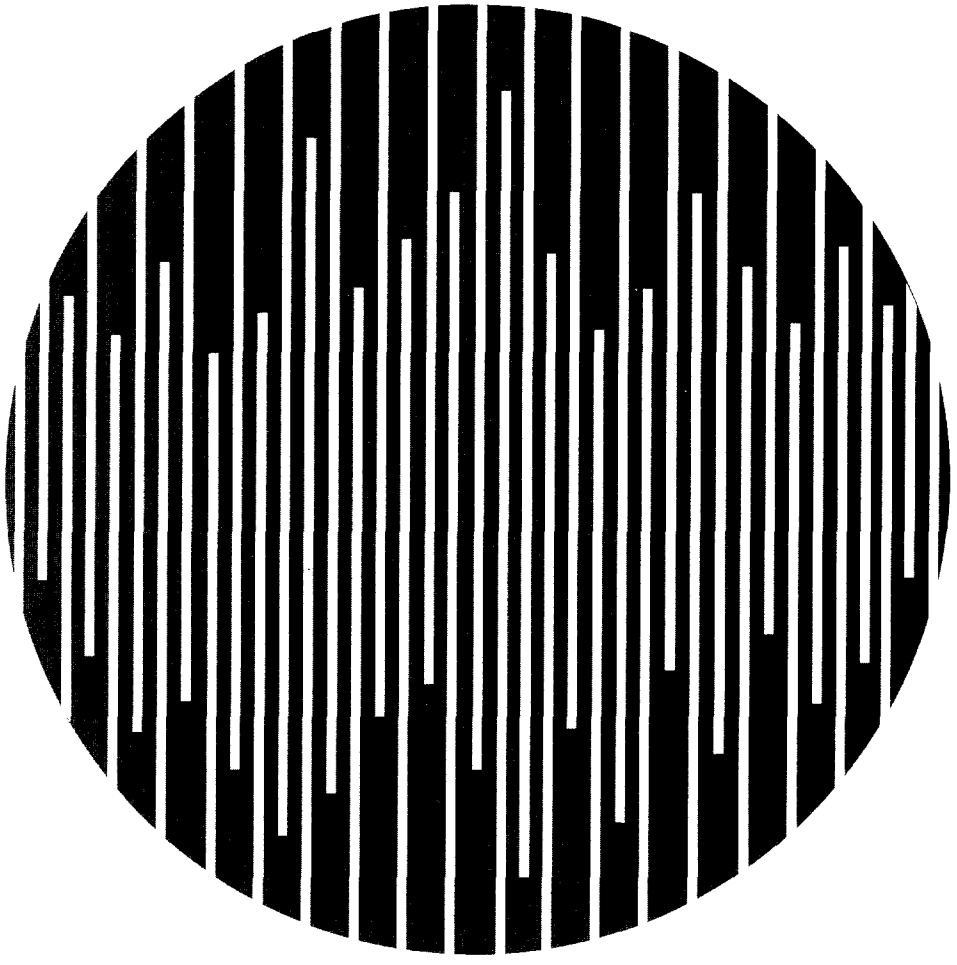
Modicon
584L Programmable Controller
Programming Guide

PI-584L-002 Rev. C



AEG

MODICON

Modicon
584L Programmable Controller
Programming Guide

PI-584L-002 Rev. C

Subject: Decription of the procedures and logic elements required to configure and program a 584L Programmable Controller.

December, 1984

Modicon, Inc. Industrial Automation Systems
One High Street
North Andover, Massachusetts 01845

PREFACE

The Modicon 584L Programmable Controller represents the finest in programmable control technology. As a replacement for relays and solid-state electronics, the controller offers an extensive array of control functions applicable for a variety of industries. Logic execution, timing, sequencing, calculations — all are efficiently performed for numerous industrial control applications.

The 584L Programming Guide is a comprehensive instruction manual of the controller's programming capabilities. From the basic concepts of network logic to the advanced subject of data transfer programming, the guide serves as an easy to use reference for each of the controller's program functions. Not only is each function thoroughly explained, but program examples are also provided to illustrate the function's use. Most importantly, this manual is written for the reader with a limited, if any, knowledge of relay ladder logic.

List of Related Documents:

584L System Planning and Installation Guide

P190 CRT Programmer User's Manual

584 PID User's Manual

584 Remote I/O User's Manual

584 ASCII Programming Guide

584 Register Access Panel User's Guide

J211 Redundancy System User's Guide

The following are trademarks of Modicon, Inc.:

Modicon	184	584L
Micro 84	384	884
Modbus	484	P180
Modvue	584	P190
Modway	584M	

Printed in U.S.A.

TABLE OF CONTENTS

FIGURES

CONTENTS (cont.)

TABLES

SECTION 1
INTRODUCTION TO RELAY LOGIC

The 584L Programmable Controller is programmed using basic relay logic programming language. The logic elements used include relay contacts, coils, references, registers, and function blocks. The controller's logic is structured into networks and segments, and programmed into the 584L via a P190 Programmer. The controller scans each network and solves the logic which can control the input to other logic in the program, or control an output (i.e., turning ON a switch, stopping a process, resetting a meter, etc.).

1.1 NETWORKS

A network is a set of interconnected logic elements which represents all or part of the user's 584L program. Each network has a maximum width of 10 nodes and a maximum length of 7 nodes. An eleventh column is provided exclusively for coils. See Figure 1-1.

A network can contain any combination of relay contacts, coils (eleventh column only), counters, timers, and arithmetic, data transfer (DX), and special function blocks. The logic can occupy the whole network area or just a portion of it.

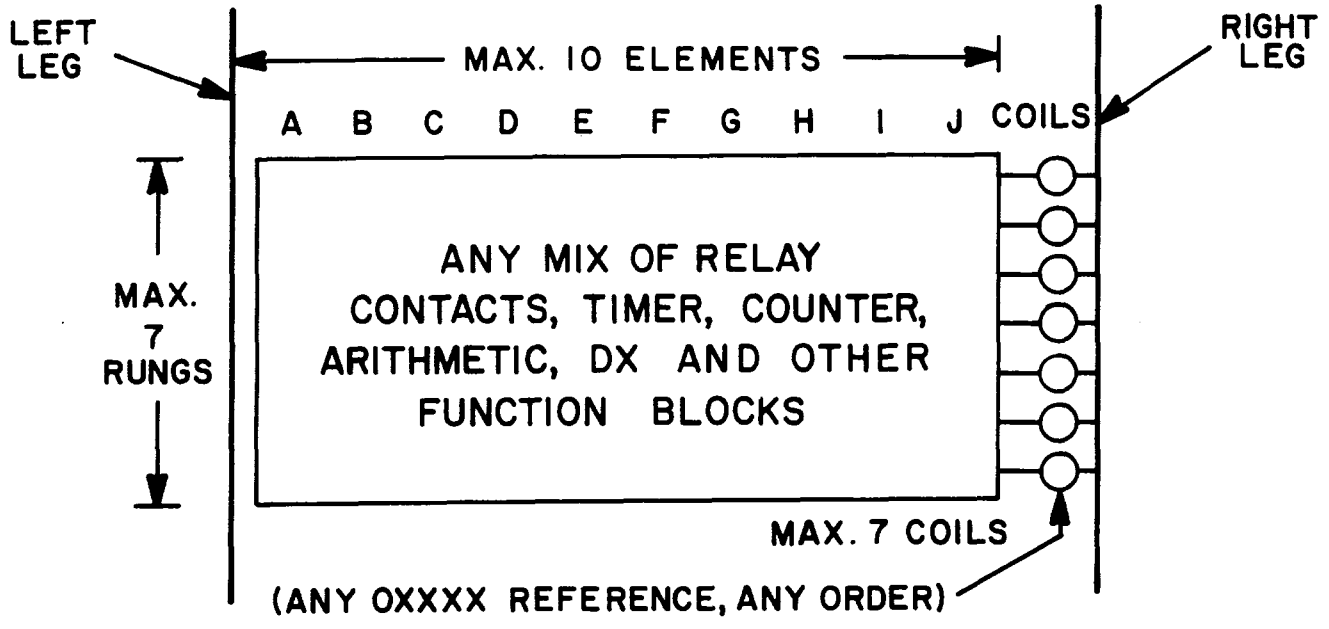


Figure 1-1. Network Parameters

1.2 SEGMENTS

Networks are separated into groups called segments. Each segment controls two I/O channels — 256 inputs and 256 outputs; 128 inputs and 128 outputs for each channel. The number of segments available is equal to the number of I/O channels divided by two. See Appendix A to determine the number of I/O channels needed for the 584L program and thereby the number of segments available.

The logic contained in each segment should control the I/O Channels it represents. For example, if logic in segment 3 controls an output in Channel 1, the output is not activated until the controller starts its next scan and scans segment 1. If the logic for that output had been placed in segment 1, the output would have been activated on the same scan.

1.3 REFERENCES

Reference numbers are used to identify relay contacts, coils, inputs, outputs, and registers. There are four types of references; each one has a different code digit to identify which type it is. This digit is the first of five consecutive digits. The reference types and their functions are listed in Table 1-1.

Table 1-1. References

0XXXX — Coil/Discrete Output

- A discrete (ON/OFF) signal that is controlled by logic.
- Can be used to drive a real output through an output module.
- Can be used internally to drive one or more contacts in user logic.

1XXXX — Discrete Input

- Status of the input is controlled by an input module.
- Used to drive contacts in user logic.
- Can be used repeatedly in the program.

30XXX — Input Register

- A numerical input from an external source (i.e., thumbwheel, analog signal, or high speed counter).
- Sixteen consecutive discrete signals.
- Can be binary or binary coded decimal (BCD).

4XXXX — Holding/Output Register

- Used to store numerical information, decimal or binary, in the controller.
- Can output numerical information to an output module.

NOTE

These numbers refer to actual registers within the controller which contain numerical values or ON/OFF conditions. An X is any digit, 0 through 9; however, it may have a specific limit (e.g., max. 5) as designated in the configuration table.

1.4 RELAY CONTACTS

The relay contact is the basic programming element. It can be referenced to either a logic coil (0XXXX) or a discrete input (1XXXX). The contact is opened, no power passing through, or closed, power passing through, when a certain condition exists (i.e., logic coil is energized or de-energized; input signal is ON or OFF).

Relay contacts can be normally open, normally closed, or transitional.

1.4.1 Normal Contacts

The two most commonly used contacts are:

Normally Open (NO) Contact 

Normally Closed (NC) Contact 

When the coil or discrete input to which a contact is referenced is ON, the normally open (NO) contact is closed and passes power, and the normally closed (NC) contact is open and does not pass power. When the coil or discrete input is OFF, the NO contact is open and does not pass power, and the NC contact is closed and passes power. See Table 1-2.

Table 1-2. Normally Open and Normally Closed Contacts

	NO Contact	NC Contact
Coil or Discrete Input is ON	passes power	does not pass power
Coil or Discrete Input is OFF	does not pass power	passes power

1.4.2 Transitional Contacts

A transitional contact is turned ON by the transition, OFF to ON or ON to OFF, of the coil or discrete input to which it is referenced. It is not affected by the ON or OFF state of the logic coil or discrete input after the transition.

Sometimes it is necessary for a function to be performed only once. In this case a transitional contact is used because it only transitions once each scan.

Since there are two different transitions (OFF to ON and ON to OFF), there are two transitional contacts:

Transitional Contact (OFF to ON) 

Transitional Contact (ON to OFF) 

1.4.3 Inserting Contacts

A relay contact is inserted into a program by positioning the cursor over the desired location, entering a 0XXXX or 1XXXX reference into the Assembly Register (AR) and pressing the appropriate software label key. To change the type of relay

contact, position the cursor over the contact to be changed and press the desired software label key. To change the reference numbers below a contact, position the cursor over the contact, enter a new value into the AR, and press the ENTER key.

1.5 VERTICAL AND HORIZONTAL SHORTS

Vertical and horizontal shorts are simply straight line connections between contacts.

Vertical shorts are used to connect contacts and function blocks one above the other in a network. Vertical shorts can also be used to connect inputs or outputs in a function block to create either/or conditions. When two contacts are connected by vertical shorts, a vertical short on each side, power is allowed to pass through if either, or both, contacts receive power.

Enter a vertical short into a program by positioning the cursor over the node to the left of and above the location desired for the short and press the VERTICAL SHORT software label key. A vertical short is cleared by pressing the VERTICAL OPEN software label key. A vertical short does not take any user memory.

Horizontal shorts are used in combination with vertical shorts to expand logic within a network without breaking the power flow. They can be used to create either/or conditions using basic relay contacts. For example, if one line of logic contains two relay contacts, and the line below it only contains one contact, a horizontal short is placed beside the single contact. See Figure 1-2. A vertical short is used to connect the horizontal short to the top logic line. Power passes through to energize the coil if the two top contacts are energized or if the bottom contact is energized.

Enter a horizontal short into a program by positioning the cursor over the node desired for the short and pressing the " ——— " software label key. To clear a horizontal short, press the DELETE NODE software label key. A horizontal short uses two words of memory.

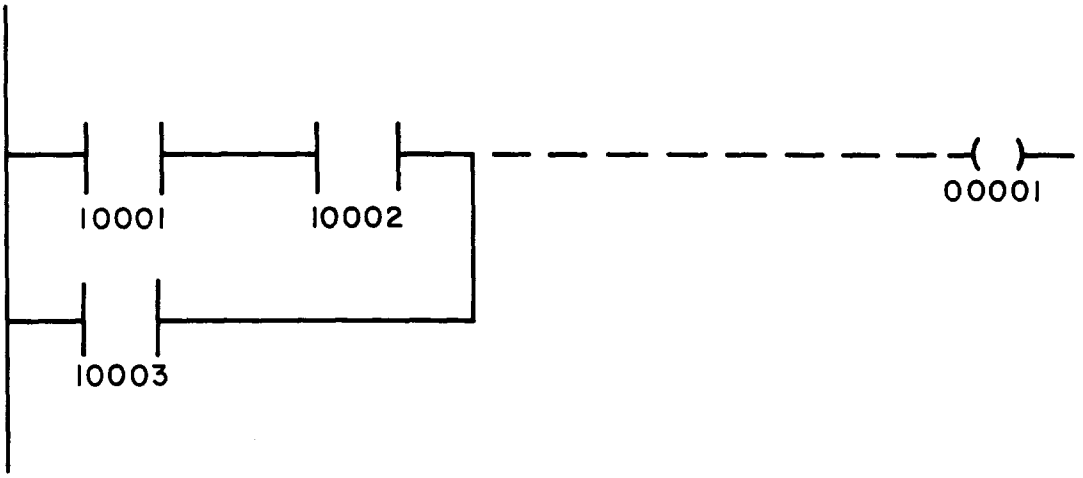


Figure 1-2. Shorts

1.6 COILS

A coil is used to activate logic within the user's program and/or to control an output circuit. It is represented by a 0XXXX reference number and either of two symbols.

A Normal Coil, —( )— , is turned OFF if power is removed and later restored.

A Latched Coil, —(L)— , retains its previous state if power is removed and later restored.

Coils are located in the far right (eleventh) column of a network. Each network can contain a maximum of seven coils.

Each 0XXXX reference can be used as a coil only once, but can be referenced to any number of relay contacts. Output coils are generally given the lower 0XXXX reference numbers and internal coils are given the higher 0XXXX reference numbers. It is not an error to intermix output and internal coil reference numbers since the 584L is able to output any valid coil.

A logic coil is inserted into a program in the same way as a relay contact except that the cursor does not have to be over column eleven. The cursor can be directly beside the last logic element in a row. When the coil software label key is pressed, dashed lines are inserted and the coil is placed in the eleventh column. The only reference allowed is a 0XXXX reference, unlike relay contacts which allow 0XXXX or 1XXXX references.

1.7 DISABLE/ENABLE

Any logic coil or discrete input in the user's program can be disabled and forced ON or OFF. To disable a coil or discrete input, do the following:

1. Ensure that Memory Protect is OFF on both the P190 Programmer and the 584L PC.

2. Position the cursor at the bottom of the P190 screen.

3. Enter the coil reference number or discrete input reference number into the Assembly Register (AR).

4. Press the ERASE/GET key.

The reference number appears at the cursor position with its state, ON or OFF, and the following software labels appear on the screen:

ENABLE	DISABLE	FORCE ON	FORCE OFF

5. Press the DISABLE software label key.

The coil or input is now DISABLED ON or DISABLED OFF, depending on its state when the DISABLE key was pressed.

Press the FORCE ON or FORCE OFF software label keys to change the state of the coil or input. Press the ENABLE software label key to enable the coil or input. If the coil or input is enabled, it cannot be forced ON or forced OFF; it retains the state it had when the ENABLE software label key was pressed.

Disabling a coil or input causes the programmed logic to bypass that coil or input. The coil's state or the input's state is now controlled by the user through the FORCE ON and FORCE OFF software label keys. Memory Protect must be OFF.

WARNING

There are exceptions to the logic bypassing a disabled coil. All the DX function blocks which allow a coil in the destination node override the coil if it is disabled. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, when the coil's state can change as a result of a particular function.

1.8 CONTROLLER SCAN

The 584L PC scans the networks in the user's program to solve the logic. The scan starts at the top left of a network and goes from top to bottom in each column working from the left column towards the right column. See Figure 1-3.

Although coils are placed in the eleventh column and are solved there, if the logic controlling them is in column ten, coils can be solved in other columns as well. If the logic controlling a particular coil is in column 6 and dashed lines connect the logic to the coil, that coil is solved in column 7. For example, in Figure 1-4, coil 00036 is solved before contacts I0012 and I0024, and coil 00033.

The scan starts in segment 1 network 1 and continues in segment 1 network 2. When the scan reaches the end of segment 1, all the logic for segment 1 is solved and the inputs are read for segment 2. The scan continues at the first network of segment 2. Segment 1 outputs are serviced and segment 2 logic is scanned and solved. The inputs are read for segment 3. This is illustrated in Figure 1-5.

The controller continues its scan until all the segments and networks have been scanned. It then returns to segment 1 network 1 and services the last network's outputs, thereby starting the cycle over again.

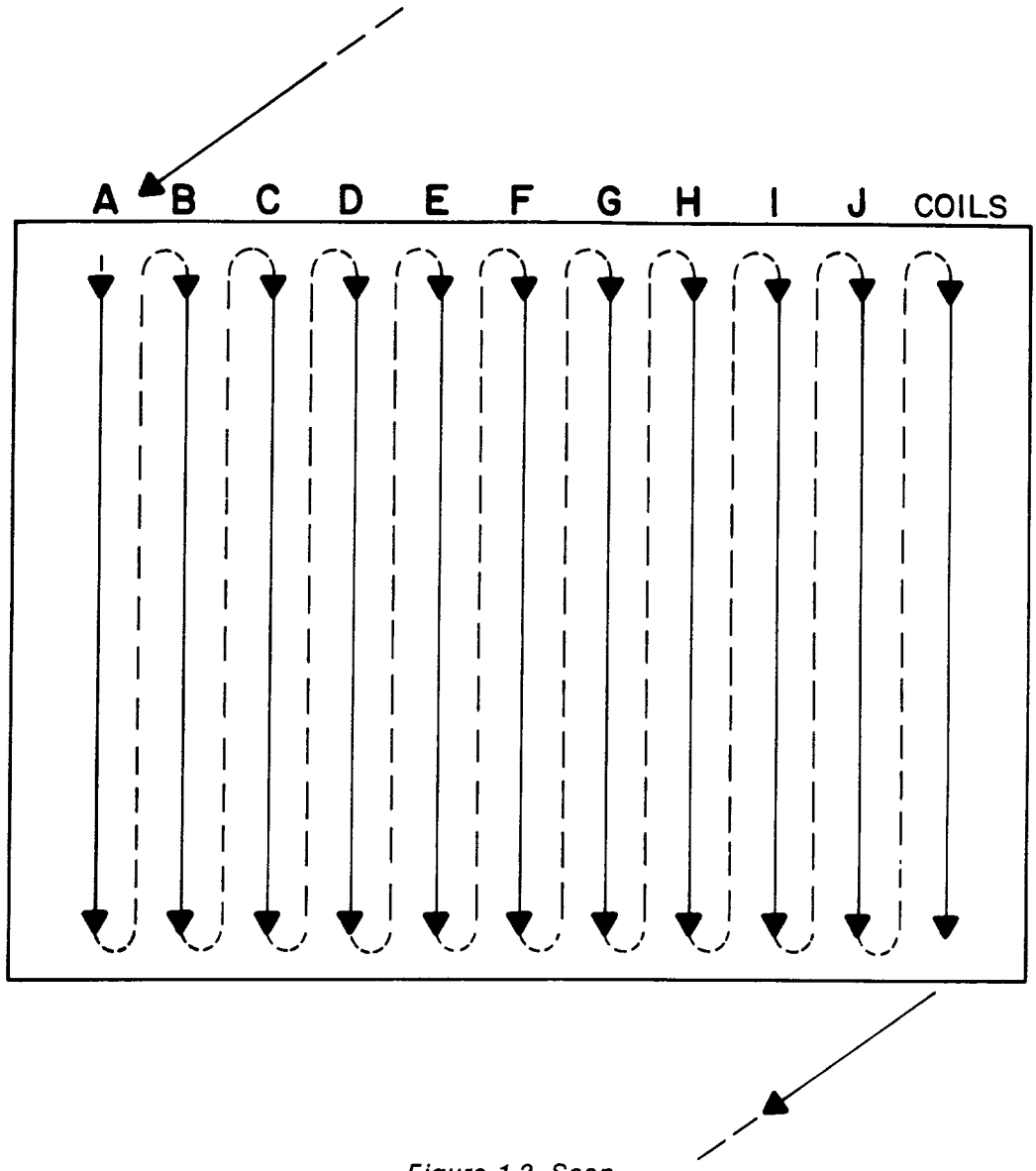


Figure 1-3. Scan

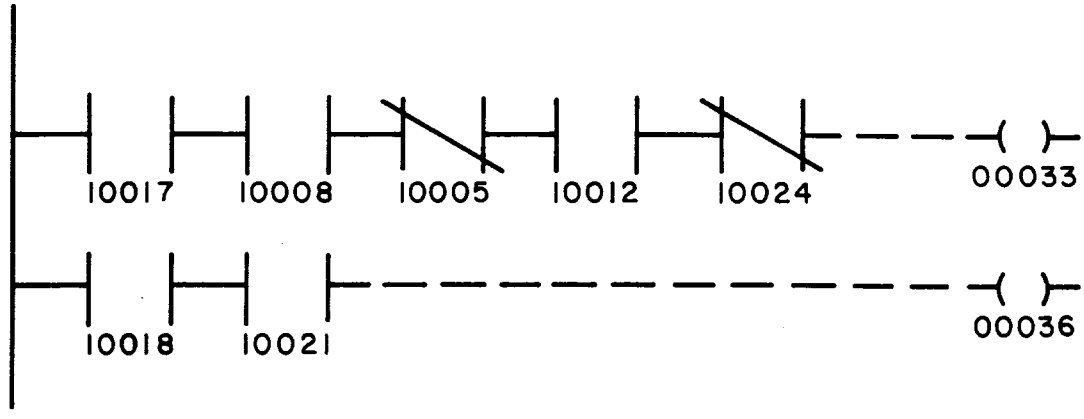


Figure 1-4. Order of Coil Solving

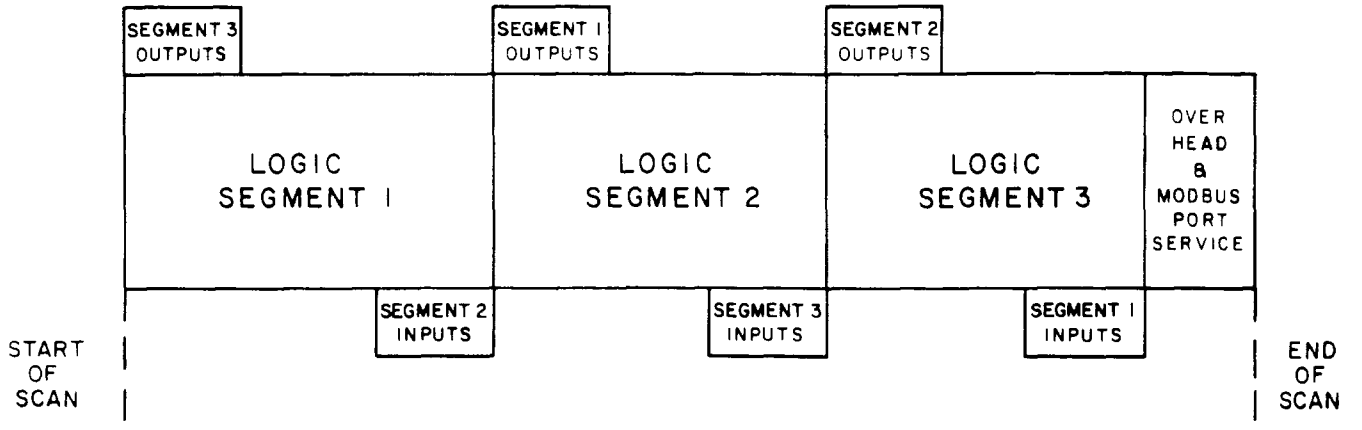


Figure I-5. Sample Logic Scan with I/O Servicing

1.9 PROGRAMMING THE 584L

User programs for the 584L PC are entered using a P190 Programmer. See the P190 CRT Programming User's Manual for installation instructions. The following must be done to enter a program:

1. The 584L PC must be configured to the user's specifications. Appendix A provides the Configuration and Traffic Cop information needed to accomplish this.

