER

Block all the inputs to the Adder, which effectively places all zeros (0000) in its input. This is then complemented, so that in the pyramid output there will be a 7777g. The 7777 is the address that will be placed in S and referred to for the operand. This procedure for preparing the address will be used in all Specific Addressing mode instructions.

2 A'---S---MCS Cont.

The output of the Adder is placed into S via the A' register so that this address (the one prepared in either 1, 1A, 1B, 1C, and 1D) can be referred to for the operand.

3 (Clear Z) MCS---Z---

Read the address specified by the S register and place its contents into the Z register, after clearing the previous contents from Z. The address read now contains all zeros.

4 +Z---ADDER---A'---A
(Could be \pm Z & A---Pyr.)

The contents of Z are sent to the A register via the Adder and A' register. (No operation is performed on the number as it passes through the Adder). This is a load type operation where the operand is to be loaded from MCS into the A register. Other operations (Arithmetic, etc.) could be performed during this time. If, however, the instructions being executed was a memory modifying type, this time could be passed through with no operation occurring.

5 Z---MCS

Write the operand back into MCS at the address specified by S and that was prepared in either step 1 or 1a. The Z register retains the operand until cleared in the next D cycle.

6 Go to D cycle & set SBC
to REL

In all instructions, except the memory modifying type, the B cycle is the last cycle of the instruction, therefore, a new instruction will be read using a D cycle referring to the Relative Bank.

6a Go to C cycle & keep SBC
set to the same bank

In memory modifying type instructions (40-57XX), a C cycle is required to store the contents into MCS and will reference the same bank, either Indirect or Bank 0, used in the B cycle. The B cycle is used to read the original operand and prepare the address where the new content (operand) will be stored; therefore, the same address must be used in the C cycle.

SAMPLE C CYCLE - 6.4 micro-seconds

C1 S---MCS Cont.

The C cycle will reference the same address used in the B cycle and since the S register contents hasn't changed, it is again sent to MCS Cont. to operate the same drive lines as in the B cycle.

2 (Block MCS---Z) Read MCS

Read the address prepared in step 1, but do not place in the Z register. Reading the address turns all its 12 cores to zero and by not placing its contents in Z the contents are destroyed. This step is used in all memory modifying instructions.

3 A---ADDER---A' (Clear Z)---

Used in a Memory Modifying instruction to store the new word, prepared in the A register,

in the address cleared in step 2. The Z register is cleared of its original contents to prepare it for the new word.

4 Z---MCS

Transfer the contents of A' to MCS via the Z register into the address cleared in step 2.

5 Go to D cycle & Set SBC to REL

The C cycle is the last cycle of all memory modifying instructions (except the Input/Output Instructions); therefore a D cycle is executed next in order to read the next instruction. All instructions, therefore all D cycles, reference the Relative Bank for the 12 bit binary word.

SAMPLE BUFFER CYCLE - 6.4 u Sec.

BUF 1 Main Cont.---Buf Cont.

Initiate the Buffer Cycle operation and block all normal transfers (Normal channel operation has previously been stopped). The normal transfers are blocked so that the A, P, F, & Z register contents will not be changed, since the normal operation can be stopped at the end of either a D, A, or C cycle. If the contents of these registers were changed, normal operation could not return to the point where it was stopped.

2 BER---ADDER---A'---S---
MCS Cont.

Used to prepare the address that will reference

- 2 BER---ADDER---A'---S---
MCS Cont. Used to prepare the address that will reference the Buffer Bank where a new word will be stored. (Using the 0100 instruction as an example). SBC was set to Buffer Bank just prior to initiating the Buffer cycle.
- 3 (Block MCS---BFR) Read MCS Read the address in MCS (Buffer Bank) prepared in step 2 but do not place in BFR. This action turns all 12 cores of the address to zeros and by not placing the contents in the BFR, the contents are destroyed. The BFR is then cleared of its original contents in readiness for the new word.
- 4 BER & +1---ADDER---A'---BER Used to update the BER address so that in the next Buffer cycle the new word can be stored in the next sequential location from where the present word is being stored.
- 5 BFR---MCS The word in the BFR (placed there in the previous D cycle) is stored in MCS at the address prepared in Step 2 and cleared in Step 3.
- 6 BER & BXR---Compare---
BUF Cont. The BER (containing the address where the word will be stored) and the BXR (containing the address +1 where the last word will be stored)

are compared after each word is stored to check if the operation is completed. If the result is zero, it is complete, therefore the Buffer cycle will be terminated and normal operation started where it was stopped. If the result of the comparison wasn't zero, the operation must continue, therefore the operation will return to Step 1 and another Buffer cycle will be executed.

| D cycle | |
|--------------|----------------------------------|
| 1st. Qtr. | $P + 1 \rightarrow S$ |
| 2nd. Qtr. | Read $S \rightarrow P$ |
| 3rd Qtr. | $Z_u \rightarrow F$ Translate |
| 4th Qtr. | Execute IF "n" Adr. |

Read Next Instruction

| A cycle | |
|--------------|----------------|
| 1st. Qtr. | $Z_1 - S$ |
| 2nd. Qtr. | Read Load Z |
| 3rd Qtr. | |
| 4th Qtr. | |

Read Address

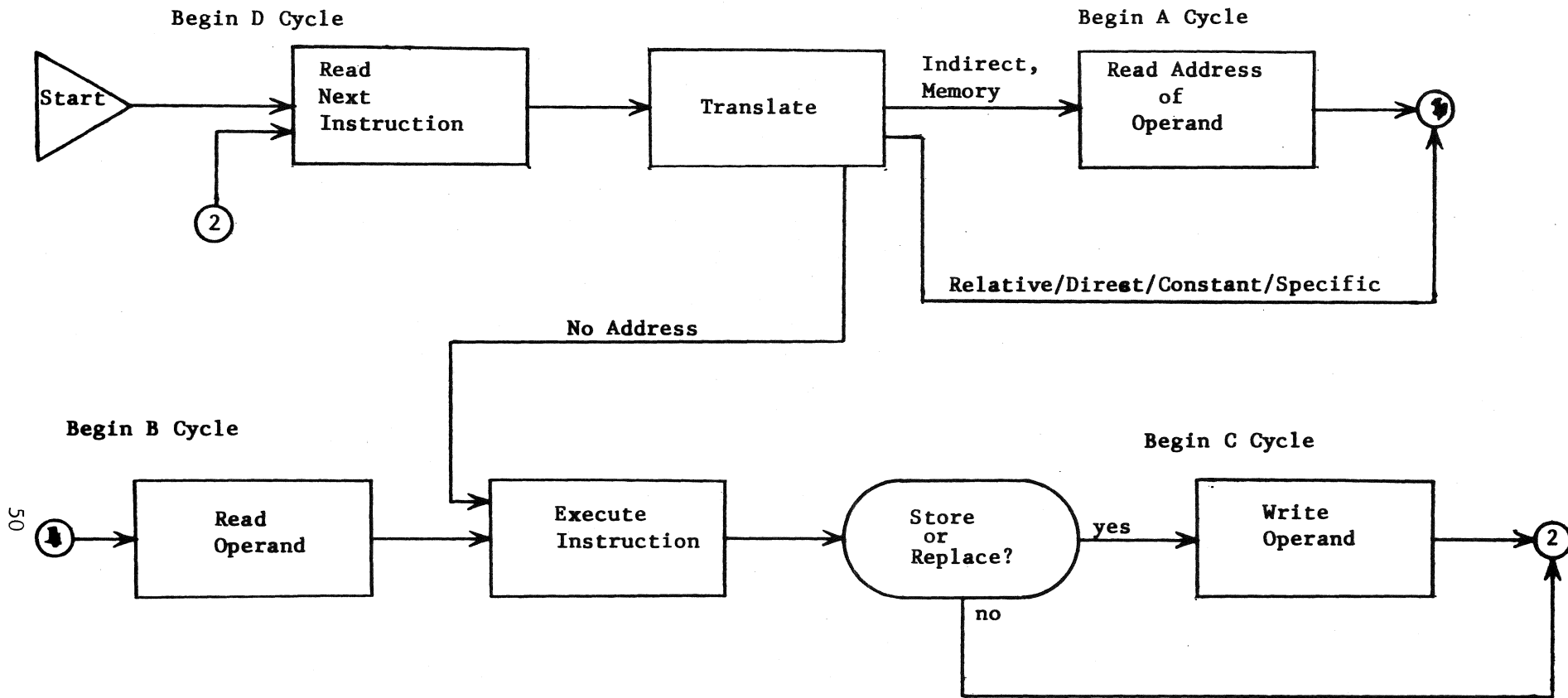
| B cycle | |
|--------------|--|
| 1st. Qtr. | Z $P + Z - S$ $P - Z_1$ Z_1 |
| 2nd. Qtr. | |
| 3rd Qtr. | |
| 4th Qtr. | |

Read Operand

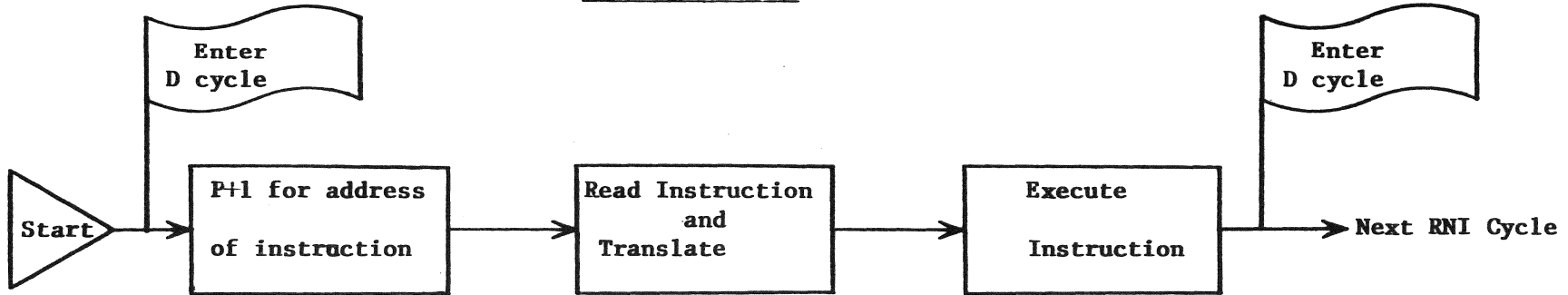
| C cycle | |
|--------------|------------------------------|
| 1st. Qtr. | |
| 2nd. Qtr. | Block Read to Z, En A - Z |
| 3rd. Qtr. | |
| 4th. Qtr. | |

Write

INSTRUCTION CYCLE PROGRESSION

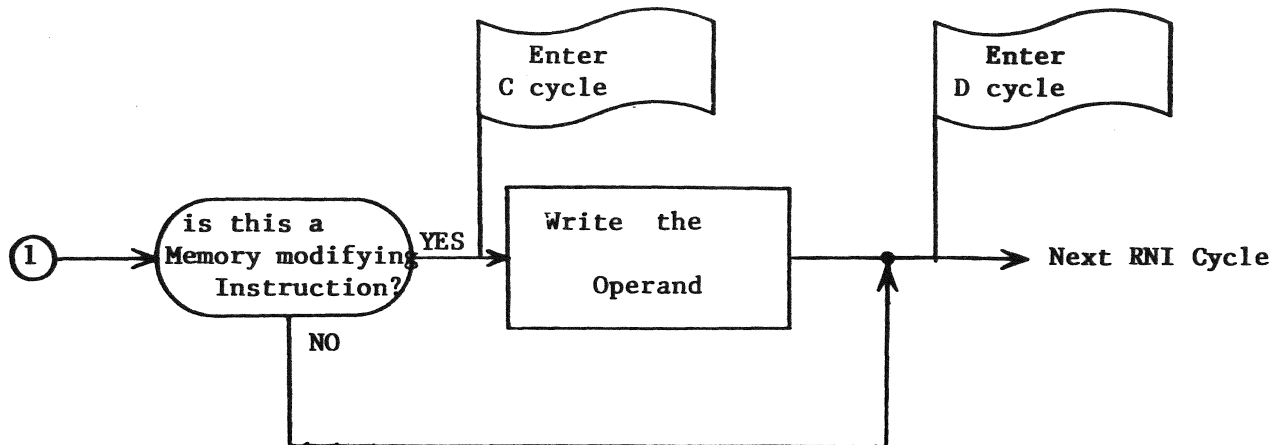
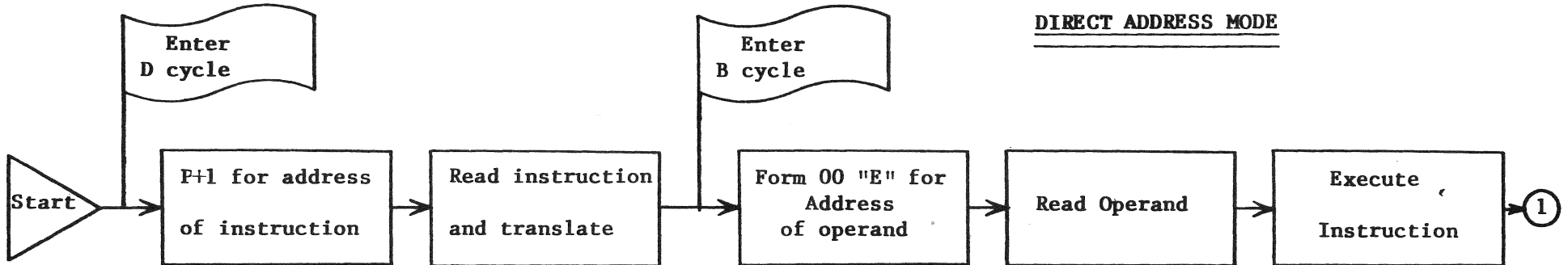


NO ADDRESS MODE

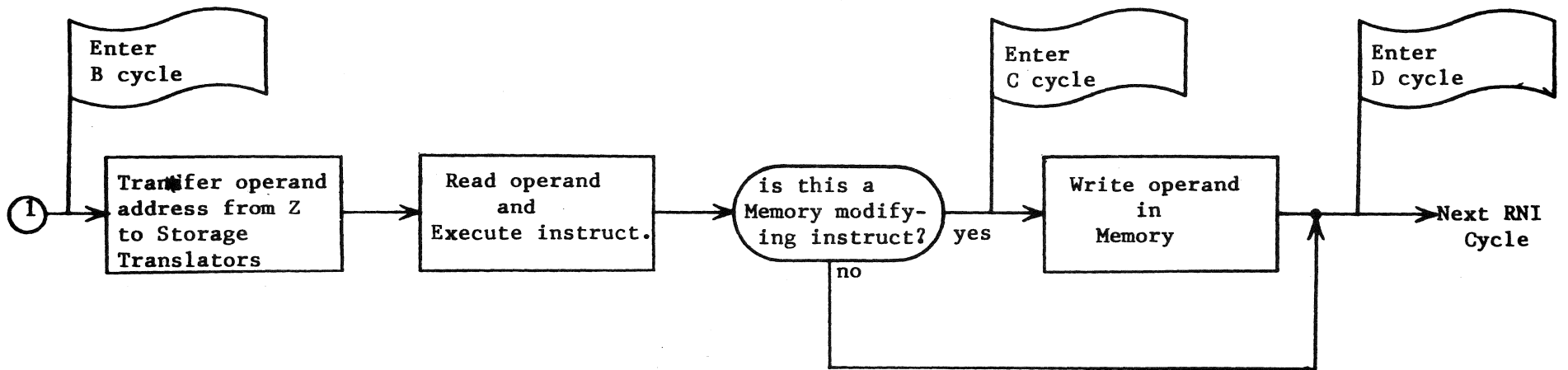
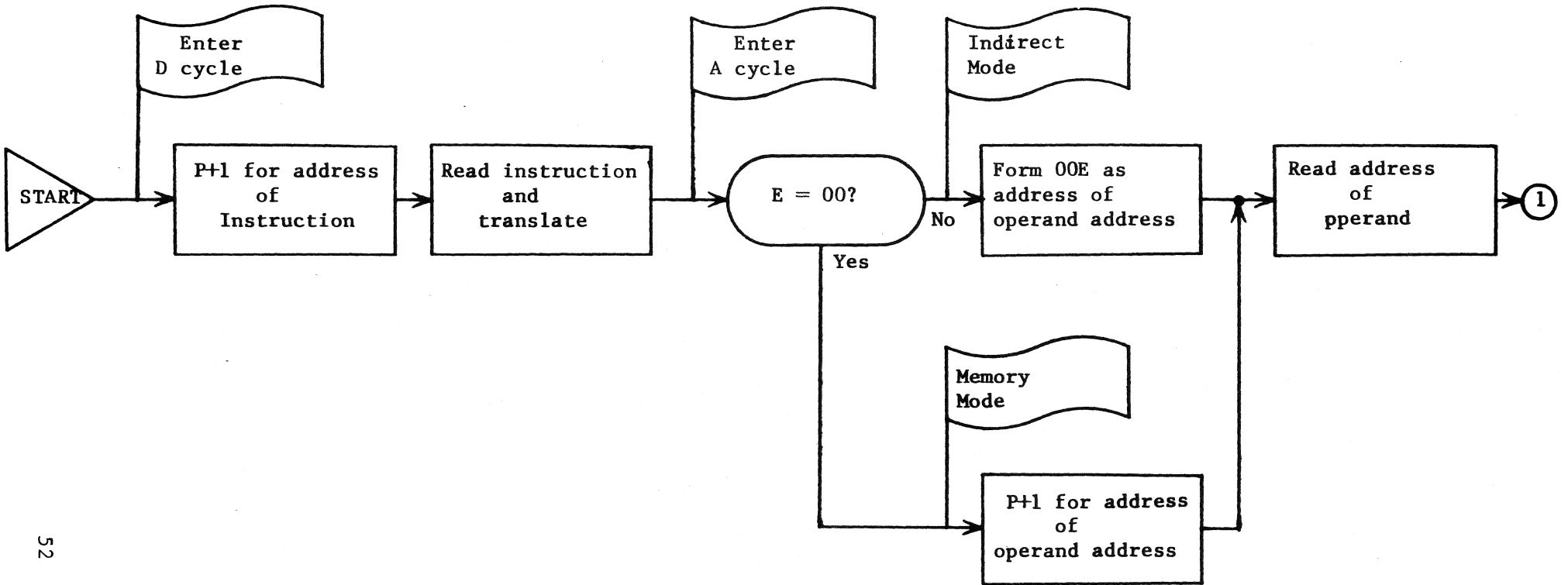


DIRECT ADDRESS MODE

51

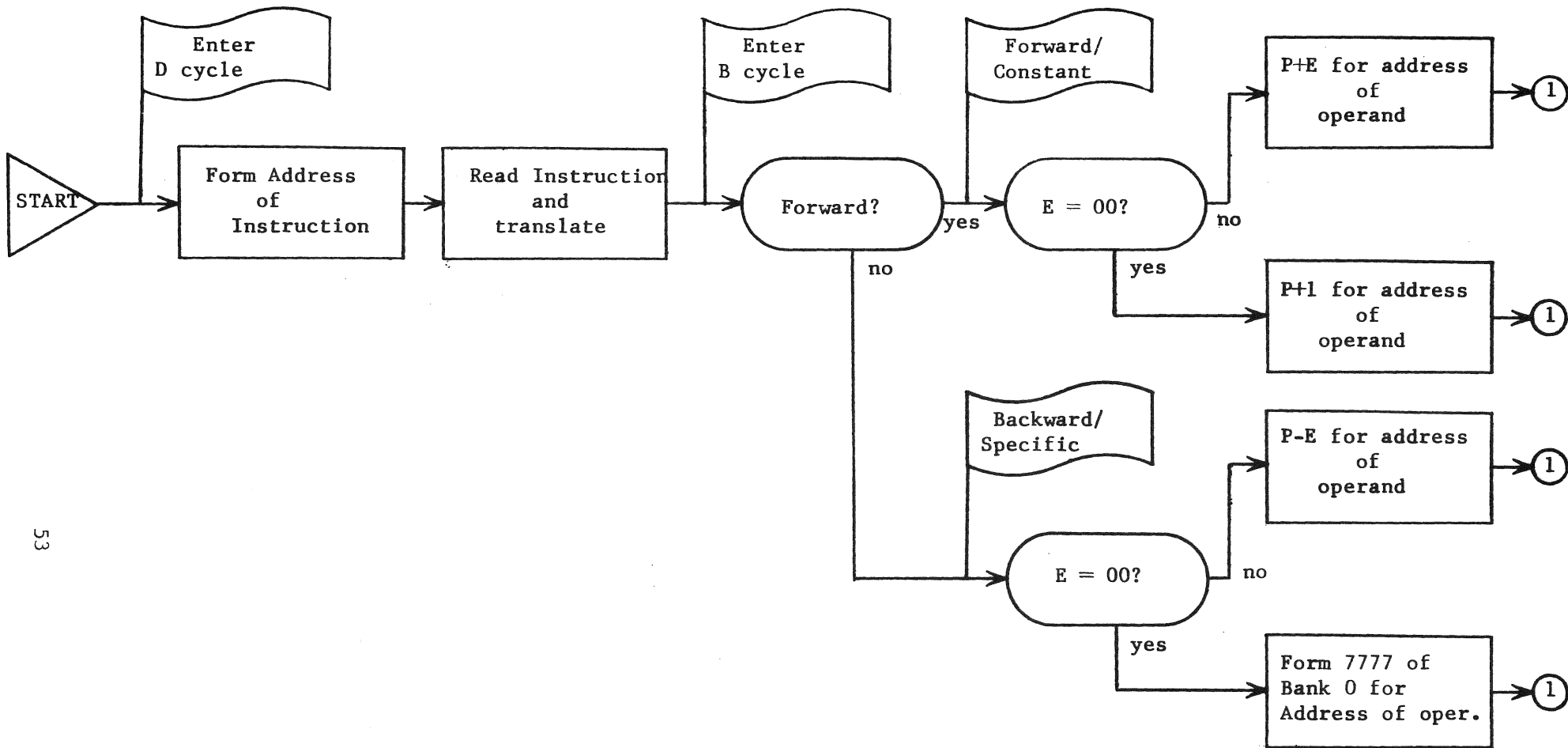


INDIRECT AND MEMORY ADDRESSING MODES

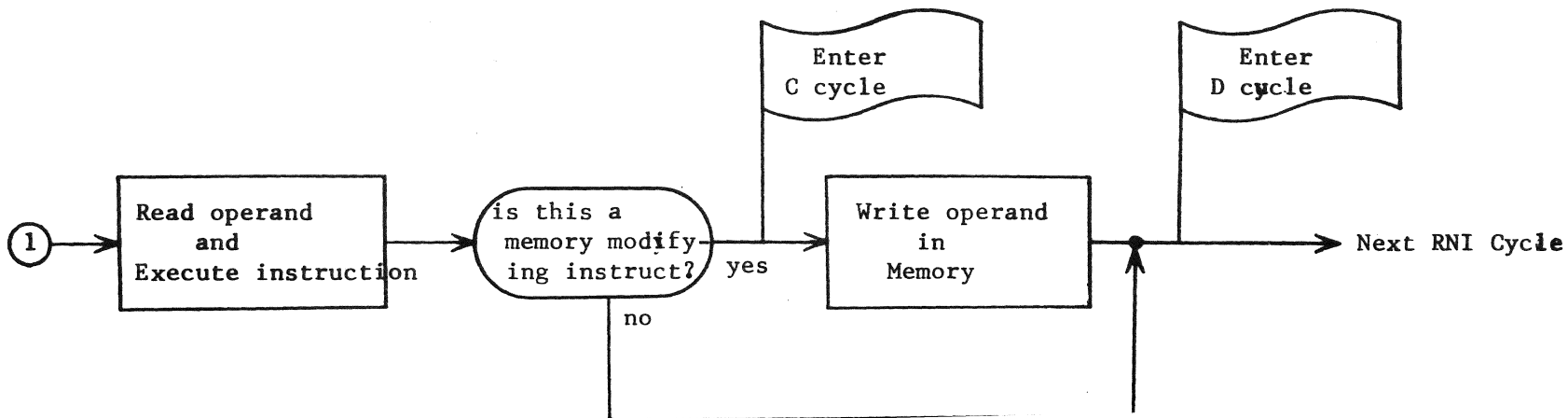


RELATIVE ADDRESSING MODES

FORWARD/CONSTANT//BACKWARD/SPECIFIC



53



INFORMATION SHEET

REGISTERS AND INVERTERS

Z REGISTER

The Z (Main Communications or Storage Restoration) register is a 12-bit unit and performs many functions. All data read from or written into memory core storage to be used in the normal channel or for internal operation must pass through the Z register. It restores the word read from MCS back into the same address and holds one of the operands in arithmetic operations. All data sent to or received from external equipment via the Normal Input-Output channels passes through Z.

There are three inputs to the Z register: the console Z manual enter buttons, the output from the 350 paper tape reader, and the output of a rank of inverters. The inverters receive inputs from MCS, Normal Channel Input from external equipment, and the A' register each of which are 12-bits.

Z is cleared every cycle in the last part of the second quarter in readiness for the input from MCS during the read phase or from A' during a store instruction. For an input instruction (normal channel), Z is cleared during the last part of the fourth quarter in a B cycle to clear out the previous contents in readiness for the input word in the following C cycle and in the last part of the first quarter in the C cycle to prepare for the new word waiting on the input lines.

The output from Z goes to the Borrow Pyramid, Normal Channel Output, BXR-Z-BER display on the console, "0" Register, and output rank of Inverters. All of these outputs are 12-bits except to the "0" Register which is the 8 least significant bits. The output inverters transfer data to the F register (upper 6-bits), inhibit drivers (MCS), and Buffer output lines.

A' REGISTER

The A' register is used to receive and pass on the results of all operations involving the pyramid. It is a 12-bit register and is called the Auxiliary Arithmetic or Transmission Register. This register is actually used as a part of the arithmetic operation performed in the pyramid.

The output from the pyramid is the only input to A' and is a 12-bit word. A' is cleared after 0.4 microseconds in each quarter of every cycle (D, A, B, & C), therefore, the data placed in it will be transferred in that same quarter or destroyed.

The output of A' goes to the A, S, Z (via the I^{-1} inverters), and BER Registers. These transfers can be accomplished only when enabled by main control. A' to S is enabled in the last part of the first quarter in D, A, and B cycles to store the reference address in S before the read phase in memory. The A' to A register transfer is used during an arithmetic or load operation to place the result or operand in A. An A' to Z transfer is allowed during a store, replace, or output from A instruction when the contents of A are to be sent either to MCS or to external equipment via Z.

A REGISTER

The A register is the principle arithmetic register called the Accumulator. This 12 bit register is used to hold the result of most arithmetic operations and in these operations actually holds one of the operands used in the pyramid. A also holds the word read from MCS during a load type instruction or brought in from external equipment during an input to A instruction.

There are three inputs to the A register from the A' and SBC registers. Each flip-flop (FF) in A receives an input from its corresponding FF in A'. The input from SBC is transferred as follows: Buffer Bank register (3 bits) MSB to bit 11 and LSB to bit 09 of A, Direct Bank Register (3 bits) MSB to bit 08 and LSB to bit 06 of A, Indirect Bank Register (3 bits) MSB to bit 05 and LSB to bit 03 of A, Relative Bank Register (3 bits) MSB to bit 02 and LSB to bit 00 of A. Also there is an input from the console "A" manual enter buttons.

A register outputs are to the P and BXR registers and the pyramid. A to P is enabled only during the bank change instructions when the Relative bank is the one to be changed. A to BXR is used to store a new LWA+1 in the BXR. A to the pyramid is allowed during the store, replace, arithmetic, and input-output instructions. An output from A is also sent to main control where it is used to determine the condition of A for conditional jump instructions. The A'-A-BFR display on the console also receives an output from the A register.

S REGISTER

The Storage Address Register is a 12-bit unit and holds the address that is read for an operand, another address, or an instruction. It is through this register that the program address register is updated at the beginning of each instruction.

The S register receives only an input from the A' register. The output from S goes to the P register and to MCS Control. MCS Control uses the 12-bit address to select the proper drive lines to read the contents of the desired memory location. A third output is transferred to the F-P-S display on the console. The S register is a forced transfer register therefore is never cleared. Each new input changes the desired flip-flops.

P REGISTER

The 12-bit Program Address Register holds the address of the instruction that is currently being executed. It retains this address throughout the execution of the instruction which could be from 6.4 to 25.6 microseconds in duration. There are a few instructions in which P does not contain the address of the instruction but these are mainly the normal channel input-output instructions.

The three inputs to P are from the S and A register and the console P manual enter buttons. The S to P transfer is enabled in the second quarter of D, at all times, and during the second quarter of B and fourth quarter of C cycles in a 7100 (forced or not) instruction.

The outputs from P are to the Pyramid and F-P-S display on the console. The P to Pyramid transfer is allowed when it is desired to place a P, P+1, P+2, $P+Z_L(E)$, or $P-Z_L(E)$ in the S register to reference the address that is the result of the operation in the pyramid.

O REGISTER

The O register or PSR is called the punch storage register since it holds the 8 bit word that is to be punched out on paper tape by the BRPE-11 punch.

The register is an 8-bit unit that receives its input from the least significant 8-bits of Z. After each word is punched by the BRPE-11, a control signal from the punch logic will clear the 9 register to prepare it for the next word from Z. As soon as the word is transferred to the 0 register the computer is started again so that operation may resume while the slower operating punch is punching out the contents of the 0 register. If a 74XX or 7677 was used as the output instruction, new instructions could be read and executed while waiting for the word to be punched. If a 73XX was used, however, starting the computer would only read up another word, place it on the output lines from Z, and stop computer operation until the previous word was punched and the 0 register was cleared, ready for the waiting word.

INFORMATION SHEET

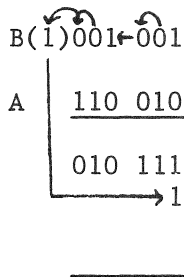
ARITHMETIC SECTION

ARITHMETIC OPERATIONS

Arithmetic in the computer is accomplished through a subtractive accumulator. That is, it adds A to B by subtracting the complement of A from B. For example, adding the binary numbers

B 001001 (9 decimal, 11 octal)
 A 001101 (13 decimal, 15 octal)
 010110 (22 decimal, 26 octal)

is actually accomplished by subtracting as shown below.



The long arrow indicates that an "end-around borrow" must be made from the first stage, since the last stage was unable to satisfy the borrow made when 1 was subtracted from 0. The final total, then is

010110 (22₁₀) or (26₈)

In the above example, the curved arrows indicate that the borrows must be made from the next higher-order stage. The arrow between the two octal groups represents the existence of a "group borrow", where the last stage in a group cannot satisfy a borrow made upon it (or within that stage itself), and so must propagate the borrow to the next higher stage.

Subtraction is performed by subtracting the true value of the subtrahand (not its complement) from the minuend, as shown below.

| | | |
|------------|---------------|------------------------|
| Minuend | 010110 | (22 decimal, 26 octal) |
| Subtrahend | <u>001010</u> | (10 decimal, 12 octal) |
| | 001100 | (12 decimal, 14 octal) |

The curved line again indicates a "stage borrow".

ADDER

The Adder is a device used to generate and recognize the borrow signals. These are pyramided from each stage level to an octal group level and back to the state level so the final stage of each bit in the answer can be decided. In accomplishing the above, the Adder performs basic functions of arithmetic, transfer, and shifting.

The Adder consists of two ranks of inverters (12 stages each) as inputs circuitry, eight-level inverter ranks (exclusive "OR, AND, and OR functions) to determine borrow or no borrow conditions, and "AND" gate inputs to the A' register stages to Toggle (subtract without borrows) the partial result into A'. The Probe operation changes the appropriate bits of A' to reflect the final result considering the bits of the minuend that borrows have passed into or through. Three main pulses occur four times in each cycle (6.4 micro-second period), once each quarter (1.6 microsecond period), thereby allowing four arithmetic, transfer, or shifting operations in each cycle. These pulses are; Clear (clears the A' register), Toggle (subtracts without borrows and places partial result in A'), and Probe (corrects each bit in A' required to obtain the final result).

The inputs to the Adder are from the Z, A, P, and BER registers and hardware +1, +2, and +10, 20, 30, or 40. The above inputs are connected to the inverter rank

input circuitry from the appropriate clear or set outputs of the listed registers and are all gated inputs. The gate enables are received from Main Control and are determined by the 12-bit instruction word translated in the F &/or F' Translators.

The Z register supplies three inputs, two +Z and one -Z, that are used for add, subtract, logical product, selective complement, replace, transfer, and shift replace type instructions. In actual circuitry the -Z input is sent through the Z output inverter ranks before entering the Adder inverter ranks.

The P register input is connected to only one Adder inverter rank and is used only for program address updating or transfers.

There are four inputs from the A register to the Adder inverter ranks. These are used for add, subtract, multiply, logical product, selective complement, replace add, store, shifting, and transfer type instruction. The shifting operation is accomplished by hardware transfer, ie, for a shift left one, bit 11 and 00 of A are connected to bit 00 and 01 of the Adder input, respectively.

Only one input of the BER is sent to the Adder and is used for program address updating, during a buffer cycle or BER transfer to other registers.

The +1, +2, and +10, 20, 30, 40 are all hardwire inputs that are enabled from Main Control during the appropriate part of a cycle. The +1 is used for updating the program address and the replace add one instruction. For the memory and constant addressing mode instructions and certain jump-no jump instructions the +2 is used to go two memory locations forward for the next instruction.

The +10, 20, 30, or 40 is used during the interrupt sequence to place the address where main program address will be stored, while the interrupt sub-routine is being executed.

The output from the Adder is into the A' register, the latter being a part of the Adder operation. The results of all transfer, shift, arithmetic, and logical operations are placed in A' before being transferred to the appropriate register. One thing should be noted at this point, since A' is cleared every quarter of a cycle and most operations through the Adder occur in the third quarter, A' display will not always indicate the results of the supposed operation.

ADDER OPERATIONS

The arithmetic, logical, shift, and transfer operations are effectively the following actions. The (a) column indicates a normal octal arithmetic, (b) indicates octal arithmetic which is effective computer operation, (c) actual computer arithmetic, and (d) decimal arithmetic of the same numbers.

1) Arithmetic

- a) Addition - Effectively, addition is accomplished by subtracting the complement of the addend from the augend.

To add octal numbers, since the average individual thinks in decimal, add two digits (addend to augend) at a time (decimally), and if the result is greater than seven, add two to the result. The carry produced by this operation is added to the digits to the left, and if no digits are present do an end-around addition of the carry and the right most digit of the answer. This is done to reflect the computer operation of end-around borrow.

Example:

$$\begin{array}{r} 5432 \\ 2436 \\ \hline 0070 \\ \hline \end{array}$$

1

Mentally add two to the first, third, and fourth digit additions.