2. Insert the 584L Programmer Tape (T584-001) into the PI90 tape drive (front upper right corner).

3. Press the red INIT and INIT LOCK keys on the P190 panel simultaneously.

4. Enter a unit number into the AR. The unit number can be within the range of 1 to 247. It refers to the location of the controller in the data line communicating with the P190.

5. Press the ATTACH software label key. The following PC status software labels appear on the screen:

These software label keys are used to change the 584L PC's status and can be reached at any time while programming by pressing the SHIFT and RESET/EXIT keys simultaneously.

6. Stop the 584L by pressing the STOP 584L software label key.

7. Press the EXIT key followed by the START NEXT key to begin entering a program.

A power rail appears on the left of the P190 screen and the following software labels are displayed:

Each software label key, when pressed, brings up another set of software labels. Figure 1-6 contains all the software label keys needed to enter a program illustrated in a broken down flow chart. To return to the original set of software labels, shown at the top of Figure I-6, from any level of software labels except the PC status level, press the CHG NODE (Change Node) key on the P190 panel.

To start a network, press the START NEXT key. A power rail appears on the left side of the P190 screen and the cursor is located in the top left node. A network can now be entered.

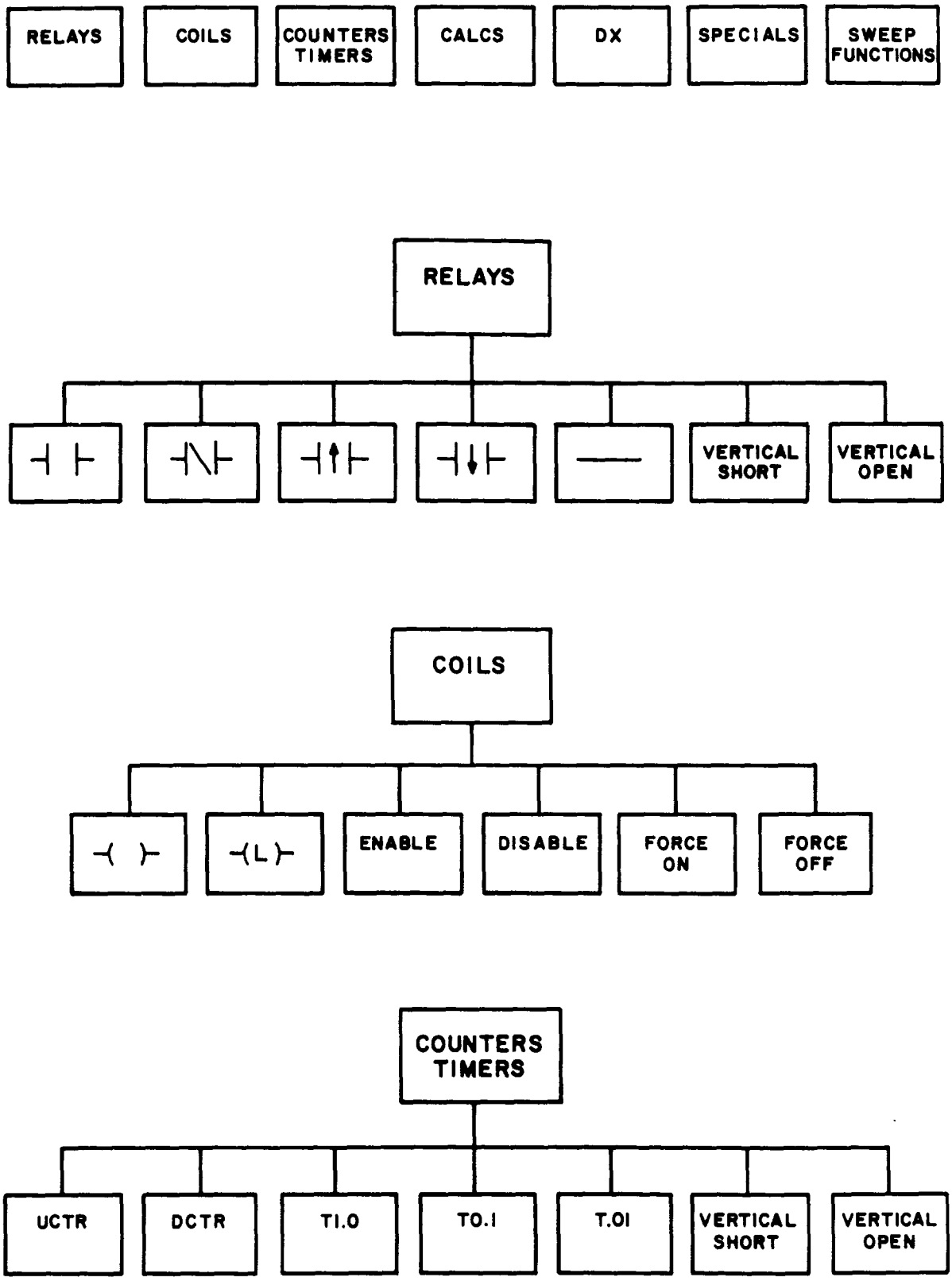


Figure 1-6. Software Label Flow Diagram

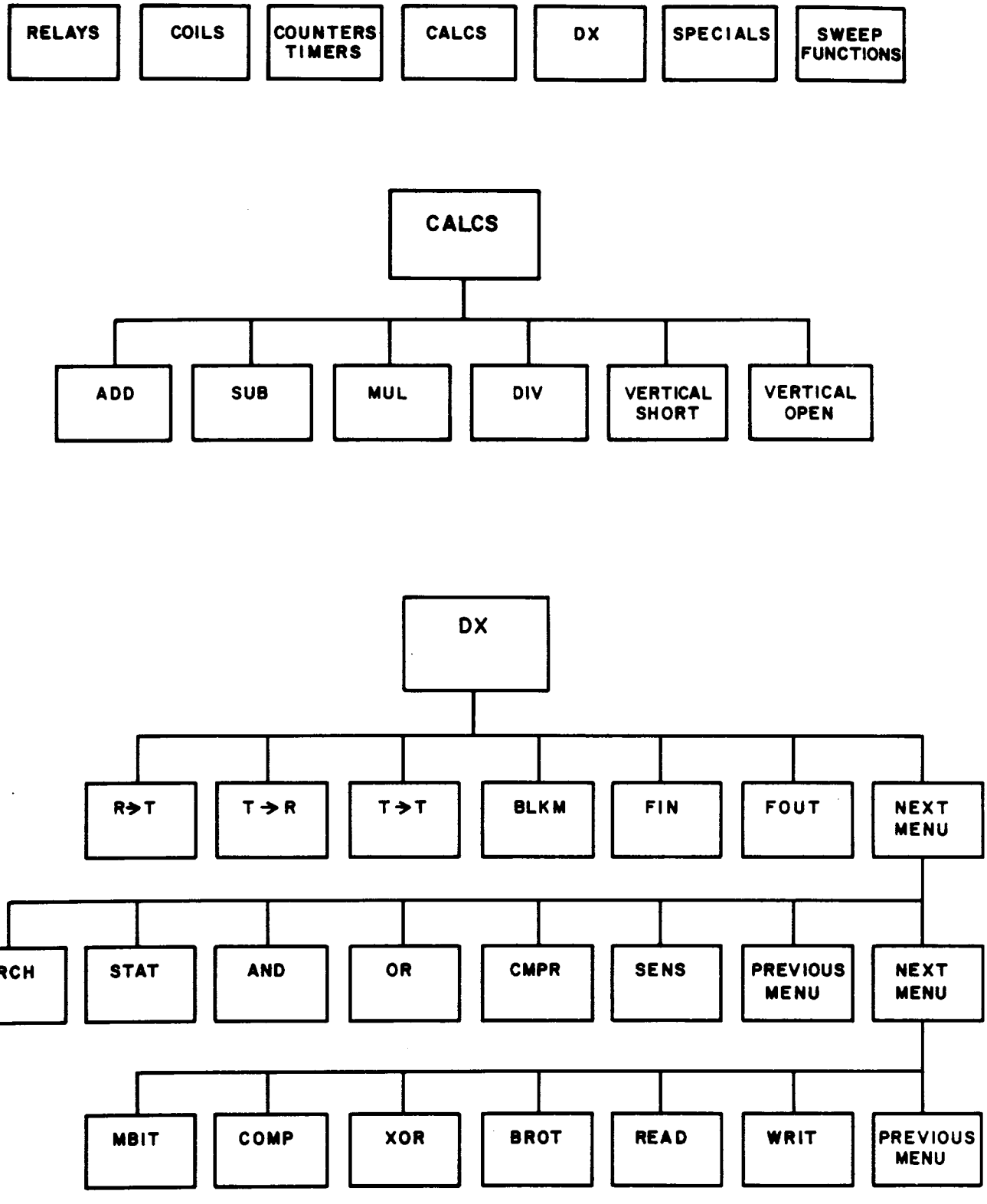


Figure 1-6. Software Label Flow Diagram (Cont.)

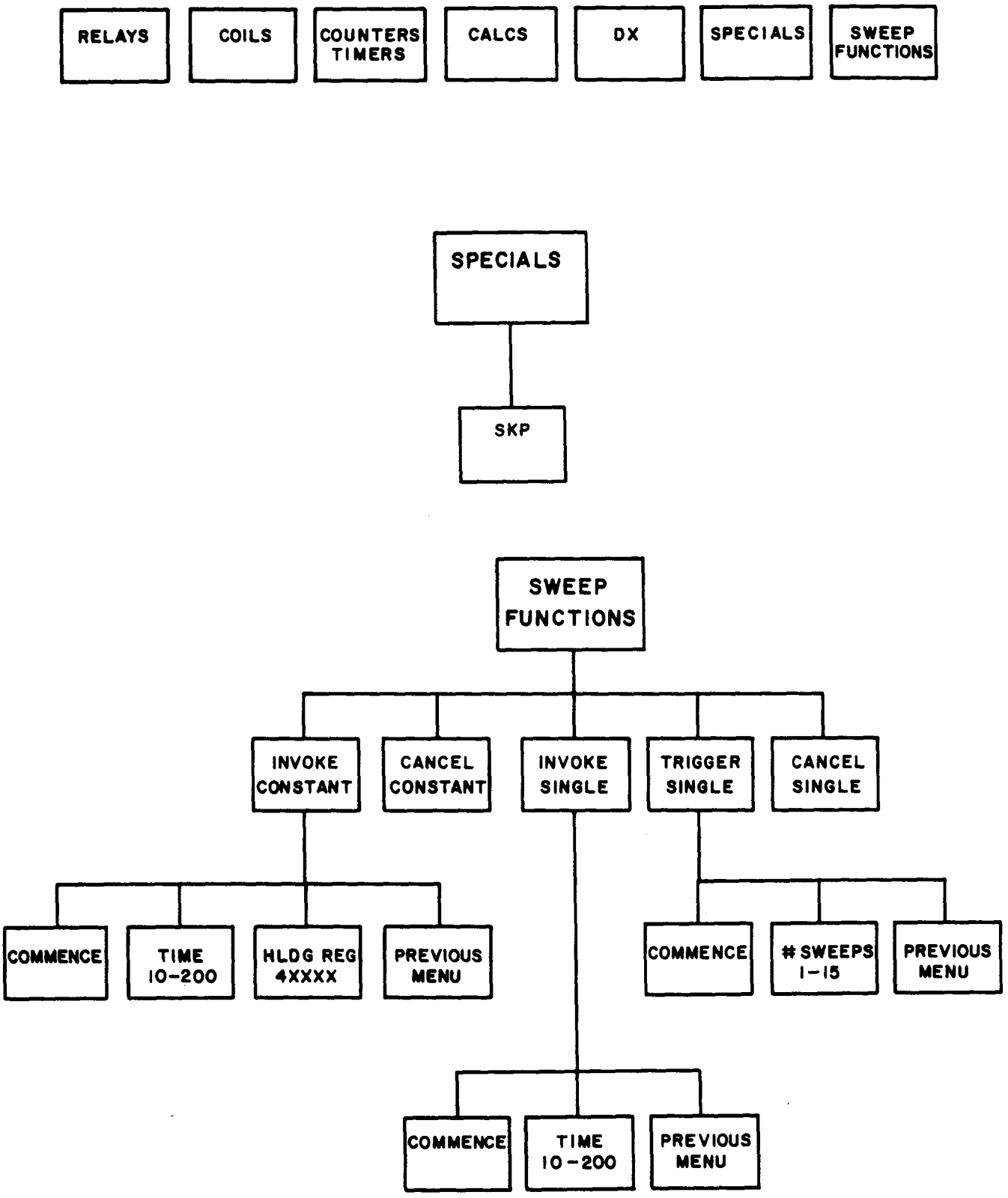


Figure 1-6. Software Label Flow Diagram (Cont.)

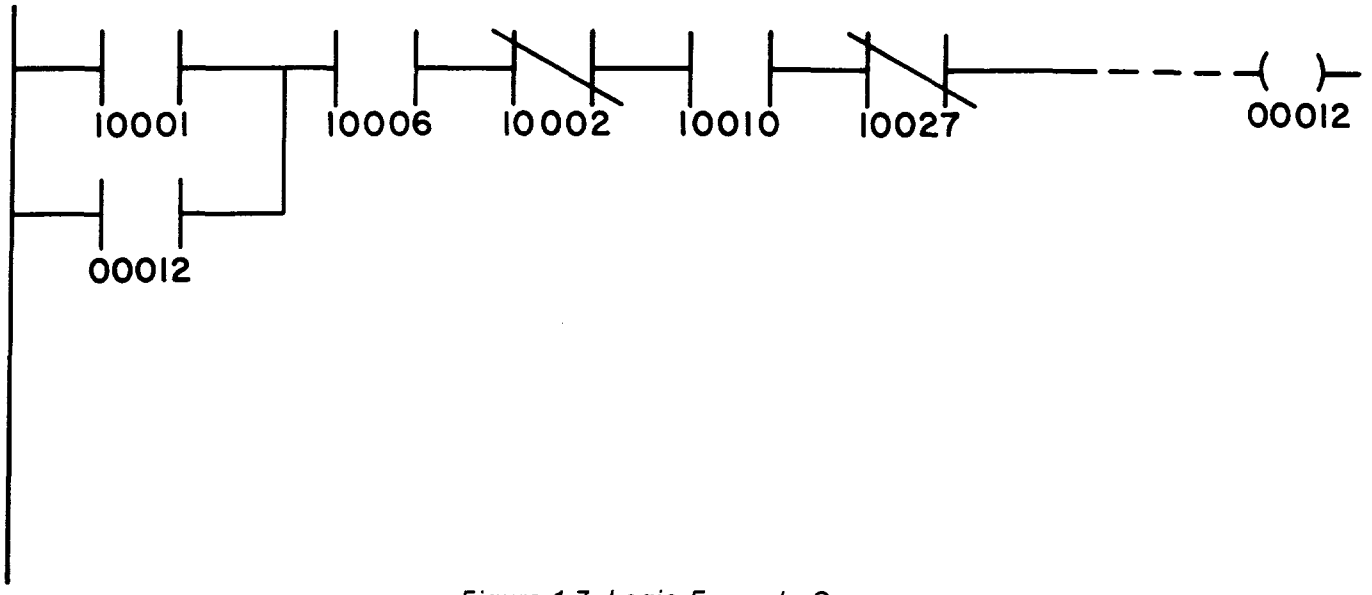


Figure 1-7. Logic Example One

Enter the logic in Figure 1-7, as follows:

1. Press the RELAYS software label key.

   Software labels of the relay contacts appear on the screen.

2. Enter 10001 into the Assembly Register (AR).
3. Press ⊣ ⊢ .
4. Press → .
5. Enter 10006 into the AR.
6. Press ⊣ ⊢ .
7. Press → .
8. Enter 10002 into the AR.
9. Press ⊣/⊢ .
10. Press → .
11. Enter 10010 into the AR.

12. Press —| |— .
13. Press → .
14. Enter 10027 into the AR.
15. Press —|/|— .
16. Press → .
17. Press the CHG NODE (Change Node) key.

    The original set of software labels appears.
18. Press COILS.

    Software labels of coils appear on the screen.
19. Enter 00012 into the AR.
20. Press —( ) .
21. Press → .

    The cursor moves to the first node in the same row (wrap-around).
22. Press VERTICAL SHORT.
23. Press ↓ .

    (Enter 00012 into the AR.)
24. Press —| |— .

    This network is complete. To start a new network press the START NEXT key.

NOTE

To get to a previous network, press the SHIFT and GET PREV/GET NEXT keys simultaneously. To go to the next network, press the GET PREV/GET NEXT key.

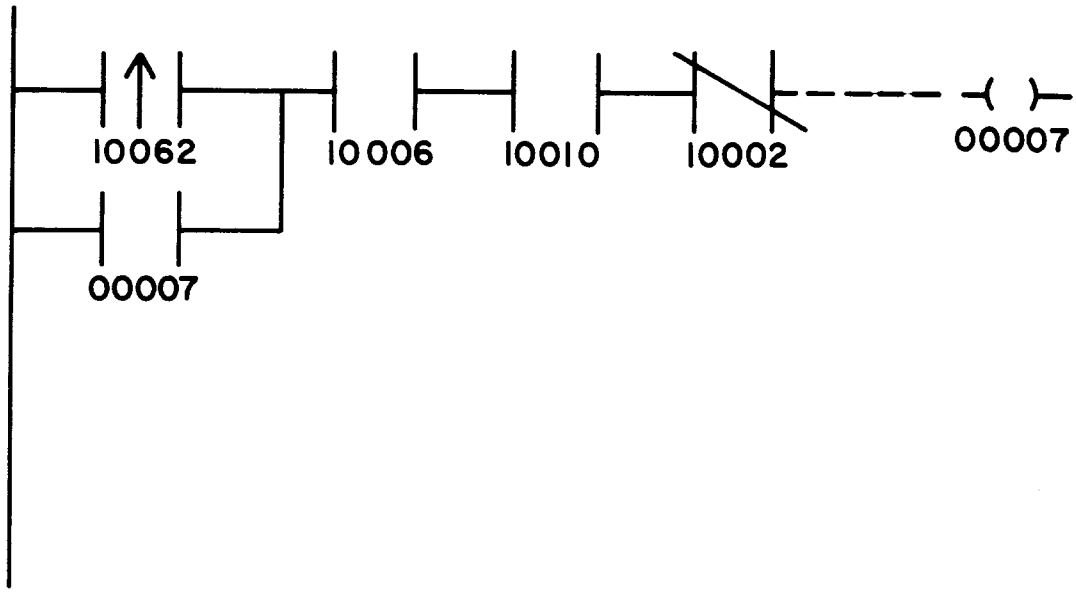


Figure 1-8. Logic Example Two

Enter the logic in Figure 1-8 as follows:

1. Press the RELAYS software label key. (If the RELAYS software label is not displayed, press the CHG NODE key first.)
2. Enter 10062 into the AR.
3. Press ⊣↑⊢ .
4. Press → .
5. Enter 10006 into the AR.
6. Press ⊣ ⊢ .
7. Press → .
8. Enter 10010 into the AR.
9. Press ⊣ ⊢ .
10. Press → .
11. Enter 10002
12. Press ⊣/⊢ .
13. Press → .
14. Press the CHG NODE key.

15. Press COILS.
16. Enter 00007 into the AR.
17. Press —( )— .
18. Press → .
19. Press RELAYS.
20. Press VERTICAL SHORT.
21. Press ↓ .

    (Enter 00007 into the AR.)
22. Press —| |— .

This network is complete.

1.10 BASIC LOGIC EXAMPLES

1.10.1 Oscillator

The following paragraphs provide an explanation of the logic illustrated in Figure 1-9 to program an oscillator.

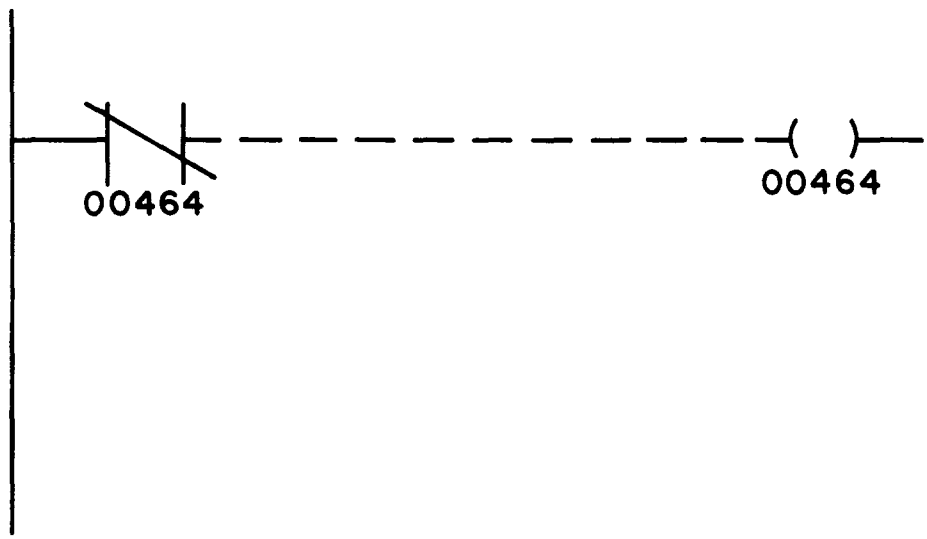


Figure 1-9. Oscillator

Many applications require the use of a signal which oscillates ON-OFF-ON-OFF. (On the first scan the coil is ON; on the second scan the coil is OFF; on the next scan the coil is ON; OFF, ON, OFF, etc.). Since the signal is ON every other scan, it can be used to flash lights or cause something to take place every other scan.

An oscillator is made using a logic coil and a normally closed contact with the same reference number.

1.10.2 Cascaded Logic

Very often, with a complex control function, more than ten nodes are needed to satisfy the function. In such a case, an internal coil is used to represent a partial result. To continue the logic flow, use a contact which is referenced to the internal coil. The second series of nodes can be ended with an output coil to represent the result of the logic or an internal coil to extend the logic again. The logic can be extended within a network and/or to another network.

NOTE

It is best to use a new network to continue cascaded logic. If a new network is not used, resulting logic solutions may be different than those desired. This is due to the order in which logic is solved.

The following paragraphs provide an explanation of the cascaded logic illustrated in Figure 1-10.

The first ten nodes in the top network illustrated in Figure 1-10 activate coil 00059. The logic is extended by referencing coil 00059 to the normally open contact in the next network. The logic is output by latched coil 00032.

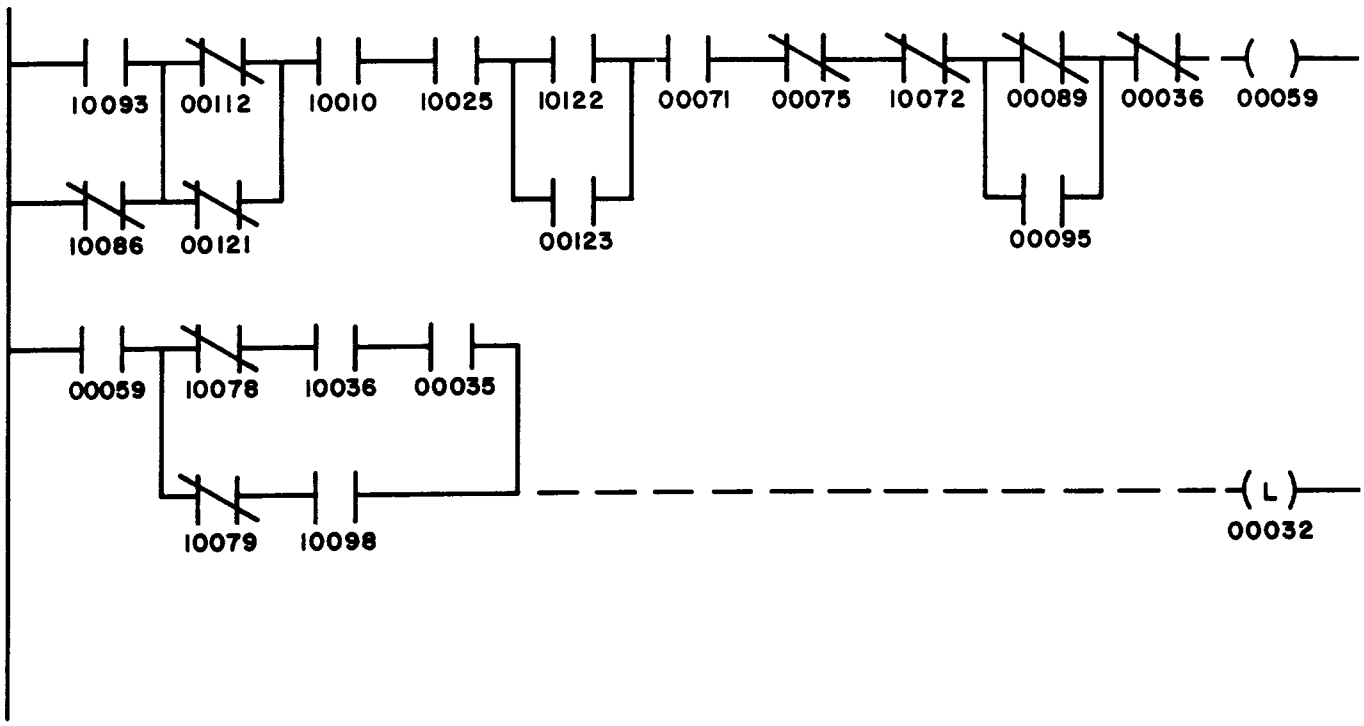


Figure 1-10. Cascaded Logic

SECTION 2
BASIC PROGRAMMING FUNCTIONS

This section describes how to edit a network as well as how to program a counter or timer. Instructions are also provided to perform arithmetic functions. These functions can be used independently or with the more advanced functions which are introduced in Sections 3 through 5.