0071 Correct Answer

1. A, P, Z, or BER & +1---Adder---A'

| | | | | | | | | | | | | | | | |
|------|-------------------------|---|-----|-------------------------|---|-----|------------|---|------------|---|------------|---|------------|---|-----------------------|
| BER- | 1234 ₈ | - | (1) | 1234 ₈ | - | (1) | 001 | ← | 010 | ← | 011 | ← | 100 | - | 668 ₁₀ |
| +1- | <u>0001₈</u> | - | | <u>7776₈</u> | - | | <u>111</u> | | <u>111</u> | | <u>111</u> | | <u>110</u> | - | <u>1₁₀</u> |
| | | | | 1236 | | | 001 | | 010 | | 011 | | 110 | - | 669 ₁₀ |
| | | | | <u>1</u> | | | | | | | | | <u>1</u> | | |
| | | | | 1235 ₈ | | | 001 | | 010 | | 011 | | 101 | | |

2. P & +2---Adder---A'

| | | | | | | | | | | | | | | | |
|----|-------------------------|---|-----|-------------------------|---|-----|------------|---|------------|---|------------|---|------------|---|--------------------------|
| P- | 6543 ₈ | - | (1) | 6543 ₈ | - | (1) | 110 | ← | 101 | ← | 100 | ← | 011 | - | 3427 ₁₀ |
| | <u>0002₈</u> | - | | <u>7775₈</u> | - | | <u>111</u> | | <u>111</u> | | <u>111</u> | | <u>101</u> | - | <u>0002₁₀</u> |
| | 6545 ₈ | | | 6546 | | | 110 | | 101 | | 100 | | 110 | - | 3429 ₁₀ |
| | | | | <u>1</u> | | | | | | | | | <u>1</u> | | |
| | | | | 6545 ₈ | | | 110 | | 101 | | 100 | | 101 | | |

3. A or P & Z_L---Adder---A'

| | | | | | | | | | | | | | | | |
|-------------------|-------------------------|---|-----|-------------------------|---|-----|------------|---|------------|---|------------|---|------------|---|--------------------|
| P - | 3501 ₈ | - | (1) | 3501 ₈ | - | (1) | 001 | ← | 101 | ← | 000 | ← | 001 | - | 1857 ₁₀ |
| +Z _L - | <u>0077₈</u> | - | | <u>7700₈</u> | - | | <u>111</u> | | <u>111</u> | | <u>000</u> | | <u>000</u> | - | <u>63</u> |
| | 3600 ₈ | | | 3601 | | | 011 | | 110 | | 000 | | 001 | - | 1920 ₁₀ |
| | | | | <u>1</u> | | | | | | | | | <u>1</u> | | |
| | | | | 3601 ₈ | | | 011 | | 110 | | 000 | | 000 | | |

4. A & +Z---Adder---A'

| | | | | | | | | | | | | | | | |
|------|-------------------------|---|-----|-------------------|---|-----|------------|---|------------|---|------------|---|------------|---|--------------------------|
| A - | 4356 ₈ | - | (1) | 4356 ₈ | - | (1) | 100 | ← | 011 | ← | 101 | ← | 110 | - | 2286 ₁₀ |
| +Z - | <u>2476₈</u> | - | | <u>5301</u> | - | | <u>101</u> | | <u>011</u> | | <u>000</u> | | <u>001</u> | - | <u>1342₁₀</u> |
| | 7054 ₈ | | | 7055 | | | 111 | | 000 | | 101 | | 101 | - | 3628 ₁₀ |
| | | | | <u>1</u> | | | | | | | | | <u>1</u> | | |
| | | | | 7054 | | | 111 | | 000 | | 101 | | 100 | | |

b) Subtraction - Subtraction is accomplished by subtracting the subtrahend from the minuend as in normal subtraction, but any borrow created in the most

significant digit is accomplished by borrowing from the least significant digit or "end-around borrow".

To subtract using octal numbers again subtract using decimal numbers. If the digit in the minuend, due to the borrow, is greater than seven, subtract two from the result of the subtraction. Should a borrow be necessary in the most significant digit, borrow from the least significant digit or end-around.

Example:
$$\begin{array}{r} (1)1634_8 \\ \underline{6543_8} \\ 3071 \\ \underline{1} \\ 3070_8 \end{array}$$
 Mentally subtract two from the result of the second and fourth digits. Correct answer

1. P or A & -Z---Adder---A'

A -
$$\begin{array}{r} 7142_8 \\ \underline{1234_8} \\ 5706_8 \end{array}$$

$$\begin{array}{r} \overbrace{714} \overbrace{2} \\ \underline{1234} \\ 5706_8 \end{array}$$

$$\begin{array}{r} 111 \leftarrow 001 \\ \underline{001 \quad 010} \\ 101 \quad 111 \end{array}$$

$$\begin{array}{r} \overbrace{100} \leftarrow 010 \\ \underline{011 \quad 100} \\ 000 \quad 110 \end{array}$$

$$\begin{array}{r} 3682_{10} \\ \underline{668_{10}} \\ 3014_{10} \end{array}$$

2. P or A & -Z_L---Adder

P -
$$\begin{array}{r} 4000_8 \\ \underline{0077_8} \\ 3701_8 \end{array}$$

$$\begin{array}{r} \overbrace{4000} \\ \underline{0077} \\ 3701_8 \end{array}$$

$$\begin{array}{r} \overbrace{100} \quad \overbrace{000} \\ \underline{000 \quad 000} \\ 011 \quad 111 \end{array}$$

$$\begin{array}{r} \overbrace{000} \quad \overbrace{000} \\ \underline{111 \quad 111} \\ 000 \quad 001 \end{array}$$

$$\begin{array}{r} 2048_{10} \\ \underline{0063_{10}} \\ 1985_{10} \end{array}$$

2) Transfer - In order to load a number from MCS into the A register it is necessary to pass the number through Z, Adder, A', and into A. The number, therefore, must be passed (transferred) through the Adder without changing its value. This is effectively accomplished by either subtracting the complement of the number from zero or subtracting zeros from the number (in both cases end-around operation). The transfers allowed in the computer are as follows:

a) +Z---Adder---A'

| | | | | | | |
|----------------------|---|--|---|--|---|-------|
| (a) | - | (b) | - | (c) | - | (d) |
| +Z 6543 ₈ | | $\begin{array}{r} (1)0000_8 \\ \underline{1234_8} \\ 6544 \\ \rightarrow 1 \\ \hline 6543_8 \end{array}$ | | $\begin{array}{cccc} (1)000 & 000 & 000 & 000 \\ \underline{001} & \underline{010} & \underline{011} & \underline{100} \\ 110 & 101 & 100 & 100 \\ \hline 110 & 101 & 100 & 011 \end{array}$ | | - N/A |

b) +10,20,30, or 40---Adder---A'---S Number is either 10, 20, 30, or 40; depending upon the interrupt recognized by the Interrupt circuitry in Main Control.

| | | | | | | |
|-------------------------|---|--|---|--|---|-------|
| (a) | - | (b) | - | (c) | - | (d) |
| +30 - 0030 ₈ | | $\begin{array}{r} (1)0000_8 \\ \underline{7747_8} \\ 0031 \\ \rightarrow 1 \\ \hline 0030_8 \end{array}$ | | $\begin{array}{cccc} (1)000 & 000 & 000 & 000 \\ \underline{111} & \underline{111} & \underline{100} & \underline{111} \\ 000 & 000 & 011 & 001 \\ \hline 000 & 000 & 011 & 000 \end{array}$ | | - N/A |

c) LOAD COMPLEMENT Number to be complemented is 7777.

The transfer is MCS---Z---Adder---A'---A

| | | | | | | |
|-------------|---|--|---|--|---|-------|
| (a) | - | (b) | - | (c) | - | (d) |
| -Z - 7777 - | | $\begin{array}{r} (1)0000_8 \\ \underline{7777_8} \\ 0001 \\ \rightarrow 1 \\ \hline 0000_8 \end{array}$ | | $\begin{array}{cccc} (1)000 & 000 & 000 & 000 \\ \underline{111} & \underline{111} & \underline{111} & \underline{111} \\ 000 & 000 & 000 & 001 \\ \hline 000 & 000 & 000 & 000 \end{array}$ | | - N/A |

d) P, A, or BER---Adder---A'---S or Z Number transferred is 2345.

This transfer is allowed to prepare the address to reference in MCS.

| | | | | | | |
|-------------------------|---|---|---|---|---|-------|
| BER - 2345 ₈ | - | $\begin{array}{cccc} 2 & 3 & 4 & 5_8 \\ \underline{0 & 0 & 0 & 0_8} \\ 2 & 3 & 4 & 5_8 \end{array}$ | - | $\begin{array}{cccc} 010 & 011 & 100 & 101 \\ \underline{000} & \underline{000} & \underline{000} & \underline{000} \\ 010 & 011 & 100 & 101 \end{array}$ | - | - N/A |
|-------------------------|---|---|---|---|---|-------|

3) Shift Operation - The shift left and right operations are hardware transfers from the A register to the Adder. The number is shifted as it is placed into the Adder and then either zeros are subtracted from the number or the complement of the shifted number is subtracted from zero, with the results in A'. The results in A' are transferred to A, where the results of the shift replace the original number. The A register will hold this result until it is forced out by some new operation involving A.

a. Left Shift - The basic left shifts are a one and three bit position shift using an end-around operation. To do this (left shift one), bit 11 and 00 of A are placed into bit 00 and 01 of the Adder respectively. The other bits are also shifted left one bit position in the same manner.

| | | | | | |
|----------|--------|-----|-----|-----|------------|
| Example: | 101 | 010 | 011 | 100 | original # |
| | (1)010 | 100 | 111 | 001 | shifted # |

The left shift two is accomplished by performing the left shift one twice in the same cycle (6.4 microseconds).

| | | | | | |
|----------|-----|-----|-----|-----|---|
| Example: | 101 | 010 | 011 | 100 | original # |
| | 010 | 100 | 111 | 001 | shifted # first time through Adder and placed into A. |
| | 101 | 001 | 110 | 010 | shifted # second time through Adder and placed in A. |

The left shift three places bit 11 and 00 of A into bit 02 and 03 of the Adder, respectively. The shift left six is accomplished by allowing a shift left three to occur twice in the same cycle.

Example shift left one: A---Adder---A'---A transfer.

The number is 5234₈.

$$\begin{array}{r}
 \text{(A) } 5123_8 \text{ - (Adder input) } 2471_8 \text{ -} \\
 \begin{array}{r}
 \text{(b)} \\
 \text{(1)0000} - \\
 \hline
 5306_8 \\
 2472 \\
 \hline
 \rightarrow 1 \\
 \text{(A')} 2471_8
 \end{array}
 \end{array}
 \quad
 \begin{array}{r}
 \begin{array}{r}
 \text{(c)} \\
 \text{(1)000} \leftarrow \text{000} \leftarrow \text{000} \leftarrow \text{000} \\
 \hline
 101 \quad 011 \quad 000 \quad 110 \\
 010 \quad 100 \quad 111 \quad 010 \\
 \hline
 \rightarrow 1 \\
 010 \quad 100 \quad 111 \quad 001
 \end{array}
 \end{array}
 \end{array}$$

Correct answer

Example shift left three: A---Adder---A'---A transfer.

The number is 1234₈.

$$\begin{array}{r}
 \text{(A) } 1234_8 \text{ - (Pyr. input) } 2341_8 \text{ -} \\
 \begin{array}{r}
 2341_8 \text{ -} \\
 \hline
 0000 \\
 \text{(A')} 2341_8
 \end{array}
 \quad
 \begin{array}{r}
 010 \quad 011 \quad 100 \quad 001 \\
 \hline
 000 \quad 000 \quad 000 \quad 000 \\
 \hline
 010 \quad 011 \quad 100 \quad 001
 \end{array}
 \end{array}$$

Correct Answer

b. Shift Right - The shift right operation is an end-off action with the sign bit (bit 11) carried with the shift. The basic shift is one bit position, but by performing this twice a shift right two is achieved. The shift right one, places bit 11 of A into bit 11 and 10 of the Adder, bit 01 of A into bit 00 of the Adder, and bit 00 of A is not used, therefore, effectively dropped end-off.

Example shift right one: A---Adder---A'---A transfer

The number is 3457₈ - 011 100 101 111 original #
 001 110 010 111 shifted #

$$\begin{array}{r}
 \text{(A) } 3457_8 \text{ - (Adder input) } 1627_8 \text{ -} \\
 \begin{array}{r}
 \text{(b)} \\
 1627_8 \text{ -} \\
 \hline
 0000 \\
 \text{(A')} 1627_8
 \end{array}
 \quad
 \begin{array}{r}
 \text{(c)} \\
 001 \quad 110 \quad 010 \quad 111 \\
 \hline
 000 \quad 000 \quad 000 \quad 000 \\
 \hline
 001 \quad 110 \quad 010 \quad 111
 \end{array}
 \end{array}$$

Example shift right two:

| | | | | | | |
|---------------------------|---|-----|-----|-----|-----|--|
| 4531 ₈ | - | 100 | 101 | 011 | 001 | original # |
| | | 110 | 010 | 101 | 100 | shift right 1 first time |
| (Result)7126 ₈ | - | 111 | 001 | 010 | 110 | shift right 1 second time (shift right two) |

(c) Multiply - This operation is actually performed by shifting; using the basic shift left functions. A multiply by 10₁₀ and 100₁₀ is used by the machine and executed in a 6.4 microsecond cycle. The multiply by 10₁₀ is performed by shifting left one and three at the same time and then subtracting the complement of the shift left one from the left three to place the results of the multiply by 10₁₀ in A'. The multiply by 100₁₀ is merely the multiply by 10₁₀ executed twice in the same cycle.

Example of multiply by 10₁₀: A---Adder---A'---A transfer.

The number is 0007₈.

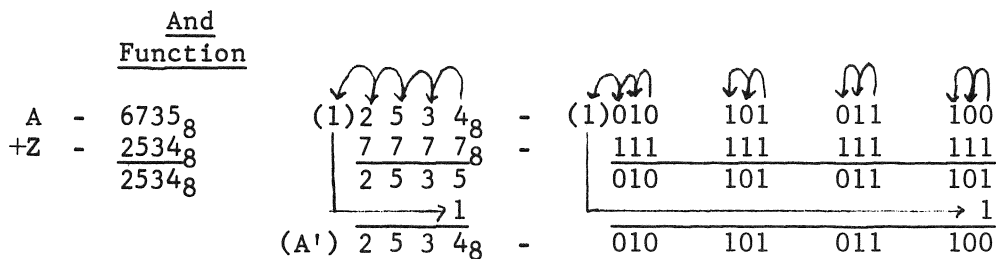
| | | | | | | | | | |
|-----|----------------------------|-------------------|---|--------------------------|---|--------|-----|-----|-----|
| (A) | 0007 ₈ (left 3) | 0070 ₈ | - | (1)0 0 7 0 ₈ | - | (1)000 | 000 | 111 | 000 |
| | (left 1) | 0016 ₈ | - | 7 7 6 1 ₈ | - | 111 | 111 | 110 | 001 |
| | | | | 0 1 0 7 | | 000 | 001 | 000 | 111 |
| | | | | → 1 | | → 1 | | | |
| | | | | (A')0 1 0 6 ₈ | | 000 | 001 | 000 | 110 |

4) Logical Operation - The two logical operations are the logical product, an AND function, and selective complement, exclusive OR function. The logical product operation is used to check the A register for a certain number, and if present, mask or remove all bits of the A register except that

number (if it is present in A). The selective complement is used to remove a certain number from the contents of the A register, if it is present. Both operations take the number from the A register; place it in the Adder, perform the operation with the results in A' and then transfer the results back to the A register.

- a. Logical Product - The numbers used for this operation are obtained from the A register and MCS via the Z register. Z contains the number to be checked for and A contains the number that will be checked to see if it contains the desired number. The result is placed back into the A register, replacing the original number. The AND function is accomplished as the number is placed into the pyramid and all ones subtracted from it. The result is formed in A' and transferred to A.

Example: A & +Z---Adder---A'---A transfer



- b. Selective Complement - The exclusive "OR" function for this operation is accomplished in the input to the Adder and the result is formed in A'. The effect is that of a partial

subtraction assuming borrows were made, but not actually making them. Again, the A register contains the number to be checked for a quantity and Z contains the number that will be removed from the original number in A. A and Z are placed in the Adder, the exclusive "OR" function performed, the result formed in A', and then transferred to A, where the result replaces the original number.

Example #1: A & +Z---Adder---A'---A transfer

Exclusive "OR": 0.0=0
 0.1=1
 1.0=1
 1.1=0

Exclusive "OR"

| | | | | | | |
|-----|----------------------|---|-----|-----|-----|-----|
| A - | 6 7 3 5 ₈ | - | 110 | 111 | 011 | 101 |
| | 2 5 3 4 ₈ | - | 010 | 101 | 011 | 100 |
| | 4 2 0 1 ₈ | | 100 | 010 | 000 | 001 |

*No borrows are made or sensed in this operation, therefore, no Probe pulse is used.

INFORMATION SHEET

F & F' REGISTERS

This unit holds the instruction word throughout its execution. Since many of the instructions require more than one cycle to execute and the instruction word is read only during the D cycle, a register is needed to hold the word for the remaining cycles of its execution.

The input to the F register is the upper 6-bits of the Z register (Z_U) that comes to F via a rank of inverters. This input is allowed only during the first part of the third quarter in a D cycle. The register is cleared of its previous contents only 0.2 microseconds before the transfer from Z to F. The output from F is sent to the F Translator and to the F portion (2 upper octal digits) of the F-P-S Display on the console.

The F' register receives its input from the rank of inverters that feed into the Z register (Transfer not shown on the block diagram), therefore, even though F and F' contain the same upper six bits the input to F' does not pass through the Z register. This input consists of all 12 bits of the instruction word read from memory core via the sense amplifiers and rank of inverters. The input transfer to F' is allowed during the last 0.2 microsecond of the second quarter in a D cycle and F' is cleared 0.2 microsecond before the transfer. The output from F' is sent to the F' Translator with no enables necessary for the transfer.

If F' is to force a new instruction into F, generally a 71, 40, or 41, enables will be made to allow F' Translator inputs to transfer into F. During the bank change instructions, 001X - 007X, the least significant bits

of F' are transferred to the four Storage Bank Control Registers to allow the selected control register to change bank numbers. The lower 6-bits of F' are connected to the E portion of the F-P-S Display on the console.

The F' register receives its input from the rank of inverters that feed into the Z register (Transfer not shown on the block diagram), therefore, even though F and F' contain the same upper six bits the input to F' does not pass through the Z register. This input consists of all 12 bits of the instruction word read from memory core via the sense amplifiers and rank of inverters. The input transfer to F' is allowed during the last 0.2 microsecond of the second quarter in a D cycle and F' is cleared 0.2 microsecond before the transfer. The output from F' is sent to the F' Translator with no enables necessary for the transfer.

If F' is to force a new instruction into F, generally a 71, 40, or 41 enables will be made to allow F' Translator inputs to transfer into F. During the bank change instructions, 001X - 007X, the least significant bits of F' are transferred to the four Storage Bank Control Registers to allow the selected control register to change bank numbers. The lower 6-bits of F' are connected to the E portion of the F-P-S Display on the console.

FUNCTION TRANSLATOR

The function translators (F translators) performs several levels of translations upon the operation code (instruction) held in the F register in order to determine what machine commands must be fulfilled during that instruction. Translation is actually carried out in either or both the F and F' translators. This is required because the F translator samples only

the upper six bits of the instruction and some instructions differ by only the least significant bit of the 12 bits.

The input to the F translator is from the F register and consists of the upper six bits of the instruction. By analyzing several combinations of the six bits, the proper enables can be set to determine what specific instruction has been read from memory. These enables are sent to main control and Storage Bank Control so that these units know what instruction must be prepared for and executed.

F' TRANSLATOR

Many of the 130 instructions can not be represented by a unique combination of the upper 6 bits (F) of the instruction word. For example, there are 19 variations of the basic 01g function code. It is necessary, therefore, to describe most instructions in terms of all 12 bits of the instruction word. A 12-bit supplementary register, F' (F prime), accomplishes this through a F' Translator.

The F' Translator functions are identical to the F translator, except F' analyzes all 12-bits of the instruction word. In most of the 00g and 01g instructions as well as the constant, specific, and memory addressing instructions, the F' Translator provides the enables to main control and SBC. In the jump-no jump type instructions, 0105 for example, and a few other instructions, the F' Translator forces a new 6-bit function code into the F register and translator. This causes main control to change the enables so that another type of instruction can be executed to accomplish the required results. In the example, 0105, if the buffer channel is busy, a 71XX is forced into F and the necessary jump executed, since the F translators now interpret the instruction as a jump forward indirect.

INFORMATION SHEET

INPUT/OUTPUT SECTION

Introduction

The input/output section of the computer provides the methods for data exchange and for proper control of information transmission between the computer, the various external equipment, and the reader and punch.

Two cables, one input and one output, link the external equipment to the computer. All information from externally located peripheral equipments must enter or leave the computer through one of these cables. The paper tape reader (CDC-350) and punch (BRPE-11) however, being integral to the console cabinet, do not connect to the I/O cables. These two equipments generate the necessary control signals artificially, and by the use of internal cabling send or receive data to or from the main computer chassis.

Information passes between the computer and the external equipment as a block of information at a word-by-word rate. The speed of the particular equipment in communication with the computer determines the data exchange rate (maximum rate of BOKC).

The computer allows two means of communication with external equipment. One method is called Normal Channel Input/Output, where the data and control signals are exchanged via the Z register. The second is called Buffer Channel Input/Output with data and control signals exchanged via the Buffer Data register. The normal channel operation, while being executed, stops computer computation until it is completed; thereby slowing down data processing rates.

Buffer channel operation, however, allows normal computation to continue while a word is being sent or received to or from external equipment connected to the buffer channel. Notice at this point, however, that in order to store in MCS or to have read from MCS the word received or sent, the normal computer computation is stopped for a 6.4 microsecond period (cycle) for that operation, and then computation resumed, while the word is sent or the next word is received from external equipment. These two communication methods allow the computer to be used for real time application, since data processing can continue while the slower equipments are sending or receiving data to or from the computer via the buffer channel.

Information & Control Cables

Connected to the normal channel input/output (Z register) and buffer channel input/output (Buffer data register) via gates are two cables. Each cable contains 23 twisted-pair information or control lines and one common ground line. Line assignment for the two cables appears in table 1-1. The cables are identical in construction and are interchangeable with respect to physical connection to the computer as well as to the function they perform. The function of each line and signal on that line is outlined in table 1-2.

Data appears on the communication lines as one of two d-c voltages: The binary "1" condition is "0" volts and the binary "0" condition is -16 volts. All binary digits of a word appear, simultaneously, on the wires of the output or input cable.

Double connectors in each external equipment (CDC-161 typewriter and CDC-163 magnetic tape system) allow more than one device to be connected to the computer. Unique select codes sent by the computer determine which equipment attached to a cable shall communicate with the computer.

Equipment Selection

The controls for input/output operations are located in the Main Control Block of the block diagram.

The computer controls the operation of its external equipment by issuing 12-bit binary function codes. This process, initiated by the EF (75) instruction, selects a 12-bit code located E positions forward of the current instruction address and places it on the output lines together with a function ready signal. Although each external device examines the code, only the particular unit directed by code responds to it. After recognizing the code, the external equipment returns a function (output) resume signal to the computer.

The upper six bits of the function code (unit designator) select a specific external equipment and the lower 6-bits (function designator) specify the function. The unit designator provides the computer with 63 distinct combinations for selecting various external control units. The function designator tells the selected unit what action is requested of it.

The 75 instruction is interpreted in the F & F' Translators and the necessary voltages sent to Main Control to make the enables necessary to execute the instruction. When the code is read from MCS and placed in the Z register,

a function ready signal is sent out on the output cable along with the 12-bit function code. Computer operation is stopped (not buffer channel operation, if it is in progress), until the external equipment responds with an output resume indicating it has recognized its function code and is ready to begin the desired operation. When the output resume is received, computer operation is resumed, and generally some input or output instruction is executed with the appropriate control signals generated and sent.

The control signals (function ready, input request, and information ready) are generated by the input/output circuitry in Main Control and sent to the output or input jacks to be transferred or received to or from the external equipment via the input/output cables. The function code and/or data is placed in the Z register and sent or received via the cables to the selected equipment. If the Buffer Channel is not busy, the function code and ready will also be sent to the buffer equipment via cables connected to the buffer data register since gates connect the buffer cables to the Z register at this time.

The Block Diagram shows the logic connection of the Normal and Buffer Channel to the input/output cables. If buffer channel is not busy, buffer control will disable the input/output cables to the Buffer Data Register (BFR) and enable them to the Z register so that effectively they are in parallel with the Normal Channel cables 1J17, 18, 19, & 20. In order for a piece of equipment to be selected, even for operation over the buffer channel, the function code and ready must be sent over the Normal Channel (via the Z register and Main Control). Once the buffer equipment is selected, the data transfer can be over either buffer or normal channel.

If it is to be via buffer channel, the execution of a 7200 or 7300 instruction will cause the gates into the buffer channel to be enabled and the gates to normal channel disabled.

From the above discussion three conditions will hold true:

- 1) Normal or buffer equipment can be selected & operated over the normal channel. Computation is stopped.
- 2) Buffer equipment can be selected over normal channel and then operated via the buffer channel while computation is in progress in the operational registers.
- 3) Buffer equipment can be selected via the normal channel and operated via the buffer channel after which normal equipment can be selected and operated over the normal channel. Computation is stopped during the normal channel input/output operation.

Notice, that once normal channel input/output is in progress, buffer equipment can not be selected even for buffer channel operation, until the normal channel operation is complete. Also note, normal equipment can not be operated via the buffer channel, but either normal or buffer equipment can be operated via the normal channel.

TABLE 1-1 PIN ASSIGNMENTS, INPUT/OUTPUT CABLES

| INPUT CABLE | | PIN | OUTPUT CABLE |
|------------------------|------------------------------|-----|--|
| Buffer Chan. 1J21 & 22 | | | Buffer Chan. 1J23 & 24 |
| Normal Chan. 1J17 & 18 | | | Normal Chan. 1J19 & 20 |
| Bit 0 | Input Status and Information | A | Bit 0 Output Function Code and Information |
| 1 | | B | 1 |
| 2 | | C | 2 |
| 3 | | D | 3 |
| 4 | | E | 4 |
| 5 | | F | 5 |
| 6 | | H | 6 |
| 7 | | J | 7 |
| 8 | | K | 8 |
| 9 | | L | 9 |
| 10 | | M | 10 |
| 11 | | N | 11 |
| | | P | |
| | Input Ready | R | Information Ready |
| | Input Request | S | Output Resume |
| | | T | Function Ready |
| | | U | Master Clear |
| | Input Disconnect | V | |
| | | W | Interrupt 10 (Advent only) |
| | | X | |
| | | Y | Interrupt 30 |
| | | Z | Interrupt 40 |
| | | a | |
| | Ground | b | Ground |

TABLE 1-2 INPUT AND OUTPUT CABLE LINE SIGNALS

INPUT CABLE LINE SIGNALS

Input Data and Input
Status
(12 lines)

Dual purpose:

- 1) As data lines they hold equipment input register contents which the computer may sample.
- 2) As input status lines they indicate equipment's response to status request sent by the computer over the output cable.

Input Ready
(1 line)

Indicates that the external equipment contains information which the computer may sample.

Accompanies the input data from external equipment. (Computer resync circuits are oriented about the leading edge of the ready signal.)

Input Disconnect
(1 line)

Indicates to computer that the input device has no more data to deliver. Computer is then free to resume main program with no further delay if normal channel is used.

OUTPUT CABLE LINES

Output Data and
Output Function
(12 lines)

Continuously monitored by all equipment.

Dual purpose:

- 1) As output data lines they hold the output word which the external device may sample.
- 2) As output function lines they carry external function (EF) codes from the computer to external equipment. Function ready alerts the

OUTPUT CABLE LINES (cont.)

| | |
|-------------------------------|--|
| Function Ready (1 line) | equipment to sample the function code. Accompanies EF code; turned on by instruction 75 and causes the equipment to examine EF code. It is turned off by an output resume from the external equipment. |
| Information Ready (1 line) | This signal accompanies the output data word from the computer and is turned on when the computer has a word of information ready for the previous selected external equipment. It is turned off by an output resume from the equipment. |
| Output Resume (1 line) | This signal is turned on when the external device has accepted the output word or EF code. (The computer resync circuitry orients itself about the trailing edge of the resume; when the signal drops the word is exchanged.) The computer prepares a word while the signal is up. |
| Master Clear (1 line) | This signal clears all external equipment. It occurs when the Load-Clear switch at the console is in the Clear (down) position. |
| Interrupt 10 (1 line) | Used with Advent Program only. If no Interrupt Lockout, store P at address (d) 0010 and obtain next instruction from address (r) 0011. |

OUTPUT CABLE LINES (cont.)

Interrupt 30
(1 line)

If not Interrupt Lockout, store P at address (d)
0030 and obtain next instruction from address
(r) 0031.

Interrupt 40
(1 line)

If not Interrupt Lockout, store P at address (d)
0400 and obtain next instruction from address
(r) 0041.

INFORMATION SHEET

Normal Input-Output Control

After the external equipment has been selected and an output resume received by the computer, the computer will initiate an input or output operation via the normal channel. To enable the computer to function with asynchronous equipment, the input/output circuitry must synchronize the long duration incoming signals with computer signals. This resynchronizing circuit with the input/output sequence control (for gating) is included in Main Control. The data transfer is to and from the Z register for normal channel operation.

Main Control receives voltages from the F & F' translators indicating either an input or output operation. It then sends the necessary control signals, makes the required input/output gating for the Z register, and stops computer operation. When the response control signals return from external equipment, the resynchronizing circuit synchronizes these signals and starts computer operation allowing either the storage of the 12-bit word received or the reading of a new 12-bit word to be sent to external equipment.

Input Data Transmission

Input operations are generated by the execution of a 72XX and 7600 instructions interpreted by the F & F' Translators. The former is a block input of one to 7777₈ 12-bit words to MCS and the latter a one 12-bit word input to the A register, both via the Z register.

When Main Control senses the 72XX instruction from the F & F' Translators, it makes necessary transfer enables, places the starting (storage) address in the A register, issues an input request signal via the input cable (1J 17 & 18) or

to Reader logic and stops the computer timing chain (normal operation). The previously selected external equipment excepts the input request, generates and issues an input ready, and sends the data word to the computer. The computer receives the input ready into the input/output circuit in Main Control via the input cable. When the resync circuit gates this signal, it will start the timing chain and gate the data word into the Z register. The computer will then prepare the storage address (taken from A to S via Adder & A'), clear the old contents from the address, store the data word in MCS at that address, and update the storage address in A by one, then read the last storage address +1 from MCS, compare with the present storage address and if not done issue another input request from Main Control via the input cables and stop the computer until the next word is received. This continues until all input words requested are brought into the computer, after which a new instruction is read and interpreted for execution.

The operation described could have been from buffer equipment over cables 1J 21 & 22 into the Z register with the control signals generated in Main Control and sent via Buffer Control over cables.

When the status of an external piece of equipment is requested by the function code, a 7600 instruction is the next instruction executed. Main Control again receives the type of instruction voltages from the translator, makes necessary transfer enables, issues an input request via 1J 17, 18, 21 or 22, and stops the timing chain. The external equipment receives the signal, generates an input ready, and gates the status response 12-bit word to the computer via the input cables. The input ready is received by Main Control, the timing chain started, the data gated into Z, and transferred to A via the Adder and A'. After this, the next instruction is read, interpreted, and its execution started.

The inputs into Z are via a rank of inverters and are all 12-bit words, except the input from the paper tape reader that is directly into Z and is only a maximum of 8-bit words. The reader input under a special load mode is 6-bits into Z_U (upper 6-bits) followed by six bits that is placed into Z_L (lower 6-bits) making a 12-bit word before it is stored into MCS.

Output Data Transmission

Output Data is read from MCS or transferred from the A register to external equipment via the output cables or to the paper tape punch register, and punch logic. These outputs are allowed with the execution of either the 73XX, 74XX, or 76XX instruction. The 73XX instruction is a block of words from 1 to 7777_8 words, 74XX one word of the lower 6 bits of Z only, and 76XX the 12-bits from the A register. Again, these operations will be preceded by the selection of the desired equipment by the use of the 75 instruction.

The output from the normal channel is through the Z register to the output cables via a rank of inverters and to the punch directly from Z to the "0" register. The latter output is only the lower 8-bits of Z; while the former is all 12-bits of Z, except during the 74XX which is only the lower 6-bits of Z.

When the 73XX instruction is read and translated, Main Control receives the translated voltages, prepares the transfer enables, reads and stores the first word address in the A register, reads the first data word from this address, places the data word in Z and on the output cables, issues an information ready and stops computer operation. The previously selected external equipment, either buffer or channel, receives the information ready control signal and data word, gates the data word into its registers, and generates an output resume that is sent to the computer. The reception of the output resume by the computer starts the timing chain and causes the computer to read the last data word storage

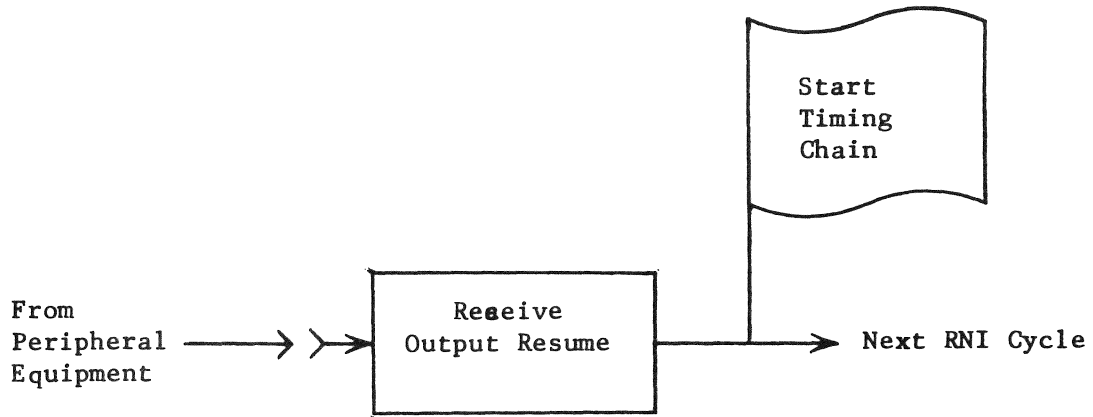
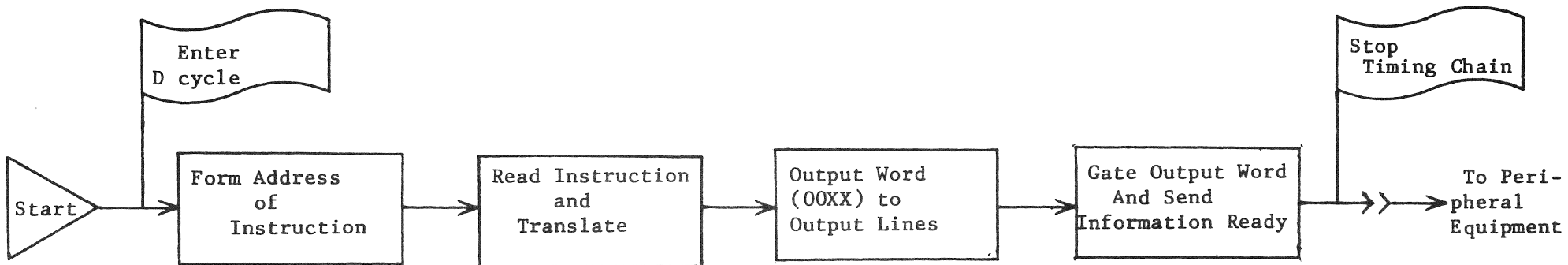
address and compare the present data word storage address with the last data word storage address to check for completion. If they were equal, the operation would terminate and the next instruction would be read from MCS. If they were not equal, the next data word would be read and sent to the equipment along with another information ready control signal. This operation would continue until the two storage address are equal, at which time the operation would terminate.

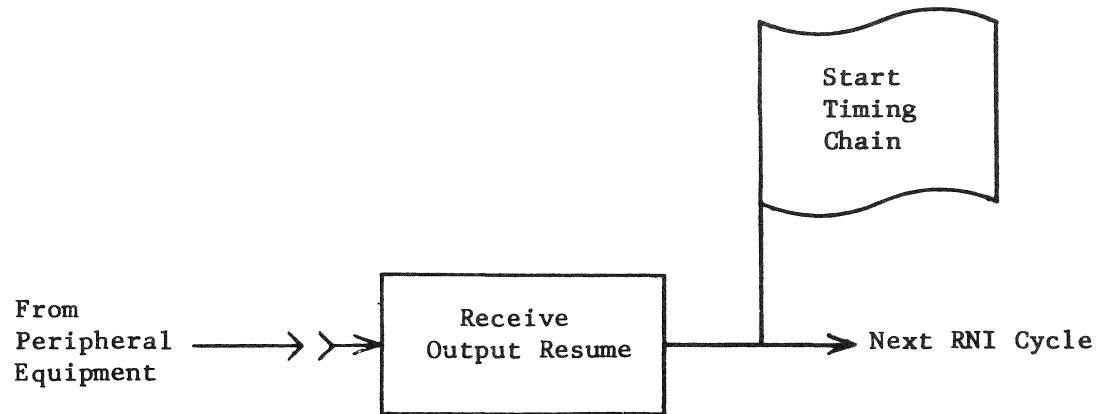
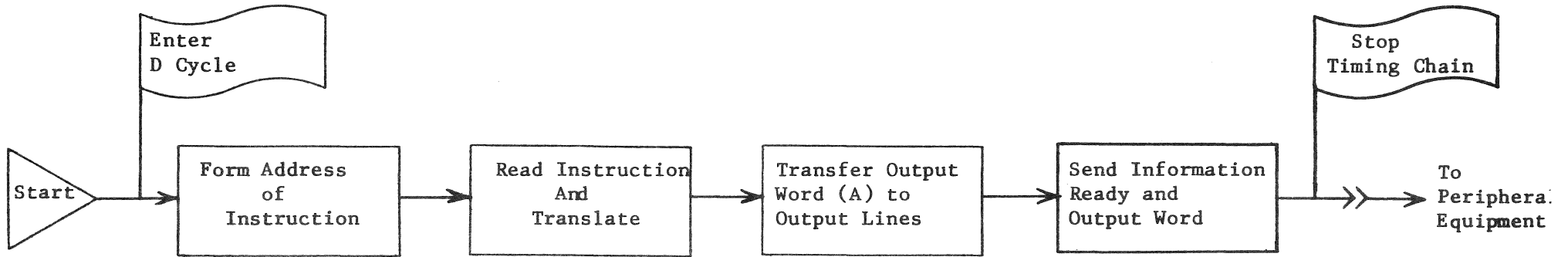
The one word output operations are basically the same, with the computer stopping when the data and control signal are sent out over the output cables and starting again when the output resume is received from the selected external equipment.

It is important to note at this point, for both the input and output operations if the last piece of equipment selected was other than the reader or punch and other instructions had been executed since that selection was made, the execution of an input or output instruction will cause the computer to stop indefinitely. This is caused since no piece of equipment is selected, therefore, none to answer with an output resume or input ready and input data word. The reader and punch are excluded here, because once selected they stay connected, logically, to the input-output channel until another equipment is selected. Therefore, if the last equipment selected was the reader and several instructions had been executed since that selection, the execution of an input or output instruction is executed the operation will be carried out properly.

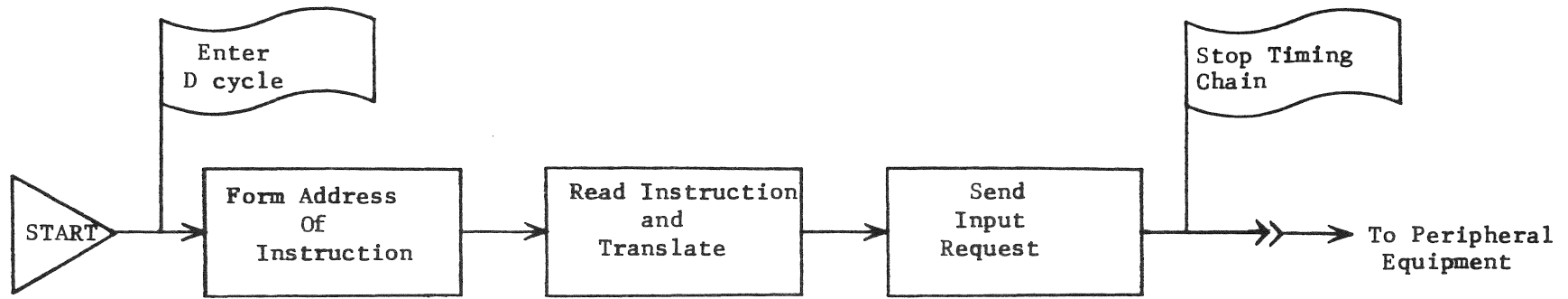
The reader and punch are always connected to the normal channel because of the logic connections in the internal computer desk.

74XX OUTPUT NO-ADDRESS

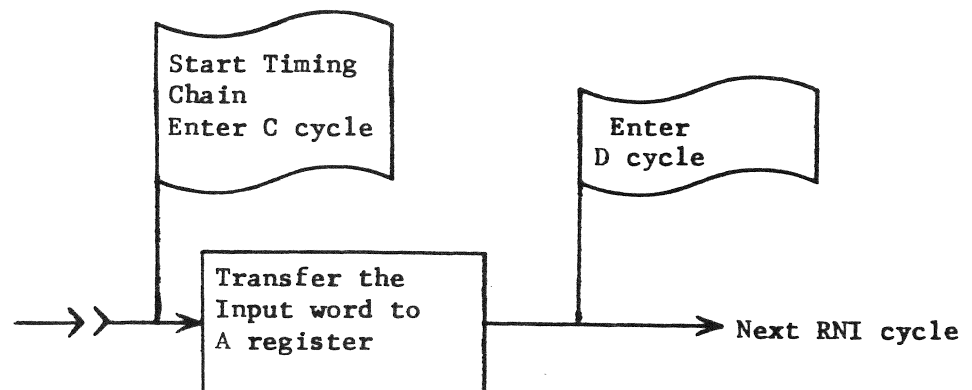


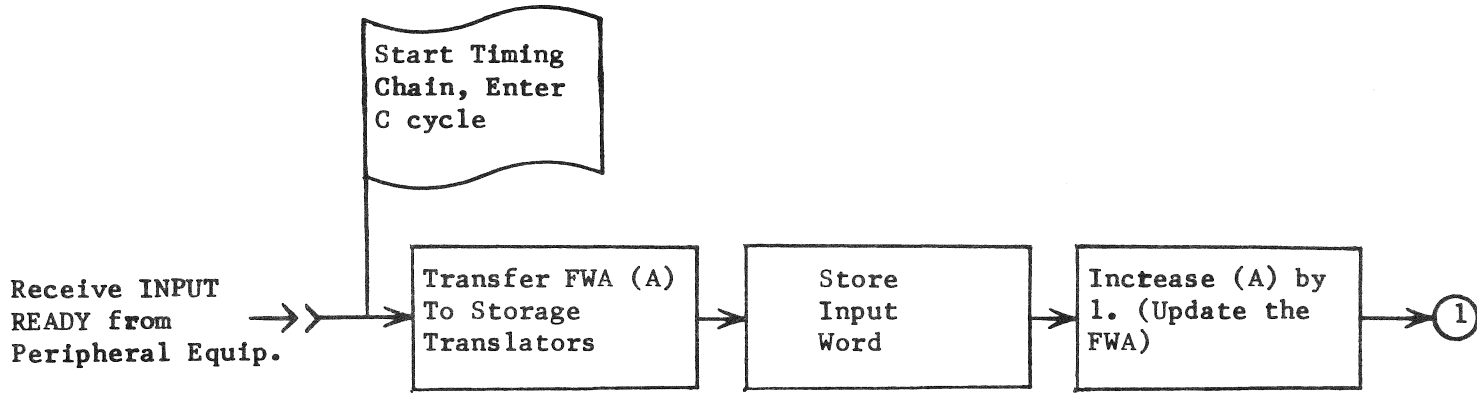
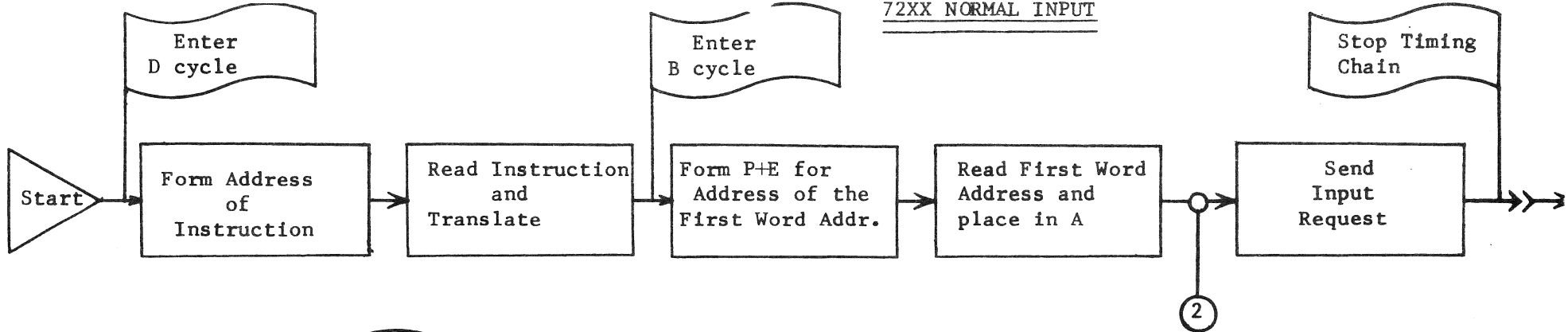


7600 INPUT TO A

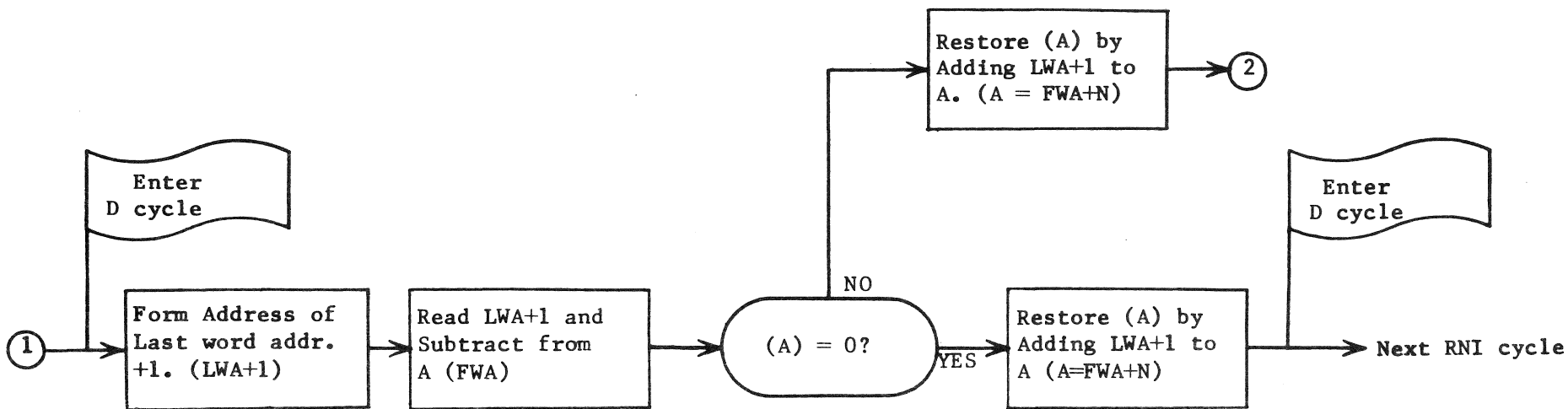


Receive INPUT
READY from
Peripheral
Equipment

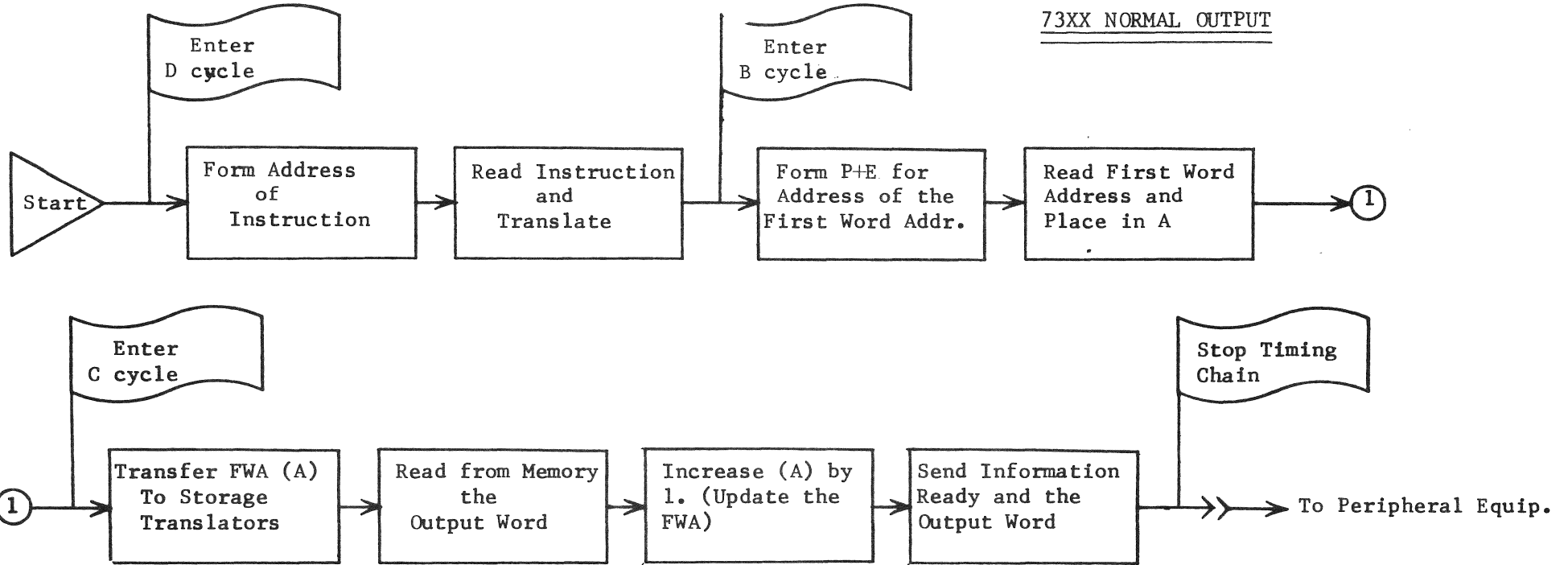




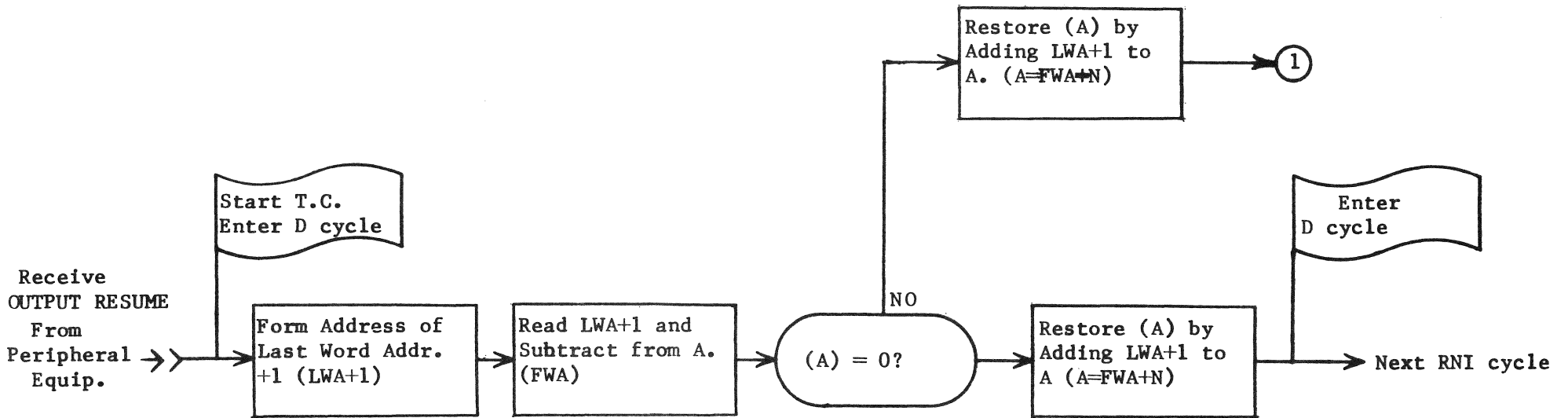
06



73XX NORMAL OUTPUT



16



INFORMATION SHEET

Buffer Channel Input-Output

Buffer Channel operation is the second computer method of communicating with external equipment and is basically the same as the normal channel operation. The main difference is, that once the buffer channel input or output operation is initiated, computation can continue, while the data is being sent to or received from external equipment. When the input word is received along with the input ready or an output resume indicating buffer equipment is ready for the next word, buffer control will stop computation or normal channel operation for only a 6.4 microsecond period, in which the buffer input or output cycle is initiated. This cycle stores the input word or reads up the next output data word, updates the storage address, issues the required control signals, and compares the storage addresses to check for completion, all in the above mentioned 6.4 microsecond period, after which it releases the normal channel or computation for resumed operation whether the buffer operation was completed or not. This same operation over normal channel takes at least 12.8 microseconds, when the word is received or sent and computation can't resume until the input or output is complete.

From the above it can be concluded that buffer channel is not totally independent of the normal computer operation since it must time share the MCS with the normal computer circuits, but it does allow partial independence that allows either two input/output operations together or one input/output operation and computation at the same. This is a basic requirement for real time computer operations involving many input/output sources.

a buffer input/output operation was in progress, buffer control would signal main control of this busy condition, allowing main control to cause a jump condition to read the next main program instruction, waiting for a later time to initiate the desired buffer operation.

The control signal inputs tell buffer control to initiate a buffer cycle operation, to store the new input word, or read and send the next output data word.

Outputs from buffer control are to main control, BFR, BER, BXR, and buffer channel input/output lines. The output to main control consists of many single voltages indicating buffer channel status and also the voltages needed to stop normal operation, when a buffer cycle is required during a buffer operation. These voltages can also start the timing chain for buffer operation, even though normal operation has been stopped by the Run-Step switch being placed in the center or neutral position. The outputs to the registers are gating voltages to allow the required data transfers. Also these allow the buffer cables to be connected to normal channel (Z register) for a BFR busy condition and connected to the BFR (Buffer channel) for a BFR busy condition.

Buffer Data Register (BFR)

This 12-bit register holds all data words sent or received over the buffer channel and the data stored in MCS during a block store instruction. It is one of the only two means of reading or writing data into MCS.

The inputs to the BFR are from the buffer input cables and the same rank of

inverters that feed the input to the Z register. The inverter rank has inputs from A' and MCS (read amplifiers) that are allowed into the BFR. The BFR is cleared in either the second or fourth quarter of a D or Buffer cycle depending upon the instruction executed.

The outputs from the BFR are to MCS via a rank of inverters (same as Z output inverters), the buffer output cables and the BFR-A-A' display on the console.

The inputs and outputs of the BFR are all gated by the voltages received from the buffer control. Since both the inverter rank from MCS (Read Amplifiers) and the inverter rank to MCS (Inhibit drivers) are used by both Z and BFR for access to MCS, it can be understood why these two channels must time share in order to store or read data.

Buffer Entrance Register (BER)

The BER is a 12-bit register and its primary use is to hold the first word storage address for the buffer input or output operation. This address is loaded into the A register and transferred to the BER using other instructions, before the buffer input-output instruction is executed. The BER can also be used for a storage location, if the buffer channel isn't used.

The only input to the BER is from the A' register and is a forced transfer. This means that both the clear and set sides of A' are connected to the clear and set sides of the BER. For this reason, the BER is not cleared by other means.

Three outputs from the BER are connected to the compare circuit (direct) to the BER-Z-BXR Display on the console, and to the Adder. The BER to Adder transfer is allowed to update and prepare the storage address during a buffer cycle and to transfer the contents of the BER to the A register.

Buffer Exit Register (BXR)

The only function of this 12-bit register is to hold the last word storage address +1 (LWA +1) during the buffer input/output operations. This address is loaded into the A register and transferred to the BXR by using other instructions prior to executing the buffer input or output instruction.

The input to the BXR is from the A register and is a forced transfer. Again, the BXR is not cleared, due to the forced transfer being used.

The two outputs from the BXR are to the compare circuit (direct) and to the BER-A-BXR Display on the console.

Compare Circuit

This circuit is used to constantly compare the output of the BER and BXR; to allow buffer control to check the status of the block transfers via the buffer channel. The circuit consists of three inverters with AND and OR inputs.

The inputs to the compare circuit are from the BER and BXR and are constantly applied. When all 12-bits of each register are equal, a voltage will be sent to buffer control to indicate $BER = BXR$. This condition is used, during a buffer input or output, to signal completion of the operation.

Buffer Channel Input Operation

Before the 7200 instruction (initiate buffer input) is executed, the first word address (FWA) must be placed in the A register and with the 0105 instruction transferred to the BER and the last word address +1(LWA+1) must be placed in A and with a 0106 instruction transferred to the BXR. This prepares the number of words to be sent or received, so that after the equipment has been selected via the normal channel, the 7200 instruction can be executed.

The 7200 instruction is read and translated by the normal computer circuits then buffer control is checked for its status--busy or not busy. If the buffer is busy-- waiting on an input word or an output resume--the computer (main control) will cause the computer to go one location forward of the address containing the 7200 instruction and obtain an address that the computer will jump to for the next instruction. If the buffer is not busy, the BFR will be cleared and a signal sent from main control to buffer control to cause the latter to issue an input request to the previously selected buffer equipment and prepare to initiate a buffer input cycle when the data and input ready are received by the BFR and buffer control, respectively. Main control then causes the next instruction to be found two locations forward of the location containing the 7200 instruction. Normal operation continues while the buffer channel performs the input operation, with buffer control stopping normal operation only ever so often for the 6.4 microsecond buffer input cycle.

Each time buffer control receives an input request, it tries to stop normal operation by sending a voltage to main control. Main control stops normal operation at the end of any cycle, except the end of a B cycle that is

by a C cycle. The exception has to be considered, since under this condition the normal channel storage address is contained only in the "S" register and would be destroyed if the buffer cycle was allowed. When normal operation is stopped, all enables are made to go into the next normal channel cycle when the buffer cycle is complete.

Buffer control; starts the timing chain, sets SBC to buffer bank, prepares the address by allowing a $BER \rightarrow \text{Adder} \rightarrow A' \rightarrow S$, allows the input word into the BFR, clears the prepared address in MCS, stores the word in that address, updates the FWA in the BER by allowing a $BER \& +1 \rightarrow \text{Adder} \rightarrow A' \rightarrow BER$, checks if completed, clears BFR, and causes main control to continue operation.

If the $BER \neq BXR$, another input request is issued by buffer control after which it waits for the new data and an input ready. If the $BER=BXR$, buffer control would block the input request and after releasing main control for normal operation, it will send a voltage to the interrupt circuitry in main control to cause an interrupt #20 to be generated.

Buffer Channel Output Operation

This operation is initiated by the execution of a 7300 instruction but must be preceded by the setting up of the FWA in BER and $LWA+1$ in BXR. Again, if the buffer is busy, main control will sense this and cause the next instruction to be read two locations forward of the location containing the 7300 instruction. If the buffer is not busy, main control will initiate a buffer operation by setting a FF in buffer control, after which it will cause resumption of normal operation.

Buffer control will now send a voltage to main control to stop normal operation at the earliest possible time. This time will be as described in the buffer input above.

The buffer cycle will be initiated, when buffer control stops normal operation. Buffer control; sets SBC to Buffer Bank, prepares the FWA and places it in the "S" register, clears BFR, reads the address in MCS and places the word in the BFR, updates the FWA in BER, issues an information ready via the output cables, sends the word out on the cable, restores the data word to MCS, checks if completed ($BER = BXR$), releases main control so that it resumes normal operation, and stops buffer operation, until the output resume is received. If $BER \neq BXR$, buffer control will initiate another buffer cycle when the next output resume is received. When $BER = BXR$, buffer control will terminate the buffer operation and after releasing main control, will generate and send to main control a voltage that causes the interrupt circuit to initiate an interrupt #20 if an interrupt lockout has not been imposed.

INFORMATION SHEET

INTERRUPT

The use of a computer in real time tracking systems require that external equipment be recognized for input/output operations, when certain conditions are created by the equipment supplying data to these external equipments. Since the computer could be in any part of the main program and at any address in memory core storage, a means must be provided to store, automatically, the address of the instruction currently being executed in the main program, when an interrupt signal is recognized from a piece of external equipment. The storage locations used for the storage of the main program address are permanent locations that can be referred to at a later time for this address, so that the computer can return to the main program without loss of data or difficult programming requirements.

The computer provides four such interrupt lines; two internally initiated, 10 (could be adapted for external use) and 20, and two externally initiated, 30 and 40. With the recognition of an interrupt signal by the interrupt circuitry in main control, the contents of the P register (address of the instruction about to be executed) is automatically stored in either locations 0010, 0020, 0030, or 0040 in the direct bank. The location used is determined by the interrupt recognized. Without stopping, the computer reads, translates, and executes the instruction found at locations 0011, 0021, 0031, or 0041, respectively, in the relative bank and continues operation from that point. This sub-routine is generally one to send or receive data from the interrupting equipment and return to the main program as soon as possible. In some cases more than one equipment may be on one interrupt line, therefore, the computer

must check to see what equipment sent the interrupt signal. In all cases, the computer, if programmed, will return to the main program via the interrupt storage address.

Interrupt 10

This interrupt, normally, is initiated by depressing, momentarily, any combination of Selective Jump and Stop switches on the console but is sometimes modified to be activated by the presence of an external interrupt voltage. When interrupt 10 is initiated, the contents of P are stored at location 0010 in the direct bank (d) and the next instruction taken from 0011 in the relative bank (r). The external interrupt voltage could be from either normal or buffer cables.

Interrupt 20

The completion of a buffer operation indicates that a piece of equipment, possibly one of many, on the buffer channel has completed its input and/or output function. This requires, in some systems, that house-keeping operations determine what buffer equipment is next in line for operation. Therefore, a buffer channel termination, except one caused by a clear Buffer Control, causes the generation of an interrupt 20. If or when the signal is recognized by the interrupt circuitry, the contents of P will be stored at location (d) 0020 and the next instruction obtained from (r) 0021.

Interrupt 30 and 40

Interrupts 30 and 40 are initiated by signals from external equipment, either buffer or normal, setting the appropriate interrupt FF in main control. In cases where more than one piece of equipment is connected an interrupt line,

the equipments must be checked to determine which one activated the signal line. A different operation could occur with each equipment but still be a part of the interrupt 30 or 40 sub-routine. When the interrupt is recognized, the contents of P are stored at either location (d) 0030 or (d) 0040 and the next instruction obtained from (r) 0031 or 0041 according to the interrupt initiated.

Interrupt Circuit Rules

Priority of Recognition

The four interrupt signals are recognized on a priority basis, i.e., if 20 and 40 occur simultaneously, 20 will be initiated first. If the proper conditions are set up with no other interrupts occurring, interrupt 40 will be recognized. Some equipments, generating interrupts, leave them on the lines indefinitely, but other for only a few milliseconds.

Interrupt Lockout

In order to keep an interrupt sub-routine from being interrupted, a lockout feature is used. This lockout, when imposed, prevents the recognition of any waiting interrupts, regardless of their priority, and continues to do so until cleared.

The lockout is imposed by two conditions: recognition of any interrupt signal or the execution of an external function (external equipment selection) instruction. Interrupt lockout will remain in effect until cleared by one of two conditions: the execution of a CIL (Clear Interrupt Lockout) instruction during run status or by Master Clear from the console in neutral or stop status. After clearing the lockout, all waiting interrupts are checked for priority and

the proper one then initiated. It must be pointed out, that if one of the two mentioned conditions creates a lockout and one of the two clearing conditions is not executed, the lockout will remain indefinitely.

With the above items in mind, it can be concluded that the procedure for an interrupt sub-routine should be as follows. After the interrupt has been initiated and the main program address stored, the next instruction will be a jump forward indirect to some address in the relative bank to begin the sub-routine. At this address execute an instruction to store the contents of A, since A will probably be used during the sub-routine and any contents thereby destroyed. Execute the sub-routine and at the end of the sub-routine: load the original contents back into A, execute the CIL instruction (0120), and jump indirect to the main program address, via the address where it was stored (d 0010, 0020, 0030, or 0040), for the next instruction.

Interrupt Flow Chart

An interrupt can only be recognized in the first 0.4 microsecond of a D cycle, after the interrupt has been received from external equipment, manual or buffer terminate interrupt circuits. As soon as it is recognized (interrupt FF and interrupt lockout FF set), the D cycle will be interrupted - before the new address is placed in "S" in the first quarter - and a 40 forced into the F register and translator from the F' translator. The remainder of the D cycle is the same, except the instruction read and restored from and to MCS is not used. Since the 40 is forced into the F translator, the remainder of the D cycle will be executed as a modified store direct instruction. This will cause a D-B-C cycle progression in the REL-DIR-DIR banks, respectively, with

the F' translator providing the necessary modifying voltages to main control to carry out the storage of the main program address.

| Data Flow | Comments |
|--|--|
| <u>D1</u> P & + 1---Adder | - Prepare address of regular instruction |
| <u>2</u> Interrupt FF set | - Interrupt signal recognized and Interrupt Lockout FF set. |
| <u>3</u> Adder--- A'--- S--- P & MCS cont. | - Place results of D1 in S, P, and MCS Cont. to read the regular instruction. |
| <u>4</u> Clear Z & F | - Prepare for instruction |
| <u>5</u> MCS--- Z (Block Z--- F & Inv.--- F') | - Read instruction from MCS at address specified by "S" and place in Z, but not in F. Z--- F & Inv.---F' is caused by the setting of the interrupt circuits, so that the regular instruction will not be translated. |
| <u>6</u> Force 40 into F | - Caused by interrupt circuits in main control. Allows the interpretation of a store indirect instruction instead of the regular instruction. |
| <u>7</u> Main Cont. & SBC | - Make necessary enables to execute modified 40X instruction. Prepare for a D-B-C progression in REL-DIR-DIR banks. |
| <u>8</u> Z--- MCS | - Restore unused instruction to address specified by "S". |
| <u>9</u> Go to B cycle & Set SBC | - B cycle is used to prepare interrupt storage address either 0010, 20, 30, or 40. |

- | | |
|---|--|
| <u>B1</u> Block regular enables to Adder force either 0010, 0020, 0030, or 0040--- Pyr. | - Prepares the address in the direct bank where the main program address (in P) will be stored. |
| <u>2</u> Adder--- A'--- S--- MCS Cont. | - "P" still contains the main program address of the instruction about to be executed, when interrupted in D2. |
| <u>3</u> (Clear Z) MCS--- Z | - Read the address specified by "S". |
| <u>4</u> Z--- MCS | - Restore the old contents to the address specified by "S". |
| <u>5</u> Go to C cycle & keep SBC set to DIR | - C cycle is used to store the main program return address. |
| <u>C1</u> S--- MCS Cont. | - "S" retains the address prepared in B2, where the main program return address will be stored. |
| <u>2</u> P---Adder--- A' | - Prepare to store main program address (P) in MCS. |
| <u>3</u> (Block MCS--- Z) Read MCS | - Read MCS, but do not place in Z. This clears the old contents from the selected address. |
| <u>4</u> (Clear Z) A'--- Z | - Place contents of "P" in Z to prepare for storage. |
| <u>5</u> S--- P | - Update "P" with either 0010, 20, 30, or 40. |
| <u>6</u> Z--- MCS | - Store main program address in MCS at address specified by "S". |
| <u>7</u> Clear Interrupt FF | - Interrupt lockout is still imposed. |
| <u>8</u> Go to D cycle & set SBC to REL | - Advance to (r) P + 1 for the next instruction which will be either location 0011, 21, 31, or 41 in the REL Bank. |

Clearing Interrupt Lockout

Interrupt lockout will immediately be cleared by the use of Master Clear, as will the majority of the operational registers of the system. However, when the CIL instruction is used, a different situation is created.

The CIL instruction (0120) is executed in 6.4 microseconds, but the input gating to the interrupt circuits is disabled until the second D cycle following the execution of the CIL instruction. This allows the execution of one instruction, of one to four cycles in duration, before the next interrupt will be recognized.

Refer to the programming manual for additional information on Interrupt Operation.

1.0 INTRODUCTION

Ault power supplies are 90% control systems and 10% power handling devices. This is necessary in order to obtain the very high degree of regulation required by the loads for the supplies. As in any control system, Ault power supplies are merely a collection of black boxes, any one of which is easily understood. An understanding of the parts results in a clear understanding of the whole system, and allows for easy maintenance.

1.1 DESCRIPTION of CONTENTS

This manual is divided as follows:

Section two describes some of the less familiar devices employed in Ault power supplies.

Section three describes the theory of operation of the parts of the Peripheral Equipment supply.

Section four discusses the supply itself.

Section five provides maintenance tips.

Also included are a schematic circuit diagram and a selection of drawings showing oscilloscope patterns from various locations in an operating supply.

Just one hour spent with this manual should help understand and repair any malfunction in the power supply.

2.0 DEVICE DESCRIPTIONS

The less common devices used by Ault Magnetics in the CDC Peripheral Equipment power supply are the zener diode, the

silicon controlled rectifier, the tunnel diode and the unijunction transistor.

2.1 AENER DIODES

A normal diode voltage current characteristic looks like that shown in Figure 1.

For silicon diodes, particularly of the diffused junction types, there is an abrupt breakdown in the reverse direction as shown by the dotted line in the figure.

As can be seen from the sketch, any current in excess of I_{R1} drawn through the device in the reverse direction will result in a constant voltage appearing across the device. This constant voltage can, and is used as a voltage reference source in all Ault power supplies.

The symbol for a zener diode is shown in Figure 2.

In a circuit application the zener diode is connected in series with a current determining resistor across some dc voltage as in Figure 3.

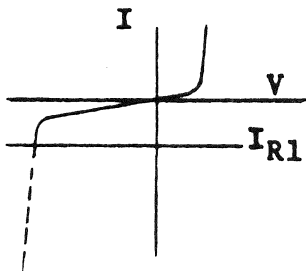


Figure 1.
Diode Voltage
Current Characteristic



Figure 2. Symbol
for a Zener Diode

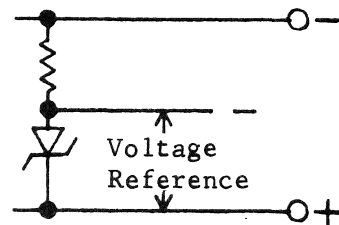


Figure 3. Zener Diode
Circuit Connection

2.2 SILICON CONTROLLED RECTIFIER

A silicon controlled rectifier is a device which blocks current flow in both directions until a pulse is put into what is called the "gate". At this time the device begins to conduct in one direction while continuing to block in the reverse direction.

The symbol for a silicon controlled rectifier is shown in Figure 4 with "G" indicating the gate lead.

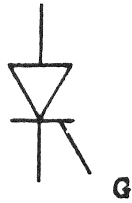


Figure 4.
Symbol of a Silicon Controlled Rectifier
with G indicating the gate

2.3 TUNNEL DIODE

The tunnel diode has a voltage-current characteristic described in Figure 5.

When a current less than I_p is forced through the device, a voltage less than E_p appears across the device.

As the input current is increased to a value greater than I_p , the voltage across the device switches to the value of E_f .

The symbol for the tunnel diode is shown in Figure 6.

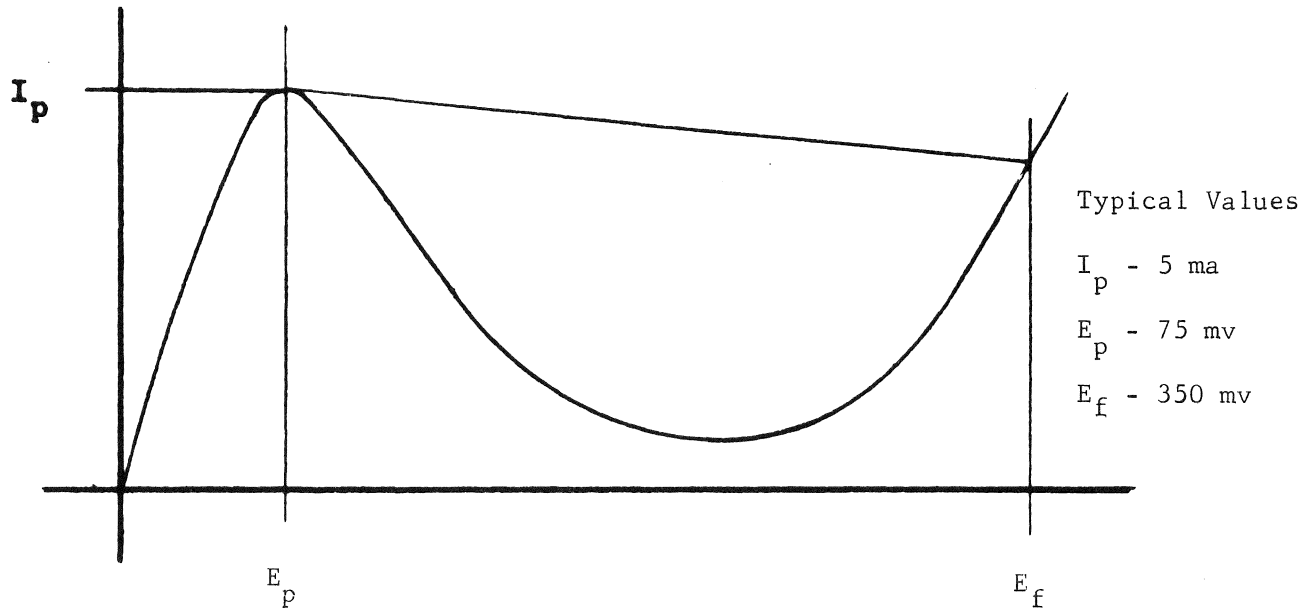


Figure 5 Forward Voltage - Current Characteristic of a Tunnel Diode

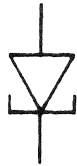


Figure 6 Tunnel Diode Symbol

2.4 UNIUNCTION TRANSISTOR

The equivalent circuit of the unijunction transistor is sketched in Figure 7.