Each function block in this section uses registers to hold and store data. A register is a location in the controller's memory in which a numerical value is stored. This value can be binary or binary coded decimal (BCD). In a 584L PC, the maximum decimal value is 9999 and the maximum number of bits is sixteen.

A function block is inserted into a program by entering the value desired for the top node of the function block into the AR and pressing the appropriate software label key. The function block appears with the desired value in the top node, and question marks in the other node(s).

How to edit a function block is explained in Table 2-1.

Table 2-1. Editing a Function Block

Type of Edit	Instructions
To replace the question marks with a value.	1. Position the cursor over the question marks. 2. Enter a value into the AR. 3. Press the ENTER key.
To change a value in a function block.	1. Position the cursor over the value. 2. Enter the new value into the AR. 3. Press the ENTER key.
To change the type of function block.	1. Position the cursor in the top node of the block. 2. Press the desired software label key.
To delete a function block.	1. Position the cursor in the top node of the block. 2. Press the DELETE NODE key.

2.1 EDITING

Once a whole or partial network has been entered, spacing changes can be made. This section highlights the four types of changes.

To make changes in a network:

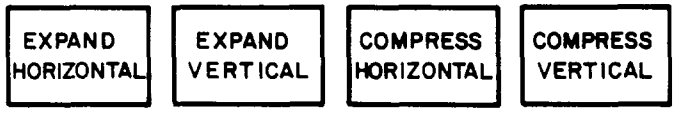
1. Ensure that the Memory Protect key on the P190 is OFF and that Memory Protect on the 584L is OFF.

2. Display the network needing changes on the P190 screen.

3. Press the EXIT key.

4. Press the EDIT NETWORK software label key.

The following software labels are displayed:



To return to the programming software keys (e.g., RELAYS, COILS, etc.), press the CHG NODE (Change Node) key.

The Memory Protect keylock is located on the lower right front of the P190 Programmer, beneath the tape drive. The Memory Protect key is used to prevent accidental or unauthorized changes to the 584L PC's memory. If the key is in a vertical position, as shown in Figure 2-1a, Memory Protect is ON and the user's logic cannot be altered by an external device, only examined.

The key is turned clockwise to the unlock position as shown in Figure 2-1b. In this case, Memory Protect is OFF and changes can be made to the user's logic.

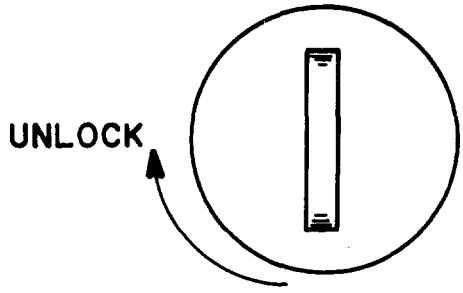


Figure 2-1a

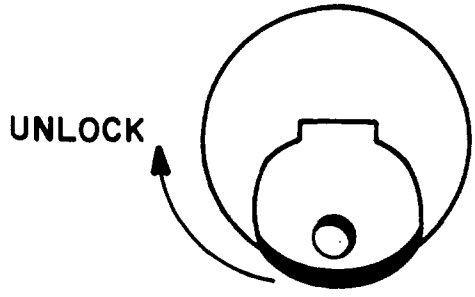


Figure 2-1b

Figure 2-1. Memory Protect

2.1.1 Expand Horizontal

This software label key is used to create a column in a network. When the key is pressed, the programming elements at the cursor position, and below, above, and to the right of the cursor, are moved one column to the right.

The logic in Figure 2-2 shows a network before pressing the EXPAND HORIZONTAL software label key. The shaded area is the cursor. The revised network is shown in Figure 2-3.

NOTE

The cursor can be placed anywhere in the column to be created. It does not have to be over an element. There must be a blank column at, or to the right of, the cursor or the EXPAND HORIZONTAL is not allowed.

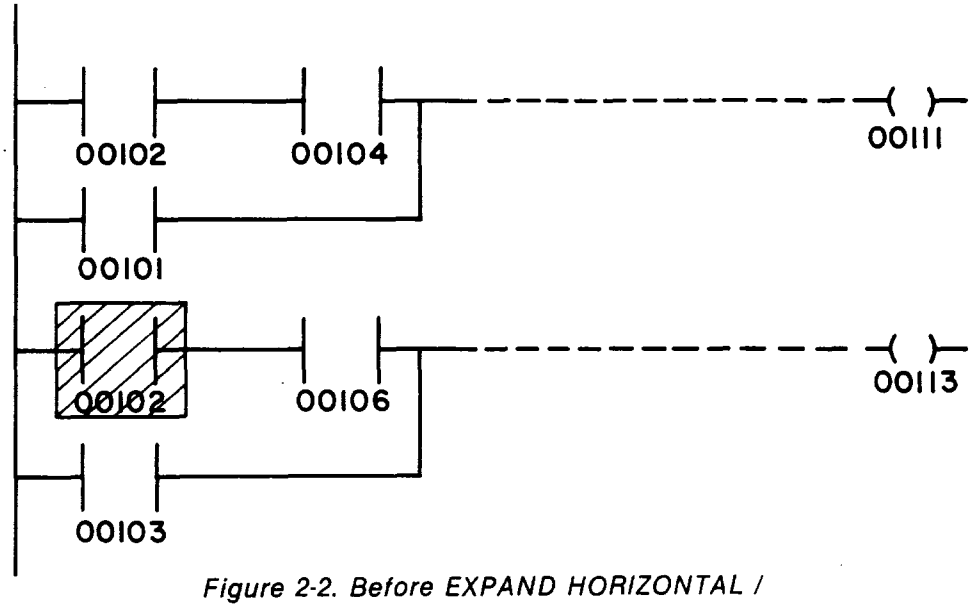


Figure 2-2. Before EXPAND HORIZONTAL / After COMPRESS HORIZONTAL

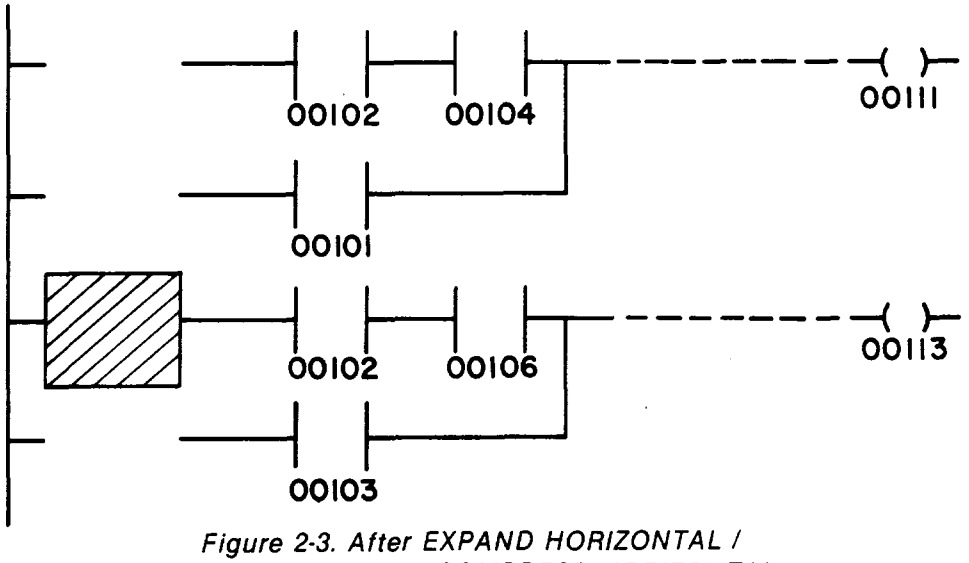


Figure 2-3. After EXPAND HORIZONTAL / Before COMPRESS HORIZONTAL

2.1.2 Compress Horizontal

The result of pressing this software label key is the opposite of pressing EXPAND HORIZONTAL. When the key is pressed, the column in which the cursor is located is deleted.

The logic in Figure 2-3 shows a network before pressing the COMPRESS HORIZONTAL software label key. The revised network is shown in Figure 2-2.

NOTE

COMPRESS HORIZONTAL is not allowed if the cursor is located over a programming element or if the column to be deleted contains a programming element.

2.1.3 Expand Vertical

This software label key is used to create a row in a network. When the key is pressed, the programming elements at the cursor position, and below, to the right, and to the left of the cursor, are moved one row down.

The logic in Figure 2-4 shows a network before pressing the EXPAND VERTICAL software label key. The revised network is shown in Figure 2-5.

NOTE

The cursor can be placed anywhere in the row to be created. It does not have to be over an element. Also, there must be a blank row under the last row containing programming elements or the EXPAND VERTICAL is not allowed.

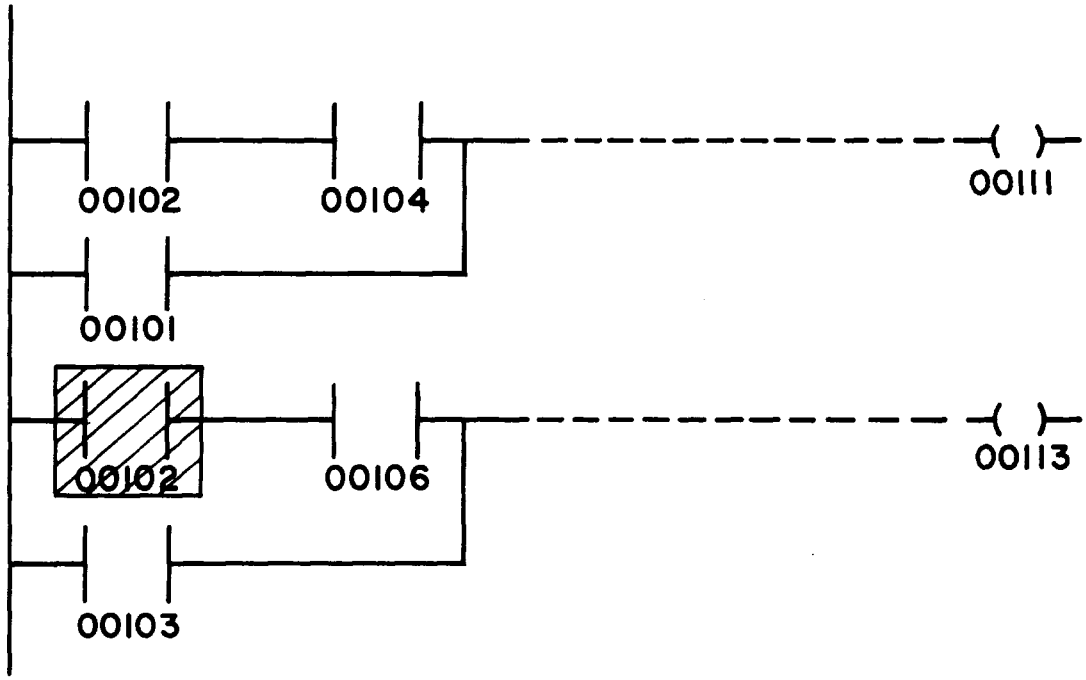


Figure 2-4. Before EXPAND VERTICAL / After COMPRESS VERTICAL

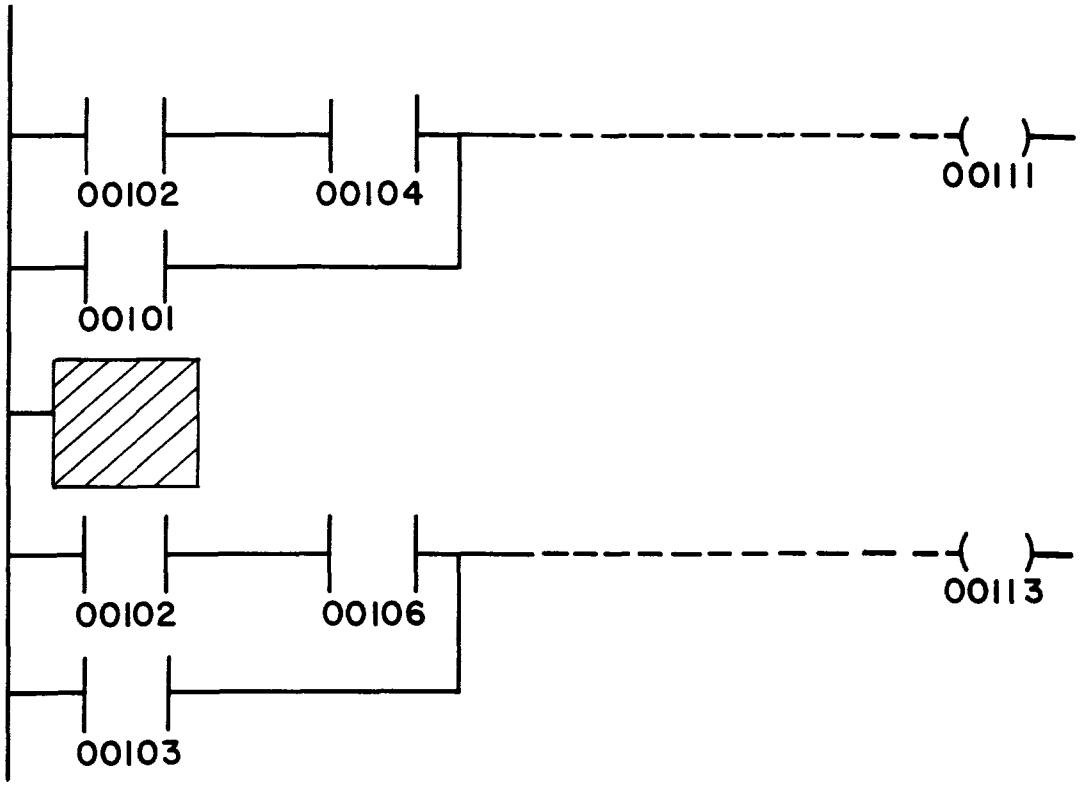


Figure 2-5. After EXPAND VERTICAL / Before COMPRESS VERTICAL

2.1.4 Compress Vertical

The result of pressing this software label key is the opposite of pressing EXPAND VERTICAL. When the key is pressed, the row in which the cursor is located is deleted.

The logic in Figure 2-5 shows a network before pressing the COMPRESS VERTICAL software label key. The revised network is shown in Figure 2-4.

NOTE

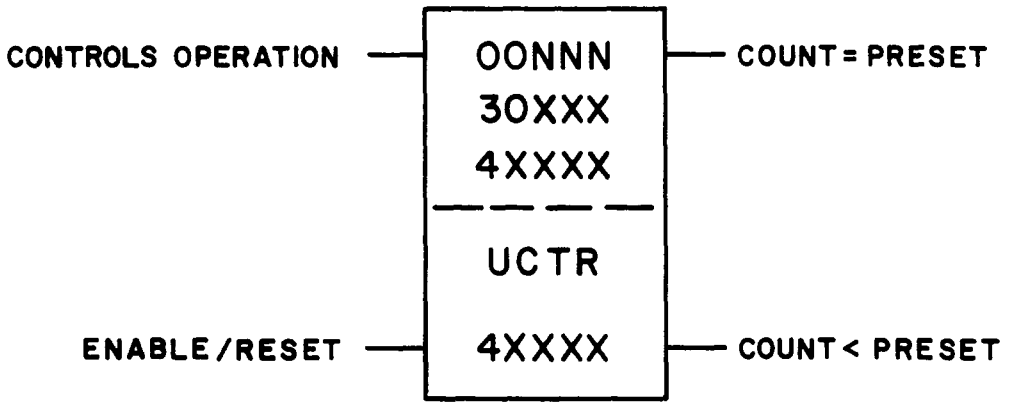
COMPRESS VERTICAL is not allowed if the cursor is placed over a programming element or if the row to be deleted contains a programming element.

2.2 COUNTERS

2.2.1 Up Counter (UCTR)

FUNCTION

The Up Counter function counts the OFF to ON transitions of the control input. This counter increases by one upon each positive transition of the control input.



FUNCTION BLOCK

- The top node contains the preset value for the counter. It can be a constant, up to 999 in a Level 1 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference. The contents of these register references or holding registers are the preset value.

- The bottom node contains the symbol UCTR and a unique 4XXXX holding register reference. This holding register contains the count value and increases, starting at zero, upon each OFF to ON transition of the control input.

NOTE

The controller sets the count value to the preset if an attempt is made to enter a value greater than the preset.

INPUTS

- The top input controls the operation. When it receives power, transitions from OFF to ON, the count value increases by one.

- The bottom input, when receiving power, enables the counter. When the input is not receiving power, the count value is reset to zero and any transitions of the top input are ignored.

OUTPUTS

- The top output passes power when the count value is equal to the preset value.

- The bottom output passes power when the count value is less than the preset value.

NOTE

Only one output passes power at a time.

EXAMPLE

The following paragraphs provide a detailed explanation of the logic used in Figure 2-6.

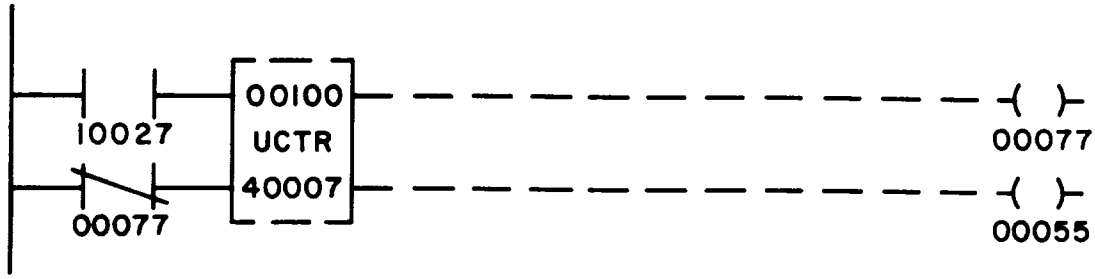


Figure 2-6. Up Counter

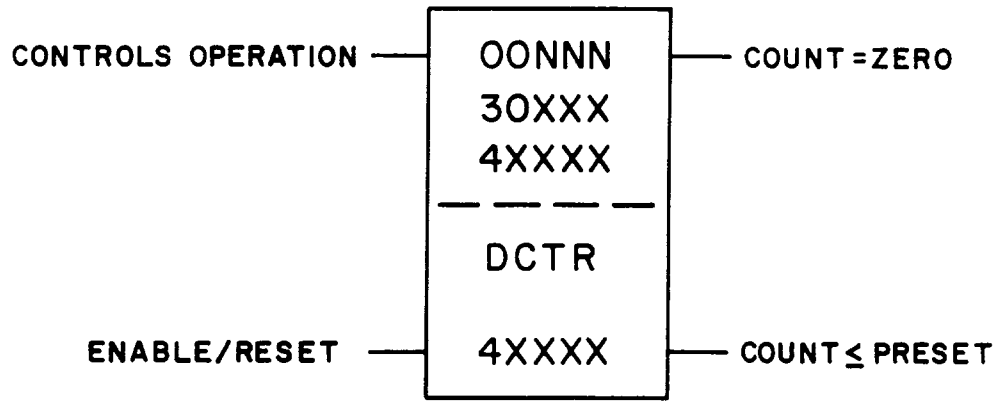
When input 10027 is energized, the top input receives power and, since the bottom input is also receiving power, the counter is enabled and counting begins.

Each time input 10027 transitions from OFF to ON, the value in register 40007 increases by one. When this value reaches 100, the top output passes power. Coil 00077 is energized and coil 00055 is de-energized. The bottom input loses power when coil 00077 is energized (the normally closed contact does not pass power), and the counter value is reset to zero.

On the next scan that input I0027 transitions OFF to ON, coil 00077 is de-energized, thereby energizing input 00077 and re-enabling the counter.

2.2.2 Down Counter (DCTR)

FUNCTION
The Down Counter function counts the OFF to ON transitions of the control input. The counter decreases by one upon each positive transition of the control input.



FUNCTION BLOCK

- The top node contains the preset value for the counter. It can be a constant, up to 999 in a Level 1 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference. The contents of this register are the preset values.

- The bottom node contains a unique 4XXXX holding register reference. This holding register contains the count value and decreases, starting at the preset value, upon each OFF to ON transition of the control input.

NOTE

The controller sets the count value to the preset if an attempt is made to enter a value greater than the preset.

INPUTS

- The top input controls the operation. When it receives power, transitions from OFF to ON, the count value decreases by one.

- The bottom input, when receiving power, enables the counter. When the input is not receiving power, the count value is reset to the preset value and any transitions of the top input are ignored.

OUTPUTS

- The top output passes power when the count value is equal to zero.

- The bottom output passes power when the count value is less than or equal to the preset value but not equal to zero.

NOTE

Only one output passes power at a time.

EXAMPLE

The following paragraphs provide a detailed explanation of the logic used in Figure 2-7.

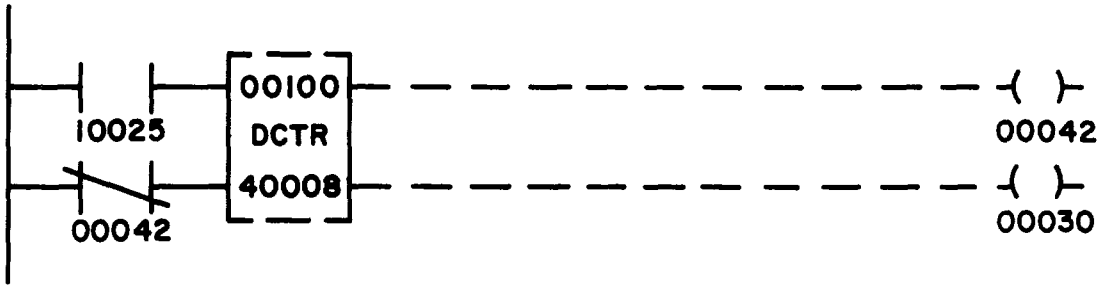


Figure 2-7. Down Counter

When input 10025 is energized, the top input receives power and, since the bottom input is also receiving power, the counter is enabled and down-counting (decreasing) begins.

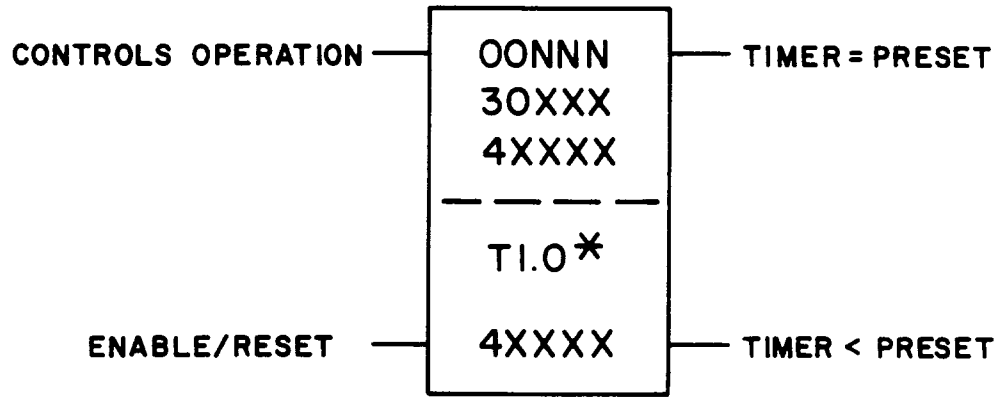
Each time input 10025 transitions from OFF to ON, the value in register 40008 decreases by one (this value starts at the preset). When this value reaches zero, the top output passes power. Coil 00042 is energized and coil 00030 is de-energized. The bottom input loses power when coil 00042 is energized (the normally closed contact does not pass power), and the counter value is reset to the preset (100).

On the next scan input that 10025 transitions OFF to ON, coil 00042 is de-energized, thereby energizing input 00042 and re-enabling the counter.

2.3 TIMER (T1.0*)

FUNCTION

The Timer function uses any one of three clocks in the 584L PC to record time. The 584L is capable of counting time in seconds, tenths of seconds, and hundredths of seconds.



* T1.0 = seconds;
T0.1 = tenths of seconds;
T.01 = hundredths of seconds.

FUNCTION BLOCK

- The top node represents the preset value for the timer. It can be a constant, up to 999 in a Level 1 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference. The contents of this register are the preset value.

  For example, if the preset value is 009, it is either 9 seconds (T1.0), .9 seconds (T0.1) or .09 seconds (T.01). If the value is 050, it is either 50 seconds (T1.0), 5 seconds (T0.1), or .5 seconds (T.01). (Hint: take the preset value and divide by either 1, 10, or 100 to find the value it represents.)

- The bottom node contains a unique 4XXXX holding register reference. This register holds the timer value and increases in time, starting at zero and going up to the preset value, as long as both the top and bottom inputs are receiving power.

NOTE

The controller sets the timer value to the preset if an attempt is made to enter a value greater than the preset.

INPUTS

- The top input controls the operation. When it receives power and the timer is enabled, the timer value increases time. When the top input loses power, the timer stops increasing.