Here R_1 and R_2 are approximately the same value.

When the signal into the emitter E is less than approximately $\frac{1}{2}E_B$, there is very low current flow I_{in} due to the blocking action of the diode.

When the voltage V_{in} is greater than $\frac{1}{2}EE$, the diode begins to conduct current and resistance R_2 becomes very small resulting in a large current flow between the emitter E and base 1.

The symbol for a unijunction transistor is shown in Figure 8.

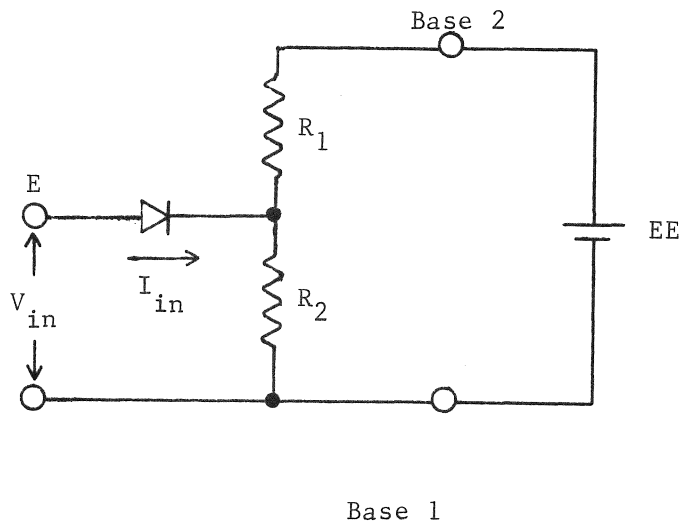


Figure 7 Unijunction Transistor Equivalent Circuit

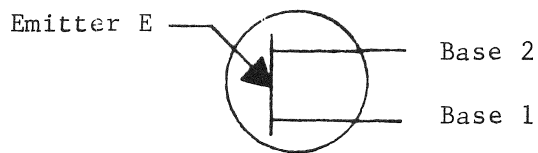


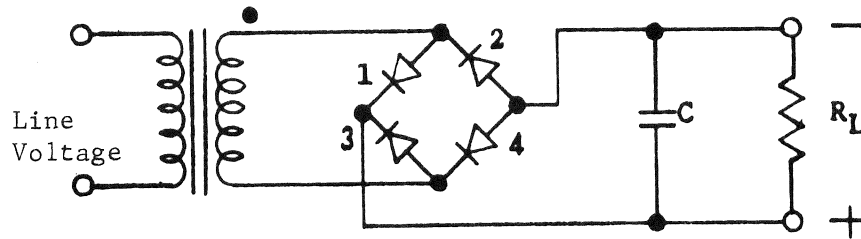
Figure 8 Symbol for a Unijunction Transistor

3.0 THEORY of OPERATION of PERIPHERAL EQUIPMENT POWER SUPPLY

This section of your instruction manual describes the theory of operation of the Peripheral Equipment power supply

3.1 NORMAL RECTIFIER CIRCUIT

A normal bridge type rectifier circuit looks like Figure 9.



When the dot end of the transformer secondary is plus, rectifiers 1 and 4 conduct allowing current to flow from + to - through the load R_L . When the other end of the transformer is plus, rectifiers 2 and 3 conduct again allowing current to flow from + to - through the load. This means that current from the transformer secondary is always flowing to the filter capacitor and the load R_L . The voltage waveform out of the rectifier looks like Figure 10.



Figure 10 Voltage Waveform Out of Full-Wave Bridge Rectifier

The voltage across the load, however, is relatively smooth dc due to the filtering action of c. The magnitude of this voltage is near the peak value of one-half sine wave under no load conditions.

As the load is increased (R_L is decreased), the current flowing through the internal resistance of the supply and poorer filtering action team up to produce internal voltage drop and poor load voltage regulation.

3.2 SILICON CONTROLLED RECTIFIER REGULATOR

The basic idea of the Ault Magnetics silicon controlled rectifier regulator bridge rectifier is sketched in Figure 11. This circuit operates the same as the standard bridge rectifier except that the control circuit only allows SCR1 and SCR2 to conduct over part of the cycle, resulting in a rectified output voltage waveform that looks like Figure 12.

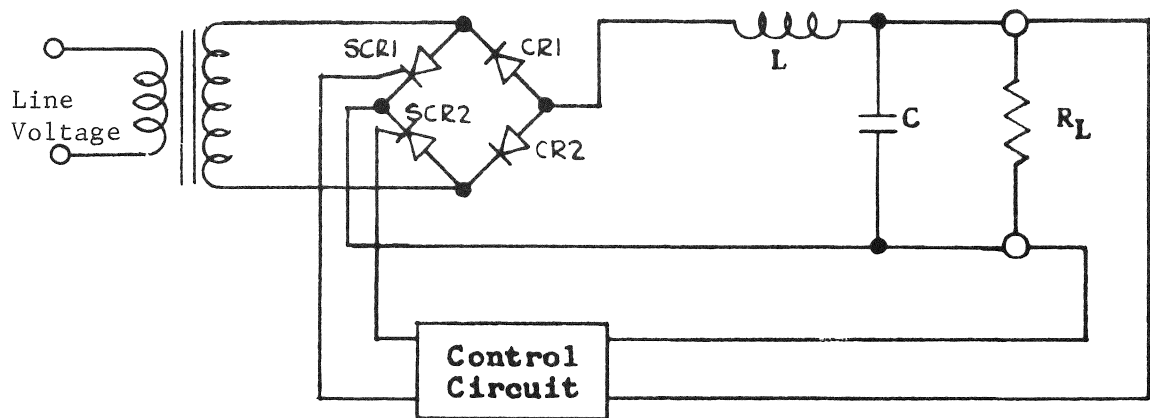


Figure 11 Ault Magnetics Silicon Controlled Rectifier Regulator



Figure 12 Output of Silicon Controlled Rectifier Bridge

The average voltage after filtering the output voltage is again somewhat less than peak depending on load. As the load is increased, the conduction angle is increased by the control circuit resulting in a larger average power handled to compensate for the inherent internal dc voltage decrease with load.

The wider the pulses, the greater the average voltage to overcome the inherent internal voltage decrease due to increased current loading. This voltage is heavily filtered to reduce the ripple from the high frequency components of the pulses.

4.0 DESCRIPTION of the PERIPHERAL EQUIPMENT POWER SUPPLY

The -20 volt and +20 volt supplies and their regulation method are separate, but almost exactly alike. The electronic overload protection circuitry is common to both supplies.

4.1 +20 VOLT POWER CIRCUIT

The power handling circuit of the +20 volt supply is shown in Figure 13.

It can easily be seen that T_2 , CR28, 29, 30, 31, L3, and C18 are exactly the circuit of Figure 11 and the operation is as described in Section 3.2.

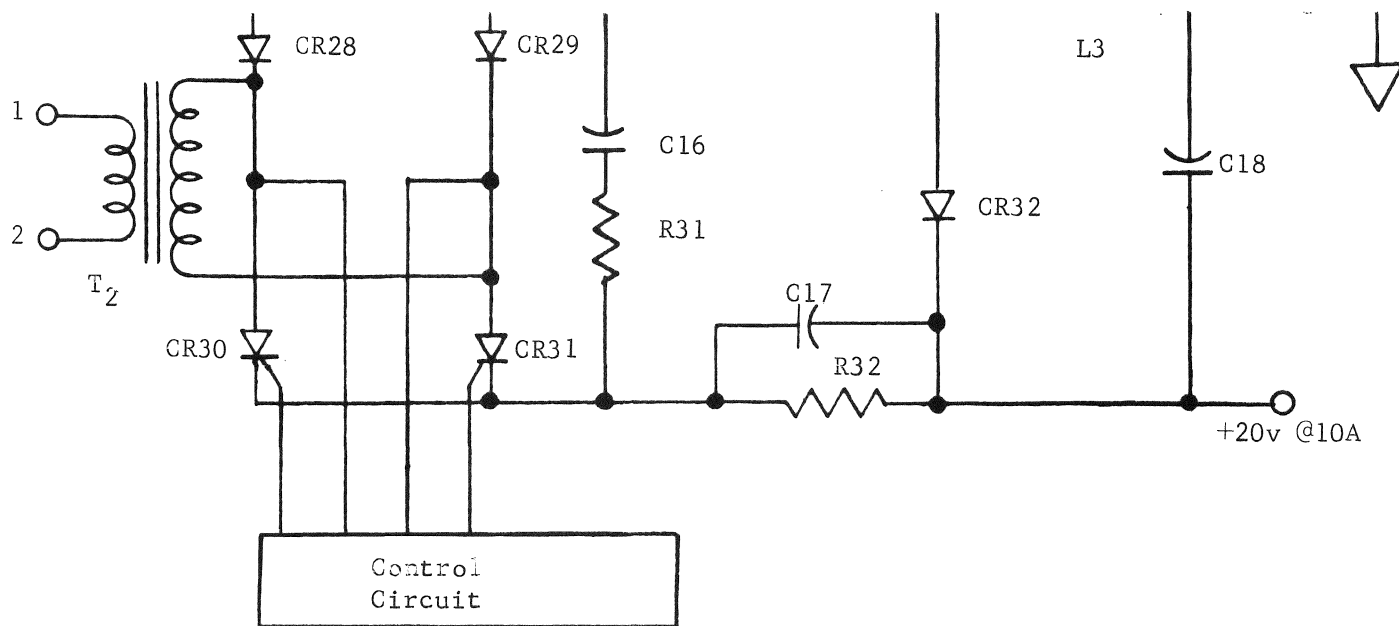


Figure 13 +20 Volt Supply Power Circuit

The damping diode CR32 allows the output current to continue flowing during the off portion of the switching cycle. Since the current thru CR32 is approximately equal to the output current, the forward drop across the diode is approximately 1 volt. This voltage is enough to keep the SCR's from being turned off during a short circuit condition. This fault is corrected by adding C17 and R32. The voltage developed across R32 charges C17 and produces a reverse current pulse, which turns off the SCR's at the end of each cycle.

C16 and R31 protect CR32 against high amplitude voltage spikes.

4.2 -20 VOLT POWER CIRCUIT

The -20 volt power circuit is the same as the +20 volt power section except that a double filter has been employed.

As shown, the output voltage is sensed at the load and applied across R46, R47 and R48. Q12 and Q13 make up a differential amplifier which senses and amplifies any slight difference between the amount of voltage selected by R47 and the amount across the reference CR 41. The output of the differential amplifier is fed through CR40 and then is amplified by Q11. The output of Q11 is then a dc voltage whose magnitude is proportional to the variation between the load voltage and +20 volts.

This output voltage is fed into a miller integrator circuit made up of Q10 amplifier with C19 capacitive feedback. The output of this circuit is a ramp whose slope is proportional to the magnitude of the input voltage or proportional to the error between the load voltage and +20 volts.

The ramp function is fed into the emitter of the unijunction transistor and eventually will reach one-half the voltage applied between base 1 and base 2. At this time, the unijunction transistor emitter junction breaks down and C19 is discharged quickly through Q9, T6 and CR39. This discharge spike is applied to CR47 through pulse transformer T6. CR47 in turn controls the silicon controlled rectifiers in the power control section.

C23, R44 and the extra winding on L3 provides extra stability and hunt free control.

T2, T5, CR45, CR46, CR33, CR34, CR35, CR36, CR37 and CR38 provide power for operation of the control circuit.

The control circuit for the -20 volt section is exactly the same.

4.4 ELECTRONIC OVERLOAD PROTECTION CIRCUITRY

The overload protection circuitry is shown in Figure 15.

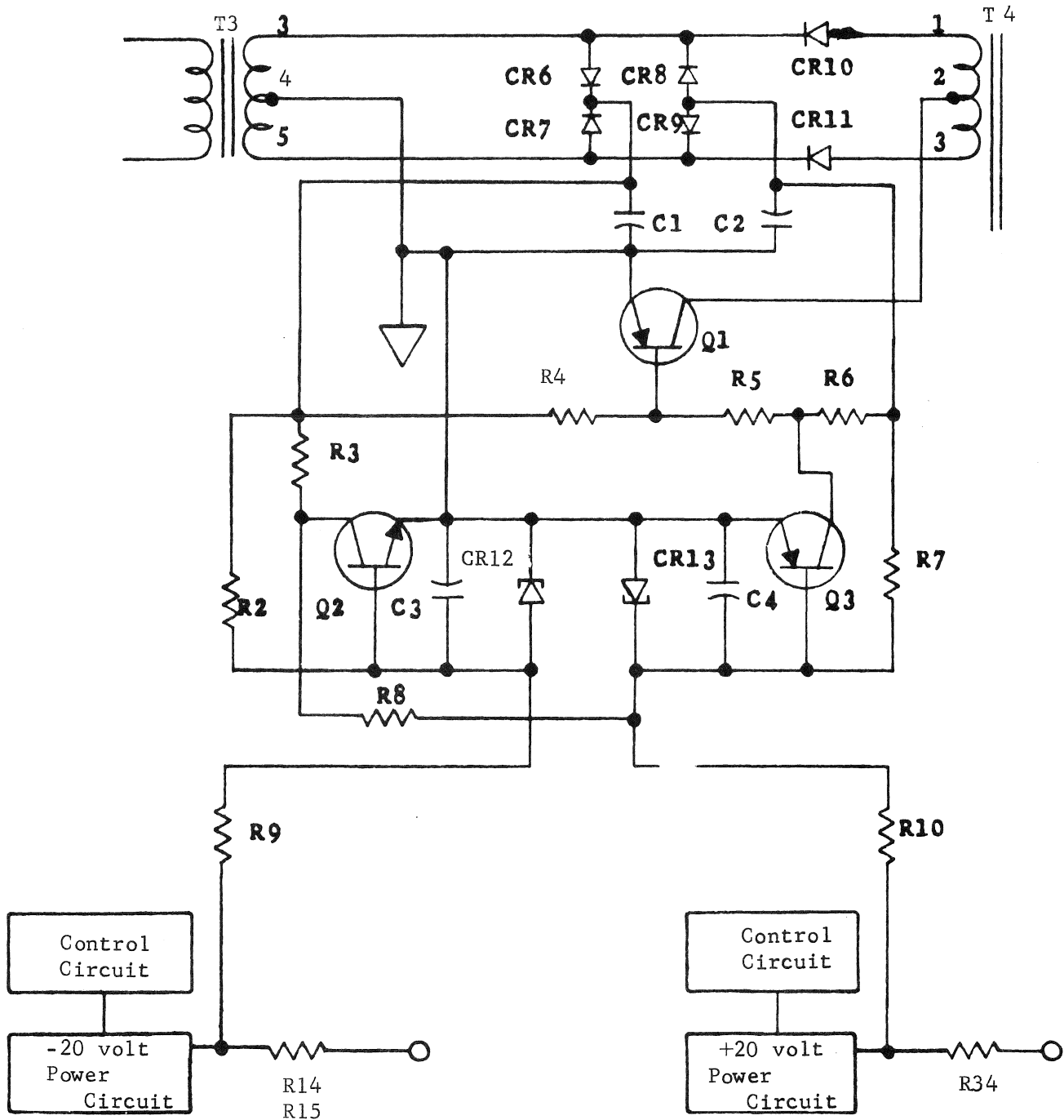


Figure 15 Electronic Overload Protection Circuitry

With the power to the control circuits disconnected, the power silicon controlled rectifiers do not conduct and no power is supplied to the load. Power to the control circuits is supplied through T4 and T5, which in turn is supplied through Q1 from T3.

As current is drawn from the -20 volt supply, a voltage appears across R14 and R15. As the current increases, this voltage increases. This voltage, together with R9, appears as a current source to tunnel diode CR12; the larger the voltage, the larger the current through CR12. As the overload condition is reached, the current through the tunnel diode exceeds I_p and the voltage across the diode goes from E_p to E_f (see Figure 5). E_f from base to emitter of Q2 allows Q2 to conduct, which turns Q3 on. When Q3 is on R5 but no voltage appears across the emitter-base junction of Q1. Q1 then opens up preventing power from reaching the primaries of T4 and T5, and the supply is turned off.

An overload on the +20 volt supply causes a large voltage across R34 which through R10 causes tunnel diode CR13 to switch to E_f turning Q3 on and consequently Q1 off without first switching Q2.

It is important to note that an overload in either supply will turn both off.

5.0 MAINTENANCE

5.1 PREVENTIVE MAINTENANCE AND GENERAL NOTES

Design procedure and part selection has been made to give

trouble-free operation with no routine maintenance required. Use of magnetic and computer grade solid-state components, and a heavy gauge steel chassis has resulted in exceptional ruggedness and a long life history. Computer type environments will not degrade performance in any way.

Information on any maintenance problems which occur should be supplied to Ault Magnetics, Inc. so that improvements may be incorporated in future models of the supply, resulting in a better overall CDC computer system.

5.2 REPLACEMENT PARTS

Replacement parts for the Peripheral Equipment power supply can be purchased directly from Ault Magnetics, Inc. at current net prices. However, since most of the components are standard electronic parts, they generally can be obtained locally in less time than required to obtain them from the factory. The schematic circuit diagram 200267-3, sheet 2, has all component values. Figure 16 shows the location of parts on the printed circuit board.

Special parts such as transformers and chokes are manufactured by, or especially for Ault Magnetics, Inc. and replacements should be ordered from Ault.

Care should be taken that transistors and diodes are not replaced unless they are actually causing a circuit malfunction. When soldering to transistor and diode leads, care should be exercised to prevent destruction of junctions by the heat.

5.3 INITIAL TROUBLESHOOTING PROCEDURE

Check the line voltage across terminals 1 and 2.

Give the supply a complete visual examination, especially looking for burned or damaged parts, foreign matter, or loose leads on the terminal strips.

5.4 ISOLATING TROUBLE

5.4.1 Instant Dropout

If the supply cannot be engaged without an instant dropout, try connecting it to the line through a variac and engaging with approximately 50v rms input.

5.4.2 No Output Voltage From Either Supply

No output voltage from either supply may be caused by the overload circuitry dropping out the supplies because of a heavy load in the load equipment. Disconnect the output voltage leads and put on a small external load of approximately 8 ohms, between terminals 5 and 4. Also, connect terminals 8 to 5 and 10 to 4.

Energize the supply.

If +20 volts appears between terminals 4 and 5, repeat the procedure between terminals 3 and 4 with terminals 9 connected to 3 and 10 connected to 4. If -20 volts appears when the supply is energized, check the Peripheral load equipment for a short circuit or load less than 1 ohm for the -20 volt supply, or less than 2 ohms for the +20 volt supply.

If both voltages continue to drop out with external test loads, the problem is most likely located in the overload circuit, but could also be an open zener reference diode or control circuit failure.

5.4.3 No Output Voltage from One Supply

If the supply is energized and ac voltage appears across lines 4 and 5 of transformer T4, but there is no output voltage from one of the supplies, then the power rectifiers or silicon controlled rectifiers of the off channel should be suspected and checked.

5.4.4 Wrong Output Voltage or Poor Regulation

Vary R47 or R27 depending on which supply is not regulating properly. If adjusting this potentiometer does not correct the difficulty, then the problem must be in the control circuitry.

5.4.5 Excessive Ripple

High ripple voltages are probably caused by the filtering network. Check capacitors for high leakage.

5.5 TROUBLE SHOOTING

5.5.1 Electronic Overload Circuit

A variable dc voltage from 0 to 1.5 volts may be applied from ground through R9 or R10 to CR12 and CR13 to check operation of this circuit. A dc scale oscilloscope across the tunnel diodes will confirm that switching occurs. If the tunnel diodes switch properly, check transistor switching

by measuring voltage from collector to emitter.

5.5.2 Control Circuit Troubleshooting

By using the oscilloscope patterns provided in this manual, check for proper triangular wave pulse when an error is introduced by varying R27 or R47, depending on the channel involved. If no triangular pulse is observed, then the problem is between Q10 and R47 or Q5 and R27, depending on which supply is involved.

If the proper ramp is observed, check the pulse out of the pulse transformer. If this is OK according to the pattern provided, check the output of CR47 or CR44 against the pattern provided.

If this checks, examine the patterns across the power silicon controlled rectifiers and the pattern out of the rectifier bridge and compare to the patterns included with this manual.

5.6 FINAL CHECK

After the supply has been repaired, test operation at full load by placing a load of 20 amperes (1 ohm) on the -20 volt output and 10 amperes (2 ohms) on the +20 volt output. Be sure that the voltage level controls R27 and R47 are properly set.

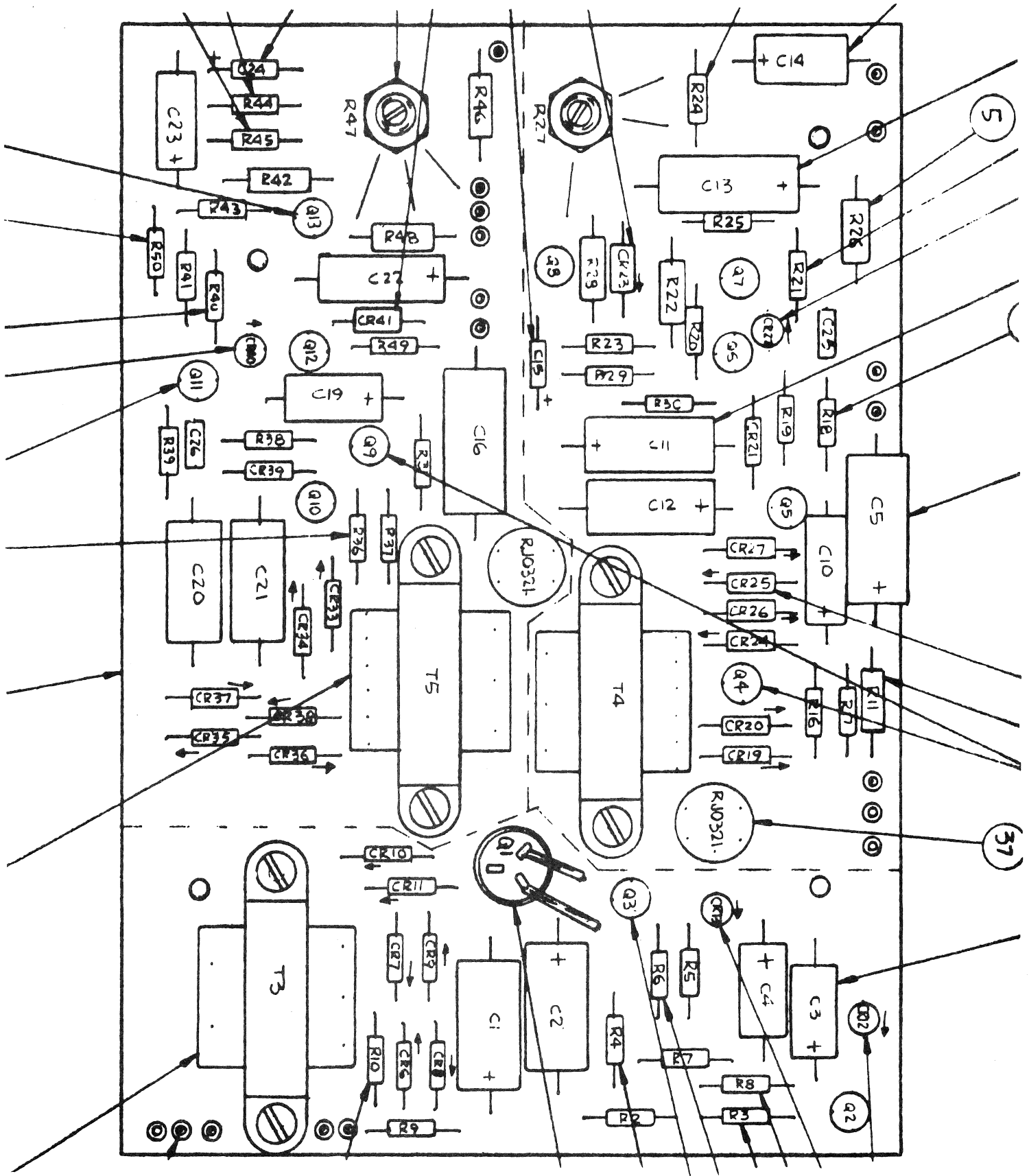
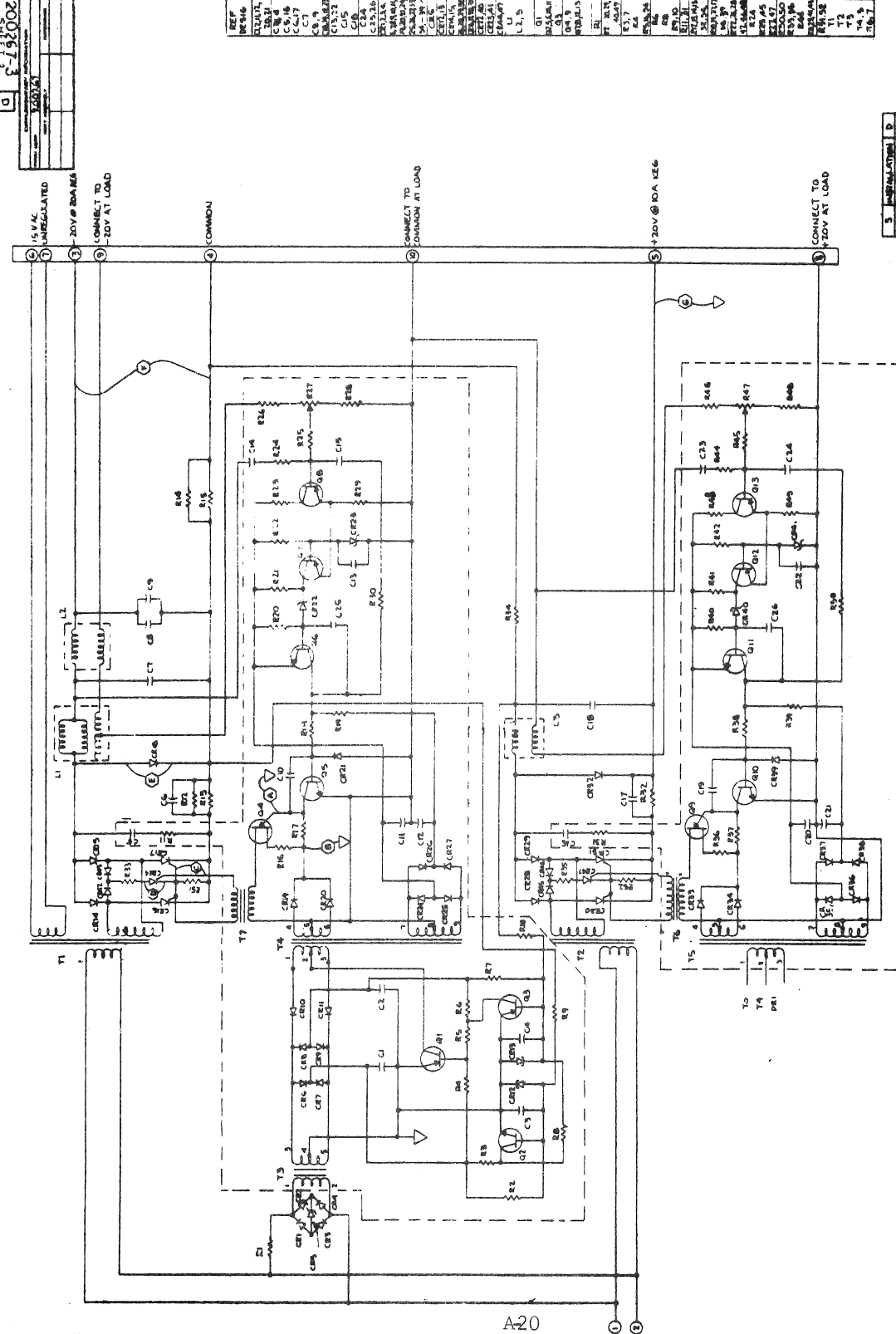


Figure 16 Location of Parts on Printed Circuit Board

| REF DESIG | DESCRIPTION | TYPE NO. | AULT PART NO. |
|--|---------------------------------------|-------------|------------------|
| C1,2,11,12, 20,21 | CAP-ELECT 100/25V | | BCA00108-2 |
| C3,4 | CAP-ELECT 100/3V | | BCA00162-2 |
| C5,16 | CAP-METALLIC PAPER 1/200V | | BCA00163-1 |
| C6,17 | CAP-ELECT 6000/15V | | CCA00031-4 |
| C7 | CAP-ELECT 15000/25V | | CCA00051-2 |
| C8,9 | CAP-ELECT 5000/30V | | CCA00031-5 |
| C10,14,19,23 | CAP-METALLIC PAPER .1/200V | | BCA00163-3 |
| C13,22 | CAP-ELECT 200/15V | | BCA00162-1 |
| C15 | CAP-TANTALIC 3.3/35V | | BCA00318-2 |
| C18 | CAP-ELECT 25000/30V | | CCA00031-1 |
| C24 | CAP-TANTALIC 1/35V | | BCA00318-1 |
| C25,26 | CAP-DISK CERAMIC 500UUF/1KV | | BCA00285-2 |
| CR1,2,3,4, 6,7,8,9,10,11, 19,20,21,24, 25,26,27,33, 34,-39 | CR42,43,45,46 ALSO SILICON DIODE - | D1-42 | BSC00310-1 |
| CR5 | ZENER DIODE - | DM5625 | |
| CR12,13 | TUNNEL DIODE - | TD3 | BSC00167-1 |
| CR14,15, 18,28,29,32 | SILICON RECTIFIER | 1N2156 | BSC00095-2 |
| CR16,17,30,31 | SILICON CONTROL RECTIFIER | 2N683 | BSC00144-1 |
| CR22,40 | ZENER DIODE - | R56 | BSC00279-1 |
| CR23,41 | ZENER DIODE - | 1N2163 | BSC00107-1 |
| CR44,47 | SILICON CONTROL RECTIFIER | 2N1597 | BSC00328-1 |
| L1 | FILTER CHOKE - 0285 | | CCH00331 |
| L2,3 | FILTER CHOKE 0198 | | CCH00221 |
| Q1 | TRANSISTOR-POWER | 2N1502 | BSC00098-2 |
| Q2,5,6,10,11 | TRANSISTOR-GERM. | 2N1302 | BSC00166-1 |
| Q3 | TRANSISTOR-GERM | 2N1303 | BSC00166-2 |
| Q4,9 | TRANSISTOR-UNIJUNCTION | 2N1671A | BSC00281-1 |
| Q7,8,12,13 | TRANSISTOR-GERM | 2N1193 | BSC00173-2 |
| R1 | RESISTOR-W.W. 20W 1K | | BRE00114-1K |
| R2 20,29, 40,49 | RESISTOR-COMP 1/2W 5.1K | | BRE00116-5.1K |
| R3,7 | RESISTOR-COMP 1/2W 2.4K | | BRE00116-2.4K |
| R4 | RESISTOR-COMP 1/2W 6.8K | | BRE00116-6.8K |
| R5,16,36 | RESISTOR-COMP 1/2W 470Ω | | BRE00116-470 |
| R6 | RESISTOR-COMP 1/2W 1K | | BRE00116-1K |
| R8 | RESISTOR-COMP 1/2W 2.7K | | BRE00116-2.7K |
| R9,10 | RESISTOR-COMP 1/2W x Ω | | BRE00116-x |
| R11,31 | RESISTOR-COMP 1W 22Ω | | BRE00115-22 |
| R12,13,14,15, 32,34 | RESISTOR-W.W. 25W .1Ω | | BRE02170-Q1 |
| R18,19,37,17 38,39 | RESISTOR-COMP 1/2W 10K | | BRE00116-10K |
| R22,26,28 42,46,48 | RESISTOR-METALLIC FILM 1W 1K | | BRE02268-1K |
| R24 | RESISTOR-COMP 1/2W 33K | | BRE00116-33K |
| R25,45 | RESISTOR-COMP 1/2W 39K | | BRE00116-39K |
| R27,47 | RESISTOR-VARIABLE 2W 200Ω | | BRE00111-200 |
| R30,50 | RESISTOR-COMP 1/2W 3K | | BRE00116-3K |
| R33,35 | RESISTOR-COMP 1/2W 100Ω | | BRE00116-100 |
| R44 | RESISTOR-COMP 1/2W 2.2K | | BRE00116-2.2K |
| R23,41,43 | RESISTOR-COMP 1/2W 6.2K | | BRE00116-6.2K |
| R51,52 | RESISTOR-COMP 1/2W 47Ω | | BRE00116-47 |
| T1 | TRANSFORMER-POWER | 0290 | CTFO0329 |
| T2 | TRANSFORMER-POWER | 0291 | CTFO0330 |
| T3 | TRANSFORMER- | 0139 | CTFO0157 |
| T4,5 | TRANSFORMER- | 0271 | CTFO0305 |
| T6,7 | PULSE TRANSFORMER | | RJ0321 |

200267-3
SHEET 3

| | |
|---|---------------------------|
| 1 | 15 VAC UNREGULATED |
| 2 | 20V @ 200 MA |
| 3 | CONNECT TO COMMON AT LOAD |
| 4 | COMMON |
| 5 | CONNECT TO COMMON AT LOAD |
| 6 | 20V @ 10A @ 256 |
| 7 | CONNECT TO COMMON AT LOAD |



A20

| REF | DESCRIPTION | TYPE | QTY |
|-----|---------------------|--------------|-----|
| R1 | RESISTOR - 10K | RES-10K | 1 |
| R2 | RESISTOR - 10K | RES-10K | 1 |
| R3 | RESISTOR - 10K | RES-10K | 1 |
| R4 | RESISTOR - 10K | RES-10K | 1 |
| R5 | RESISTOR - 10K | RES-10K | 1 |
| R6 | RESISTOR - 10K | RES-10K | 1 |
| R7 | RESISTOR - 10K | RES-10K | 1 |
| R8 | RESISTOR - 10K | RES-10K | 1 |
| R9 | RESISTOR - 10K | RES-10K | 1 |
| R10 | RESISTOR - 10K | RES-10K | 1 |
| R11 | RESISTOR - 10K | RES-10K | 1 |
| R12 | RESISTOR - 10K | RES-10K | 1 |
| R13 | RESISTOR - 10K | RES-10K | 1 |
| R14 | RESISTOR - 10K | RES-10K | 1 |
| R15 | RESISTOR - 10K | RES-10K | 1 |
| R16 | RESISTOR - 10K | RES-10K | 1 |
| R17 | RESISTOR - 10K | RES-10K | 1 |
| R18 | RESISTOR - 10K | RES-10K | 1 |
| R19 | RESISTOR - 10K | RES-10K | 1 |
| R20 | RESISTOR - 10K | RES-10K | 1 |
| R21 | RESISTOR - 10K | RES-10K | 1 |
| R22 | RESISTOR - 10K | RES-10K | 1 |
| R23 | RESISTOR - 10K | RES-10K | 1 |
| R24 | RESISTOR - 10K | RES-10K | 1 |
| R25 | RESISTOR - 10K | RES-10K | 1 |
| R26 | RESISTOR - 10K | RES-10K | 1 |
| R27 | RESISTOR - 10K | RES-10K | 1 |
| R28 | RESISTOR - 10K | RES-10K | 1 |
| C1 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C2 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C3 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C4 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C5 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C6 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C7 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C8 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C9 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C10 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C11 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C12 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C13 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C14 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C15 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C16 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C17 | CAPACITOR - 100UF | CAP-100UF | 1 |
| C18 | CAPACITOR - 100UF | CAP-100UF | 1 |
| D1 | DIODE - 1N4001 | DIODE-1N4001 | 4 |
| D2 | DIODE - 1N4001 | DIODE-1N4001 | 4 |
| D3 | DIODE - 1N4001 | DIODE-1N4001 | 4 |
| D4 | DIODE - 1N4001 | DIODE-1N4001 | 4 |
| Q1 | TRANSISTOR - 2N3055 | TRANS-2N3055 | 1 |
| Q2 | TRANSISTOR - 2N3055 | TRANS-2N3055 | 1 |
| Q3 | TRANSISTOR - 2N3055 | TRANS-2N3055 | 1 |
| Q4 | TRANSISTOR - 2N3055 | TRANS-2N3055 | 1 |

| | |
|----|---------------------------|
| 5 | UNREGULATED |
| 6 | 20V @ 200 MA |
| 7 | CONNECT TO COMMON AT LOAD |
| 8 | COMMON |
| 9 | CONNECT TO COMMON AT LOAD |
| 10 | 20V @ 10A @ 256 |
| 11 | CONNECT TO COMMON AT LOAD |

200267-3
SHEET 3

SCHEMATIC DIAGRAM
PERIPHERAL
EQUIPMENT
POWER SUPPLY

200267-3
SHEET 3

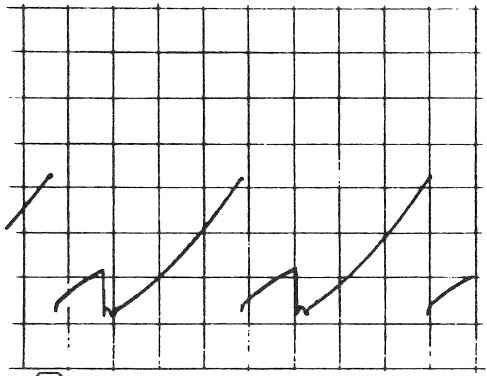
1-1 THIS SYMBOL REFERS TO WAVE SHAPES; SEE DWG # 200267-3, SHEET 4.

SHEET 4
200267-3

REVISIONS

| SYM | DESCRIPTION | DATE | APPROVAL |
|-----|-------------|------|----------|
| | | | |

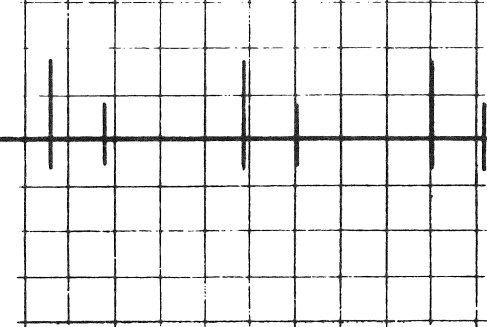
| SUPPLEMENTARY INFORMATION | |
|---------------------------|-----------|
| FIRST USED | 200267-3 |
| NEXT ASSEMBLY | REFERENCE |
| | |



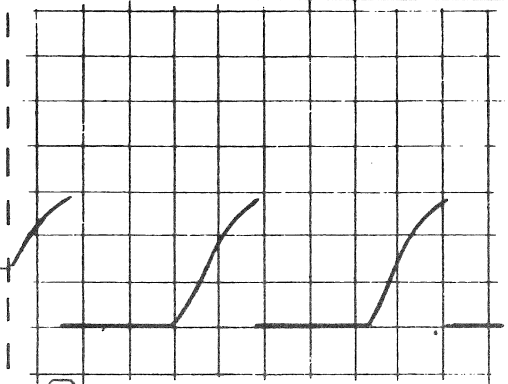
A UNIJUNCTION EMITTER
HORIZONTAL 2MS/CM
VERTICAL 5V/CM



B INPUT TO UNIJUNCTION
JUNCTION R16&R17 TO GND
HORIZONTAL 2 MS / CM
VERTICAL 10V / CM



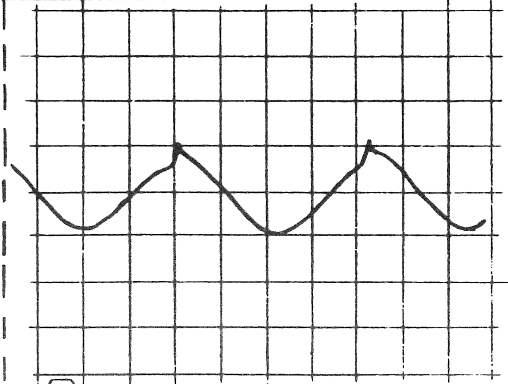
C CR44 GATE TO CATHODE
HORIZONTAL 2MS / CM
VERTICAL 1 V / CM



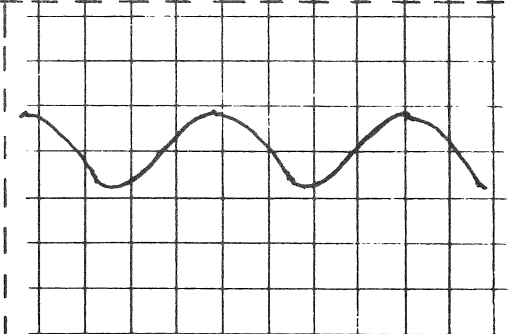
D CR44 ANODE TO CATHODE
HORIZONTAL 2 MS / CM
VERTICAL 20 V / CM



E OUTPUT OF POWER BRIDGE
HORIZONTAL 2 MS / CM
VERTICAL 20 V / CM



F - 20 VOLT SUPPLY RIPPLE
HORIZONTAL 2 MS / CM
VERTICAL 20 MV / CM



G +20 VOLT SUPPLY RIPPLE
HORIZONTAL 2 MS / CM
VERTICAL 100 MV / CM

3--WHEN +20 VOLT SUPPLY IS LOADED TO 5 AMPS (2 OHM) OSCILLOSCOPE PATTERNS ARE SIMILAR TO - 20 VOLT SUPPLY.

2--20 VOLT SUPPLY LOADED TO 10 AMPS (2 OHMS) DURING OSCILLOSCOPE PATTERN MEASUREMENTS.

1--REFER TO SCHEMATIC DIAG. 200267-3, SHEET 2 FOR TEST POINT LOCATIONS.

| TOLERANCE UNLESS NOTED | | | DRAWN BY | |
|------------------------|--------|------------|-------------------|---------|
| ONE PLACE | (.0) | ±.020 | RAJ | 3-21-62 |
| TWO PLACE | (.00) | ±.018 | DRAFTING | ✓ |
| THREE PLACE | (.000) | ±.005 | MATL & FIN. | ✓ |
| 90° FORMED ANGLES | | +2° -1° | DESIGNER | ✓ |
| DEV | 2212 | | DESIGN | ✓ |
| SUPSD | | | APPROVED | |
| GOVT MATL SPEC | | | REF MATL (SPEC) | |
| | | | FINISH (SEE NOTE) | |

WAVE SHAPES -
PERIPHERAL EQUIP.
POWER SUPPLY

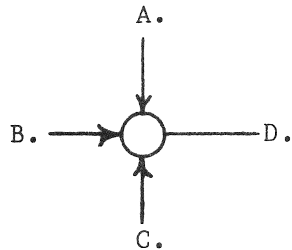
FINISH FULL SIZE WT ACT. CALC

AULT
MAGNETICS
MINNEAPOLIS, MINNESOTA

200267-3
SHEET 4

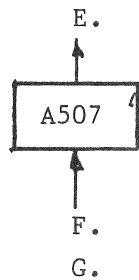
CONTROL DATA LOGICAL SYMBOLS

AND FUNCTION



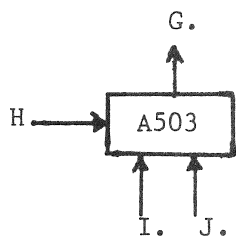
It takes A AND B AND C to get a "one" at point D.

INVERTER



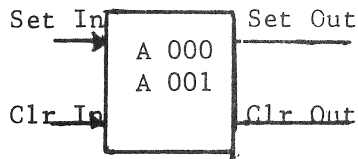
A "one" at point F will give a "zero" at point E.

OR



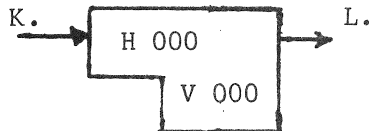
A "one" at point H OR I OR J will give a "zero" at G.

FLIP-FLOP



The set in feeds the even term, the clear in feeds the odd term, the set out comes from the odd term and the clear out comes from the even term.

CONTROL DELAY



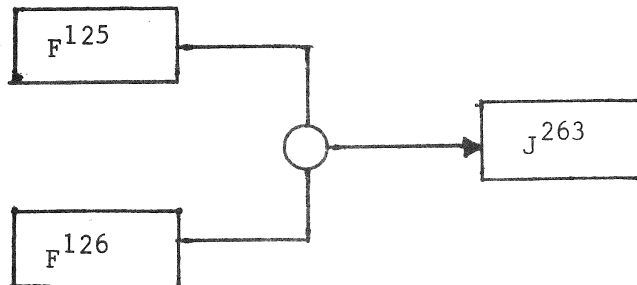
A "one" at point K will result in a "one" at point L one phase time later. The V term is the output.

The MASTER clock of the machine runs at 2.5 Meg/Cycles, which is effectively doubled, giving 5 Meg/Cycles

ASSIGNMENT SHEET

CARD TYPES

1. How is it possible to "AND" more than two outputs together when it is possible to make only two physical connections to a pin on a Buggie?
2. How would the circuit be affected if CR08 opened?
3. Referring to the following diagram, what would you expect to happen to the output of J263 if the output diode of F125 were to become open?



4. What would happen if CR01 in an 11 card opened?
5. What would be the outputs of a 31 card if, due to a malfunction, both inputs become "1" at the same time?
6. Referring to the schematic of card type 33, which pins would be "set" input pins, and which would be "set" output pins?

INFORMATION SHEET

"L" AND "M" CARDS

Many conditions are present, both in and out of the computer, that make it necessary to work with logic levels other than -3v and -.5v. The circuits designed to work with these other levels of voltage are the L and M cards.

The "L" and "M" cards have been designed to permit more efficient transfer of data over the input and output cables, and are used wherever it is necessary to convert one set of voltage levels to another. Both of these cards are similar in purpose although not in operation. The similarity exists in the fact that both will convert one set of voltage levels to another.

"L", or output, cards are used to convert C.D.C.'s logic values of -3v ("1") and -.5v ("0") to 0v and -16v respectively before sending this data out to an externally connected piece of equipment. This must be done to reduce the possibility of any noise pulses induced in the cables from being recognized as logic values. A secondary purpose is to insure that due to IR losses in the cables and connectors, an appreciable difference in voltage remains between the two logic values at the ends of the cables.

The main difference that exists between an "L" card and a standard inverter is in the size of the components used in their respective inverter circuits, although the physical placement of these components are primarily the same as in the standard inverter. Another difference is that the "L" card contains no cutoff protection since the output transistor will be driven to cut off to produce the required output voltage.

The output diode, normally existing at the outputs of standard inverters, have also been removed. This is because the output of an "L" card need never be "anded" with another term.

Figure 1 shows the outputs that will be obtained from an "L" card with either "1" or "0" in.

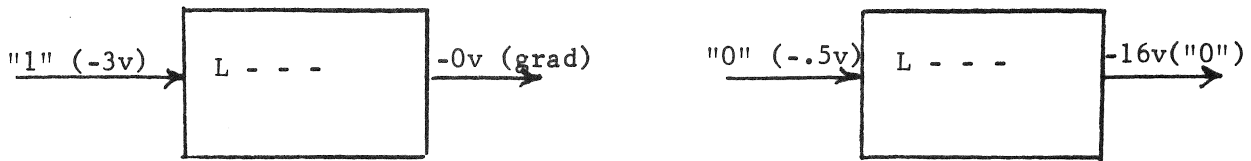


Figure 1. "L" Card

Referring to the above diagram, we can see that whenever the computer sends a binary 1 to an output cable it will appear on the cable as 0v. A binary (logical) 0 will appear as -16v. As a result the presence of 0v on a cable is construed to be a "1" on the line while -16v represents a "0". Therefore, "1" in produces "1" out, while "0" in produces "0" out.

This failure to invert logic causes no problem, however, since the input cables all terminate in "M" cards. An "M" card is designed to convert this -16v and 0v back to conventional C.D.C. logic values. See Figure 2.

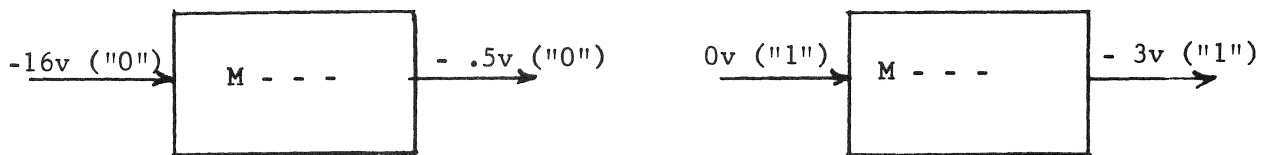


Figure 2. "M" Card

Figure 3 shows the correlation between "L" and "M" cards used on input-output cables.

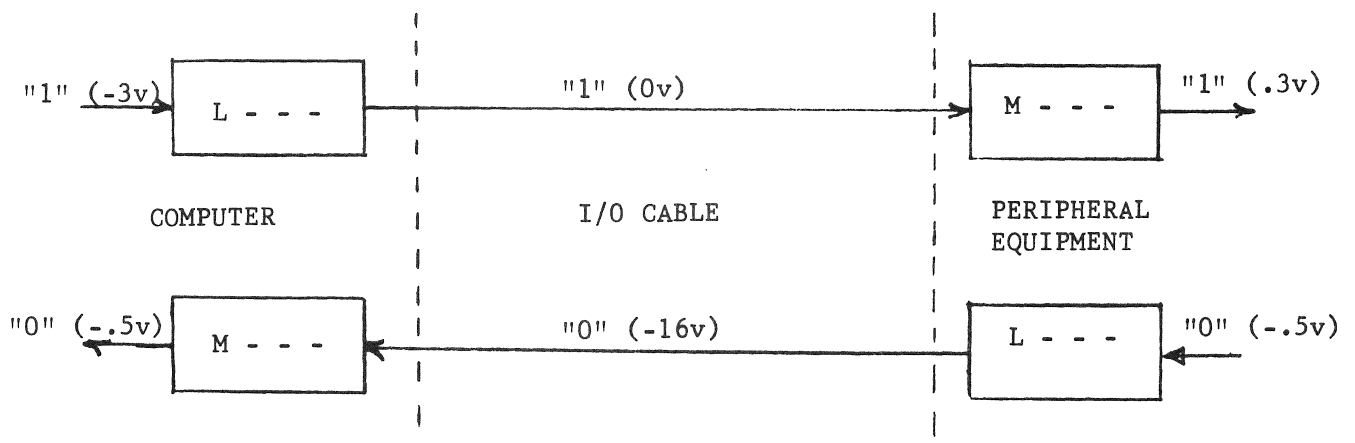


Figure 3. B-5

This use of "L" and "M" cards is not limited to input-output cables. They are used any time voltage levels other than -0.5v and -3v are required. There are two instances where this occurs. These two instances are:

The manual switches and push buttons on the console are devices that will apply a ground potential to a certain point in the 160-A. These points are the input pins of "M" cards. The "M" card then, will convert this ground (0v) to a C.D.C. logical "1" (-3v). With the switch or push button open, the input to the "M" card will be an effective -16v causing "0" (-0.5v) out. An example is shown in Figure 4.



Figure 4.

All visual displays on the 160-A console are controlled by D-C relays. The voltage necessary to energize these relays exceeds -3v . Therefore, "L" cards are always used to allow C.D.C. logic levels to produce visual indications on the console. Figure 5 gives an example of this.

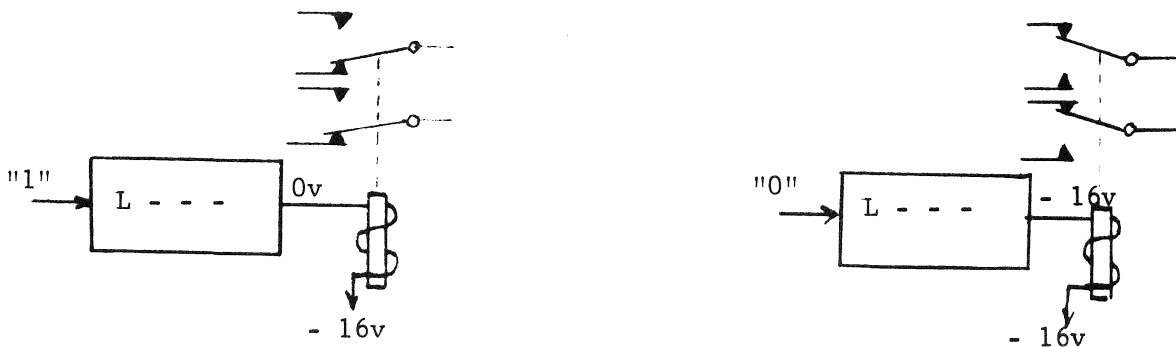


Figure 5.

Before our discussion of these input and output cards ends, let us add a few other pertinent facts regarding these cards.

1. Because of the greater voltage change which must occur in an "L" or "M" card is greater than that of a standard inverter, the switching time is longer. It is approximately 1 to 2 useconds per card.

2. Each "L" card contains 3 inverter circuits.

3. Each "M" card contains 3 inverter circuits.

4. Test points are A, B, and C.

5. "L" cards have only one usable output per inverter.

6. "M" cards have only one input per inverter.

7. "L" and "M" cards are numbered 6X.

ASSIGNMENT SHEET

"L" AND "M" CARDS

1. What would be felt on an output line if pin 13 of the "L" card feeding this line were open?

2. What would be the result if an input diode in an "L" card became open?

3. What would be the result if pin 1 of an "L" card were to become open?

4. What would be the result if Q01 in an "M" card were to short (emitter to collector)?

INFORMATION SHEET

FILE OF EQUATIONS

The file of equations is a list of Boolean expressions that represent all logic within the 160-A. Inputs and outputs to each term are listed showing their sources and terminations. All "and" functions and "or" functions are shown, along with the physical location and purpose of each term. In addition, the card type, upon which the subject term is located, is also indicated. You can see that the file of equations then contains the complete logic of the 160-A. The file of equations may rightly be called the ultimate source of information regarding all logic drawings, sequence charts, and tables of execution times. The ability to utilize the file of equations correctly and efficiently is one of the greatest aids you will have in performing corrective maintenance on the 160-A.

It cannot be overemphasized that a master of the file of equations is of the utmost importance to you. Vol. IV is the file of equations for this machine. The first four pages of this manual are devoted to explaining the contents and the usage of the rest of the manual. The clarification of certain points, and the introduction of some omitted information, is the purpose of this sheet.

The first part of a logical equation consists of a Subject Symbol (this is the term to be discussed) followed by an equation setting this symbol equal to a group (or groups) of other terms. Example: $J^{405} = K^{405} + K^{522} + K^{532} + K^{406} J^{923}$. The subject symbol in this case is J^{405} . This equation represents the logic shown in Figure 1.

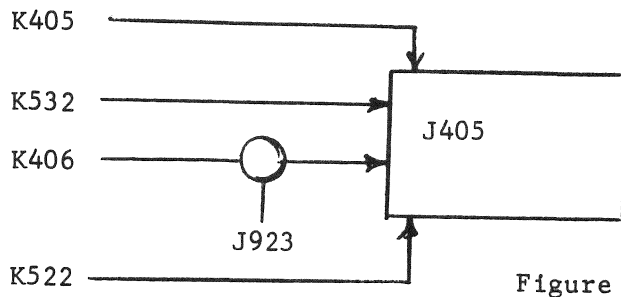


Figure 1

The term Y^{999} if it appears in a logical equation, is used to designate electrical ground. Example: $X^{123} = Y^{999} + A^{000}$ means:

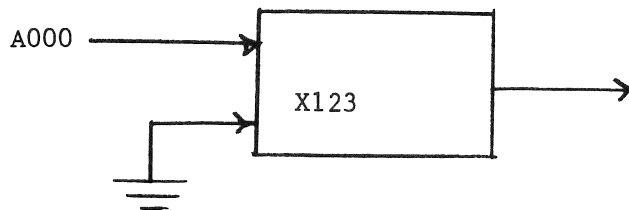


Figure 2

The first group of numbers and letters in the second line indicates the physical location of the card containing the subject term:

Example: $J^{405} = K^{405} + K^{522} K^{532} + K^{406} J^{923}$
 1E043B Z⁰⁰⁰ Z⁰¹⁰ Z⁰²⁰ Z⁰³⁰ Z⁰⁴⁰ Z⁰⁵⁰

In the example above, 1E043B shows where the card containing J^{405} is located. The first number (1) indicates the chassis on which the card is located. There are seven different chassis in the 160-A. They are as follows:

- a. Computer logic chassis which is numbered 10100.
- b. Console and relay chassis which is numbered 10200.
- c. Punch control unit chassis, numbered 10400.
- d. Power Supply chassis, numbered 10500.
- e. Punch logic unit, numbered 10600.
- f. Photoelectric Reader, numbered 10700.
- g. High Speed Punch, numbered 10800.

The only difference that exists in chassis numbers is in the third digit. i.e., 1, 2, 4, 5, 6, 7, and 8. It is this digit alone that is used to indicate the chassis number. J^{405} then, is on a card mounted on the Computer logic chassis (10100).

The next letter and 3 numbers (in the example "E043") show the vertical and horizontal coordinates at which the card is located. Every chassis containing printed circuit cards has these cards arranged in horizontal rows. Each row is given an identifying letter starting with "A". For instance, the computer logic chassis contains 6 horizontal rows of cards, A, B, C, D, E, and F. Therefore, J^{405} must be located in the horizontal row labeled "E".

Each card position in every row is numbered from left to right in ascending order starting with 01. The computer logic chassis has card positions numbered from 01 to 130. J^{405} is the 43rd card from the left in the row labeled "E".

The number of outputs available from a card always equals 12 minus the number of inputs (up to a maximum of 8). Examples: a 13 card has 3 inputs. $12 - 3 = 9$, but we know only eight will be used.

A 16 card has 6 inputs. $12 - 6 = 6$. Therefore 6 outputs will be available on this card, although not all must be used.

When a card contains two circuits (2X or 3X card) the usable 12 pins are divided to allow the "A" circuit to use pins 1 through 6, and the "B" circuit pins 7 through 12. In this case the number of outputs from one section equals 6 minus the number of inputs to that section.

In referring back to our example, the card in this position (1E043) may contain two or more inverter circuits. The last letter in our example (B) tells us that the card contains only one circuit and one test point. Let us elaborate somewhat on this last point.

As you already know, each separate circuit on a card contains its own test point. These test points are labeled in a definite manner; i.e., the number of circuits on a card can be determined by knowing one term number and the associated test point label. Following are the rules that govern the naming of test points:

- a. Any card containing only one test point will have that test point labeled "B".
- b. Any card containing two test points will have them labeled "A" and "C".
- c. A card containing more than two test points has them labeled in order, i.e., "A", "B", "C", "D".)

Now then, let us go a step further with our reasoning. Some of the cards containing more than two circuits are the 57, 61, 67, 75, and 86 cards. These cards are all "special cards" insomuch as they do not work strictly with C.D.C. logic levels (-0.5v and -3.0v). Standard logic cards never have more than 2 test points and therefore must contain either a single "B" point or both an "A" and "C" point.

Card Placement

As shown above, the card type, number of circuits on the card, and test points may be found by using the section of the file of equations showing card placement. (Refer to pages following page 4 in Vol. IV). We find from this that in card location E43 of the computer logic chassis (main logic chassis) is J⁴⁰⁵, that the card type is 14 and that no other circuits exist on that card.

The terms listed in the file of equations are in Alpha-numeric order making it quite easy to find the desired term. Also given above each term is the purpose of the term, such as A⁰⁰⁰ through A¹¹³. These are all terms associated with the "A" register.

Pin Assignments

Each printed circuit card contains 15 pins. Three of these are reserved for +2-v, -20v and ground. The remaining 12 pins may be used for inputs and outputs. The maximum number of outputs available from any card may never exceed eight. Regardless of how many outputs are available, pin 12 is always the first output, followed by 11, then 10, etc.

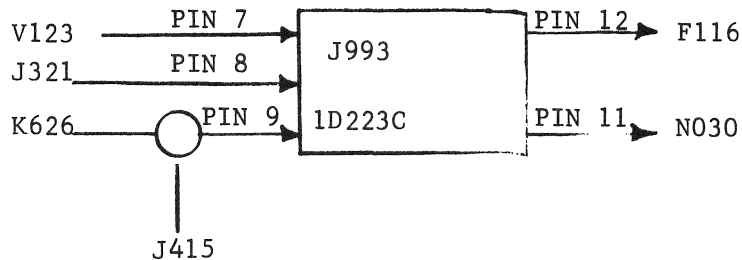
The inputs, on the other hand, always start on pin 1 and progress in ascending order, i.e., 1, 2, 3, etc.

ASSIGNMENT SHEET

FILE OF EQUATIONS

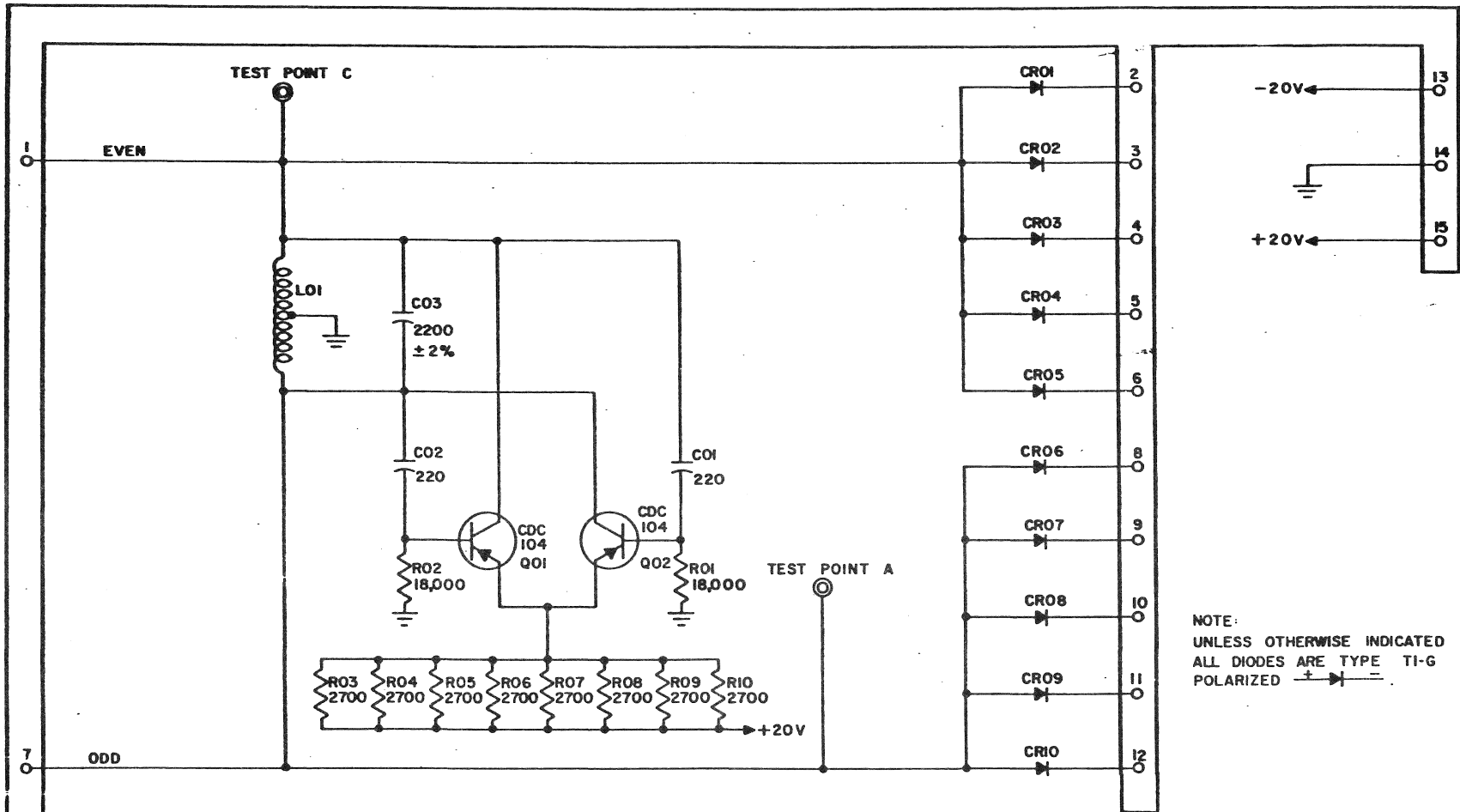
1. Using the file of equations for the 160-A (160-A, Vol. IV), draw the logic diagram showing inputs and outputs for E303. Label pin numbers and give the card location and type of card.

2. Write the logical equation for the following diagram.

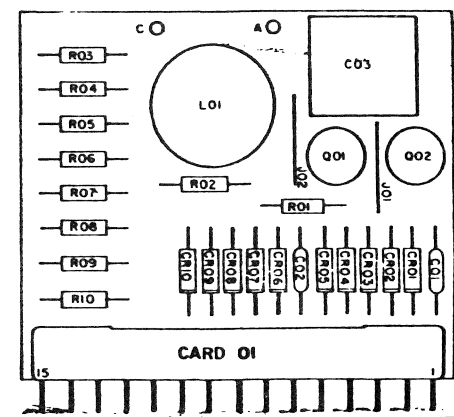


3. Assume that you are performing Corrective Maintenance on a 160-A and notice that the wire from pin 12 of F405 is not physically connected to pin 1 of W090. Instead, you find that it is connected to pin 3 of F532. Would this cause an error during operation? Is this logical error? If so, why? If not, why not? (Refer to Vo. IV).

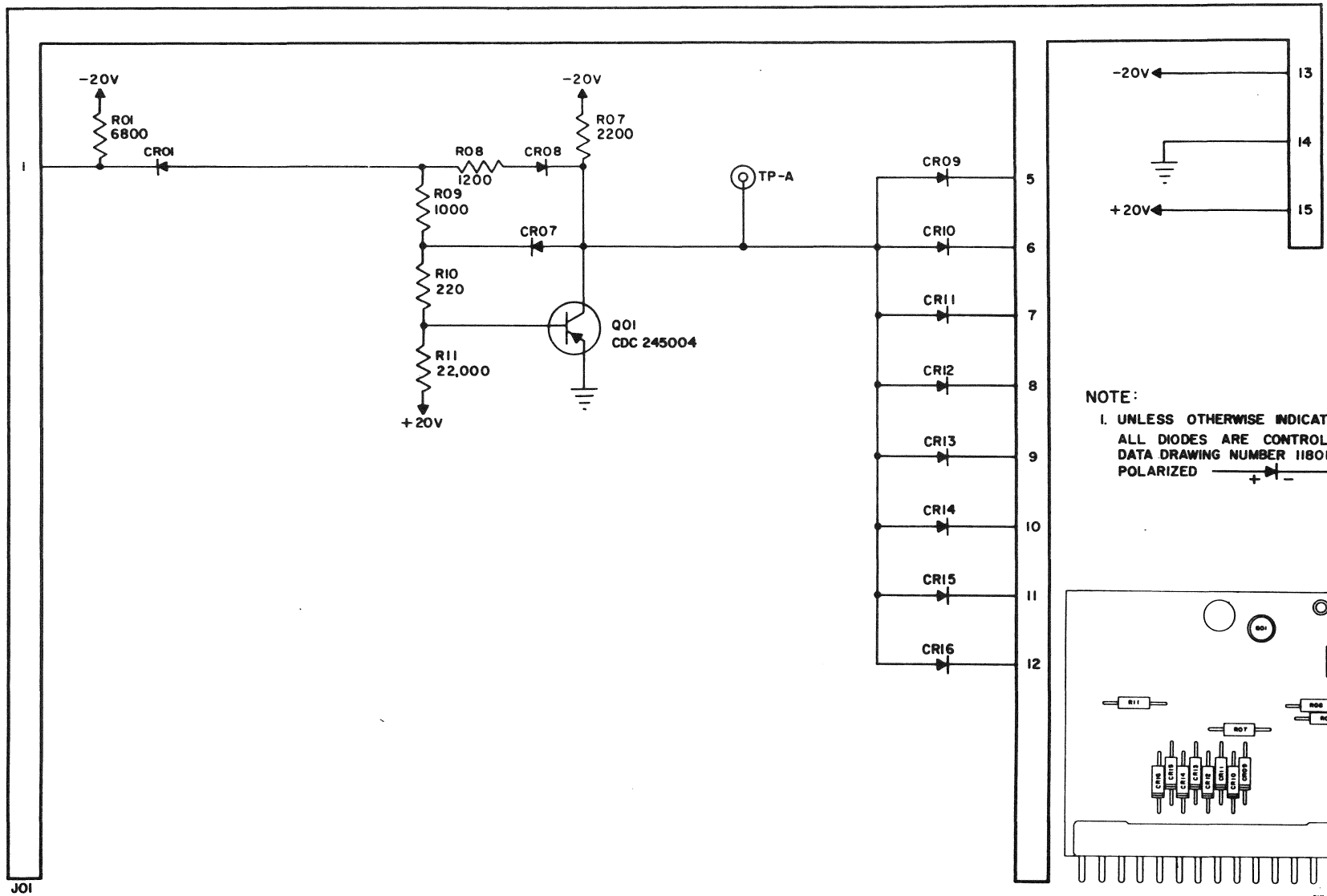
Oscillator 01
B-14

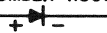


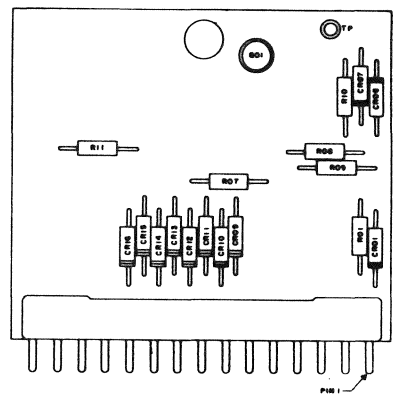
NOTE:
UNLESS OTHERWISE INDICATED
ALL DIODES ARE TYPE TI-G
POLARIZED \rightarrow