- The bottom input, when receiving power, enables the timer. When this input is not receiving power, the timer value is reset to zero. The timer does not increase in time if only the top input is receiving power.

OUTPUTS

- The top output passes power when the timer value equals the preset value.

- The bottom output passes power when the timer value is less than the preset value.

NOTE

Only one output passes power at a time.

EXAMPLE

The following paragraphs provide a detailed explanation of the logic used in Figure 2-8.

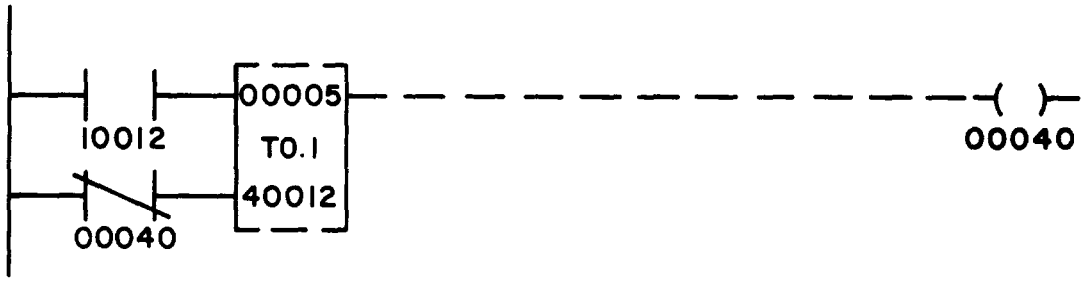


Figure 2-8. Timer

When input 10012 is energized, the top input receives power and, since the bottom input is also receiving power, the timer is enabled and register 40012 begins accumulating time in tenths of seconds. When the value in register 40012 equals 5 (.5 or ½ second), the top output passes power and energizes coil 00040. The bottom input loses power when coil 00040 is energized (the normally closed contact does not pass power), and the timer value (register 40012) is reset to zero.

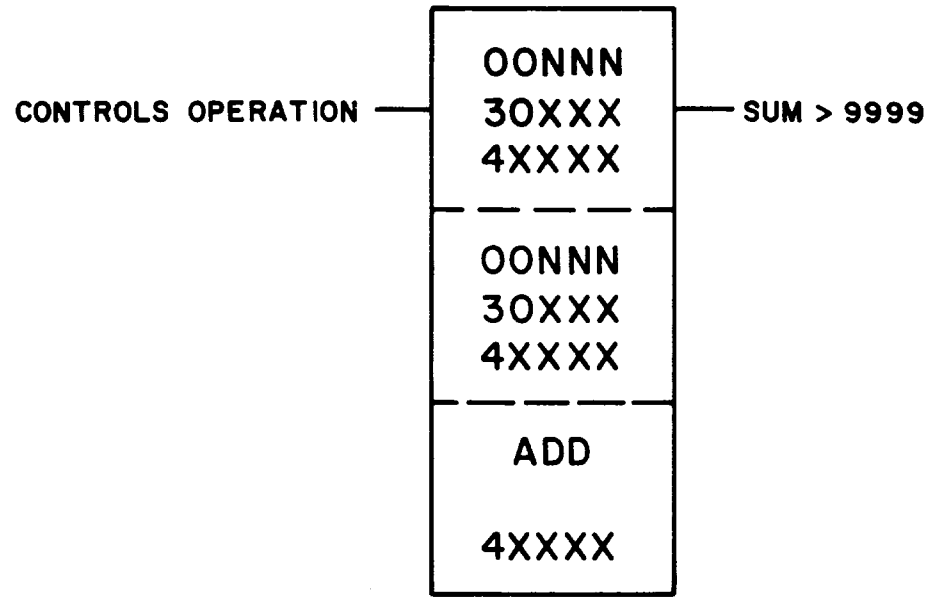
On the next scan that input 10012 is energized, coil 00040 is de-energized, thereby energizing input 00040 and re-enabling the timer.

2.4 ARITHMETIC FUNCTIONS

The following arithmetic functions are available: Addition, Subtraction, Multiplication, and Division. All four function blocks occupy three nodes in a network and place the result of the operation in the bottom node's holding register. The top input of each function block controls the operation; when it is receiving power the function is performed.

2.4.1 Addition (ADD)

FUNCTION
The Add function adds two values together (top and middle nodes) and places the sum in a holding register (bottom node).



FUNCTION BLOCK
- The top and middle nodes contain values which can be either constants, up to 999 in a Level 1 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference. The top and middle node values are added together when the control input receives power.

- The bottom node contains 4XXXX holding register reference. This holding register holds the sum of the top and middle node values.

INPUT
- The top input controls the operation. When it receives power, the value in the top node is added to the value in the middle node and the sum is placed in the bottom node's holding register.

OUTPUT
- The top output passes power when the sum is greater than 9999; it indicates that a 1 should be placed in front of the result located in the holding register. For example, if 6500 is added to 5000, the result in the holding register is 1500 and the top output passes power. The actual sum is 11,500.

NOTE

Only the top input and top output are used for this function.

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 2-9.

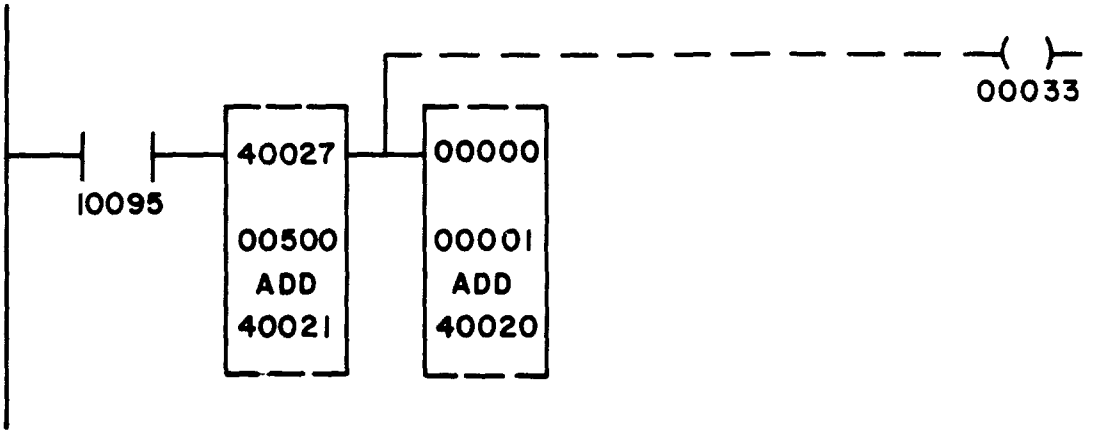


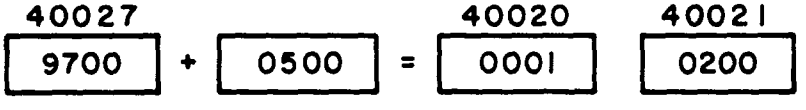
Figure 2-9. Addition

When input 10095 is energized, the top input of the first add block receives power and the content of register 40027 is added to the fixed value 00500. The sum is placed in register 40021.

If the content of register 40027 is 9700, the result placed in register 40021 is 0200 and the top output passes power. Coil 00033 is energized and the top input of the second add block receives power. This function places a one in register 40020 so the sum of the two numbers is read properly.

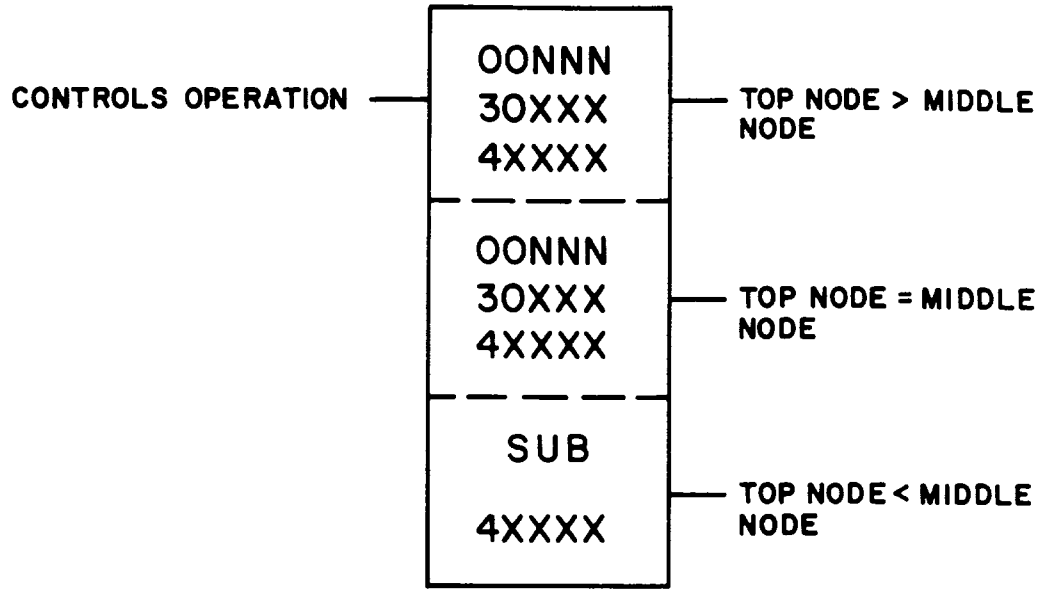
NOTE

The add block may give incorrect results with sums over 20,000.



2.4.2 Subtraction (SUB)

FUNCTION
The Subtract function subtracts the value in the middle node from the value in the top node and places the difference in the bottom node's holding register.



FUNCTION BLOCK

- The top node contains a value which can be a constant, up to 999 in a Level 1 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference. The value in the middle node is subtracted from this value when the control input receives power.

- The middle node contains a value which can be a constant, up to 999 in a Level 1 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference. This value is subtracted from the value in the top node.

- The bottom node contains the symbol SUB and a 4XXXX holding register reference. This holding register contains the difference between the top and middle node values.

INPUT

- The top input controls the operation. When it receives power, the value in the middle node is subtracted from the value in the top node, and the difference is placed in the bottom node's holding register.

OUTPUTS

- The top output passes power when the value in the top node is greater than the value in the middle node.

- The middle output passes power when the value in the top node equals the value in the middle node.

- The bottom output passes power when the value in the top node is less than the value in the middle node.

NOTE

The use of any or all outputs is optional. Two outputs can be connected together with vertical shorts to create a greater than or equal to function using the top and middle outputs, or a less than or equal to function using the middle and bottom outputs (i.e., these may be required for set point control or alarm limits).

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 2-10.

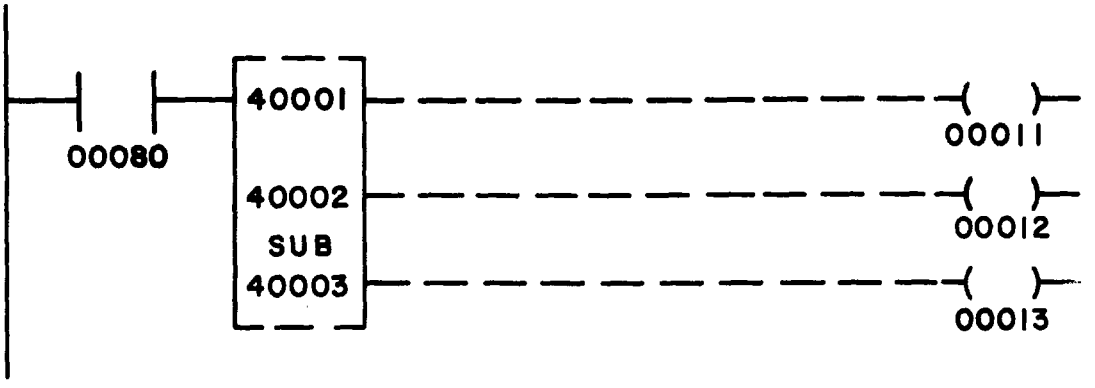


Figure 2-10. Subtraction

When contact 00080 is energized, the top input of the function block receives power and the subtract is performed. The value in register 40002 is subtracted from the value in 40001 and the result is placed in register 40003.

If the value in 40001 is greater than the value in 40002, the top output passes power and energizes coil 00011. This indicates a normal subtract function.

If the value in 40001 is equal to the value in 40002, the middle output passes power and energizes coil 00012. This indicates a difference of zero.

If the value in 40001 is less than the value in 40002, the bottom output passes power and energizes coil 00013. This indicates that the answer to the subtract should be negative.

NOTE

The value placed in register 40003 is the absolute value of the difference — no sign is associated with the content of register 40003.

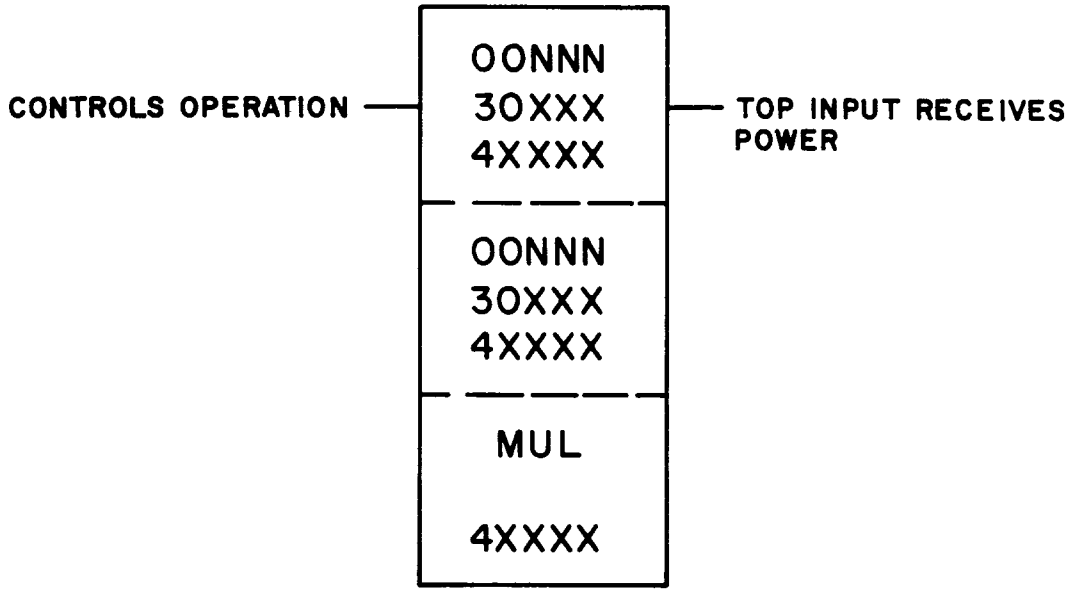
If the value in 40001 is 9000 and the value in 40002 is 0500, the result placed in 40003 is 8500.

40001		40002		40003
9000	−	0500	=	8500

2.4.3 Multiplication (MUL)

FUNCTION

The Multiply function calculates the product of the values in the top and middle nodes and places the answer in two consecutive holding registers referenced to in the bottom node of the function block. Two consecutive holding registers are used because the product of two 4-digit numbers can be up to eight digits in length.



FUNCTION BLOCK

- The top and middle nodes each contain a value which can be a constant, up to 999 in a Level 1 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference. The top and middle node values are multiplied.

- The bottom node contains both the symbol MUL and a 4XXXX holding register reference. This holding register holds the high order portion of the product even if it is zero. The next consecutive holding register (4XXXX + 1) holds the low order portion of the product. For this reason, the last available holding register cannot be used.

INPUT

- The top input controls the operation. When it receives power, the value in the top node is multiplied by the value in the middle node and the product is placed in the bottom node's holding register(s).

OUTPUT

- The top output passes power when the top input receives power. This allows the function blocks to be cascaded within a network.

NOTE

Only the top input and the top output are used.

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 2-11.

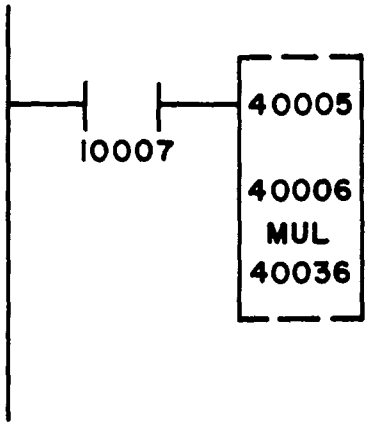
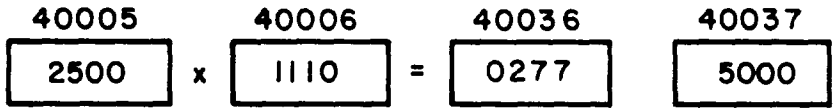


Figure 2-11. Multiplication

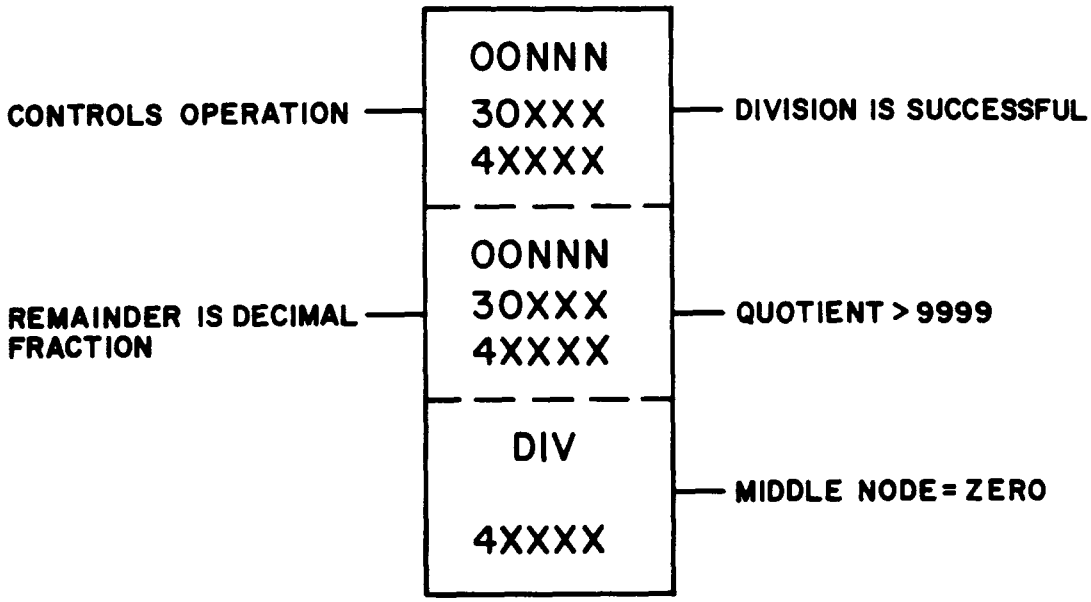
Multiply operates upon two four-digit numbers to produce an eight-digit product. In this example, when input 10007 is energized, the top input receives power and the value in register 40005 is multiplied by the value in register 40006. The resulting product is stored in registers 40036 and 40037.

If registers 40005 and 40006 contain the values 2500 and 1110 the product is 2,775,000. Thus, register 40036 is loaded with the value 0277 and register 40037 is loaded with the value 5000. The two registers are multiplied and the product is stored every scan input 10007 is energized.



2.4.4 Division (DIV)

FUNCTION
The divide function divides the value in the top node by the value in the middle node and places the quotient in the bottom node's holding register.



FUNCTION BLOCK

- The top node is the dividend. It can be a constant, up to 999 in a Level 1 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference or a 4XXXX holding register reference. Two consecutive registers are used and both are needed for a double precision number. If a single precision number is desired and registers are being used, fill the first register with zeros. When a register reference is used, the next register is implied; therefore, the last available register, input or holding, cannot be used in this node.

- The middle node is the divisor. It contains a constant, up to 999 in a Level 1 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference.

- The bottom node contains the symbol DIV and a 4XXXX holding register. This holding register holds the result of the division. The next holding register (4XXXX + 1) holds the remainder; therefore, the bottom node cannot contain the last available holding register.

INPUTS

- The top input controls the operation. When it receives power, the value in the top node is divided by the value in the middle node, and the quotient and remainder are placed in two consecutive holding registers.

- The middle input, when receiving power, causes the remainder to be a decimal fraction. If it does not receive power the remainder is a whole number. For example, 10 divided by 3 has a decimal remainder of .3333 and a whole number remainder of 1.

OUTPUTS

- The top output passes power when the division is successful. If the middle or bottom outputs pass power, the division is unsuccessful and the top output does not pass power.

- The middle output passes power when the quotient is greater than 9999. If this output passes power, zeros are placed in the bottom node's holding registers.
- The bottom output passes power when the divisor (middle node) equals zero. If this output passes power, zeros are placed in the bottom node's holding registers.

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 2-12.

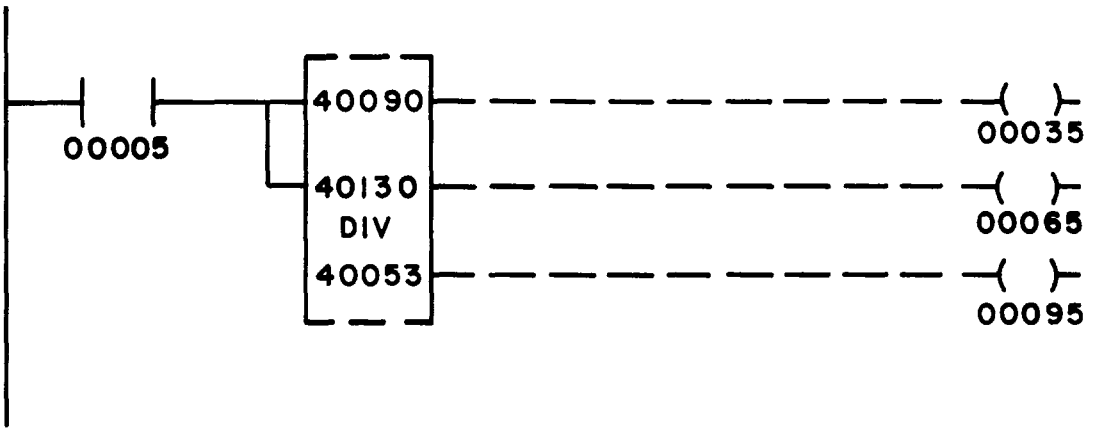
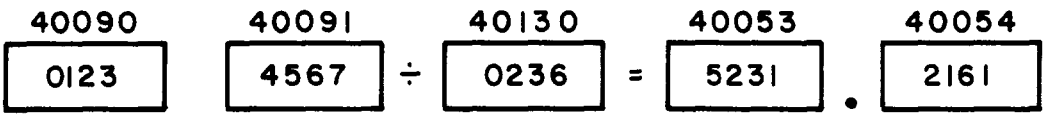


Figure 2-12. Division

When contact 00005 is energized, the top input receives power and the content of registers 40090 and 40091 (double precision number) is divided by the content of register 40130. The result is placed into register 40053 and the remainder into register 40054. Since the middle input is receiving power, the remainder is a decimal fraction rather than a whole number.

If the double precision number is 1,234,567 (40090 = 0123, 40091 = 4567) and the divisor (40130) is 0236, the value 5231 is placed in register 40053 and the decimal remainder 2161 is placed in register 40054.



If the middle input was not receiving power, the whole remainder 0051 would be placed in register 40054. Coil 00035 is energized if the division is successful.

Coil 00065 is energized if the quotient is greater than 9999. Coil 00095 is energized if the divisor (middle node) equals zero.

2.5 SUMMARY OF BASIC FUNCTIONS

	TOP NODE	MIDDLE NODE	BOTTOM NODE	TOP INPUT	MIDDLE INPUT	BOTTOM INPUT	TOP OUTPUT	MIDDLE OUTPUT	BOTTOM OUTPUT
UCTR	00NNN, 30XXX, or 4XXXX Preset	— 4XXXX	4XXXX	Control	—	Enable/Reset	Count = Preset	—	Count Preset
DCTR	00NNN, 30XXX, or 4XXXX Preset	—	4XXXX	Control	—	Enable/Reset	Count = Zero	—	Count Preset
TIMER	00NNN, 30XXX, or 4XXXX Preset	—	4XXXX	Control	—	Enable/Reset	Timer = Preset	—	Timer Preset
ADD	00NNN, 30XXX, or 4XXXX	00NNN, 30XXX, or 4XXXX	4XXXX	Control	Not Used	Not Used	Sum > 9999	Not Used	Not Used
SUB	00NNN, 30XXX, or 4XXXX	00NNN, 30XXX, or 4XXXX	4XXXX	Control	Not Used	Not Used	Top Node > Middle Node	Top Node = Middle Node	Top Node < Middle Node
MUL	00NNN, 30XXX, or 4XXXX	00NNN, 30XXX, or 4XXXX	4XXXX	Control	Not Used	Not Used	Top Input Receiving Power	Not Used	Not Used
DIV	00NNN, 30XXX, or 4XXXX	00NNN, 30XXX, or 4XXXX	4XXXX	Control	Remainder is Decimal Fraction	Not Used	Division is Sucessful	Quotient > 9999	Middle Node Equals Zero

2.6 LOGIC EXAMPLES

2.6.1 Real Time Clock

The following paragraphs provide a detailed explanation of the logic illustrated in Figure 2-13 to program a real time clock.