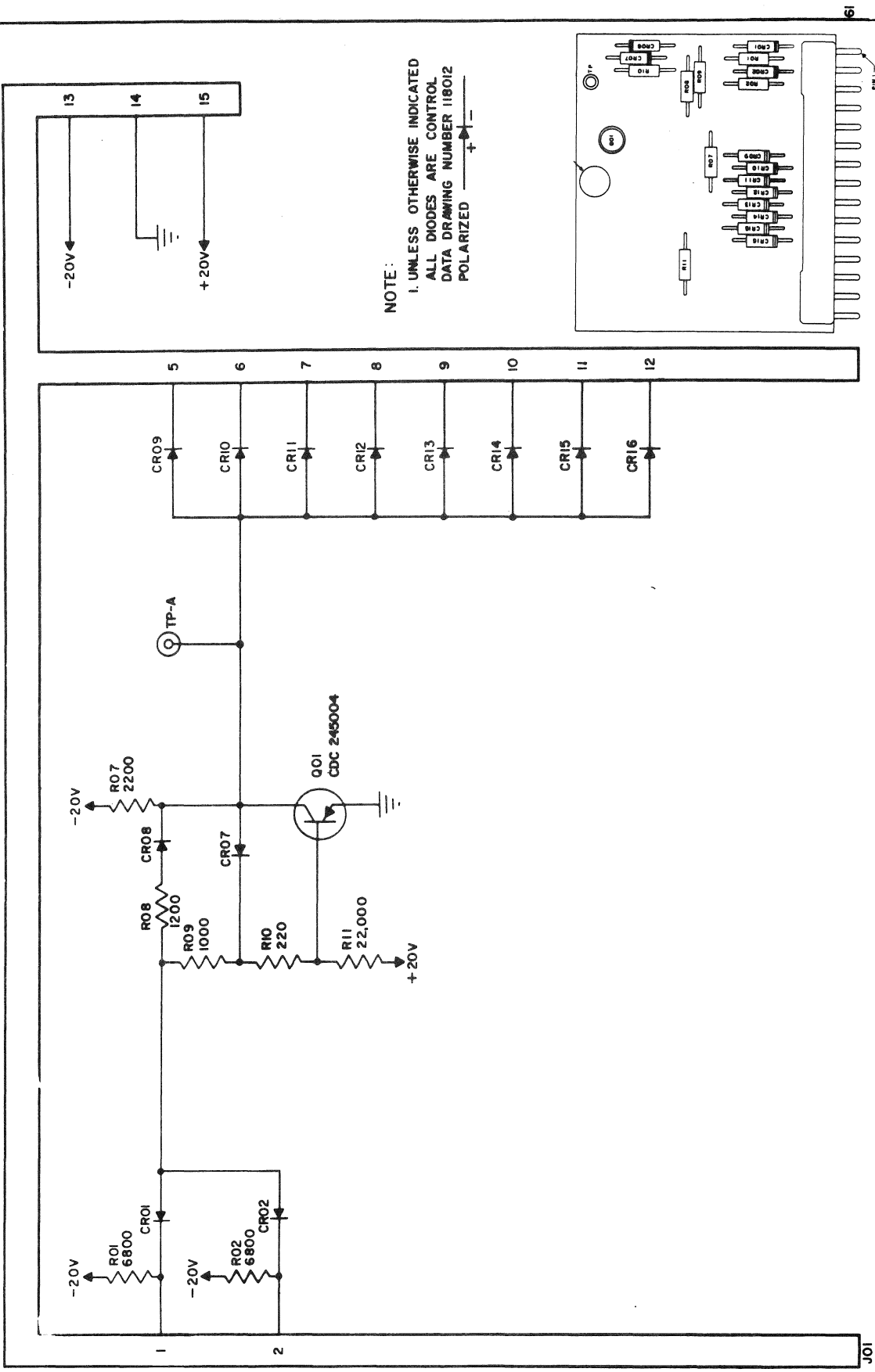


Single Inverter 11A
B-15



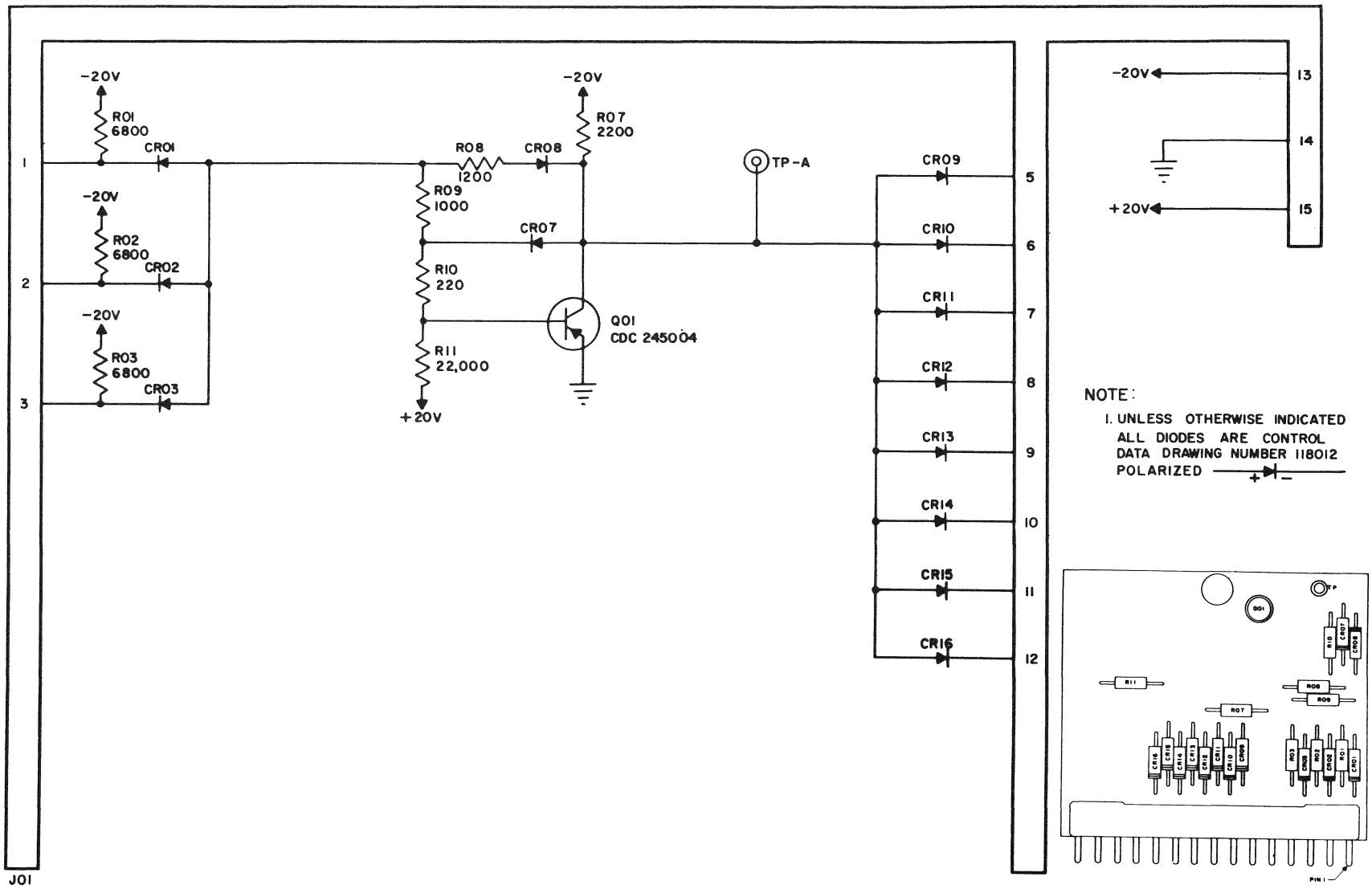
NOTE:
I. UNLESS OTHERWISE INDICATED
ALL DIODES ARE CONTROL
DATA DRAWING NUMBER 118012
POLARIZED 





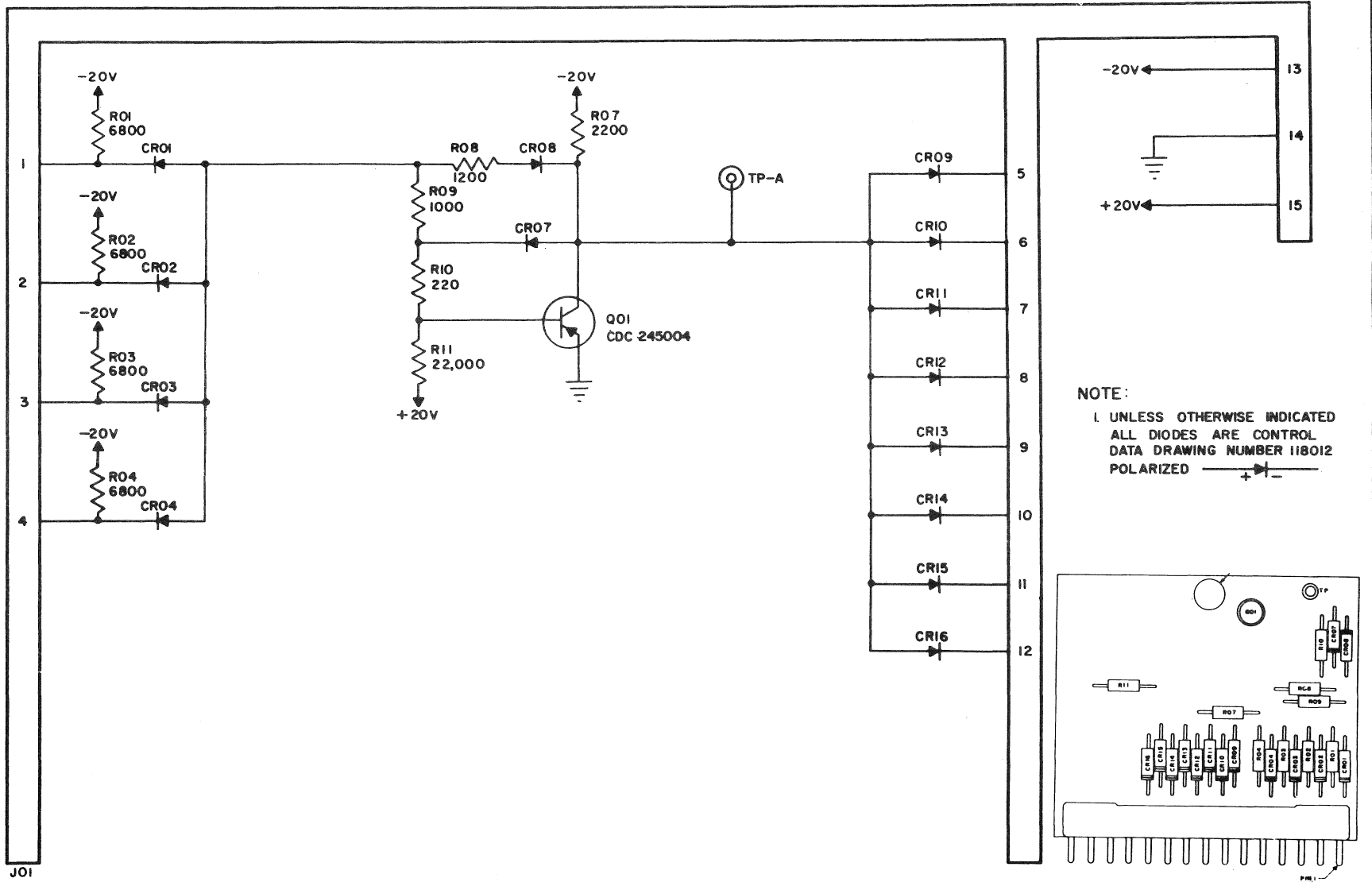
Single Inverter 12A
 B-16

Single Inverter 13A
B-17

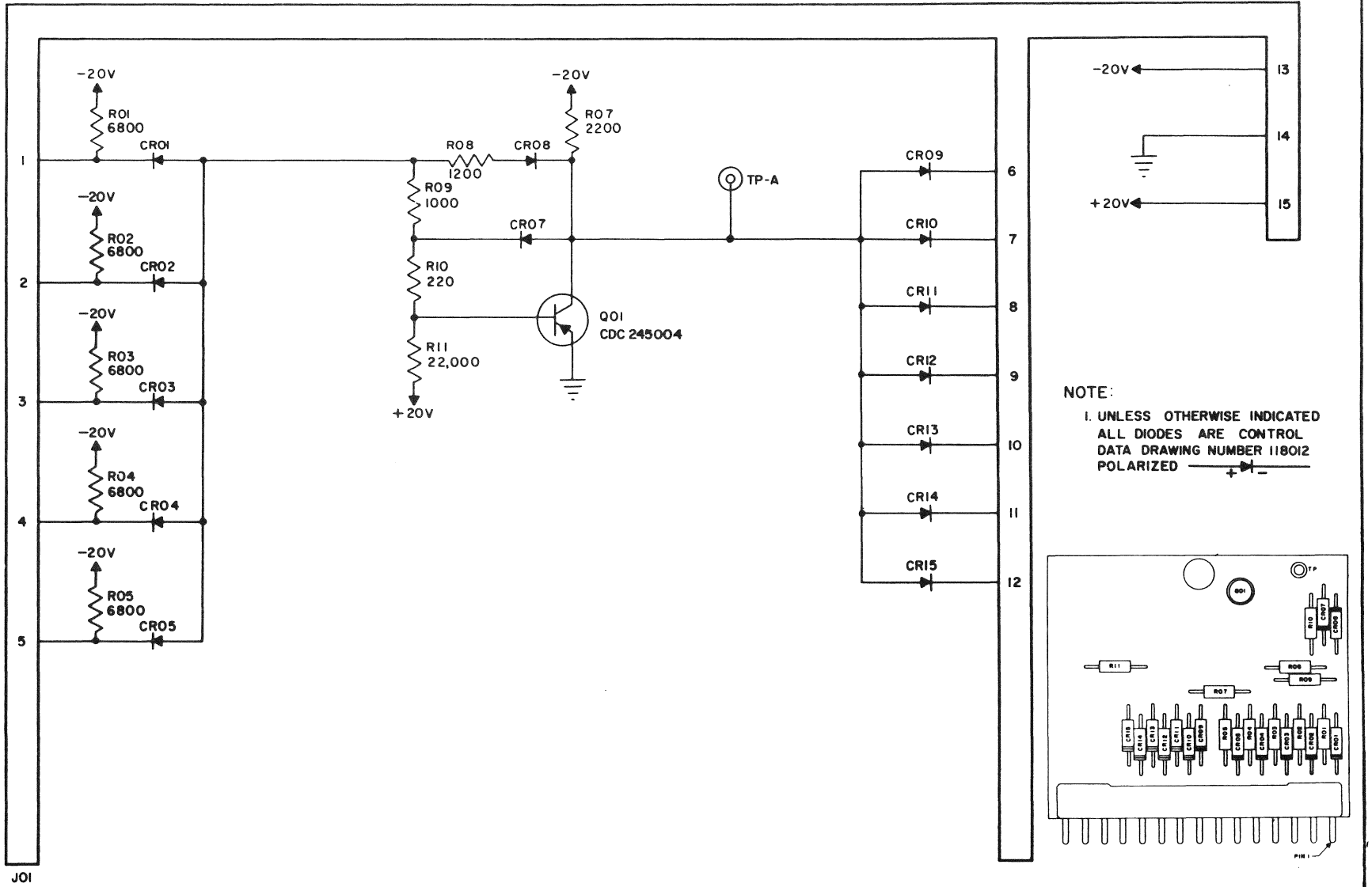


NOTE:
1. UNLESS OTHERWISE INDICATED
ALL DIODES ARE CONTROL
DATA DRAWING NUMBER 118012
POLARIZED

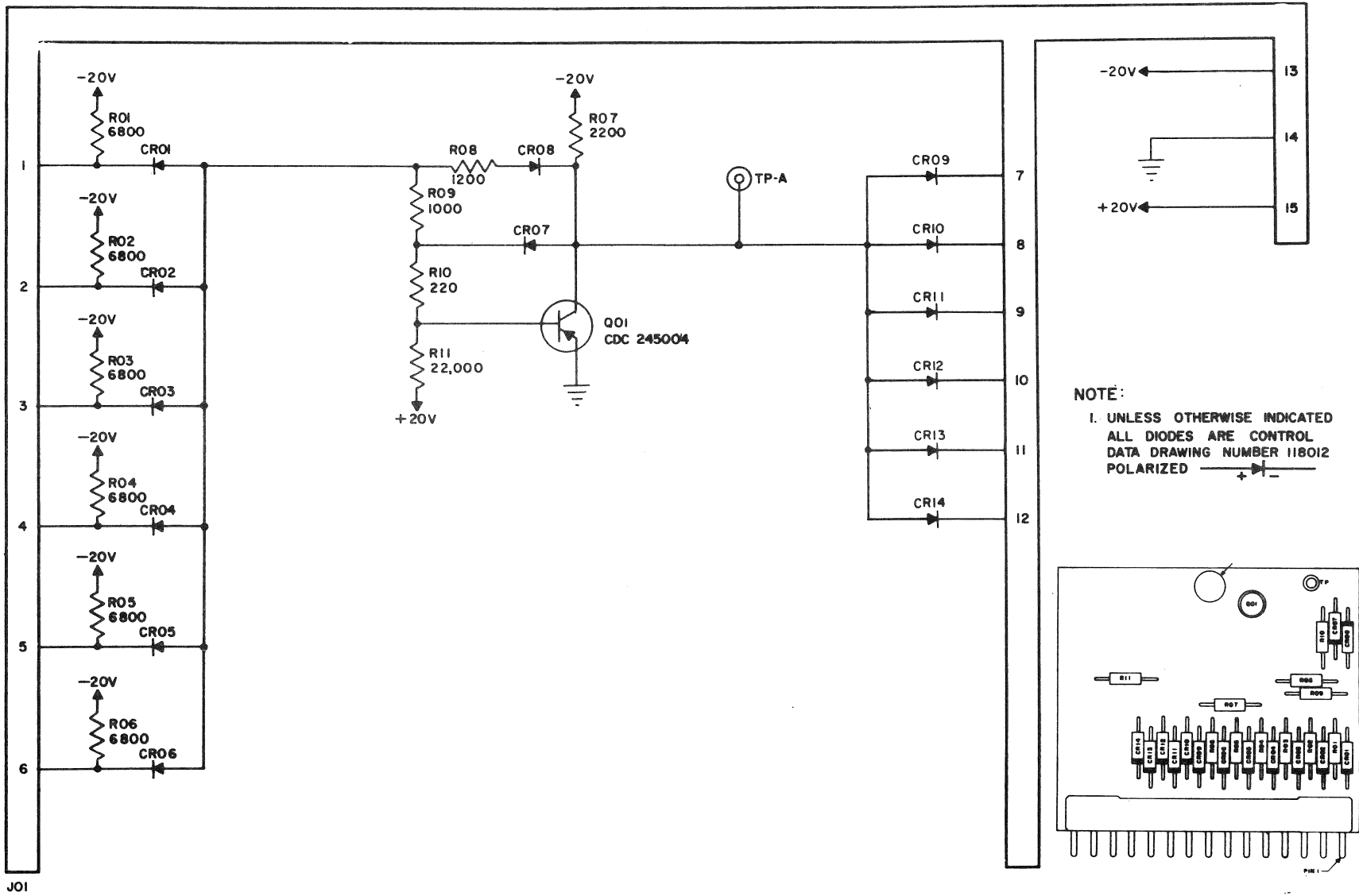
Single Inverter 14A
B-18



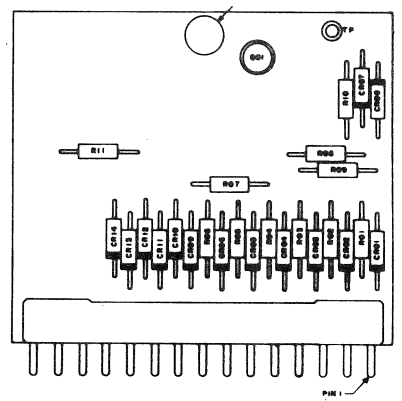
Single Inverter 15A
B-19



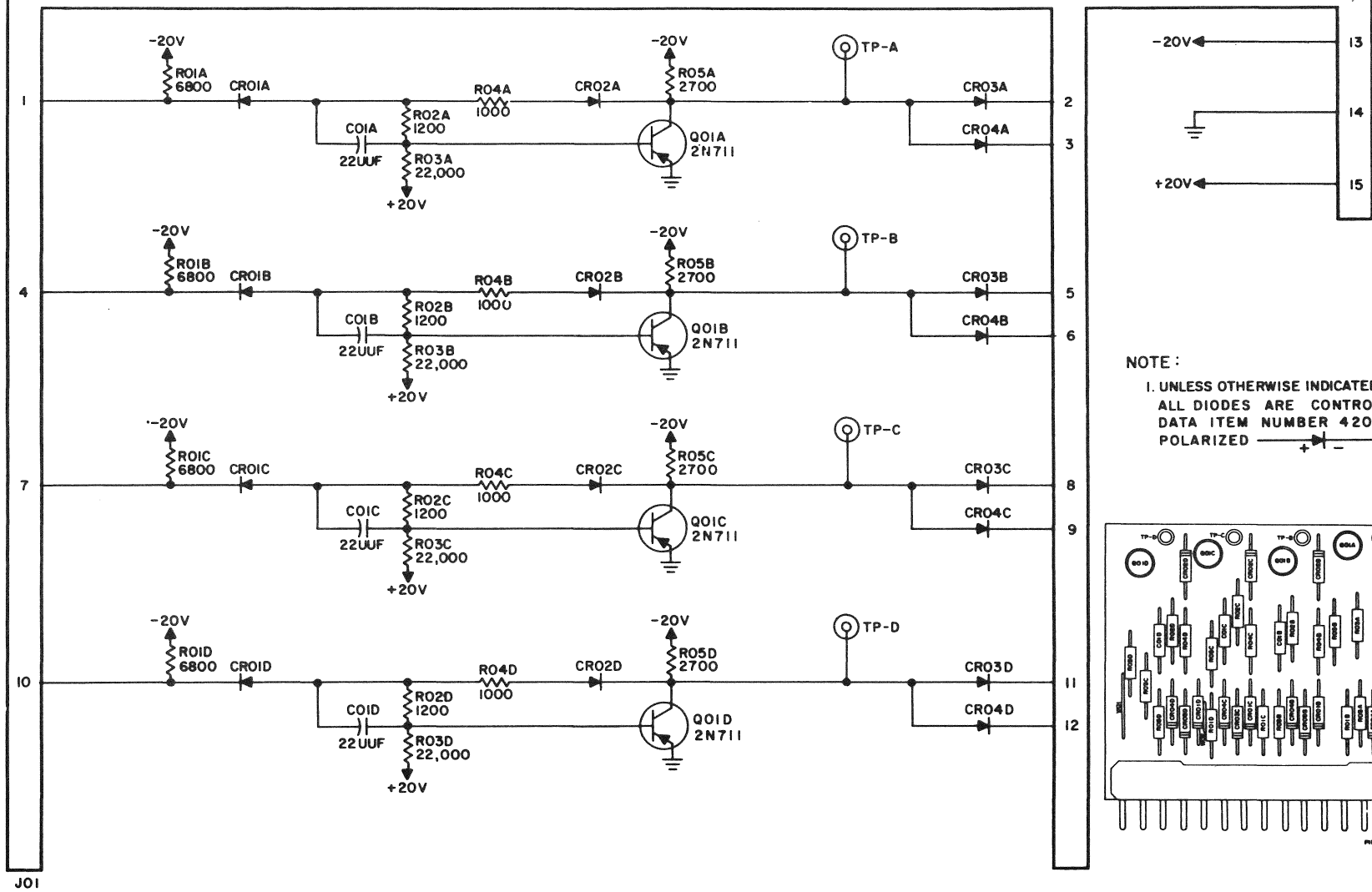
Single Inverter 16A
B-20



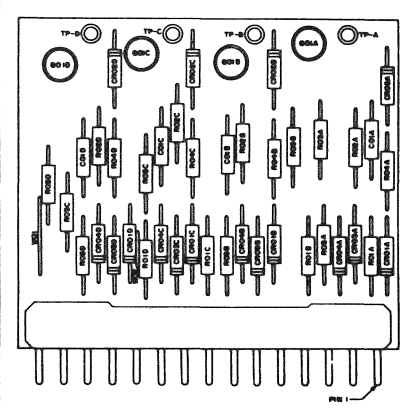
NOTE:
1. UNLESS OTHERWISE INDICATED
ALL DIODES ARE CONTROL
DATA DRAWING NUMBER 118012
POLARIZED



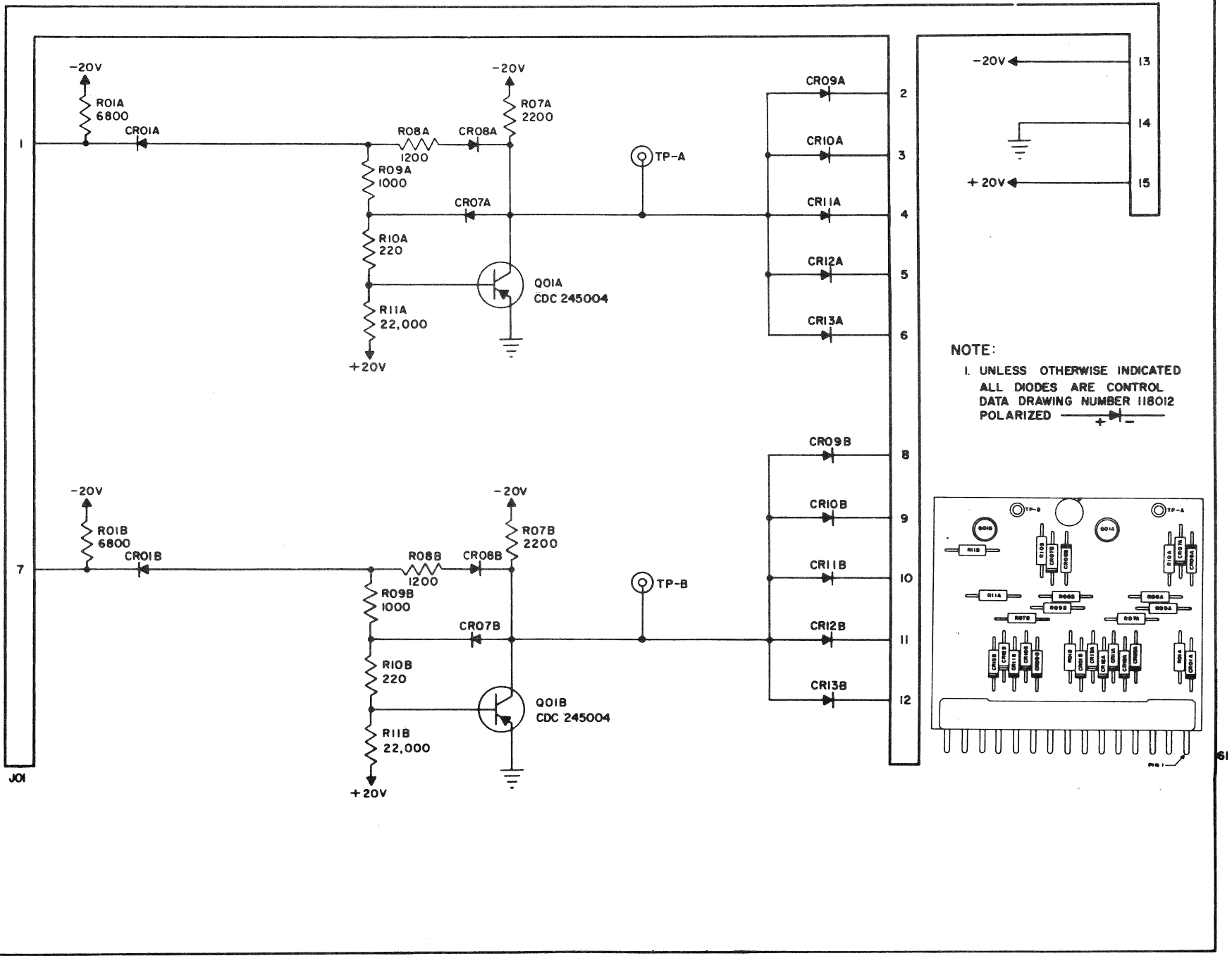
Quadruple Inverter 20
 B-21



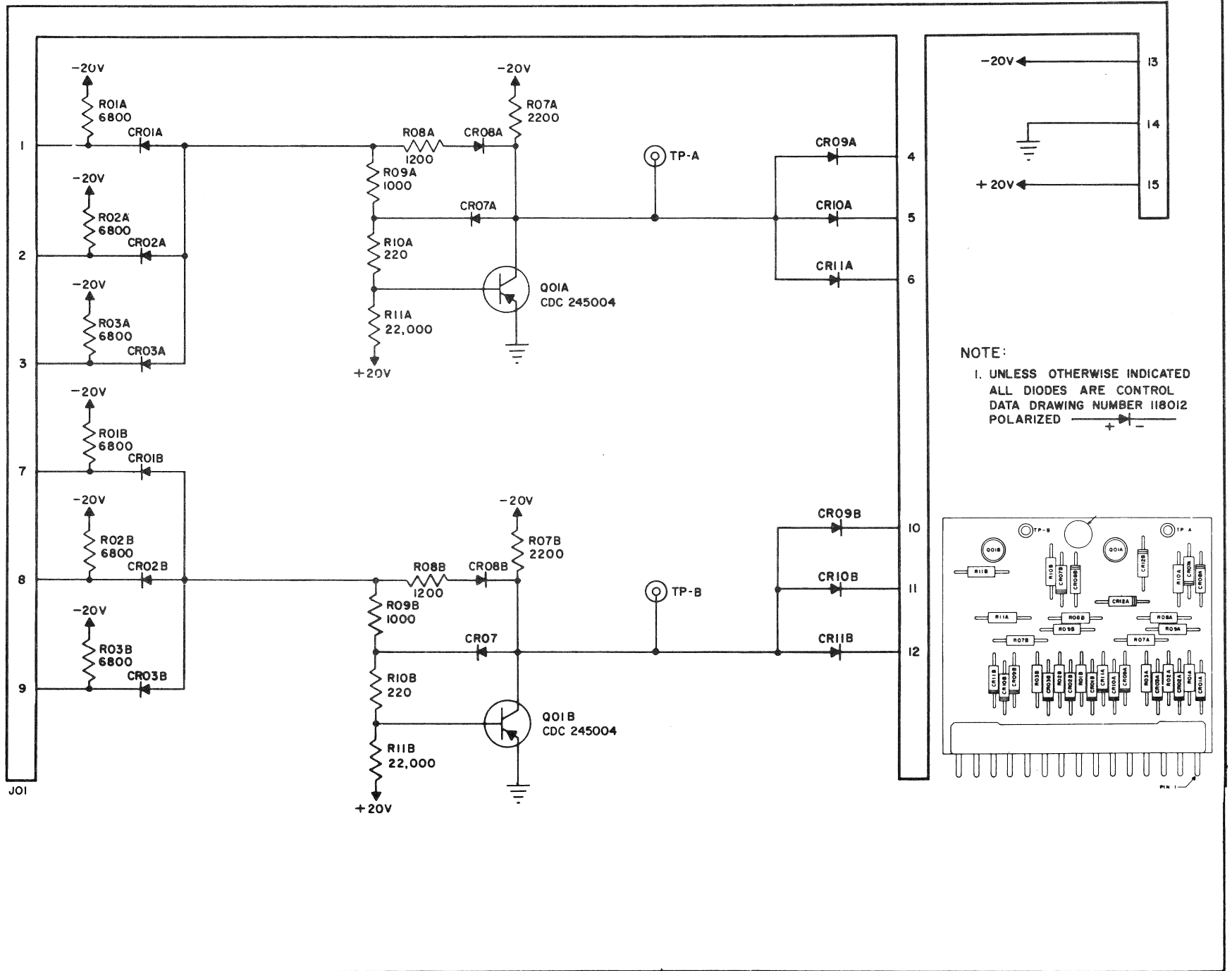
NOTE:
 1. UNLESS OTHERWISE INDICATED,
 ALL DIODES ARE CONTROL
 DATA ITEM NUMBER 4200
 POLARIZED $\begin{matrix} + & - \\ | & | \end{matrix}$




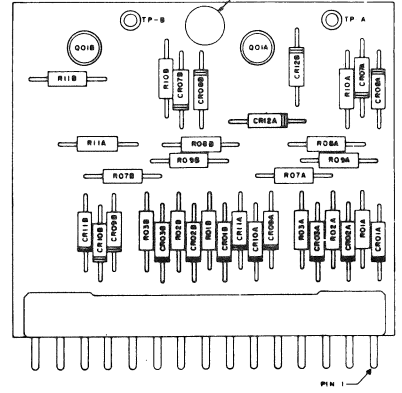
Double Inverter 21A
B-22



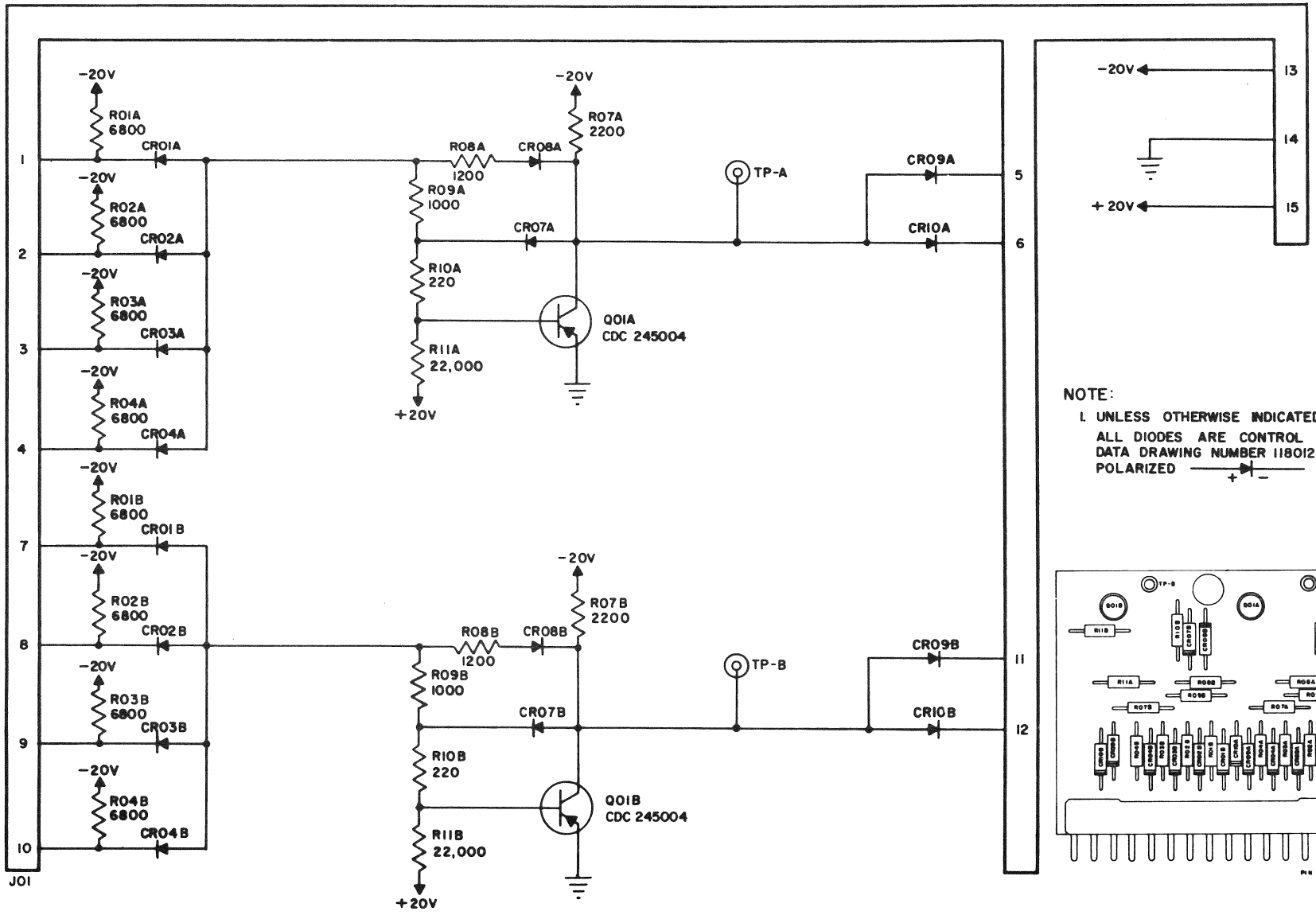
Double Inverter 23A
B-24

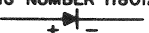


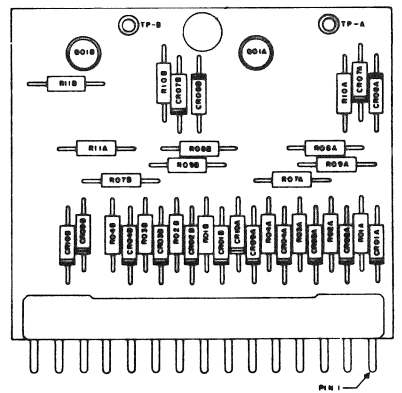
NOTE:
I. UNLESS OTHERWISE INDICATED
ALL DIODES ARE CONTROL
DATA DRAWING NUMBER 118012
POLARIZED 



Double Inverter 24A
B-25

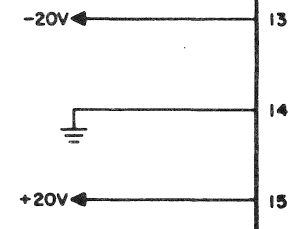
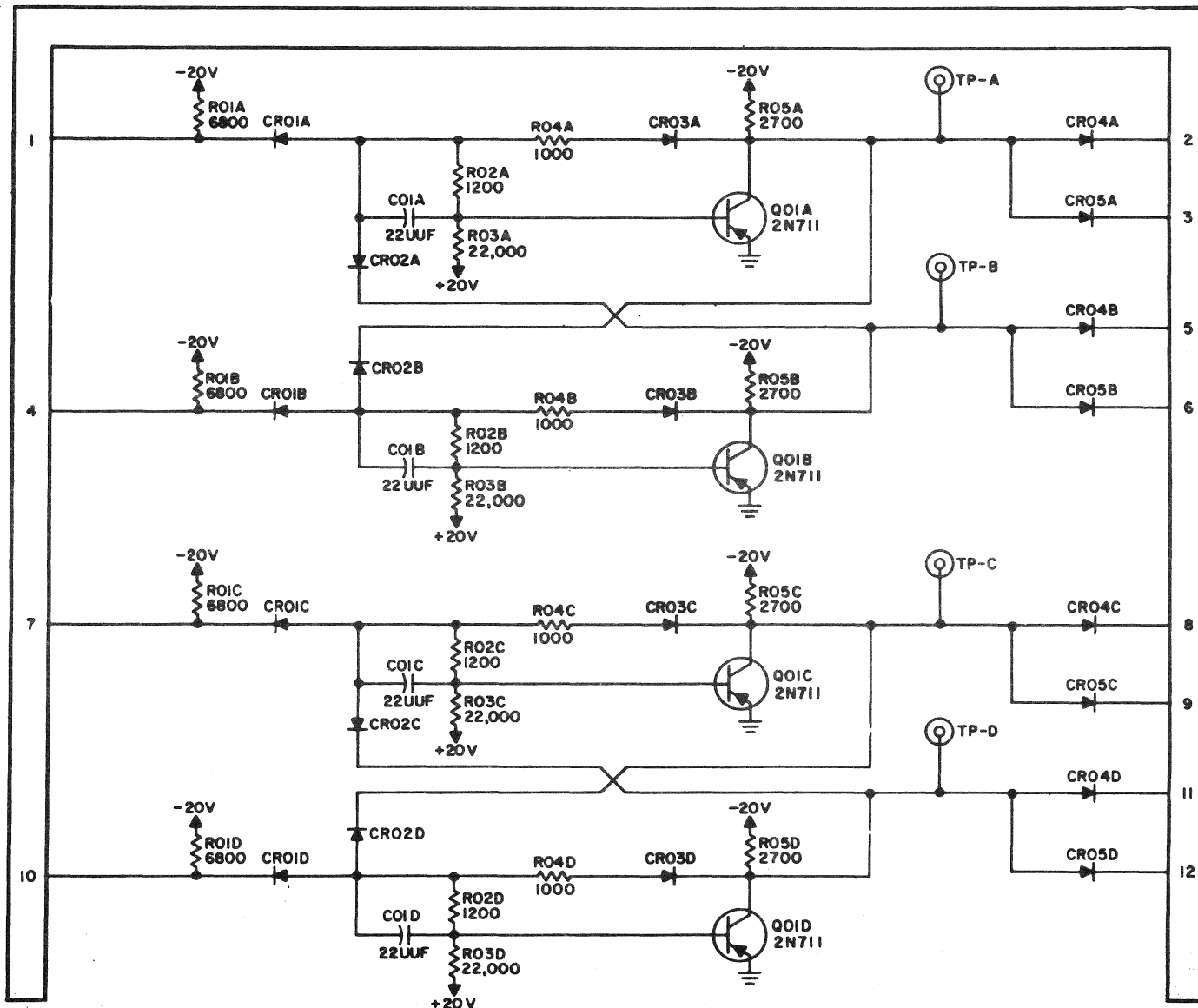


NOTE:
1. UNLESS OTHERWISE INDICATED
ALL DIODES ARE CONTROL
DATA DRAWING NUMBER 118012
POLARIZED 

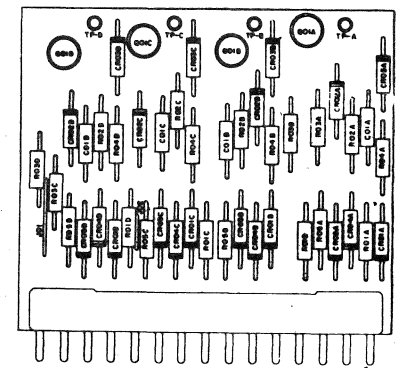


Double Flip Flop 30A

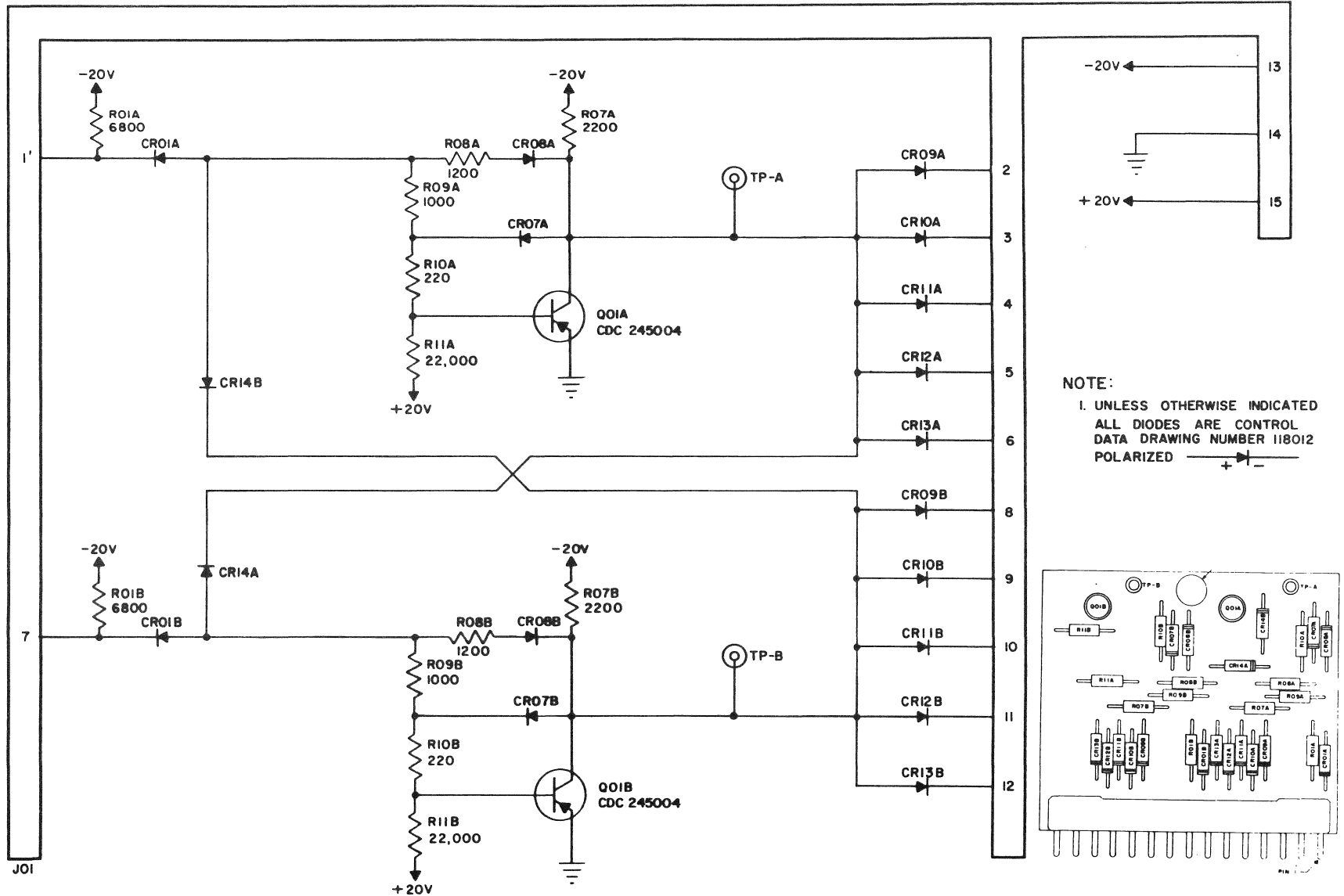
B-26

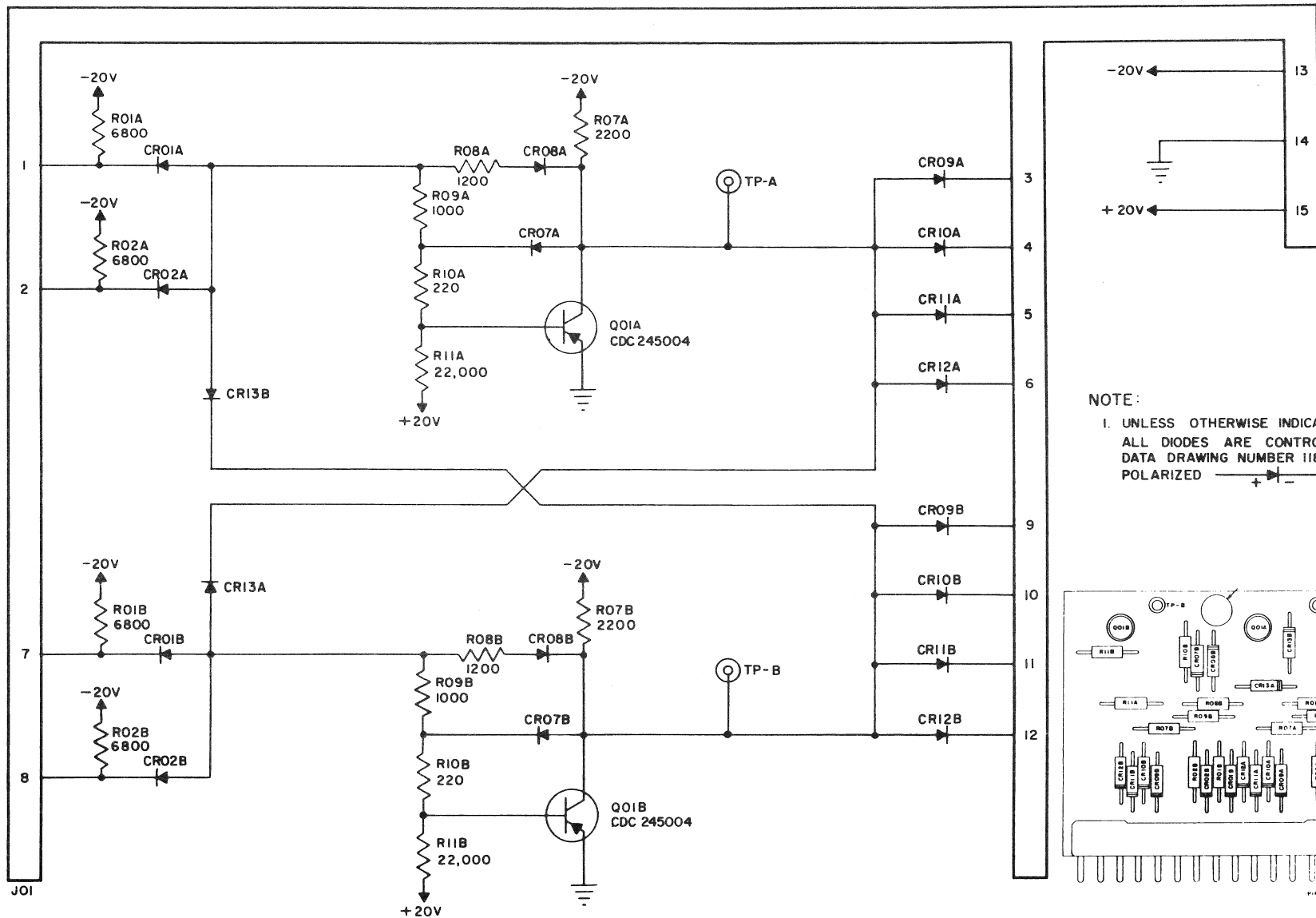


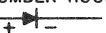
NOTE:
 1. UNLESS OTHERWISE INDICATED,
 ALL DIODES ARE CONTROL
 DATA ITEM NUMBER 4200
 POLARIZED \rightarrow

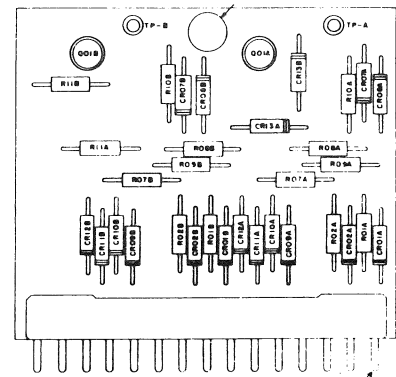


Flip Flop 31A
B-27

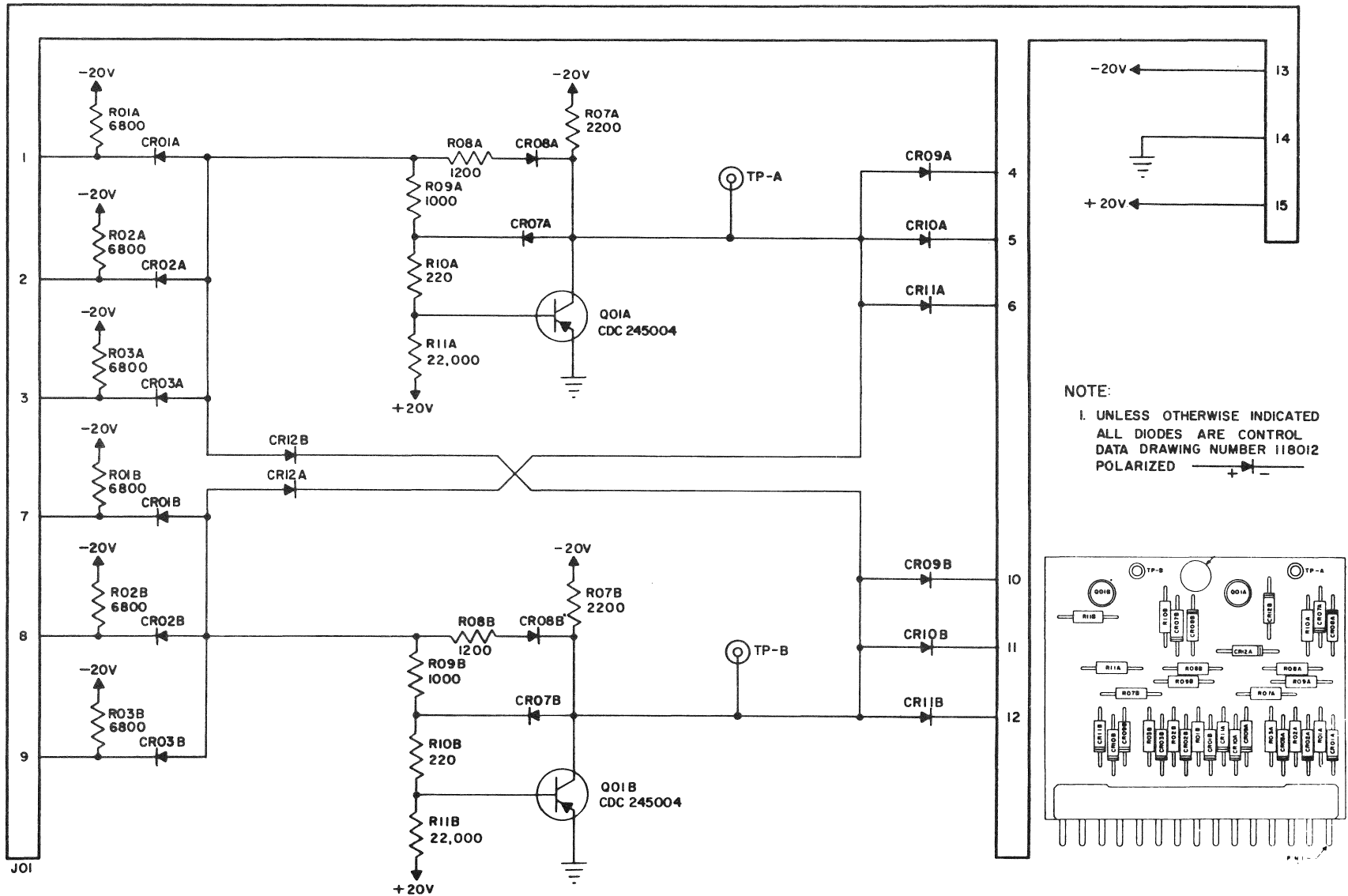




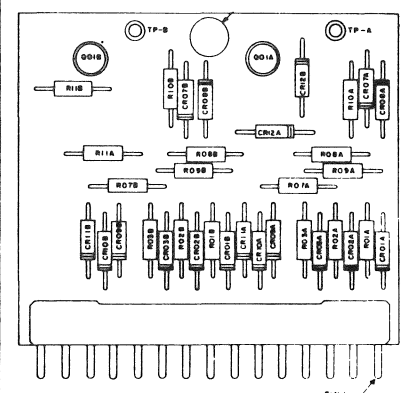
NOTE:
1. UNLESS OTHERWISE INDICATED
ALL DIODES ARE CONTROL
DATA DRAWING NUMBER 118012
POLARIZED 



Flip Flop 33A
B-29

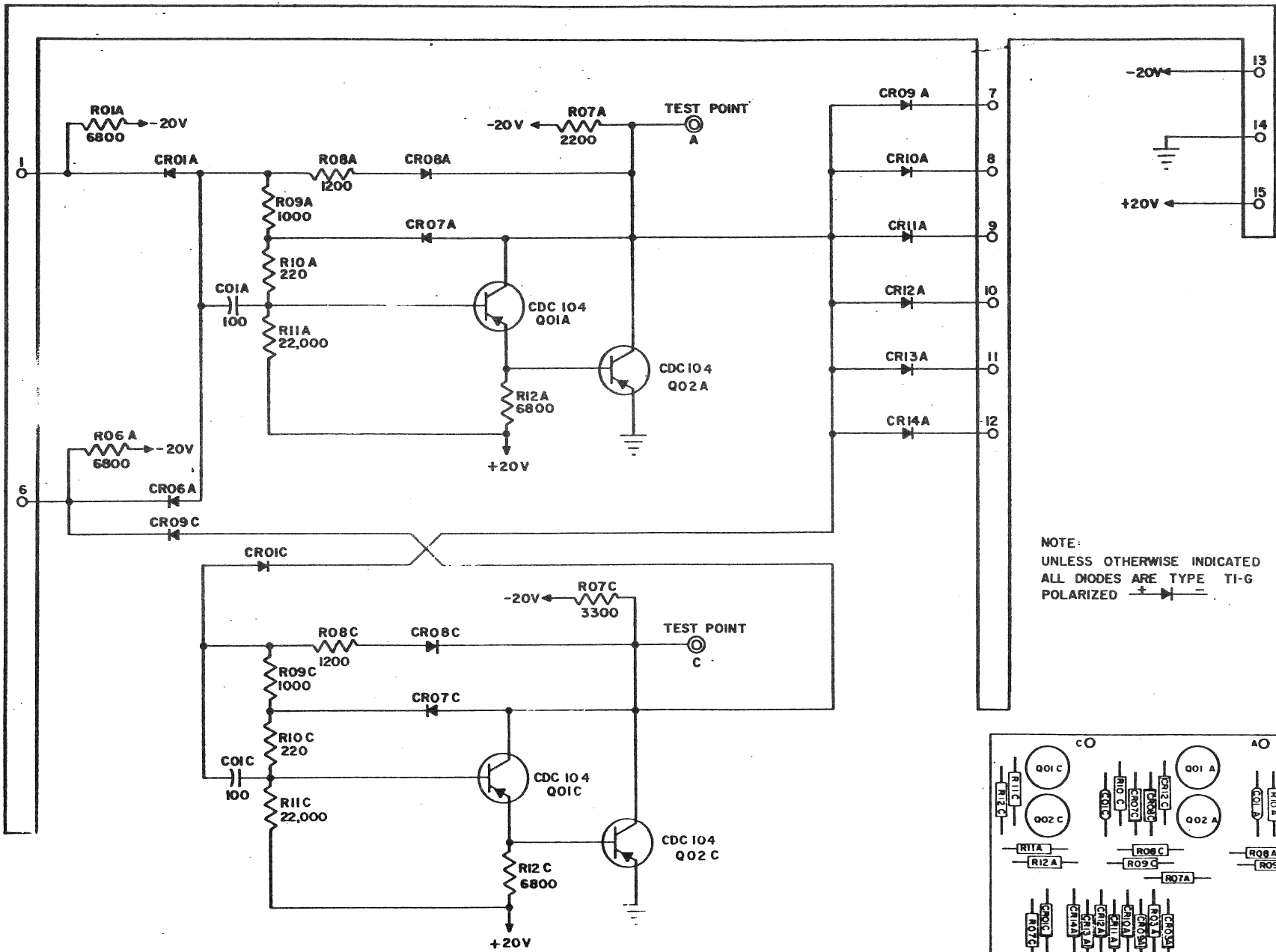


NOTE:
1. UNLESS OTHERWISE INDICATED
ALL DIODES ARE CONTROL
DATA DRAWING NUMBER 118012
POLARIZED \rightarrow

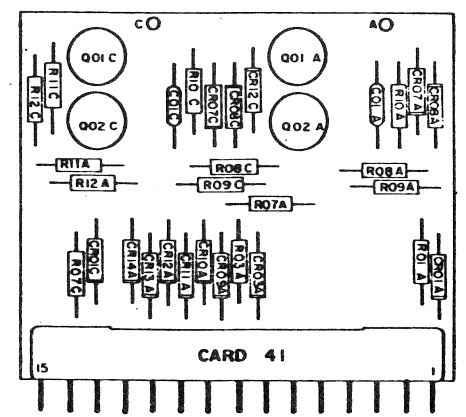


Control Delay 41

B-30



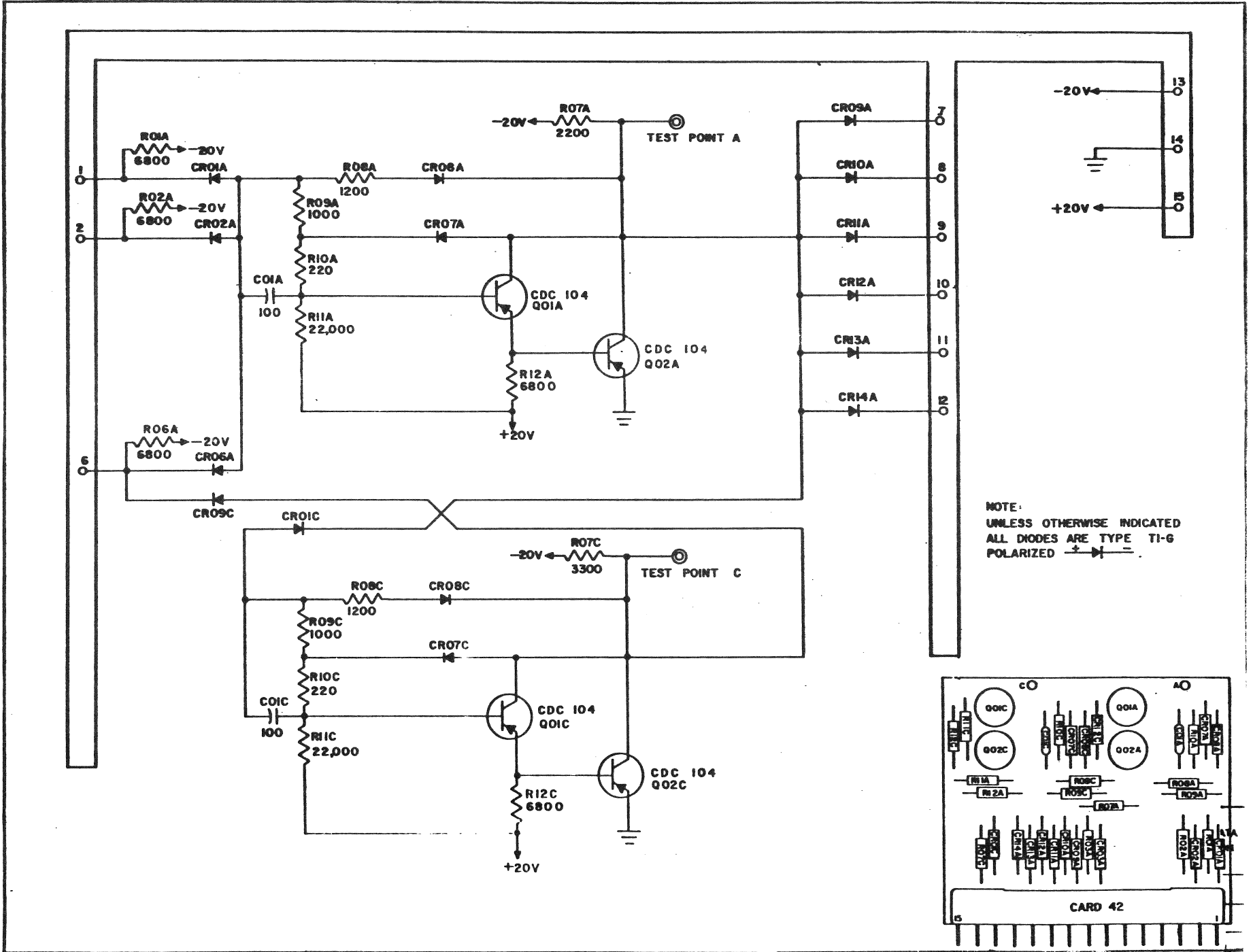
NOTE:
UNLESS OTHERWISE INDICATED
ALL DIODES ARE TYPE TI-G
POLARIZED \rightarrow



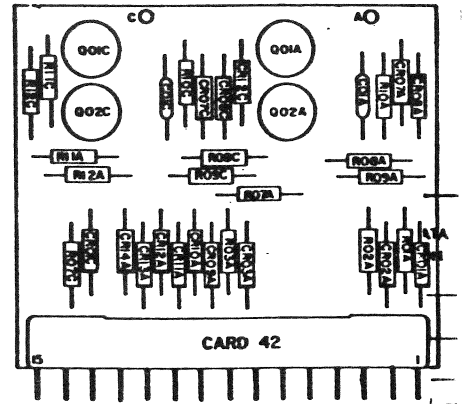
CARD 41

Control Delay 42

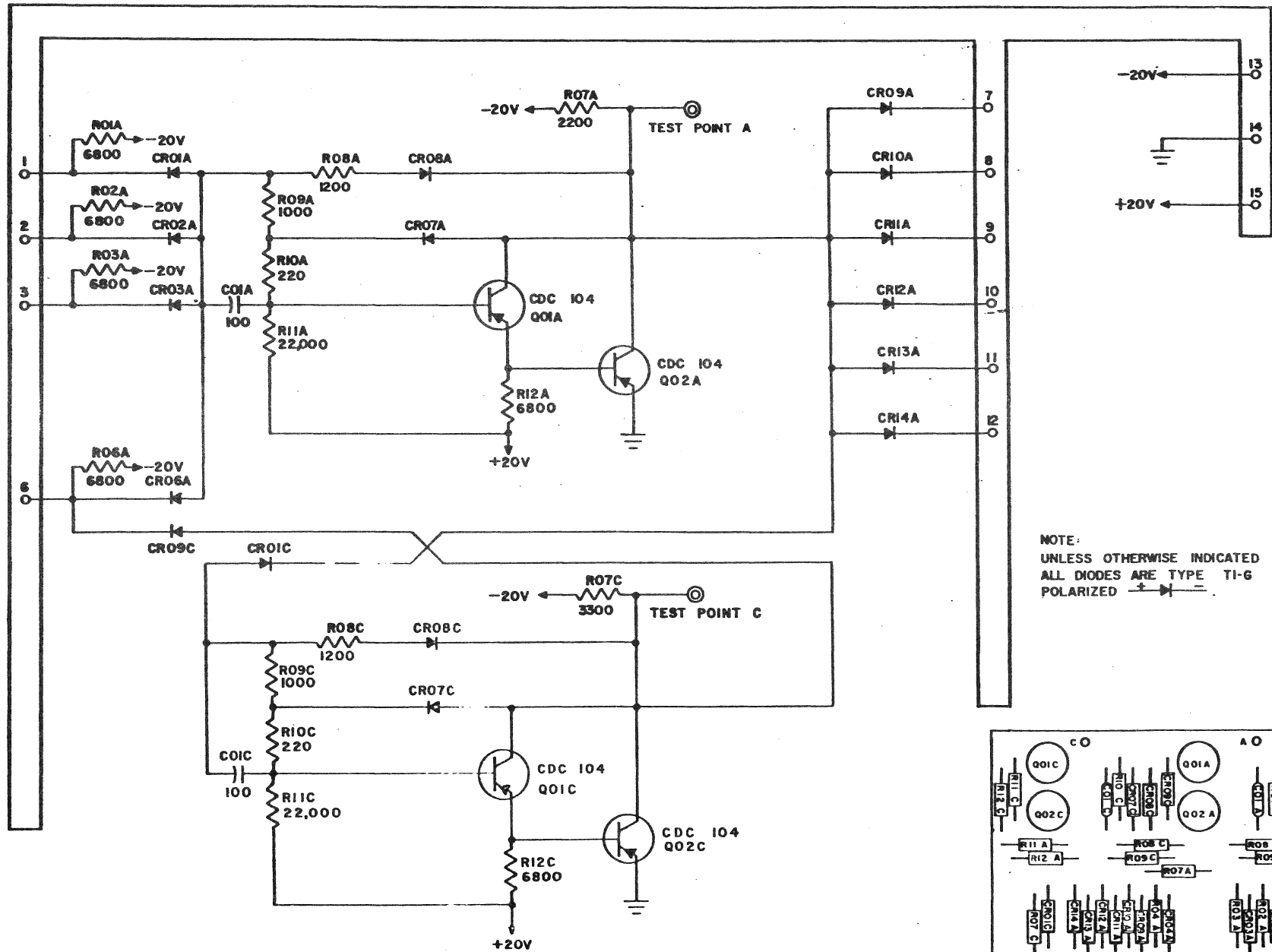
B-31



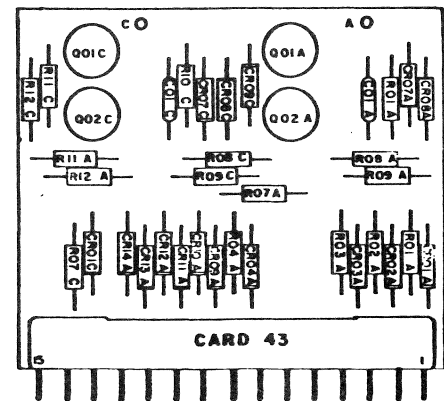
NOTE:
UNLESS OTHERWISE INDICATED
ALL DIODES ARE TYPE TI-6
POLARIZED $\begin{matrix} + \\ \text{---} \end{matrix}$



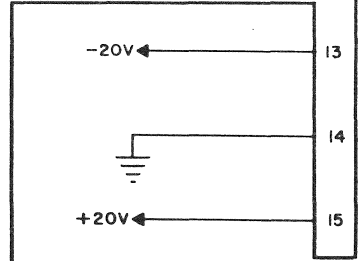
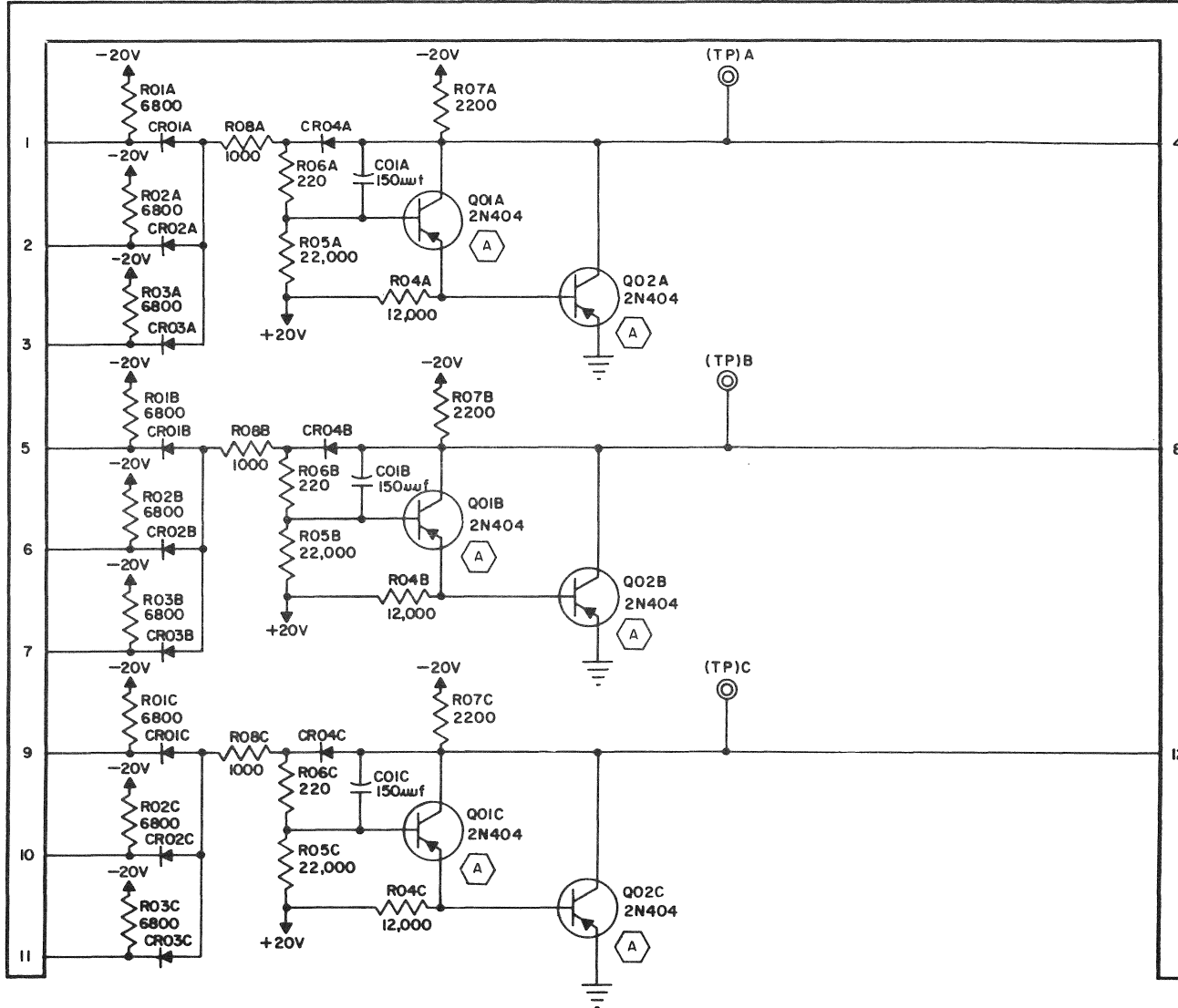
Control Delay 43
B-32



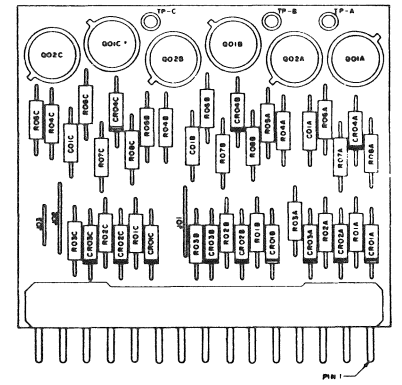
NOTE:
UNLESS OTHERWISE INDICATED
ALL DIODES ARE TYPE TI-6
POLARIZED \rightarrow