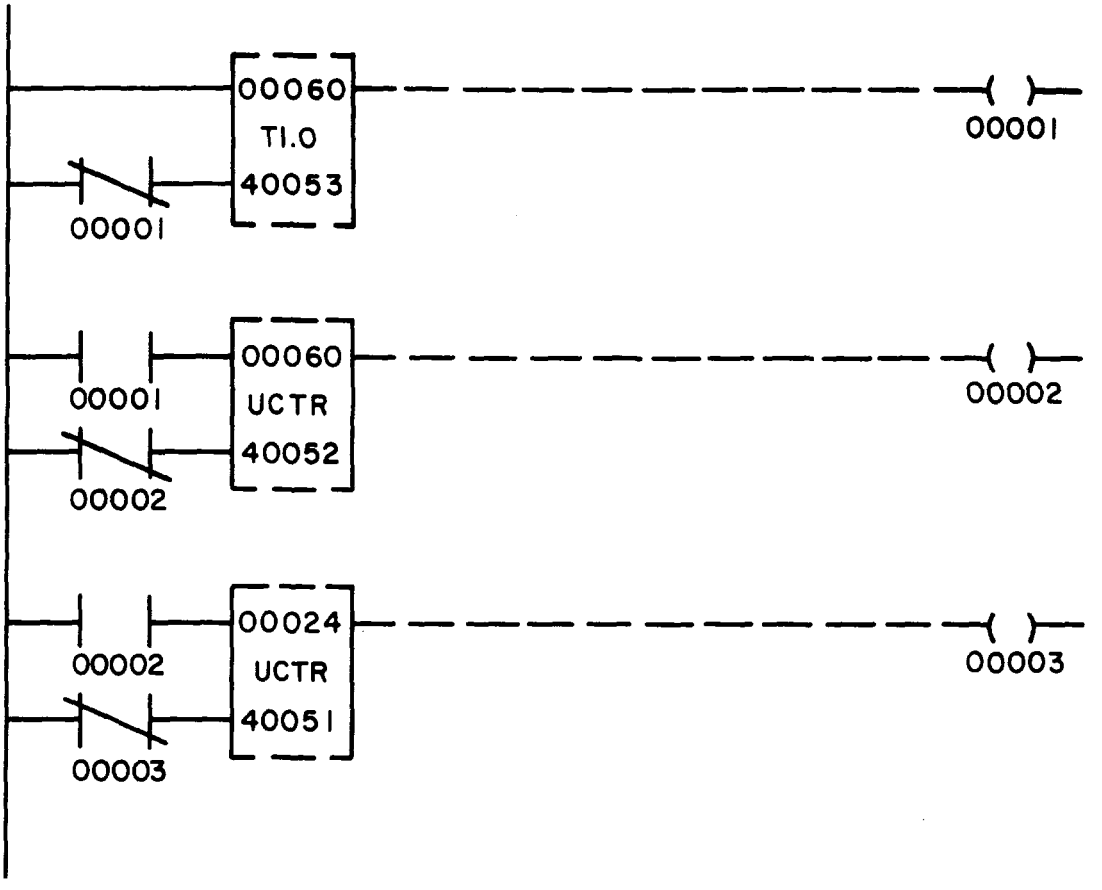


Figure 2-13. Real Time Clock

The top network in this example is the 1 minute timer. At the beginning of the logic solving, coil 00001 is OFF so both the top and bottom inputs of the timer are receiving power. Register 40053 starts increasing time in seconds until it reaches 60. At this point, the top output passes power and energizes coil 00001. Register 40053 is reset and the counter in the second network (40052) increases by one, indicating that one minute has elapsed.

Since the timer in network 1 is no longer equal to the preset, coil 00001 is de-energized and the timer resumes increasing time. Once the value in 40052 reaches 60, indicating 60 minutes, the top output passes power and energizes coil 00002. Register 40052 is reset and the counter in the third network (40051) increases by one, indicating that one hour has passed. The correct time of day can be read in registers 40051 — 40053 in hours, minutes, and seconds.

2.6.2 Fahrenheit to Centigrade Conversion

The following paragraphs provide a detailed explanation of the logic illustrated in Figure 2-14 to program a Fahrenheit to Centigrade temperature conversion.

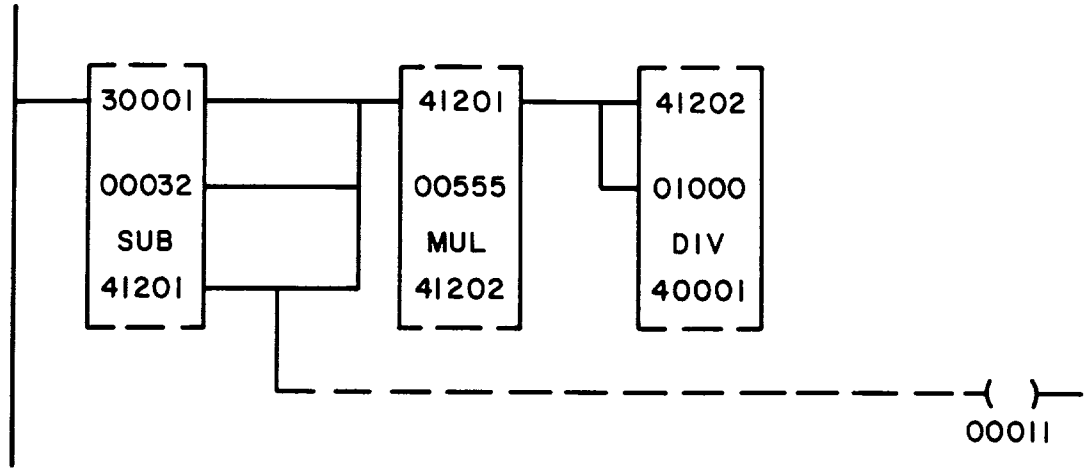


Figure 2-14. Fahrenheit to Centigrade Conversion

In this example, when the top input of the subtract function block receives power, the number 32 is subtracted from the value in register 30001 (degrees fahrenheit). The difference is placed in register 41201.

Whether the answer is positive, negative, or zero, the top input of the multiply function block receives power. If the answer to the subtract is negative, coil 00011 is energized to indicate a negative value. The value in 41201 is multiplied by 555 and the product is placed in register 41202.

The top input of the divide function block receives power and the value in 41202 is divided by 01000. The temperature conversion in degrees centigrade is placed in 40001.

2.6.3 Double Precision Add

The following paragraphs provide a detailed explanation of the logic illustrated in Figure 2-14 to program a double precision add.

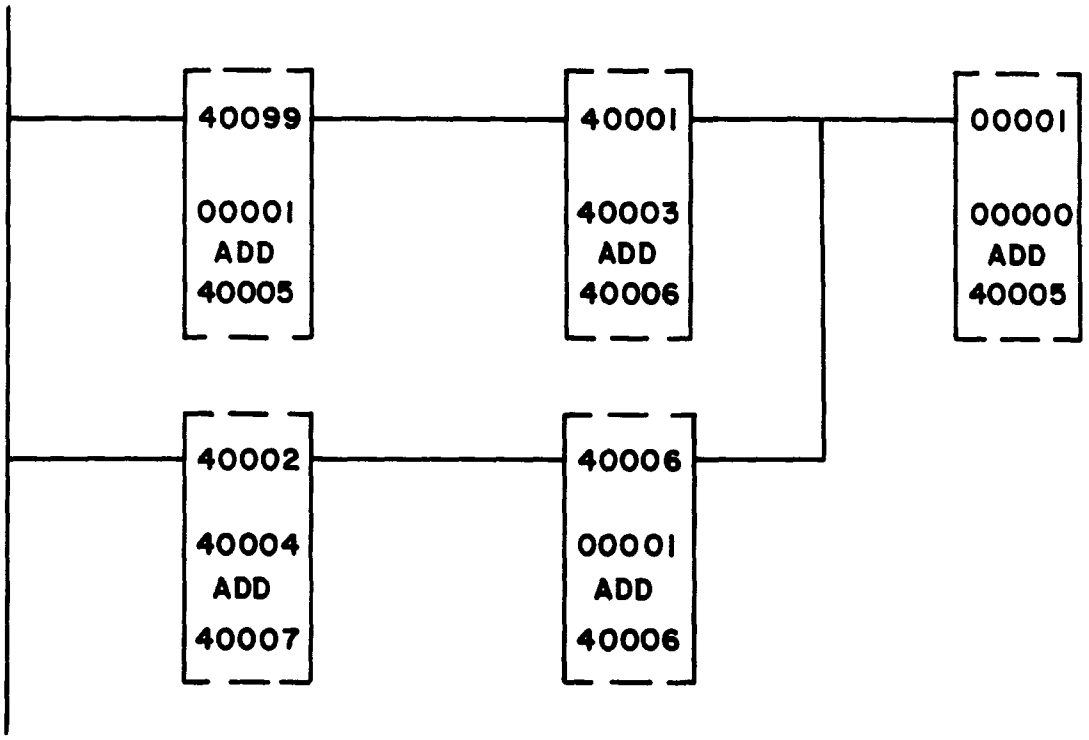


Figure 2-15. Double Precision Add

In this example, an eight-digit number in registers 40001 and 40002 is added to an eight-digit number in registers 40003 and 40004. The nine-digit result is placed in registers 40005-40007. The ADD function block at the top left of the network simply clears register 40005 to zero on each scan.

Since an overflow condition exists power is passed to the next function block. If the result of addition in this block is greater than 9999, the top output passes power to place a one in register 40005.

The second half of the addition takes place in bottom left function block, and passes power if an overflow condition exists. If an overflow exists, register 40006 is incremented by one to account for it. If 40006 overflows when this one is added to it, power is passed to increase 40005 to one.

If the following values are in the registers:

40001	40002		40003	40004		40005	40006	40007
9760	3842	+	6553	8317	=	0001	6314	2159

2.6.4 Subtract Greater Than 9999

In the Subtract function block, it is possible to have a value as large as 32,767 in the top node. This is possible if the register value is in binary format. There are sixteen bits available in a register. The first (most significant) bit is the sign bit, zero being negative and one being positive. The other fifteen bits, when all ones, equal 32,767.

SECTION 3
DATA TRANSFER (DX) MOVE FUNCTIONS

The MOVE functions copy data from registers and/or tables into other registers or tables. The data can then be examined or changed by the controller without altering the original data.

A register is a location in the controller's memory in which a numerical value is stored. This value can be binary or binary coded decimal (BCD). In a 584L PC, the maximum decimal value is 9999 and the maximum number of bits is sixteen.

A table is a group of consecutive registers or discretes. The maximum number of registers or groups of discretes in a table is 255 in a Level 1 584L PC or 999 in a Level 2 584L PC.

Each function block (except STAT) occupies three nodes in a 10 x 7 node network format and consists of a source, a destination, and a node specifying table length. The top input is the control input; when it receives power the function is performed. The top output passes power when the top input receives power. This allows function blocks to be cascaded within a network.

The input(s) to a function block can be a single relay contact, another function block, or a whole network of logic. The output(s) can be connected directly to coils, to other function blocks, to relay contacts, or left unconnected.

NOTE

If a single move operation is desired, use a transitional contact to control the top input.

The DX function blocks can perform functions with 30XXX input registers, 4XXXX holding registers, and 0XXXX and IXXXX discrete references.

If discrete references are used, remember the following:

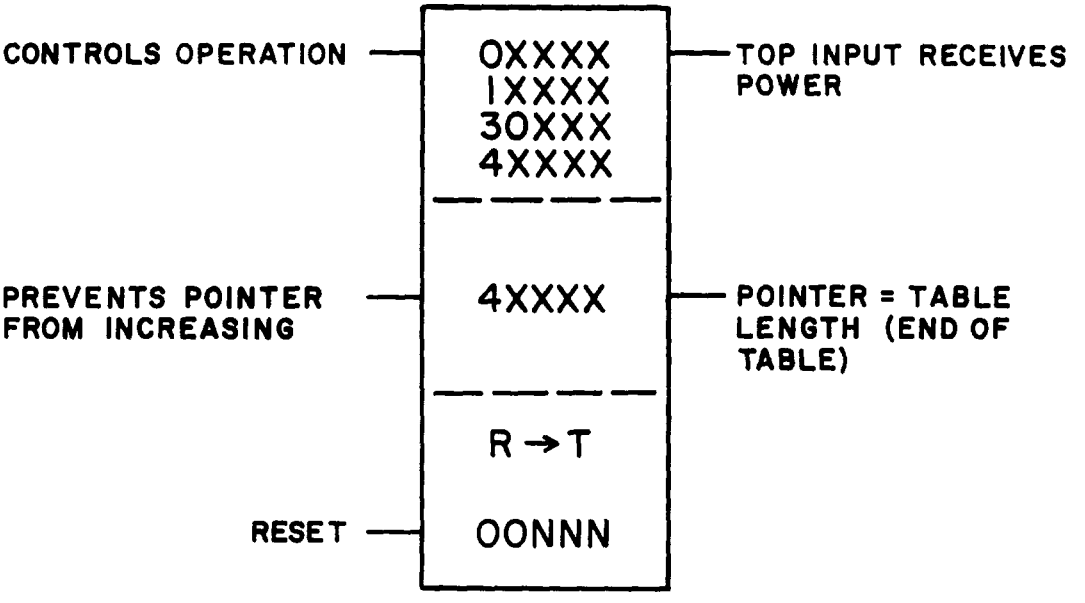
- Discretes are used in groups of sixteen.
- The reference number used is the first in the group; the other fifteen references are implied.
- Only certain reference numbers are valid; the number must be divisable by 16 with a remainder of 1. The valid reference numbers are: 00001, 00017, 00033, etc., and 10001, 10017, 10033, etc.

If a 0XXXX or a 1XXXX reference is used in a DX function block, it cannot be used anywhere else in the program. If discrete references are used to specify table length, the value refers to the number of groups of 16 discretes (i.e., 4 indicates 4 groups or 64 discretes).

3.1 REGISTER-TO-TABLE MOVE (R→T)

FUNCTION

The Register-To-Table Move function copies sixteen logic coils, sixteen discrete inputs, one input register, or one holding register into a single specific location within a table of registers.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source is a single 16 bit location (e.g., a register or group of sixteen discretes).

- The middle node is the destination node. It is a 4XXXX holding register which holds the pointer value. The pointer value plus one indicates the position of a register in a table (i.e., a pointer value of three indicates the fourth position in a table). The table starts at the next register (4XXXX + 1), not at the pointer.

- The bottom node contains the symbol R→T and a numerical value which specifies the table length. This constant can range from 1 to 255 in a Level 1 584L PC or from 1 to 999 in a Level 2 584L PC.

NOTE

If the pointer register is loaded with a value greater than the table length, the 584L PC sets the pointer value to the table length when the function block is solved.

INPUTS

- The top input controls the operation. When it is receiving power, the information in the source register is copied into a location in the table.

- The middle input, when receiving power, prevents the pointer from increasing.

- The bottom input, when receiving power, resets the pointer to zero. If the top input is also energized, the source register is copied to the first register in the table.

OUTPUTS

- The top output passes power when the top input receives power.

- The middle output passes power when the pointer equals the table length. This indicates that the table is full (end of table).

EXAMPLE

The following paragraphs provide a detailed explanation of the logic used in Figure 3-1.

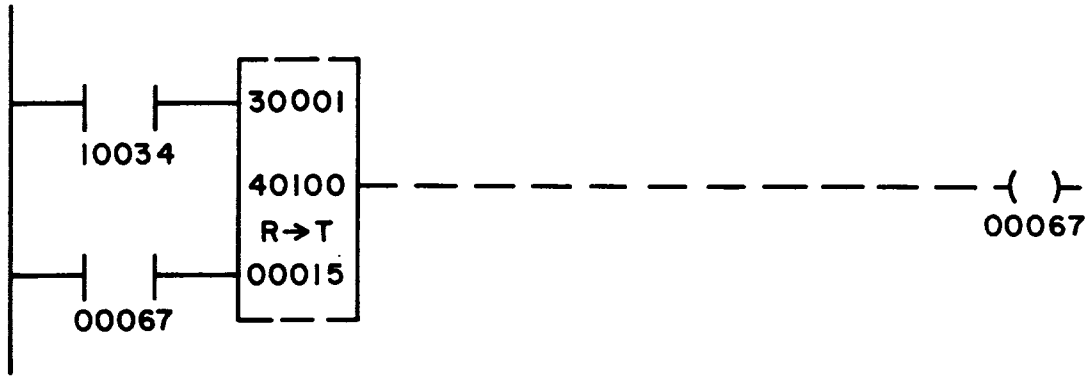


Figure 3-1. Register-To-Table Move Logic

When input 10034 is energized, the top input receives power and the content of input register 30001 is moved into table 40101-40115. Data is moved one entry per scan.

If the pointer value equals zero when the top input receives power, the content of register 30001 is moved into register 40101 and the pointer value increases to one.

On the next scan, provided the top input is still receiving power, the content of register 30001 is moved into register 40102 and the pointer value increases to two. Whenever the top input loses power, the move operation stops and the pointer holds its value.

Once the pointer has increased to the (table size as defined in the bottom node (e.g., 15), the middle output passes power and energizes coil 00067. On the next scan, the bottom input receives power because it is referenced to coil 00067. This input resets the pointer to zero, thereby de-energizing coil 00067.

Figure 3-2 is an illustration of the Register-To-Table Move described in the preceding paragraphs.

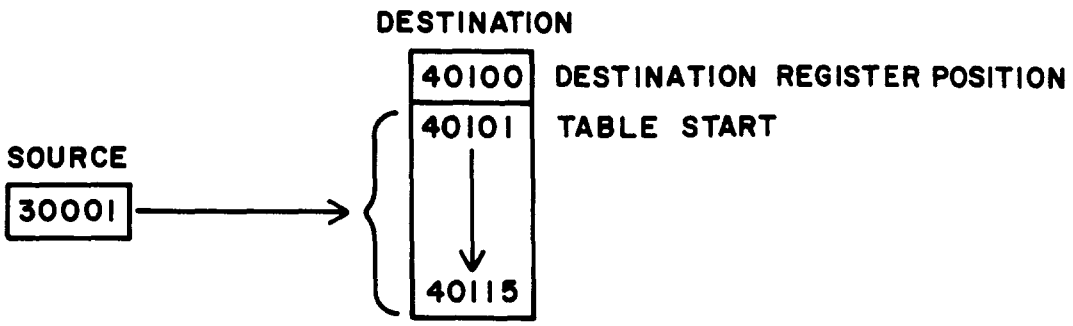
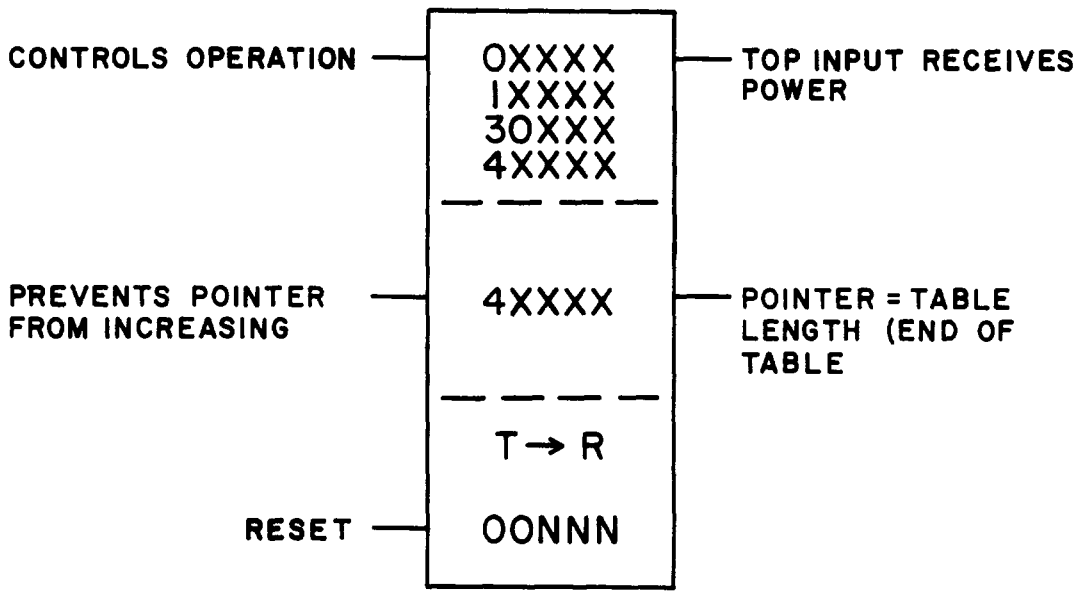


Figure 3-2. Register-To-Table Move

3.2 TABLE-TO-REGISTER MOVE (T ⟶ R)

FUNCTION

The Table-To-Register Move function copies one register or group of sixteen discretes from a table into a single holding register.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source is a table of 16 bit locations. Its size is defined in the bottom node.

- The middle node is the destination node. It is a 4XXXX holding register which holds the pointer value. The pointer value plus one indicates the position of a register in a table (i.e., a pointer value of three indicates the fourth position in a table). The next consecutive holding register (4XXXX + 1) receives the data. The pointer does not receive the data.

- The bottom node contains the symbol T→R and the numerical value that specifies the source table length. This constant can range from 1 to 255 in a Level I 584L PC, or from 1 to 999 in a Level 2 584L PC.

NOTE

If the pointer register is loaded with a value greater than the table length, the 584L PC sets the pointer value to the table length when the function block is solved.

INPUTS

- The top input controls the operation. When it is receiving power, the information in the source table's register is copied into a single holding register.
- The middle input, when receiving power, prevents the pointer value from increasing.
- The bottom input, when receiving power, resets the pointer to zero. If the top and bottom inputs are energized at the same time, the first register in the table is moved to the destination register.

OUTPUTS

- The top output passes power when the top input receives power.
- The middle output passes power when the pointer value equals the table length (end of the table).

EXAMPLE

The following paragraphs provide a detailed explanation of the logic used in Figure 3-3.

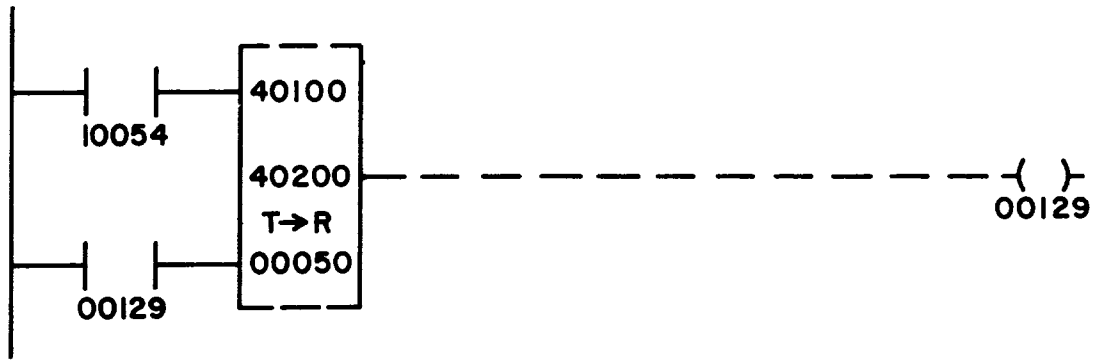


Figure 3-3. Table-To-Register Move Logic

When input 10054 is energized, the top input receives power and the content of the table starting at 40100 is moved into register 40201. Data is moved one register per scan. The pointer value increases after each move.

If the pointer value equals zero when the top input receives power, the content of register 40100 is moved into register 40201 and the pointer value (register 40200) increases to one. On the next scan, provided the top input is still receiving power, the content of register 40101 is moved into 40201 (the same register as before) and the pointer value (register 40200) increases to two. On this second scan, the value of register 40101 takes the place of the value of register 40100 in register 40201.

When input 10054 is energized, the top input receives power and the register moved is 40100 plus the pointer value (e.g., if the pointer value in register 40200 is 32, the register moved is 40132). When the pointer register 40200 equals 50, coil 00129 is energized. The bottom input is energized because it is referenced to coil 00129. The pointer is reset to zero, thereby de-energizing coil 00129.

Figure 3-4 illustrates the Table-To-Register Move described in the preceding paragraphs.

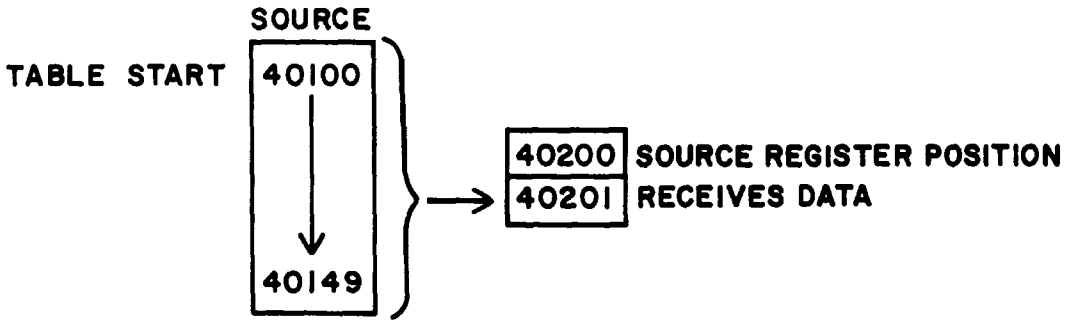
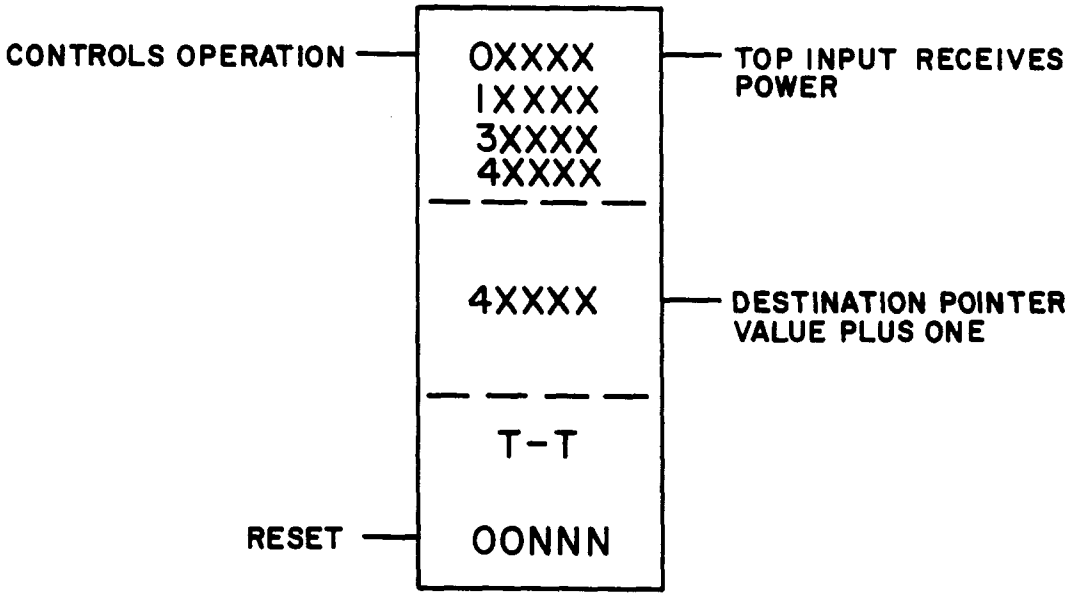


Figure 3-4. Table-To-Register Move

3.3 TABLE-TO-TABLE MOVE (T → T)

FUNCTION

The Table-To-Table Move function copies discretes or registers from one table to a table of holding registers.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source is a table of 16 bit locations. Its size is defined in the bottom node.

- The middle node is the destination node. It is a 4XXXX holding register which holds the pointer value. The pointer value plus one indicates the position of a register in a table (i.e., a pointer value of three indicates the fourth position in a table). The table starts at the next register (4XXXX + 1), not at the pointer.

- The bottom node contains the symbol T—►T and the numerical value that specifies the length for both tables. This constant can range from 1 to 255 in a Level I 584L PC, or from 1 to 999 in a Level 2 584L PC.

NOTE

If the pointer register is loaded with a value greater than the table length, the 584L PC sets the pointer value to the table length when the function block is solved.

INPUTS

- The top input controls the operation. When it is receiving power, the information in one register of the source table is copied into the corresponding register in the destination table.

- The middle input, when receiving power, prevents the pointer value from increasing.

- The bottom input, when receiving power, resets the pointer value to zero. If the top and bottom inputs are energized at the same time, the first register in the table is moved to the first register in the destination table.

OUTPUTS

- The top output passes power when the top input receives power.

- The middle output passes power when the pointer value equals the table length (end of table).

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 3-5.

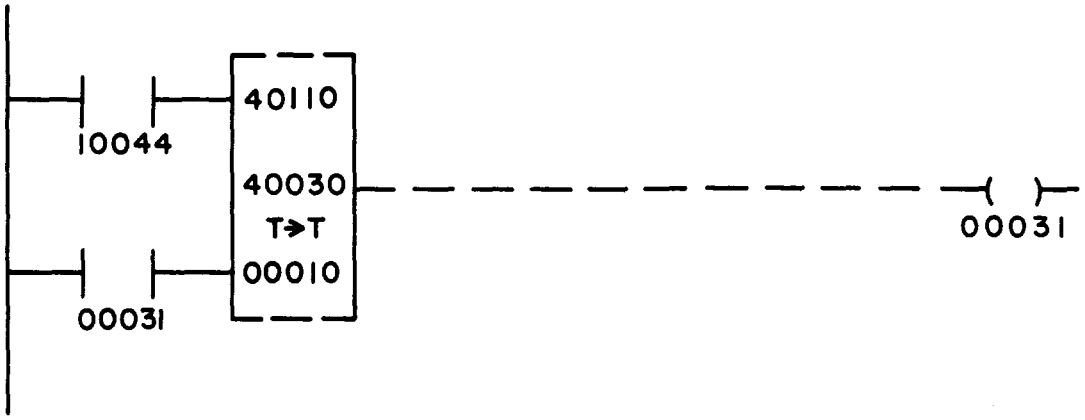


Figure 3-5. Table-To-Table Move Logic

When input 10044 is energized, the top input receives power. A register from the table which starts at register 40110 is moved into the register at the corresponding position in the table which starts at 40031. Register 40030 holds the pointer value.

One register is moved per scan and the pointer value increases by one on each scan. The middle output passes power when the pointer value equals the table size 10 in this example, thus energizing coil 00031. On the next scan, the bottom input receives power because it is referenced to coil 00031. The pointer value is reset to zero; therefore, coil 00031 is de-energized OFF.

At the start of the move, if the pointer is at zero, the content of register 40110 is copied into register 40031, and the pointer increases to one. On the next scan, register 40111 is copied into register 40032, and the pointer increases to two. If the function block stops receiving power, the pointer holds its value and recalls it when power is returned.

Figure 3-6 illustrates the Table-To-Table Move described in the preceding paragraphs.

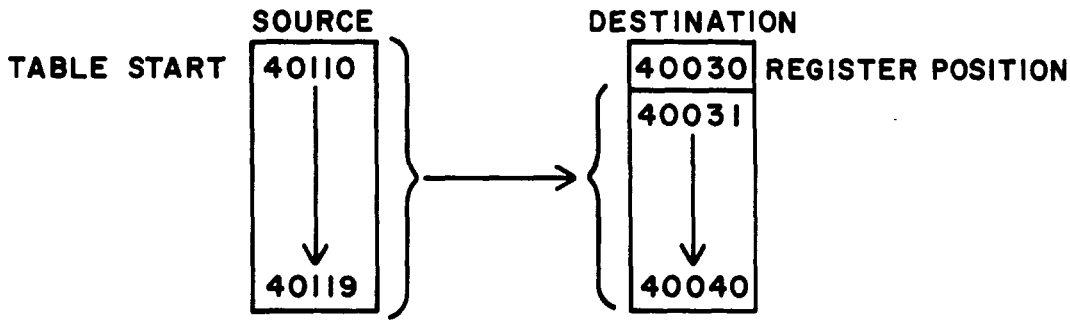
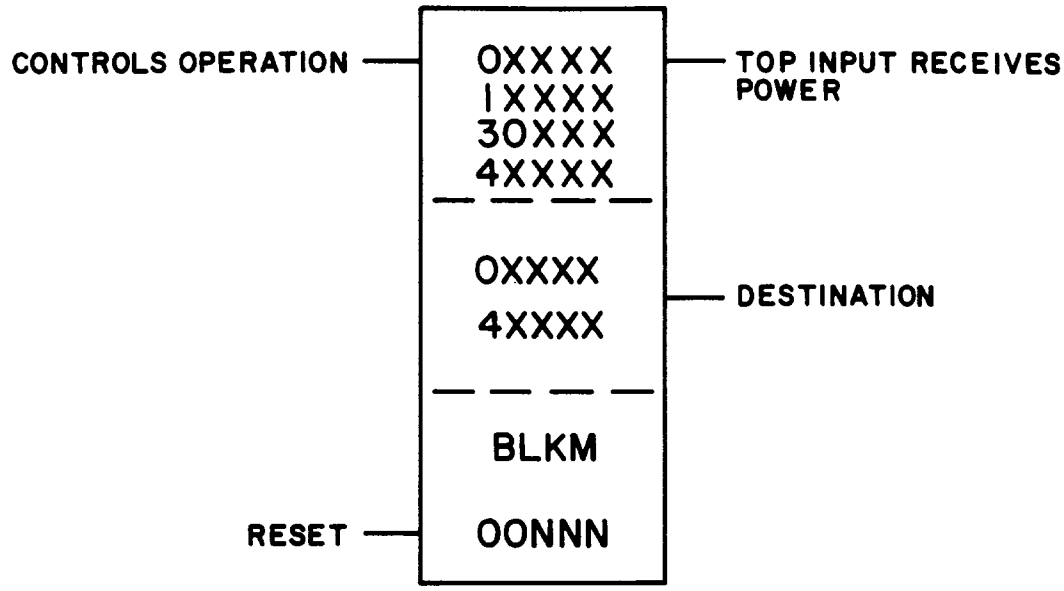


Figure 3-6. Table-To-Table Move

3.4 BLOCK MOVE (BLKM)

FUNCTION

The Block Move function copies the entire contents of a table of registers or discretes into another table on one scan. This function does not use a pointer register.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source is a table of 16 bit locations.

- The middle node is the destination node. It can be either a 0XXXX logic coil reference or a 4XXXX holding register reference. The destination is a table of 16 bit locations, the same size as the source.

WARNING

The Block Move function overrides the disable state of a coil used in the destination node of the function block. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, because the coil's state can change as a result of the BLKM function.

- The bottom node contains the symbol BLKM and a numerical value that specifies the table length for both the source and the destination. This constant can range from 1 to 100.

INPUT

- The top input controls the operation. When it receives power, one table of registers or discretes is copied into another table of the same length.

OUTPUT

- The top output passes power when the top input receives power.

NOTE

Only the top input and top output are used.

EXAMPLE
The following paragraph provides a detailed explanation of the logic used in Figure 3-7.

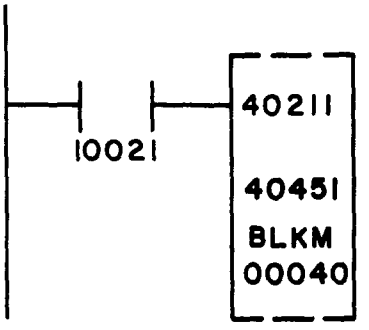


Figure 3-7. Block Move Logic

When input 10021 receives power, the contents of registers 40211-40250 are copied into registers 40451-40490. All the registers are moved in one scan, each scan the top input receives power. No output is required for this function.

Figure 3-8 illustrates the Block Move described in the preceding paragraphs.

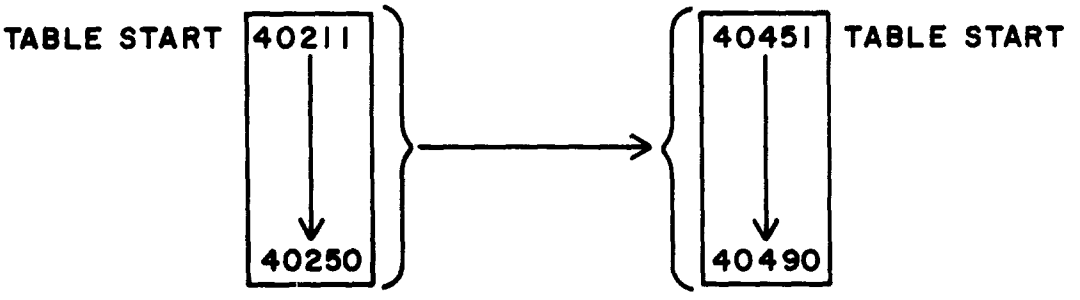


Figure 3-8. Block Move

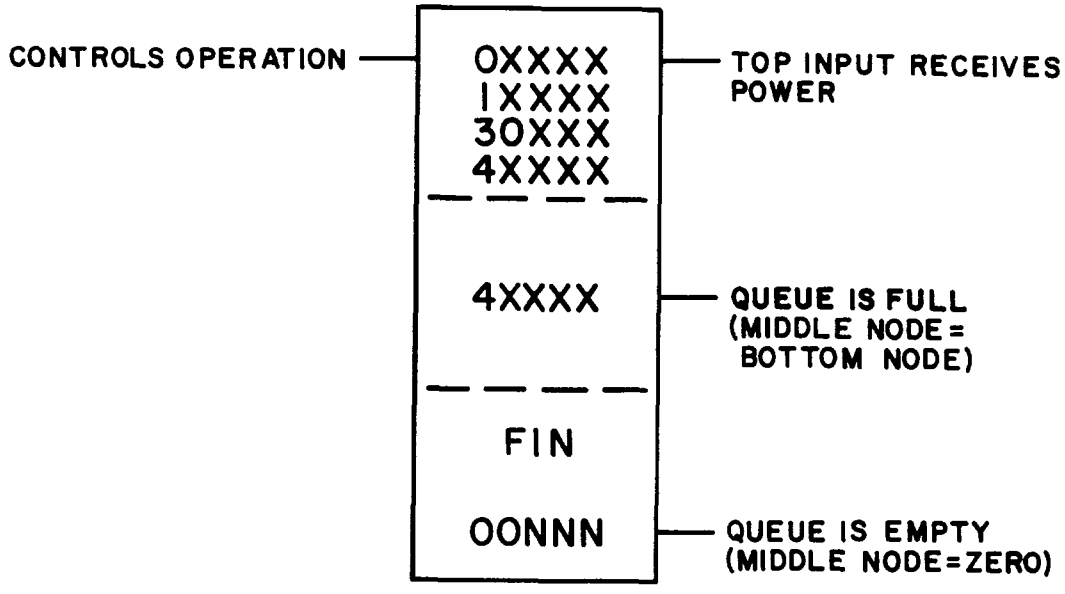
3.5 FIFO OPERATIONS

A FIFO queue is a table of 4XXXX holding registers. Individual 16 bit data items come out of the queue in the same order they entered the queue. A common analogy to a FIFO queue is a paper cup dispenser — cups are removed in the same order they were inserted.

Generally, both a First-In and a First-Out function block are used for each queue. The same pointer register is used for both the First-In and First-Out function blocks, and the queue length in each must be the same.

3.5.1 First-In (FIN)

FUNCTION
The First-In function inserts new data into the queue.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source is a single 16 bit location (e.g., a register or a group of sixteen discretes).

- The middle node is the destination node. It is a 4XXXX holding register which is used as the pointer. It also holds the pointer value (number of registers in the queue). The data is placed in the queue starting at 4XXXX + 1.

- The bottom node contains the symbol FIN and the numerical value that specifies the queue length. This constant can range from 1 to 100.

NOTE

If the pointer register is loaded with a value greater than the queue length, the 584L PC sets the pointer value to the queue length when the function block is solved.

INPUT

- The top input controls the operation. When it receives power, the information in the source register is copied to the first location in the queue and the pointer value is increased.

OUTPUTS

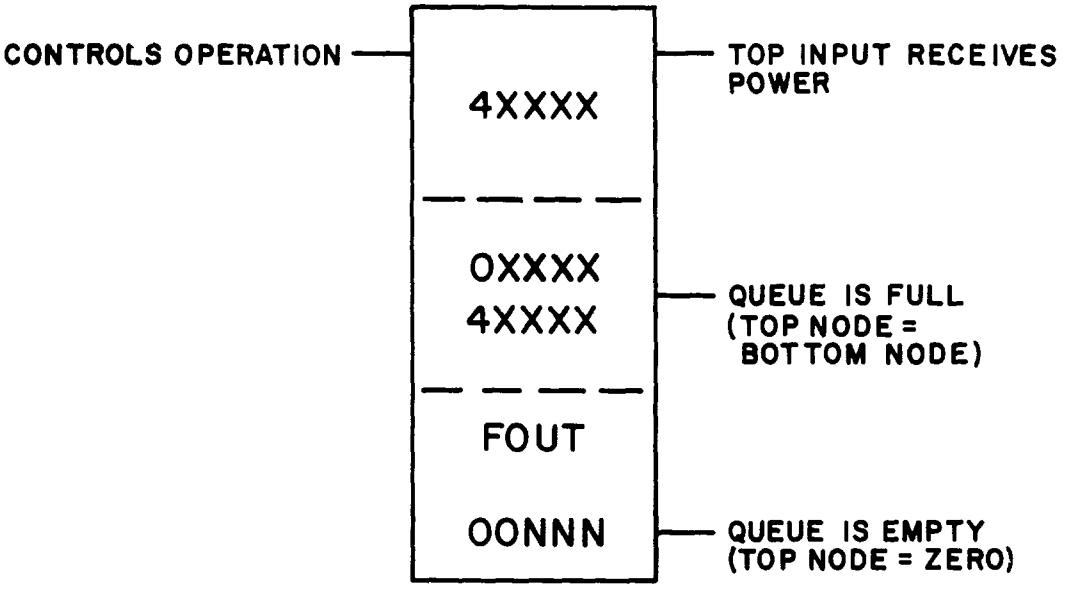
- The top output passes power when the top input receives power.

- The middle output passes power when the queue is full, pointer value equals queue length.

- The bottom output passes power when the queue is empty, pointer value equals zero.

3.5.2 First-Out (FOUT)

FUNCTION
The First-Out function removes the oldest data from the queue.



FUNCTION BLOCK

- The top node is the source node. It is a 4XXXX holding register which is used as the pointer. The queue starts at 4XXXX + 1. This register also holds the pointer value (number of registers in the queue).

- The middle node is the destination node. It can be either a 0XXXX logic coil reference or a 4XXXX holding register reference. It is a single 16 bit location such as a register or group of 16 discretes.

WARNING

The FOUT function overrides the disable state of a coil used in the destination node of the function block. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, because the coil's state can change as a result of the FOUT function.

- The bottom node contains the symbol FOUT and the numerical value that specifies the queue length. This constant can range from 1 to 100.

NOTES

The FOUT function is the only function which uses the source register as the pointer.

If the pointer register is loaded with a value greater than the queue length, the 584L PC sets the pointer value to the queue length when the function block is solved.

INPUT

- The top input controls the operation. When it receives power, the oldest information in the queue is removed and placed in sixteen logic coils or a holding register. Also, the pointer value is decreased by one.

OUTPUTS

- The top output passes power when the top input receives power.
- The middle output passes power when the queue is full, pointer value equals queue length.
- The bottom output passes power when the queue is empty, pointer value equals zero.

EXAMPLE

The following paragraphs provide a detailed explanation of the logic used in Figure 3-9.

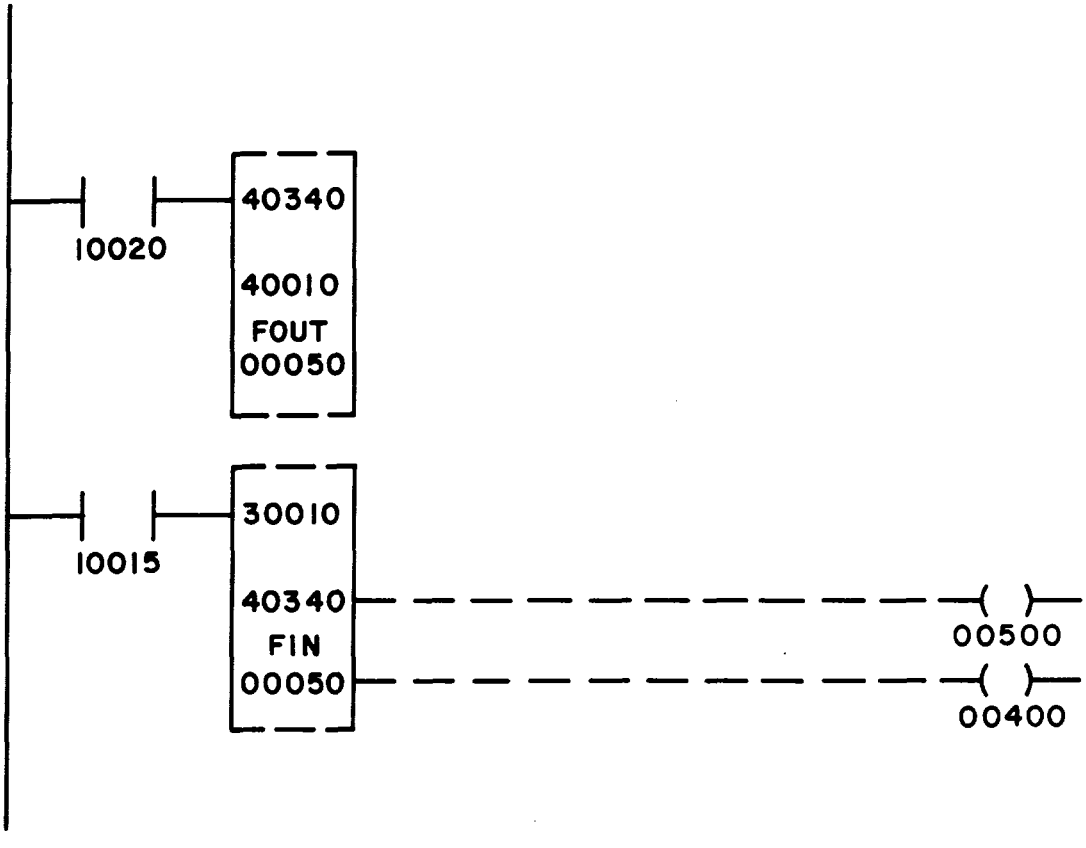


Figure 3-9. First-In/First-Out Logic

When input 10020 is energized, the oldest data in the queue is removed and placed in register 40010, and the pointer is decreased.

If the queue is full (value 0050 in register 40340), coil 00500 is energized. Attempts to insert new data are ignored. If the queue is empty, coil 00400 is energized and attempts to remove old data are ignored.

When input 10015 is energized, the top input receives power and register 30010 is copied into register 40341 in the queue (40341-40390). The pointer value in register 40340 increases by one. Each scan the top input receives power, a new value enters the queue. As new data is entered, old data is "pushed down" one register; therefore, register 40341 always contains the most recent entry.

The FOUT function block is placed before the FIN function block to ensure that, if the queue is full, the oldest data is removed before new data is entered. If the FIN block came first, an attempt to enter new data would be ignored if the queue was full. Placing the FOUT block first ensures that this will not happen; there will always be at least one empty register for new data.

Figure 3-10 illustrates the FIFO moves described in the preceding paragraphs.

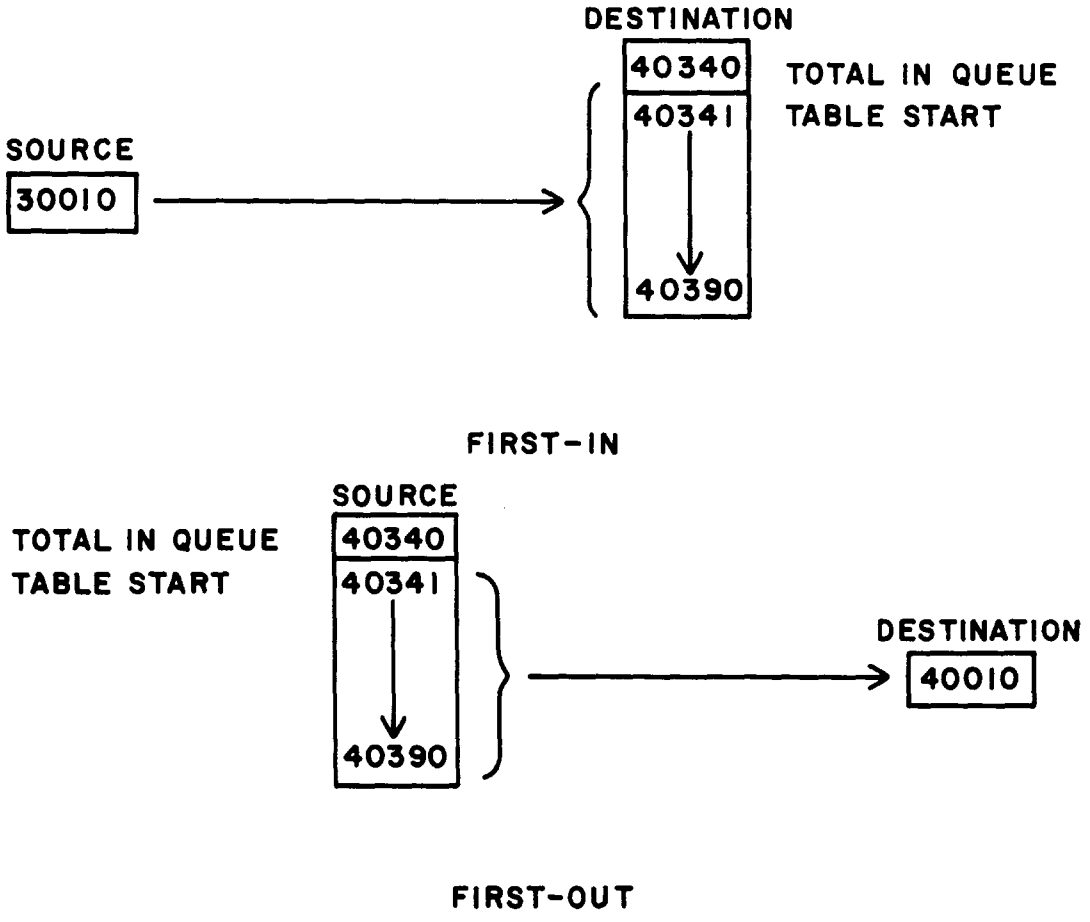
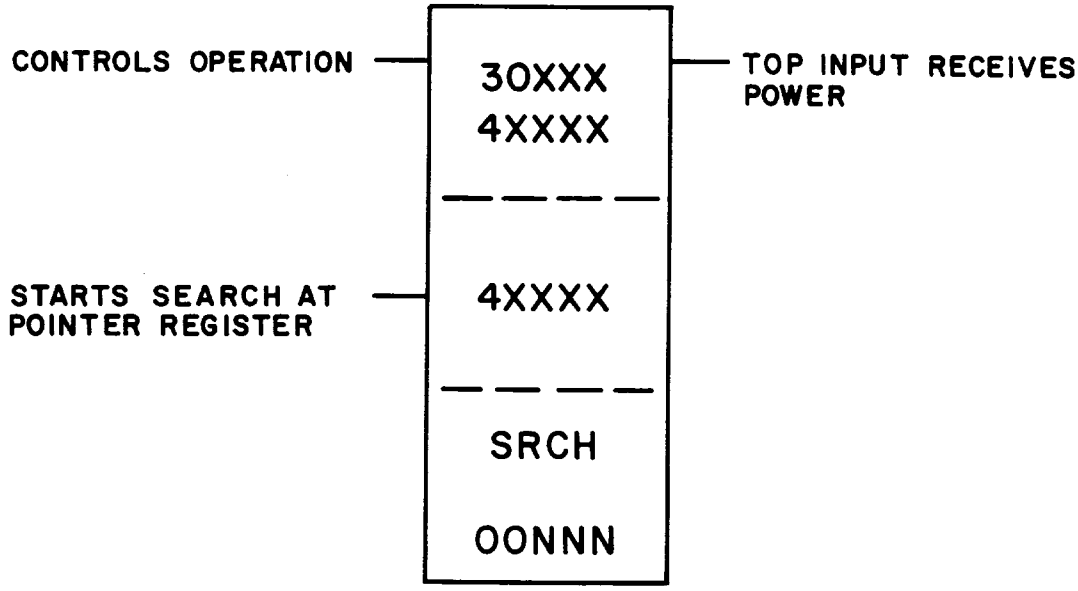


Figure 3-10. FIFO Moves

3.6 TABLE SEARCH OPERATION (SRCH)

FUNCTION

The Table Search function searches a table of registers for a specified value. When a matching value is found, the operation stops and the pointer value indicates the register in which the match is.