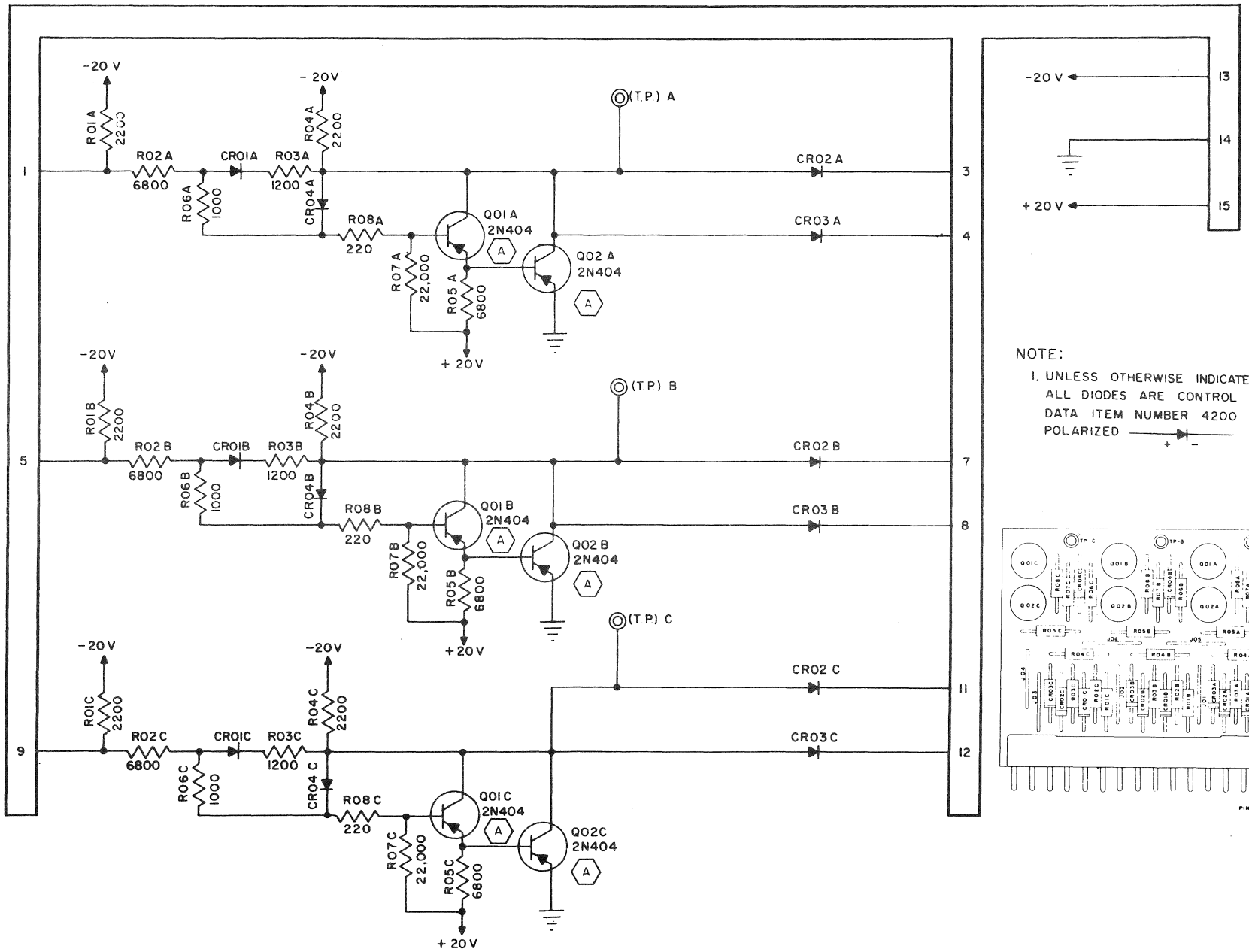



B-133
Output 60A

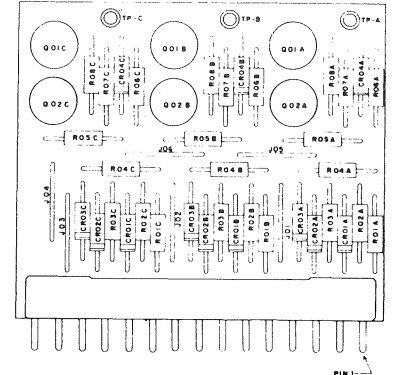


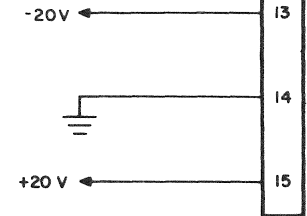
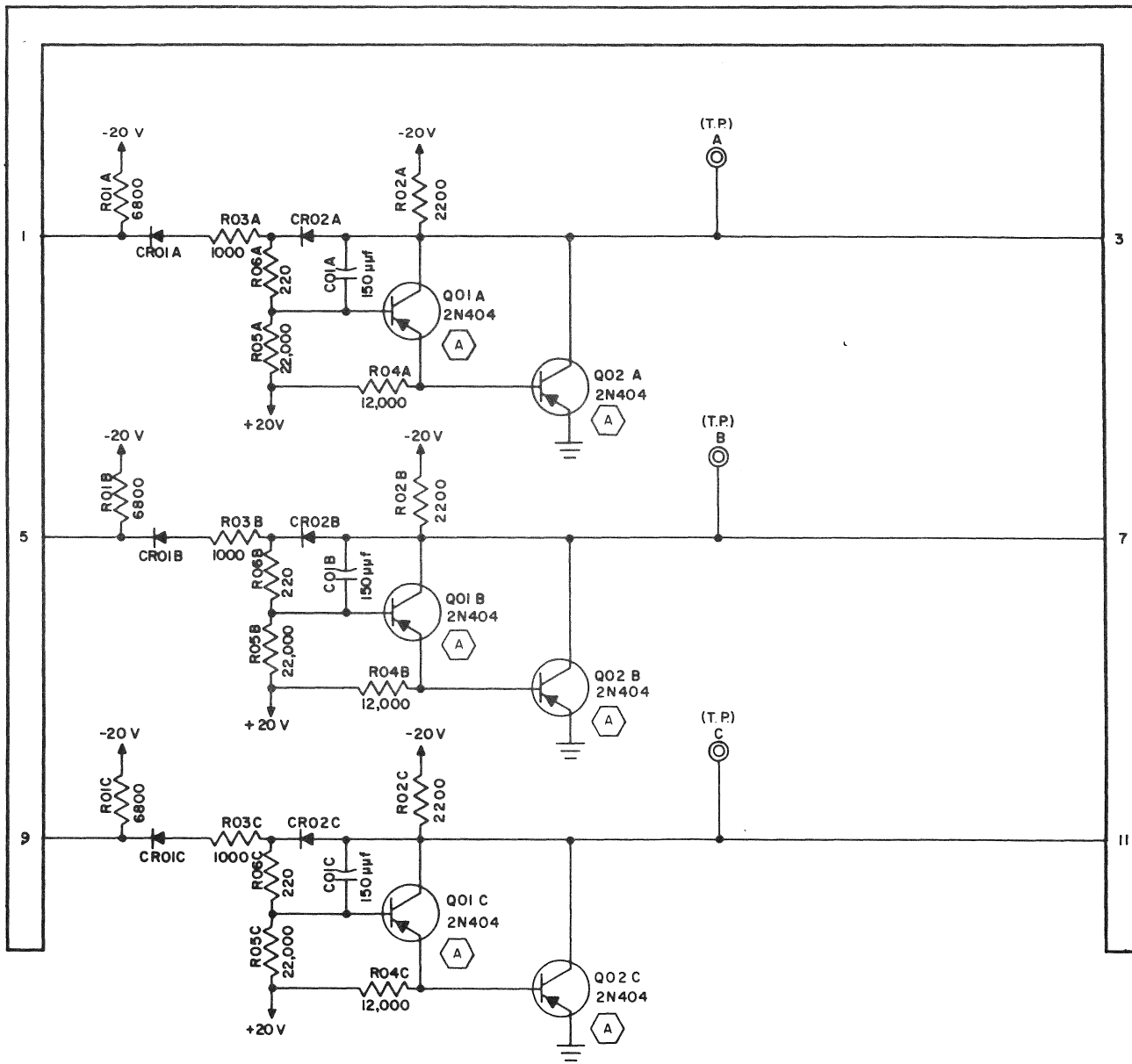
NOTE:
 1. UNLESS OTHERWISE INDICATED,
 ALL DIODES ARE CONTROL
 DATA ITEM NUMBER 4200
 POLARIZED



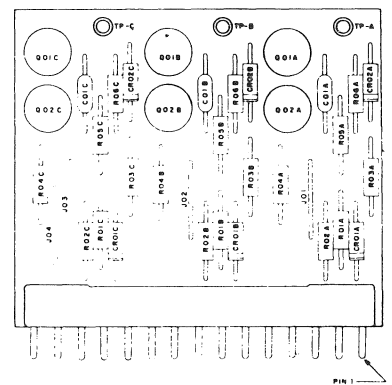


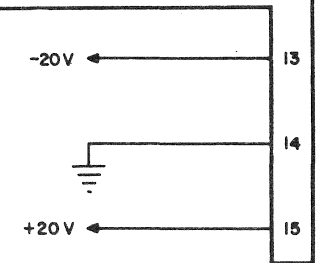
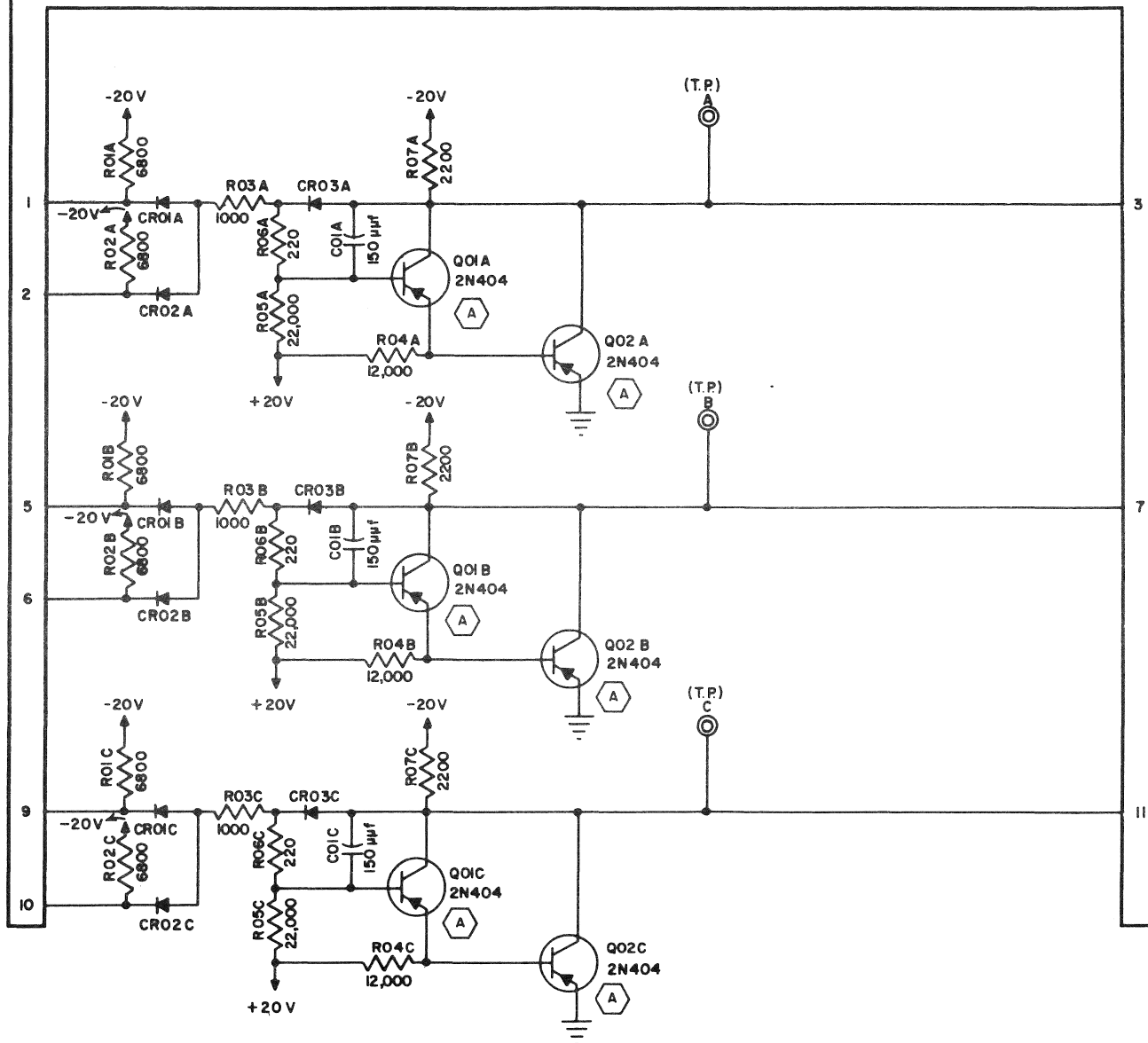
NOTE:
I. UNLESS OTHERWISE INDICATED,
ALL DIODES ARE CONTROL
DATA ITEM NUMBER 4200
POLARIZED 



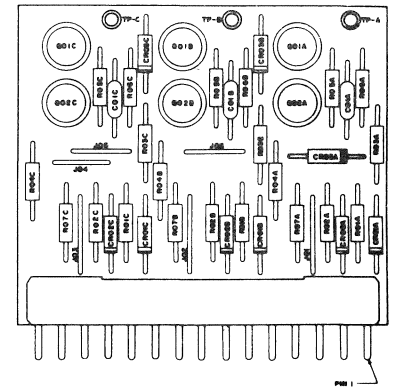


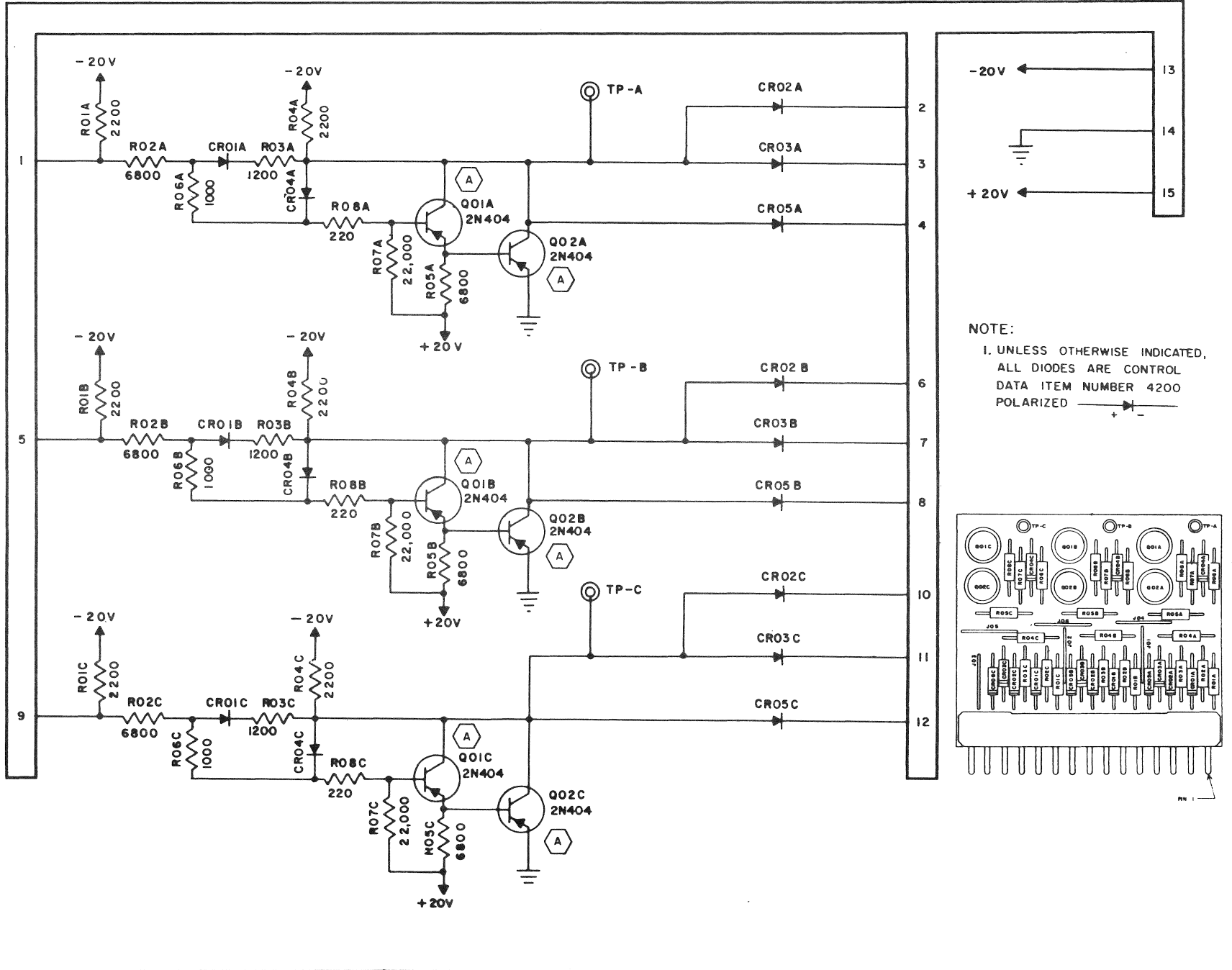
NOTE:
1. UNLESS OTHERWISE INDICATED,
ALL DIODES ARE CONTROL
DATA ITEM NUMBER 4200
POLARIZED




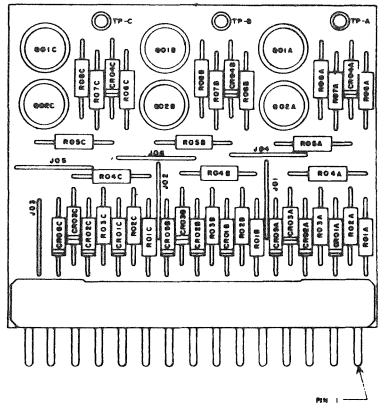


NOTE:
1. UNLESS OTHERWISE INDICATED,
ALL DIODES ARE CONTROL
DATA ITEM NUMBER 4200
POLARIZED





NOTE:
1. UNLESS OTHERWISE INDICATED,
ALL DIODES ARE CONTROL
DATA ITEM NUMBER 4200
POLARIZED 



ASSIGNMENT SHEET

1

160A OPERATIONAL CHARACTERISTICS

1. Why does the Buffer Input line come into both the Buffer Data Register and the I^{---1} inverters?
2. What would happen if the inhibit generators were inoperative during a memory reference?
3. What is meant by the number "0" and the word "REL" displayed in the MCS mode indicator?
4. What is the purpose of the A' register?

ASSIGNMENT SHEET

2

CONSOLE CONTROLS AND INDICATORS

1. What is the "LOAD" position of the load-sweep switch used for?

2. With the "A-A'- Buffer Data Register" switch in the up position, what is being displayed on the console; what, if anything, indicates we're in the up position of the switch; and what effect would master clear have on what is being observed?

3. After turning power on, you should always master clear. Why?

4. What is the purpose of the selective jump switches?

ASSIGNMENT SHEET

3

MEMORY CYCLES

1. What is the difference between a D cycle and a B cycle?

2. What type of operation would the computer be doing if a series of D to C cycle progressions were being executed?

3. Which cycle would the computer always do first? Why?

4. Which addressing mode might require all four memory cycles?

5. What is the cycle progression for a, STF (store forward) Instruction?

LDD (load direct)?

ADI (add indirect)?

6. How long will it take to execute a series of 16 instructions employing,

A. Indirect addressing, if half are store instructions?

B. Specific addressing, no store instructions?

ASSIGNMENT SHEET
4
INTRODUCTION TO PROGRAMMING

1. What is meant by "internally programmed" when related to a computer?
2. Why should a programmer use flow charts?
3. What is a program?

ASSIGNMENT SHEET

6

STOP INSTRUCTIONS

1. What will appear in the status mode indicator when a 7777 instruction is executed?

2. What would be seen in the Z register when the computer stops as a result of a HLT instruction? The P register? The A register?

3. What is the difference between a 0000 instruction and a 7700 instruction?

4. What is the difference between a 0000 instruction and a 000X instruction? Could the 000X instruction be 0000 and perform its job? Why or why not?

ASSIGNMENT SHEET

7

TRANSMITTIVE INSTRUCTIONS

1. Write a program that will take the contents of 6 memory locations starting at address 0060 to 6 memory locations starting at address 0070 and transfer the contents of address 7777 to address 1003. Do Not Destroy The Original Contents Of The "A" Register. Write your program beginning at address 5000.

ASSIGNMENT SHEET

8

ARITHMETIC INSTRUCTIONS

1. If when we subtract one number from another we add the complement would it be correct to load complement and then add a number to the result? Explain.

ASSIGNMENT SHEET
13
INPUT-OUTPUT INSTRUCTIONS

1. Assume the following conditions:
 - a. All bank controls set to 0
 - b. Program in M.L. 2000 to 2065
 - c. M.L. 2000 = 0100 (Block Store Instruction)
 - d. FWA = 1000, LWA+1 = 3000
 - e. (A) = 0000

Question: Which memory locations would be affected by the BLS Instruction?

What would happen at the completion of the BLS operation?

2. Given:

0200 = 7203
0201 = 0300
0202 = 7700
0203 = 0100

ALL INCOMING DATA EQUALS 0000

Which addresses will receive the incoming data? When the computer stops, what will the A, Z, and P registers contain?

160A PROGRAMMING PROBLEMS - REVIEW OF PROGRAMMING
SHEET 14

1. Master Clear, Set P to 0100 and Start. What is the contents of the A Register when the computer stops? What is the mode of addressing?

0100 0420
0101 0614
0102 0710
0103 7700

2. Master Clear, Set P to 0100 and Start. What does the contents of A, (A), Equal when the computer stops? What is the mode of addressing?

0075 1020
0076 0410
0077 0312
0100 2075
0101 3076
0102 3477
0103 7700

3. Master Clear, Set P to 0100 and Start. What is (A)f?

0075 0106
0076 0105
0077 0104
0100 2175
0101 3176
0102 3577
0103 7700
0104 0312
0105 0410
0106 1020

CLUES: A. What does the E portion of A load indirect instruction specify?
B. If your answer is an operand address, you better think again.

4. Master Clear, Set P to 0100 and Start. What is (A)f?

0100 2100
0101 0107
0102 3100
0103 0110
0104 3500
0105 0111
0106 7700
0107 1020
0110 0410
0111 0312

CLUE: A. The G portion of an add memory instruction is an _____.

5. Master Clear, Set P to 0100 and Start. What is (A)f? What is the mode of addressing?

0100 2204
0101 3204
0102 3604
0103 7700
0104 1000
0105 0010
0106 0100

6. Master Clear, Set P to 0100 and Start. What is (A)f and (P)f?

0100 2200
0101 1000
0102 3200
0103 0010
0104 3600
0105 7700
0106 7700

CLUES: A. The G portion of an add constant instruction is an _____.
B. You should now see the difference between the memory and constant modes of addressing.

7. Master Clear, Set P to 0100 and Start. What is (A)f? What is the mode of addressing?

0076 2222
0077 1032
0100 2301
0101 3301
0102 3704
0103 7700

8. Master Clear, Set P to 0100. What is (A)f and (P)f?

0076 0110
0077 0112
0100 2200
0101 0050
0102 4211
0103 5177
0104 2500
0105 0112
0106 4176
0107 0677
0110 0000
0111 0112
0112 0027
0113 0000

CLUE: A. Upon completion of a replace add instruction, the sum of the two operands could be found at _____ and _____.

9. Master Clear, Set P to 0100 and Start. When the computer stops, what is the contents of memory location 0105 and (A)_f?

0077 0105
0100 2600
0101 3734
0102 0115
0103 4177
0104 7700
0105 0000

CLUES: A. A right shift is an _____ shift.
B. Did you remember the sign extension?

10. Master Clear, Set P to 1000 and Start. What is (A)_f?

1000 2204
1010 0102
1002 3602
1003 7700
1004 2330

CLUE: A. A left shift is an _____ shift.

11. Master Clear, Set P to 0000 and Start. What is (A)_f?

0000 5405
0001 7107
0002 5005
0003 7700
0004 2200
0005 0002
0006 5701
0007 7011
0010 0004
0011 0002

CLUES: A. On a JFI instruction, P + E is the _____.
B. The E portion of a JPI instruction is _____.

12. Master Clear, Set P to 1000 and Start. What is (A)_f?

0776 0601
0777 6103
1000 2200
1001 7700
1002 6601
1003 0676
1004 6706
1005 6604

See next page for CLUES.

- CLUES: A. When executing a conditional jump instruction, if the condition is met the computer will _____.
If condition is not met, it will _____.
- B. If you said the sum of $7776 + 1$ was 7777 , you had better think again and remember one's compliment.

ANSWERS TO 160-A PROGRAMMING PROBLEMS

1. (A)f = 0024, No Address
2. (A)f = 1116, Direct
3. (A)f = 1116, The address of an operand address
4. (A)f = 1116, Operand Address
5. (A)f = 0710, Forward
6. (A)f = 1107, (P)f = 0106, Operand
7. (A)f = 1111, Backward
8. (A)f = 7777, (P)f = 0110, The specified memory location and the A register.
9. (A)f = 7010, ML0105 = 7010, End Off
10. (A)f = 2330, End Around
11. (A)f = 0010, Address of the Jump Address
12. (A)f = 0000, Jump, Exit

ASSIGNMENT SHEET

16

REGISTER AND INVERTER RANKS

1. A' to S is a forced transfer, S to P is a "ones" transfer. Is there any advantage in using one method over the other? Explain.

2. Couldn't more flip flops have been used with Z register instead of the inverter ranks? Is there an advantage to using inverter ranks? Explain.

3. What is the reason for having two registers to interpret the function code?

4. Why is it necessary to have both an "A" and an "A'" register?

ASSIGNMENT SHEET
17

BORROW PYRAMID

1. The toggle places a result in the A' register that treats every stage as though it were receiving a _____ and the probe corrects this condition by _____ each stage of A' that (does/does not) have a _____ into it.

ASSIGNMENT SHEET
18

INTRODUCTION TO TIMING AND RESYNC COUNTER

1. How long, (in micro seconds) does flip/flop number 3 remain set?
2. What manual operators control starts the Resync Counter?
3. What is the purpose of the Resync Counter?
4. Why is it necessary to have both a Resync and an Excursions Counter?

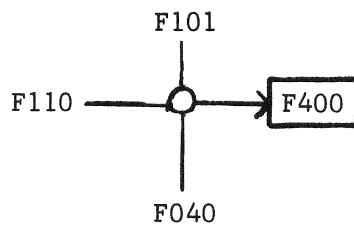
ASSIGNMENT SHEET
19

EXCURSIONS COUNTER

1. At what rate do the flip-flops in the excursions counter count?

ASSIGNMENT SHEET
21
F AND F' REGISTERS AND TRANSLATORS

1. What does F400 interpret?



ASSIGNMENT SHEET
22

"D" CYCLE AND NO ADDRESS MODE

1. What is the logical significance of J200?

ASSIGNMENT SHEET

27

SPECIAL INSTRUCTIONS SEQUENCE

1. Explain the difference between a normal D cycle 4th Quarter and the 4th Quarter of an 0101 instruction.

ASSIGNMENT SHEET

29

CYCLE BREAKDOWN #2

For the following program list the cycle breakdown required for each instruction and the contents of the A, Z, S, P, F registers after the completion of each quarter of each cycle.

M.C. Set P to 0100 and Run. Memory has been cleared.

M. L.

0076 0001
 0077 0053
 0100 0477
 0101 3302
 0102 3600
 0103 0252
 0104 6203
 0105 5307
 0106 6710

| CYCLE | QTR | A | Z | S | P | F |
|-------|-----|---|---|---|---|---|
| | 1 | | | | | |
| | 2 | | | | | |
| | 3 | | | | | |
| | 4 | | | | | |
| | 1 | | | | | |
| | 2 | | | | | |
| | 3 | | | | | |
| | 4 | | | | | |
| | 1 | | | | | |
| | 2 | | | | | |
| | 3 | | | | | |
| | 4 | | | | | |
| | 1 | | | | | |
| | 2 | | | | | |
| | 3 | | | | | |
| | 4 | | | | | |
| | 1 | | | | | |
| | 2 | | | | | |
| | 3 | | | | | |
| | 4 | | | | | |
| | 1 | | | | | |
| | 2 | | | | | |
| | 3 | | | | | |
| | 4 | | | | | |
| | 1 | | | | | |
| | 2 | | | | | |
| | 3 | | | | | |
| | 4 | | | | | |
| | 1 | | | | | |
| | 2 | | | | | |
| | 3 | | | | | |
| | 4 | | | | | |

ASSIGNMENT SHEET

30

CYCLE BREAKDOWN #3

Analyzing the previous assignment will reveal that each quarter of a cycle has a specific purpose. The column on the left below lists all cycles and the four quarters of each, match the appropriate item from the list on the right to the cycle and quarter cycle:

D CYCLE

1st quarter _____

2nd quarter _____

3rd quarter _____

4th quarter _____

- a. Read operand
- b. not used
- c. execute instruction (No Address)
- d. Clear Memory location

A CYCLE

1st quarter _____

2nd quarter _____

3rd quarter _____

4th quarter _____

- e. Not used
- f. Read operand address
- g. Not used
- h. Form address of instruction
- i. execute instruction

B CYCLE

1st quarter _____

2nd quarter _____

3rd quarter _____

4th quarter _____

- j. Not used
- k. Translate instruction
- l. Store word
- m. Read instruction, update P
- n. Not used

C CYCLE

1st quarter _____

2nd quarter _____

3rd quarter _____

4th quarter _____

- o. Form address of operand address
- p. Form address of operand

ASSIGNMENT SHEET

31

MEMORY LOGIC CARDS

1. Explain the difference between a R/W driver and an inhibit generator circuit.

ASSIGNMENT SHEET

33

ADDRESS SELECTION

1. Bits 9, 10, and 11 of the "S" Register select a R/W select. How do we select either Read or Write?

ASSIGNMENT SHEET

34

STORAGE BANK CONTROL LOGIC

1. If r, d, i, and b are all set to different values, how many times is the Relative, Direct, Indirect, and Buffer bank referenced in the following program?

```

1000 - 0400
1001 - 4010
1002 - 2040
1003 - 3100
1004 - 5000
1005 - 7700
    
```

2. The bank controls are set in the following manner:

r = 1 d = 1 i = 2 b = 0

Master Clear. Set P to 1000 and run. What will be in the A register when the computer stops?

| Bank 0 | Bank 1 | Bank 2 |
|-------------|-------------|-------------|
| 1000 - 2200 | 1000 - 0577 | 1000 - 5500 |
| 1001 - 1001 | 1001 - 3100 | 1001 - 1002 |
| 1002 - 4010 | 1002 - 1000 | 1002 - 0701 |
| 1003 - 2110 | 1003 - 0130 | 1003 - 0051 |
| 1004 - 7700 | 1004 - 7777 | 1004 - 7700 |

3. What stops the timing chain if an external module is selected, and it is busy?

4. Why is it necessary to make an "odd/even" bank selection?

5. Bank 4 has been selected. The following terms will output a "1"

- a) S121, B433, B435
- b) S121, B435
- c) S120, B435
- d) S120, B433, B435, J760

ASSIGNMENT SHEET 34 (cont.)

6. What can you say about the contents of P on the D cycle following a 0051 instruction?
- a) $P = 0000$
 - b) $P = P+1$
 - c) $P = P+2$
 - d) $P = A$

ASSIGNMENT SHEET

36

75XX SELECT I

1. How does an external equipment know if the computer output lines hold a Select Code or an information word?
2. What logical purpose does the Function Ready F/F serve?
3. How many microseconds after Sample sets does the Timing Chain Start?
4. When the computer selects the punch the resume comes into J557 from which input?
 - a) J311
 - b) J925, F203
 - c) K308, 1505, K412, 1588, and J915
 - d) 1503, J931, K308
5. What clears the Select PER F/F? (two things)

ASSIGNMENT SHEET

38

INPUT TO "A"

1. Can you trace input information from the input lines to Z register flip-flops, with the timing chain not running? Explain.

2. On an input from the reader which terms gate PER to INP?

3. What term enables us to do a D to C cycle progression for the 7600 instruction?

4. If you "step" through the 7600 instruction, what peculiarity would you notice at the end of the D cycle?

5. What does the Wait Input F/F accomplish when it sets?

ASSIGNMENT SHEET

39

LOAD MODE

1. The set and clear inputs to the "Sample" F/F may be time compared with the main timing chain.

The set input could be given as Time X _____, and the clear input as Time X _____.
2. How is the "Check Sum" formed in the A register during load mode?
3. What happens if the "Load Mode Stop" F/F fails to clear after it once sets?
4. What terms must be present to stop Load Mode?
5. What happens if the computer reads 2 consecutive seventh levels while in Load Mode?
6. How many times does the run F/F clear during the input of 100 words in Load Mode?

ASSIGNMENT SHEET

40

NORMAL INPUT

1. What is the cycle progression for a Normal Input?

2. Why isn't Block +1 to R set on the first C cycle of N input? Explain.

3. Why is the contents of P+1 the LWA + 1 and not LWA? Explain.

4. What term prevents D to B cycle progressions after the first input Request is sent?

5. Describe what happens when the peripheral device responds with an input Disconnect?

ASSIGNMENT SHEET 41 (cont.)

BUFFER OUTPUT

5. What purpose does the BFR to I⁻³ transfer serve in an Output Buffer?
6. What enables the computer to go directly into a Buffer Cycle following the 7300 instruction?

ASSIGNMENT SHEET

42

BLOCK STORE

1. What allows continuous buffer cycles, instead of buffer cycles being interlaced with program cycles?

2. In the following program what would be stored and where?

Master Clear and Run

0000 - 0450
0001 - 0105
0002 - 0001
0003 - 2200
0004 - 2000
0005 - 0106
0006 - 0012
0007 - 0100
0010 - 0012
0011 - 7700
0012 - 0000

3. In the above program what would happen if the "Compare Lockout" F/F failed to clear?

ASSIGNMENT SHEET

43

INTERRUPT

1. What would happen if the 1.5 microsecond delay input into J275 was opened?

2. During the interrupt sequence what happens to the F' register and why?

3. If a manual interrupt and an external interrupt came in simultaneously, and if J264 has a constant "1" output, will both the manual and the external F/F's be set? Explain.

4. Does the operator have to decide to use the interrupt feature? Explain.

INITIAL
CONDITIONS

NAME

| TERM NUMBER | FIRST QUARTER | | | | | | | SECOND QUARTER | | | | | | | THIRD QUARTER | | | | | | | FOURTH QUARTER | | | | | | | REMARKS | | | | | |
|-------------|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------|----|----|----|----|--|
| | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 30 | 31 | 32 | 33 | | 34 | 35 | 36 | 37 | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

INITIAL
CONDITIONS

NAME _____

| TERM NUMBER | FIRST QUARTER | | | | | | | SECOND QUARTER | | | | | | | THIRD QUARTER | | | | | | | FOURTH QUARTER | | | | | | | REMARKS | | | | | | | | | | | | | | | | | | |
|-------------|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------|----|----|----|----|--|--|--|--|--|--|--|--|--|--|--|--|--|--|
| | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 30 | 31 | 32 | 33 | | 34 | 35 | 36 | 37 | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

C-51

INITIAL _____
 CONDITIONS _____

NAME _____

| TERM NUMBER | FIRST QUARTER | | | | | | | SECOND QUARTER | | | | | | | THIRD QUARTER | | | | | | | FOURTH QUARTER | | | | | | | REMARKS | | |
|-------------|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------|----|----|
| | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 30 | 31 | 32 | 33 | | 34 | 35 |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

INITIAL _____
 CONDITIONS _____

NAME _____

| TERM NUMBER | FIRST QUARTER | | | | | | | SECOND QUARTER | | | | | | | THIRD QUARTER | | | | | | | FOURTH QUARTER | | | | | | | REMARKS | | |
|-------------|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------|----|----|
| | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 30 | 31 | 32 | 33 | | 34 | 35 |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

C-51

INITIAL _____
CONDITIONS _____

NAME _____

| TERM NUMBER | FIRST QUARTER | | | | | | | SECOND QUARTER | | | | | | | THIRD QUARTER | | | | | | | FOURTH QUARTER | | | | | | | REMARKS | | | | | | | | | | | |
|-------------|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------|----|----|----|----|--|--|--|--|--|--|--|
| | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 30 | 31 | 32 | 33 | | 34 | 35 | 36 | 37 | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

C-51

INITIAL _____
 CONDITIONS _____

NAME _____

| TERM NUMBER | FIRST QUARTER | | | | | | | SECOND QUARTER | | | | | | | THIRD QUARTER | | | | | | | FOURTH QUARTER | | | | | | | REMARKS | | | |
|-------------|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------|----|----|----|
| | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 30 | 31 | 32 | 33 | | 34 | 35 | 36 |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

C-51

INITIAL
CONDITIONS

NAME

| TERM NUMBER | FIRST QUARTER | | | | | | | SECOND QUARTER | | | | | | | THIRD QUARTER | | | | | | | FOURTH QUARTER | | | | | | | REMARKS | | | | | | |
|-------------|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------------|----|----|----|----|----|----|----------------|----|----|----|----|----|----|---------|----|----|----|----|--|--|
| | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 30 | 31 | 32 | 33 | | 34 | 35 | 36 | 37 | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| - | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

G-51