FUNCTION BLOCK

- The top node is the source node. It can be either a 30XXX input register reference or a 4XXXX holding register reference. This register is the first register of a table.

- The middle node is the destination node. It is a 4XXXX holding register which is used as a pointer and it also holds the pointer value. This value indicates the location of the register containing a match. The next consecutive holding register, 4XXXX + 1, contains the value being searched for; this value can be a 4-digit number up to 9999, a 16 bit binary pattern, or two ASCII characters.

- The bottom node contains the symbol SRCH and the numerical value that specifies the table length. This constant can range from 1 to 100.

INPUTS

- The top input controls the operation. When it receives power, each register in a table is examined to see if it contains a specified value. The search begins at the first register in a table unless the middle input is receiving power and the pointer value is greater than zero.

- The middle input, when it receives power, begins the search operation at the register whose location is specified in the pointer register, or continues the search operation from the register in which the match was found. If the pointer value is greater than zero and the middle input does not receive power, the search begins at the first register in the table.

OUTPUTS

- The top output passes power when the top input receives power.
- The middle output passes power when a match is found. If no match is found in the scan of an entire table, this output does not pass power and the pointer is reset to zero.

EXAMPLE

The following paragraphs provide a detailed explanation of the logic used in Figure 3-11.

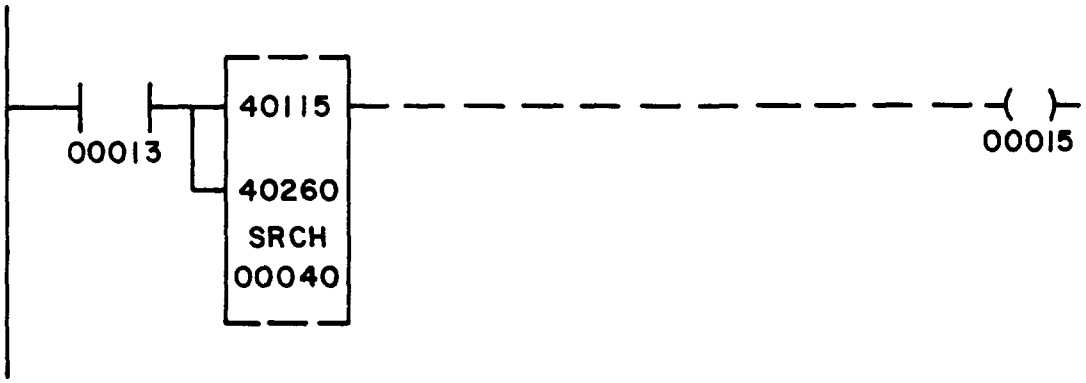


Figure 3-11. Table Search Logic

When contact 00013 is energized, the top input receives power. The table, which starts at register 40115, is searched to see if it holds the same value as in register 40261. Register 40260 holds the pointer value.

If a match is found, the search stops, the position number of the register is placed in 40260, and the middle output passes power.

To search for additional matches, both the top and middle inputs must receive power. If this condition is true, the search continues at the next consecutive register after the one containing a match. If no match is found, the middle output does not pass power and the pointer is reset to zero.

Figure 3-12 illustrates the Table Search described above.

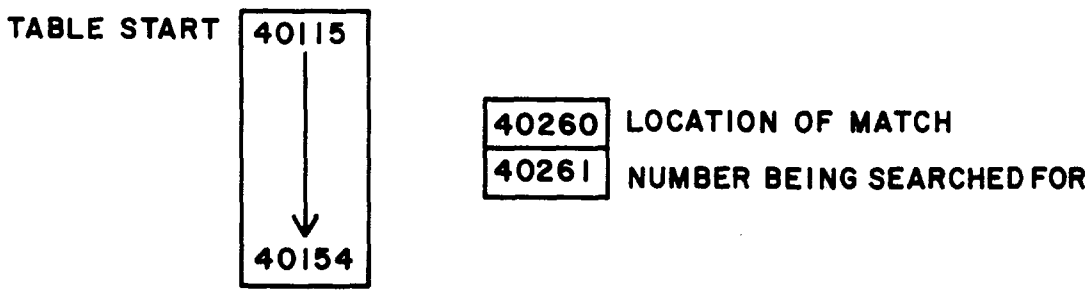
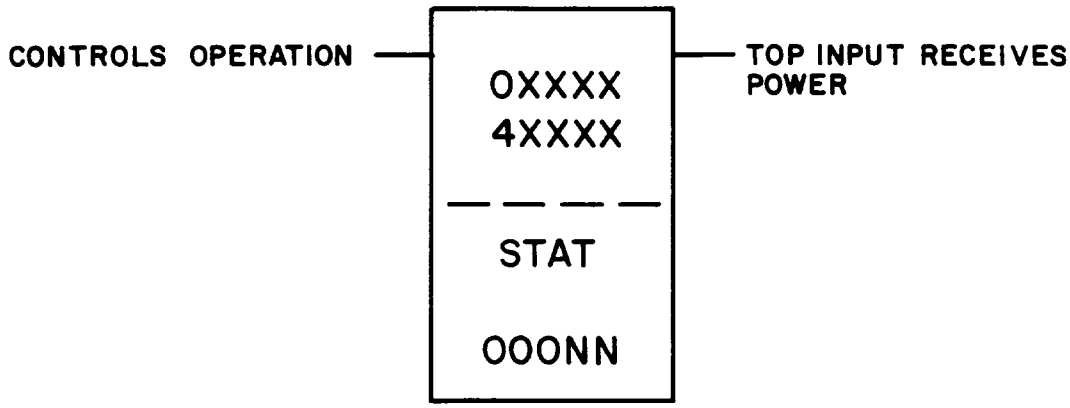


Figure 3-12. Table Search

3.7 GET CONTROLLER SYSTEM STATUS (STAT)

FUNCTION

The Get Controller System Status function obtains vital information about the controller, such as: memory protect status, battery status, I/O error, loss of active lights, and J200 Remote I/O Interface status. The information is placed in a table of registers or discretes.



FUNCTION BLOCK

- The top node is the destination node. It specifies the registers or discretes which will hold the status information. It can be either a 0XXXX logic coil reference or a 4XXXX holding register reference. It is a table of registers or groups of discretes. Each register has 16 bit locations and discretes are in groups of sixteen.

- The bottom node contains the symbol STAT and the numerical value that specifies the table length. This constant can range from 1 to 71.

WARNING

The STAT function overrides the disable state of a coil used in the destination node of the function block. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, because the coil's state can change as a result of the STAT function.

INPUT

- The top input controls the operation. When it receives power the controller status is copied into the registers specified in the top node.

OUTPUT

- The top output passes power when the top input receives power.

STATUS INFORMATION

Status information is located in specific words of memory. Each word has its own holding register. The status information available and the words containing the information are listed in Table 3-1. The registers listed in this table are just examples. Any group of seventy-one consecutive 4XXXX holding registers can be used or seventy-one consecutive groups of 16 coils.

Table 3-1. Status Words

Word(s)	Register(s)	Status
1	40101	Controller Status
4	40104	J200 Remote I/O Interface Status
12-27	40112-40127	Input Module Active Light Status
28-43	40128-40143	Output Module Active Light Status
44-71	40144-40171	Communication Status to J200 Drops (2 words/registers to each drop)

Words 2-3, and words 5-11 are for future use.

The Controller Status Word (register 40101) bit assignments are listed in Table 3-2. If the bit in the left column equals one (ON), the condition in the right column is true.

Bits in a word are numbered 1 to 16 and progress from left to right. Bit 1 is the most significant bit (MSB). Bit 16 is the least significant bit (LSB).

NOTE

The STAT block puts status words for remote channels I-4 after the status words for channel 32, instead of before the status words for channel 5.

Table 3-2. Controller Bit Status

Bit No.	Condition
1	Port 1 Set Up
2	Port 2 Set Up
3	Port 1 Dev # Entered
4	Port 2 Dev # Entered
5	Future Use
6	Enable Constant Sweep
7	Enable Single Sweep Delay
8	Max. 2048 Reference System
9	AC Power Failure
10	Run Light OFF
11	Memory Protect is OFF
12	Battery Backup Fault (CMOS Memory)
13	Future Use
14	Future Use
15	Future Use
16	Future Use

Each register or word of the I/O Module Active Light Status (registers 40112-40143) contains the input or output information for two channels. Table 3-3 illustrates this breakdown of channels into words or registers.

Table 3-3. I/O Module Active Light Word Status

Channels	Word	Register
1,2	12	40112
3,4	13	40113
5,6	14	40114
7,8	15	40115
9,10	16	40116
.	.	.
.	.	.
.	.	.
31,32	27	40127
1,2	28	40128
3,4	29	40129
5,6	30	40130
7,8	31	40131
9,10	32	40132
.	.	.
.	.	.
.	.	.
31,32	43	40143

NOTE

Words 12 through 27 contain input information. Words 28 through 43 contain output information.

Each register of the I/O Module Active Light Status (registers 40112-40143) has individual bit assignments. These are listed in Table 3-4.

A one bit indicates that the I/O slot has been enabled in the Traffic Cop and the slot does not contain a working I/O module.

A zero bit indicates that the I/O slot has either been inhibited in the Traffic Cop or the slot contains a working I/O module.

Table 3-4. I/O Module Active Light Bit Status

Odd Channels	
Bit No.	Slot No.
1	I/O Slot No. 1
2	I/O Slot No. 2
3	I/O Slot No. 3
4	I/O Slot No. 4
5	I/O Slot No. 5
6	I/O Slot No. 6
7	I/O Slot No. 7
8	I/O Slot No. 8
Even Channels	
Bit No.	Slot No.
9	I/O Slot No. 1
10	I/O Slot No. 2
11	I/O Slot No. 3
12	I/O Slot No. 4
13	I/O Slot No. 5
14	I/O Slot No. 6
15	I/O Slot No. 7
16	I/O Slot No. 8

Words 44-71 (Registers 40144-40171) represent the communication status to J200 drops. There are 14 drops, 2 channels each. Each drop uses two status words.

The bit assignments for each drop are the same. Table 3-5 contains the bit assignments for the first word in each drop (i.e., words 44, 46, 48, ...68, 70). The bit assignments for the second word in each drop (i.e., words 45, 47, 49, ... 69, 71) are listed in Table 3-6.

Table 3-5. Remote I/O Word One Bit Status

Bit No.	Condition
1-3	This field identifies the type of processing scheduled to be performed by the 584L for the addressed IOR. The following functions are defined: 000 — Normal I/O 001 — Restart Phase I (Communication Reset) 010 — Restart Phase II (Application Reset) 011 — Unassigned (INHIBIT IOR) 100 — INHIBIT 101 — Unassigned (INHIBIT IOR) 110 — Unassigned (INHIBIT IOR) 111 — Unassigned (INHIBIT IOR)
4	Sequence number received on response from the addressed IOR is different from the expected value.
5	Byte count underrun. The internal IOR response byte count is incompatible with the IOL transmitted byte count.
6	The current message response received by the 584L from the addressed IOR is not supported.
7	Future Use
8	The current 584L message has not been accepted by the addressed IOR.
9-11	The receive sequence number. Indicates the number (modulo 8) of the next information frame that is expected to be received from the addressed IOR.
12	Identifies the cable to be used by the IOL to communicate with the addressed IOR (0 = Cable 0, 1 = Cable 1).
13-15	The send sequence number. Indicates the number (modulo 8) of the current information frame that is being sent to the addressed IOR.
16	The current 584L message has been queued for processing by the addressed IOR.

Table 3-6. Remote I/O Word Two Bit Status

Bit No.	Condition
1	Future Use
2	The IOL sensed a character overrun error from the addressed IOR.
3	The IOL sensed a communication error from the addressed IOR. A communication error is a CRC failure; receives abort or residual bit count error.
4	The addressed IOR did not respond to the latest 584L command message.
5	Future Use
6	Link reject response issued by the addressed IOR indicating that the IOR has just gone through its power-up sequence.
7	Link reject response issued by the addressed IOR indicating that the current message has an incompatible sequence number.
8	Link reject response issued by the addressed IOR indicating that the current command is not being supported by the IOR.
9-16	This field counts the number of retries attempted by the 584L to the addressed IOR. The maximum value is 255_{10}.

EXAMPLE
The following paragraph provides a detailed explanation of the logic used in Figure 3-13.

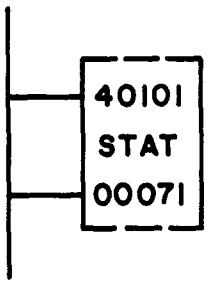


Figure 3-13. Get Controller System Status (STAT)

The status information is obtained every scan because the function block is next to the power rail. Since the length is 71, all the available status information is obtained in registers 40101 to 40171.

3.8 SUMMARY OF MOVE FUNCTIONS

	Source	Destination	Max. Length	Top Input	Middle Input	Bottom Input	Top Output	Middle Output	Bottom Output
R→T	Any Reference	4XXXX Pointer	255/999	Move, Increase Pointer	No Increasing	Enable/ Reset	Top Input Receiving Power	Table Full	Not Used
T→R	Any Reference	4XXXX Pointer	255/999	Move, Increase Pointer	No Increasing	Enable/ Reset Power	Top Input Receiving	Table Full	Not Used
T→T	Any Reference	4XXXX Pointer	255/999	Move, Increase Pointer	No Increasing	Enable/ Reset	Top Input Receiving Power	Table Full	Not Used
BLKM	Any Reference	0XXXX, or 4XXXX	100	Move	Not Used	Not Used	Top Input Receiving Power	Not Used	Not Used
FIN	Any Reference	4XXXX Pointer	100	Move, Increase Pointer	Not Used	Not Used	Top Input Receiving Power	Queue Full	Queue Empty
FOUT	4XXXX Pointer	0XXXX, or 4XXXX	100	Move, Decrease Pointer	Not Used	Not Used	Top Input Receiving Power	Queue Full	Queue Empty
SRCH	30XXX, or 4XXXX	4XXXX Pointer	100	Search	Start at Pointer	Not Used	Top Input Receiving Power	Value Found	Not Used
STAT	—	0XXXX, or 4XXXX	71	Move	—	Not Used	Top Input Receiving Power	—	Not Used

3.9 LOGIC EXAMPLES

3.9.1 Analog Multiplexing

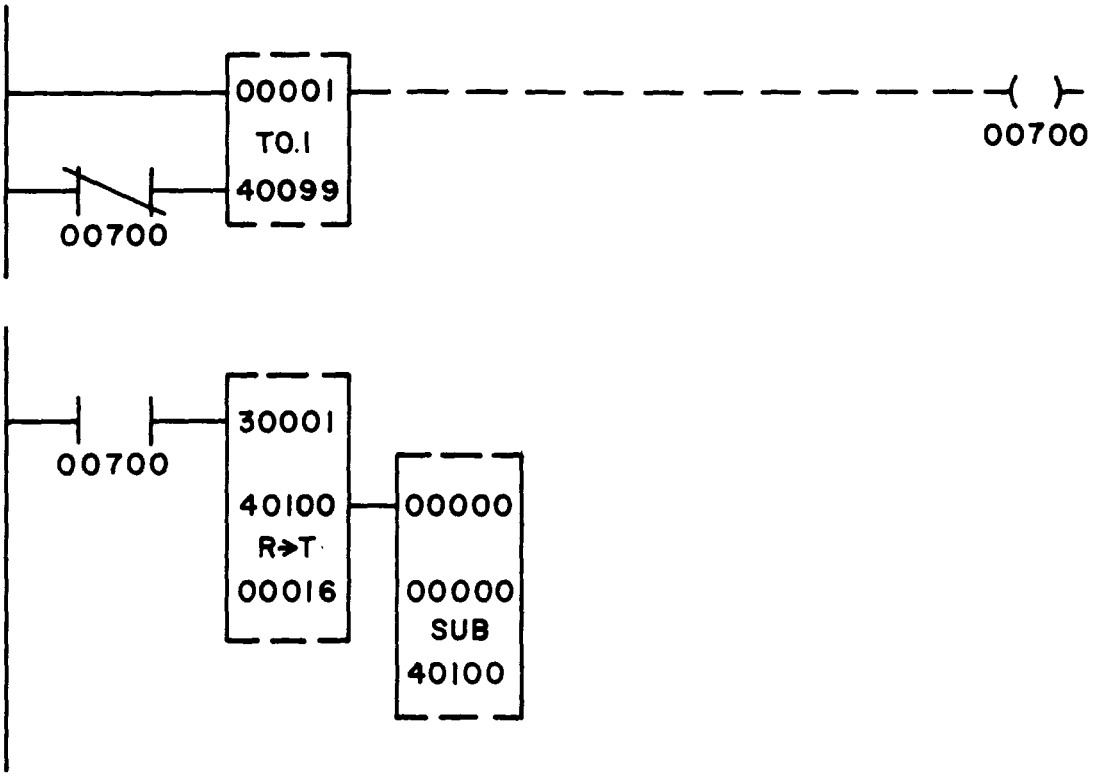


Figure 3-14. Analog Multiplexing

The B258 Analog Multiplexer module is used with a B243 Analog to Digital Converter module and the 584L PC. The B258 module multiplexes one of sixteen analog signals into one input of a B243 module.

The logic needed to perform this operation is shown in Figure 3-14. The first network contains a one-tenth of a second timer which is used to initiate the reading of the analog input. Every tenth of a second, coil 00700 is energized and the R⟶T function block in the second network receives power.

When this input receives power, the value in binary input register 30001 is moved into the first location (40101) in a table of sixteen registers starting at 40101. After the first move, the counter automatically increments so the next value is inserted into register 40102. This R⟶T move continues until the table (40101-40116) is full (pointer value equals sixteen). When the table is full the middle output passes power and the subtract block receives power. The subtract clears the pointer value (40100) to zero for the next scan.

NOTE

For this example, the 584L traffic cop must contain 30001 as a binary input register and 40100 as a BCD output register.

3.9.2 Recipe Storage

The following paragraphs provide a detailed explanation of the logic illustrated in Figure 3-15 to program a recipe storage function.

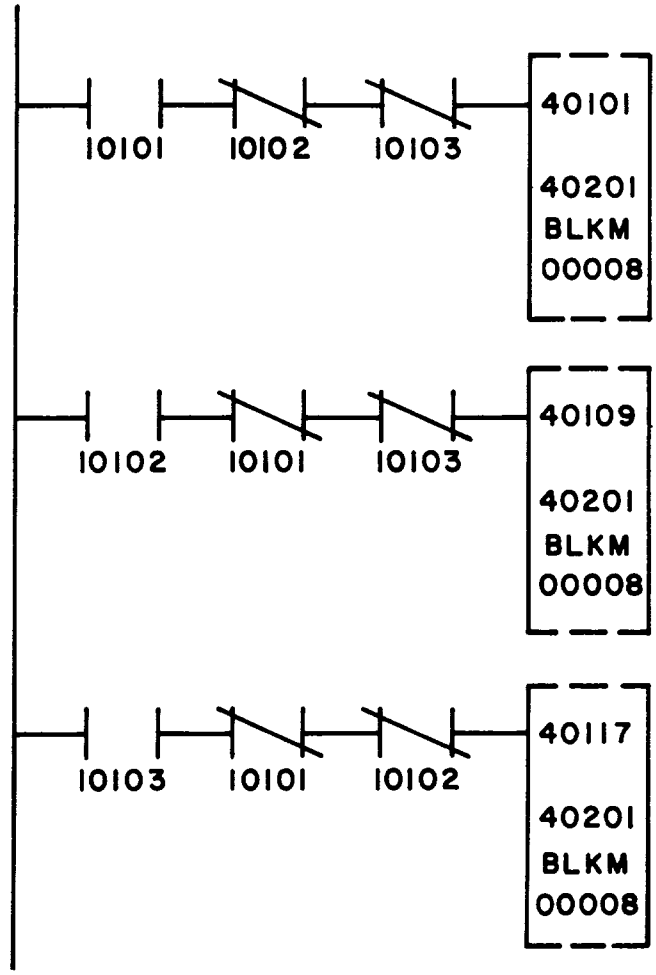


Figure 3-15. Recipe Storage

In some applications, information in various tables is needed, one table at a time, to perform various functions. An example of this is a manufacturer who manufacturers three products which are similar yet have specific differences. An example of this is soups. Soups are similar, but chicken soup is different from tomato soup.

The information or recipe for each soup is stored in a unique table. Since only one soup is made at a time, a working table is needed which can apply to any of the three soups. This is accomplished with a block move as shown in Figure 3-15. The tables for all the soups contain specific information in corresponding registers. These registers must also correspond with the working table's registers. For example, if the first register in one table contains cooking time, the first register in all the tables must contain cooking time.

The process is controlled from an operator panel. The panel can have three input switches; 10101, 10102, and 10103. To make soup A, the operator turns 10101 ON, and 10102 and 10103 remain OFF.

Following the logic in Figure 3-15, input 10101 is energized and passes power through normally closed contacts 10102 and 10103. The recipe for soup A is moved from table 40101-40108 to table 40201-40208. Table 40201-40208 is a working table. Each output register in this table is controlling a specific part of the operation.

If the original recipe tables are used as working tables, three individual programs are required. By using one working table as illustrated in this example, only one program is needed to control the output information.

SECTION 4
DATA TRANSFER (DX) MATRIX FUNCTIONS

A matrix is an array of databits made by one or more consecutive registers in the 584L. The size of a matrix, in bits, is in even multiples of 16 (e.g., 16, 32, 48, 64, 80, etc.); each register contains 16 data bits.

Figure 4-1 is an example of the format of a 3-register matrix:

Register	Bits															
40051	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
40052	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
40053	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48

Figure 4-1. Matrix Format

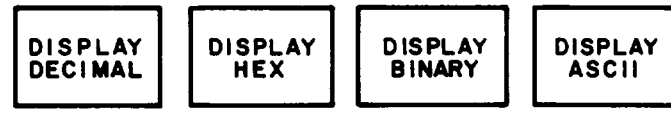
Each bit has a value of one or zero. Registers are generally displayed in decimal format.

To obtain a register value in bit format, do the following:

1. Move the cursor to the appropriate area at the bottom of the P190 screen.
2. Enter the register number (e.g., 40250) into the AR.
3. Press the ERASE/GET key on the P190 panel.

The register number appears with a value next to it in decimal format (e.g., 40250 = 0023 DECIMAL).

The following software labels are displayed:



4. Press the DISPLAY BINARY software label key.

   The following is displayed on the screen:

   40250 = 0000000000010111

Each bit has a value assigned to it. To determine the value of the register in binary format, add the individual bits.

The values are:

BITS	1	2	3	4	5	6	7	8
VALUES	32768	16384	8192	4096	2048	1024	512	256
BITS	9	10	11	12	13	14	15	16
VALUES	128	64	32	16	8	4	2	1

If the bit pattern is 0000000001100100, the value is 100 (64 + 32 + 4). If the bit pattern is 0000000111110100, the value is 500 (256 + 128 + 64 + 32 + 16 + 4).

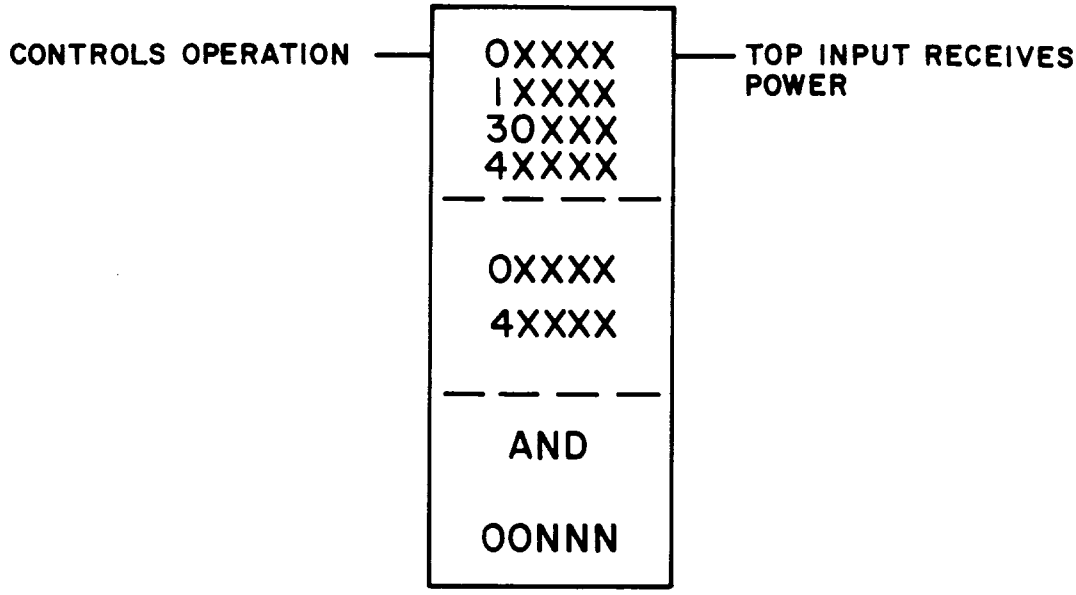
The DX MATRIX functions can revise data, shift data, or examine data in a matrix with or without altering the source.

Each DX MATRIX function block occupies three nodes in a 10 X 7 node network format and consists of a source node, a destination node, and a node specifying matrix length in registers. The top input is the control input; when it receives power the function is performed. The top output passes power when the top input receives power. This allows function blocks to be cascaded within a network.

4.1 LOGICAL AND OF TWO MATRICES (AND)

FUNCTION

The Logical AND function takes the result of a mathematical operation (AND) on two matrices and places the result in the second matrix. The value 0 or 1 of each bit in the result is determined by the values in the two matrices. A resulting bit is a one bit (ON) if both bits, one from each matrix, are one bits; if either bit or both bits are zeros, the resulting bit is a zero bit (OFF). Section 4.10.1 contains a truth table which illustrates this.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source matrix is a group of registers or discretes.

- The middle node is the second source matrix as well as the destination node. It can be either a 0XXXX logic coil reference or a 4XXXX holding register reference. The destination matrix is the same size as the source.

WARNING

The AND function overrides the disable state of a coil used in the destination node of the function block. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, because the coil's state can change as a result of the AND function.

- The bottom node contains the symbol AND and the numerical value that specifies the matrix length in registers or groups of discretes for both matrices. This constant can range from 1 to 100 (i.e., a 3 indicates 48 bits).

INPUT

- The top input controls the operation. When it receives power, the logical AND function is performed.

OUTPUT

- The top output passes power when the top input receives power.

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 4-2.

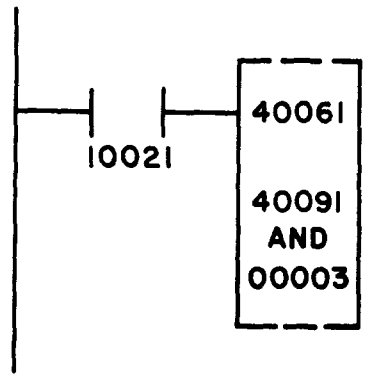
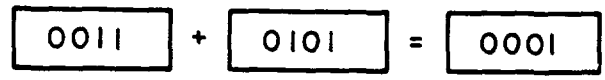


Figure 4-2. Logical AND

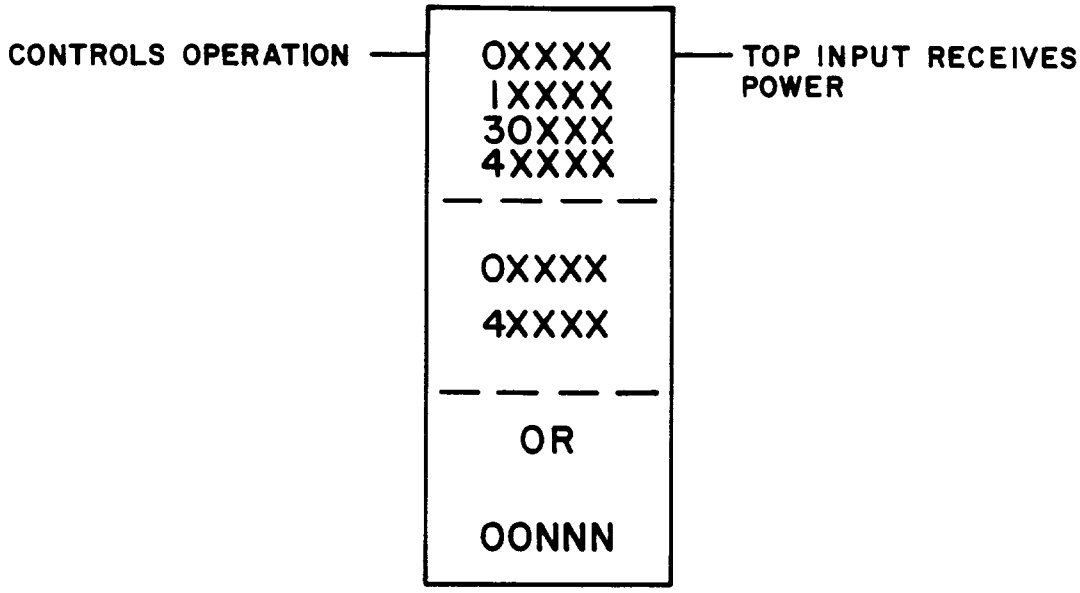
When input 10021 is energized, the top input receives power. The bit pattern in registers 40061 — 40063 is logically ANDed with the current bit pattern in registers 40091-40093. The resulting bit pattern is placed in registers 40091-40093, thereby replacing the previous bit pattern. The source registers (40061-40063) are not altered. In applications in which the original information in registers 40091-40093 cannot be lost, the information must be copied into another table before the AND takes place. One way to do this is to use a BLOCK MOVE. See Section 3.4.



If bits 1 through 4 in register 40061 are 0011, and bits 1 through 4 in register 40091 are 0101, the result placed in register 40091 is 0001.

4.2 LOGICAL INCLUSIVE OR OF TWO MATRICES (OR)

FUNCTION
The Logical Inclusive OR function takes the result of a mathematical operation (OR) on two matrices and places the result in the second matrix. The value 0 or 1 of each bit in the result is determined by the values in the two matrices. A resulting bit is a one bit (ON) if either bit or both bits, one from each matrix, are one bits; if both bits are zeros, the resulting bit is a zero bit (OFF).



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source matrix is a group of registers or discretes.

- The middle node is the second source matrix as well as the destination node. It can be either a 0XXXX logic coil reference or a 4XXXX holding register reference. The destination matrix is the same size as the source.

WARNING

The OR function overrides the disable state of a coil used in the destination node of the function block. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, because the coil's state can change as a result of the OR function.

- The bottom node contains the symbol OR and the numerical value that specifies the matrix length in registers or groups of discretes for both matrices. This constant can range from 1 to 100 (i.e., a 3 indicates 48 bits).

INPUT

- The top input controls the operation. When it receives power, the logical inclusive OR function is performed.

OUTPUT

- The top output passes power when the top input receives power.

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 4-3.

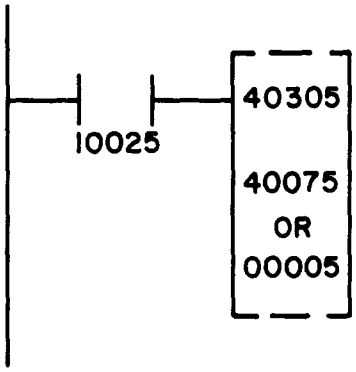


Figure 4-3. Logical Inclusive OR

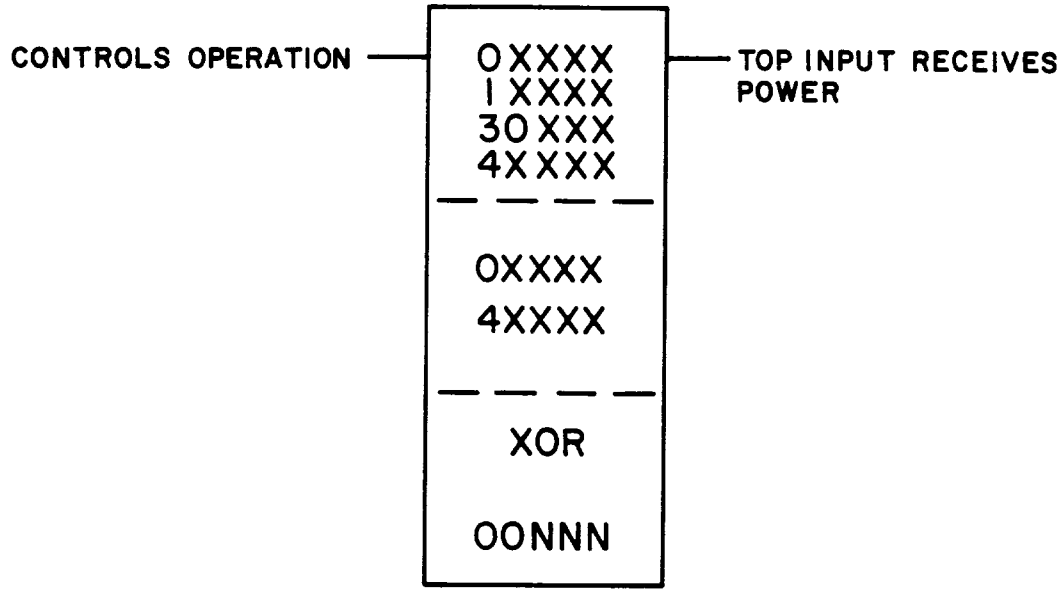
When input 10025 is energized, the bit pattern in registers 40305-40309 is logically ORed with the current bit pattern in registers 40075-40079. The resulting bit pattern is placed in registers 40075-40079, thereby replacing the previous bit pattern. The source registers (40305-40309) are not altered.

In applications in which the original information in registers 40075-40079 cannot be lost, the information must be copied into another table before the OR takes place. One way to do this is to use a BLOCK MOVE. See Section 3.4.

If bits 1 through 4 in register 40305 are 0011, and bits 1 through 4 in register 40075 are 0101, the result placed in register 40075 is 0111.

4.3 LOGICAL EXCLUSIVE OR OF TWO MATRICES (XOR)

FUNCTION
The Logical Exclusive OR (XOR) function takes the result of a mathematical operation on two matrices and places the result in the second matrix. The value 0 or 1 of each bit in the result is determined by the values in the two matrices. A resulting bit is a one bit (ON) if either bit, one from each matrix, is a one bit; if they are both zero bits (OFF) or both one bits (ON), the resulting bit is a zero bit (OFF). Section 4.10.1 contains a truth table which illustrates this.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source matrix is a group of registers or discretes.

- The middle node is the second source matrix as well as the destination node. It can be either a 0XXXX logic coil reference or a 4XXXX holding register reference. The destination matrix is the same size as the source.

WARNING

The XOR function overrides the disable state of a coil used in the destination node of the function block. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, because the coil's state can change as a result of the XOR function.

- The bottom node contains the symbol XOR and the numerical value that specifies the matrix length for both matrices. This constant can range from 1 to 100 (i.e., a 3 indicates 48 bits).

INPUT

- The top input controls the operation. When it receives power, the Logical Exclusive OR (XOR) function is performed.

OUTPUT

- The top output passes power when the top input receives power.

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 4-4.

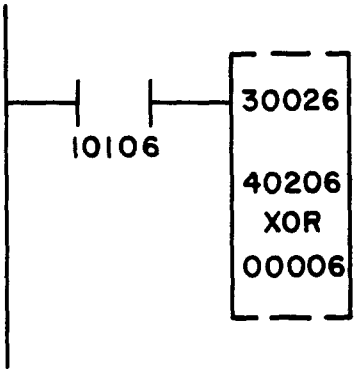
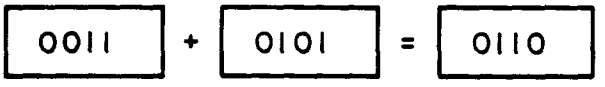


Figure 4-4. Logical Exclusive OR (XOR)

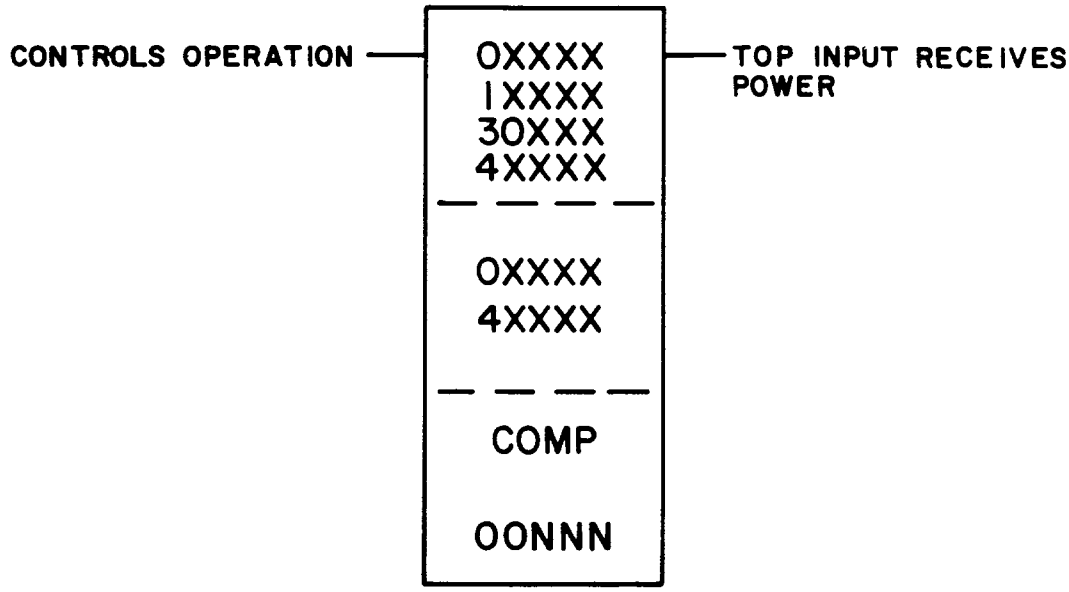
When input 10106 is energized, the top input receives power. A Logical Exclusive Or is performed between the bit patterns in registers 30026-30031 and registers 40206-40211. The resulting bit pattern is placed in registers 40206-40211, thereby replacing the previous bit pattern. The source registers (30026-30031) are not altered. In applications in which the original information in registers 40206-40211 cannot be lost, the information must be copied into another table before the XOR takes place. One way to do this is to use a BLOCK MOVE. See Section 3.4.

If bits 1 through 4 in register 30026 are 0011, and bits 1 through 4 in register 40206 are 0101, the result placed in register 40206 is 0110.



4.4 LOGICAL COMPLEMENT OF ONE MATRIX (COMP)

FUNCTION
The Logical Complement function causes the content of one matrix to be complemented — all ones are replaced by zeros, and all zeros are replaced by ones. The result is placed in another matrix.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source matrix is a group of registers or discretes.

- The middle node is the destination node. It can be either a 0XXXX logic coil reference or a 4XXXX holding register reference. The destination matrix is the same size as the source.

WARNING

The COMP function overrides the disable state of a coil used in the destination node of the function block. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, because the coil's state can change as a result of the COMP function.

- The bottom node contains the symbol COMP and the numerical value that specifies the matrix length for both matrices. This constant can range from 1 to 100 (i.e., a 3 indicates 48 bits).

INPUT

- The top input controls the operation. When it receives power, the logical complement (COMP) function is performed.

OUTPUT

- The top output passes power when the top input receives power.

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 4-5.

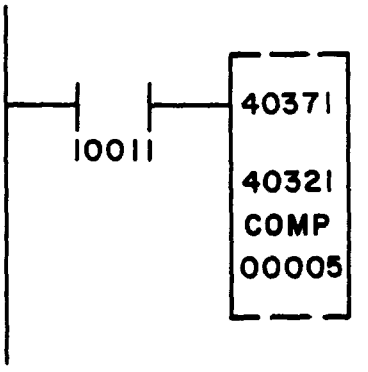


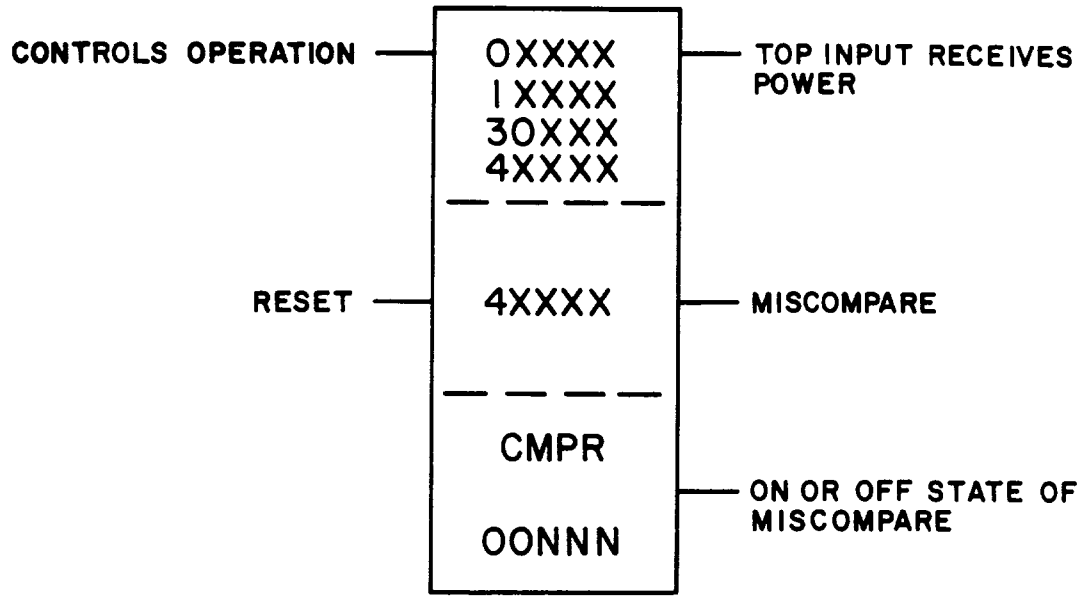
Figure 4-5. Logical Complement

When input 10011 is energized, the bit pattern in registers 40371-40375 is complemented — ones changed to zeros, and zeros changed to ones. The resulting bit pattern is placed in registers 40321-40325. The source registers (40371-40375) are not altered.

If bits 1 through 4 in register 40371 are 0011, the result placed in register 40321 is 1100.

4.5 LOGICAL COMPARE OF TWO MATRICES (CMPR)

FUNCTION
The Logical Compare function compares two matrices bit-by-bit. The contents of the matrices are only examined, not altered. If two bits agree, both zeros or both ones, then the next two bits are compared. If the two bits do not agree, the function stops and the pointer contains the position of the bit that did not agree. The result of the function, miscompare or agreement, is indicated by the middle output. If no miscompare is found, the function stops at the end of a table.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source matrix is a group of registers or discretes.

- The middle node is the destination node. It is a 4XXXX holding register reference. This register holds the pointer value that controls which bit the compare starts at, and when the compare is done, it indicates which bit is a miscompare. The matrix is located in consecutive registers immediately following the pointer, starting at 4XXXX + 1.

- The bottom node contains the symbol CMPR and the numerical value that specifies the matrix length in registers or groups of discretes for both matrices. This constant can range from 1 to 100 (i.e., a 3 indicates 48 bits).

INPUTS

- The top input controls the operation. When it receives power, two matrices are compared bit-by-bit.

- The middle input, when receiving power, resets the pointer value to zero.

NOTES

The position of the matrix bit being compared at any one time is equal to the pointer value plus one.

If the pointer value is greater than or equal to the matrix length, the controller resets the pointer value to zero before performing the function.

OUTPUTS

- The top output passes power when the top input receives power.

- The middle output passes power if a miscompare is found. This output does not pass power when: the top input is not receiving power, no miscompare is found, or the end of the matrix is reached.

- The bottom output is controlled by a miscompare, middle output. If a miscompare is found, the bottom output indicates the ON or OFF state of the bit in the first matrix. If the bit is a one, this output passes power. The output does not pass power if the bit is a zero.

EXAMPLE
The following paragraphs provide a detailed explanation of the logic used in Figure 4-6.

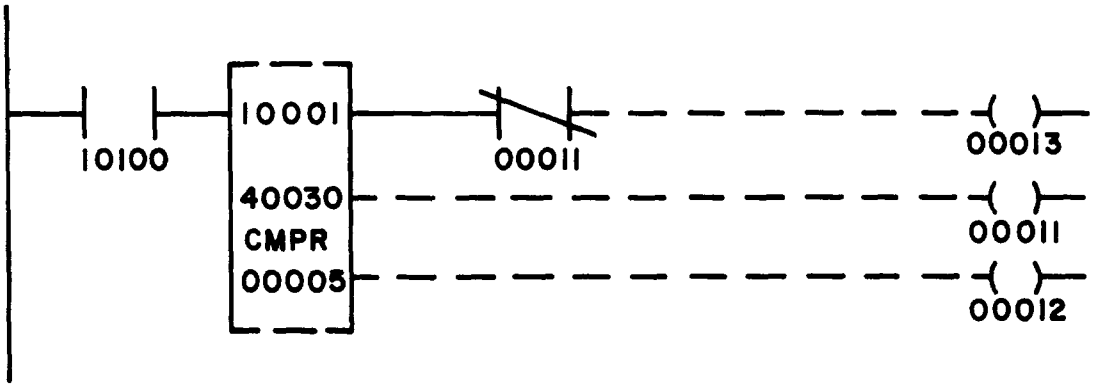


Figure 4-6. Logical Compare

When input 10100 is energized, the top input receives power. Discrete inputs 10001-10081 are compared bit-by-bit with registers 40031-40035. The registers and discretes are not altered. If two bits, one from each matrix, do not agree, the function stops and coil 00011 is energized. Coil 00012 indicates the status of the miscompare.

If a bit in 10001 is a one bit and it is compared with a zero bit in register 40031, the bottom output passes power and energizes coil 00012. If discrete 10001 contains a zero bit which is compared to a one bit in register 40031, the bottom output does not pass power and coil 00012 is not energized. In both cases, the location of the bit that miscompared is placed in register 40030 and the compare function continues from that point.

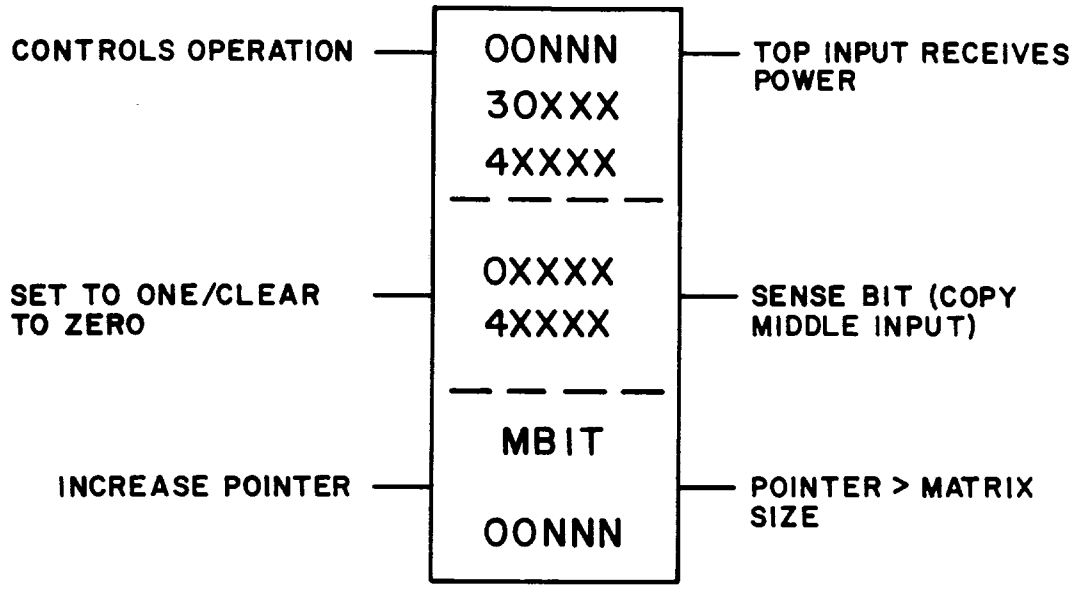
If a normal relay contact is used, the compare function continues after finding a miscompare so the pointer value changes. If the location of the miscompare is bit 5, the compare continues at bit 6.

To detect the end of the table if no miscompare has been found, a vertical short and a normally closed contact referenced to coil 00011 are placed beside the top output of the function block. If no miscompare is found, coil 00011 remains OFF and therefore normally closed contact 00011 passes power and energizes coil 00013, no miscompare and end of table.

4.6 LOGICAL BIT MODIFY (MBIT)

FUNCTION

The Logical Bit Modify function alters the state of individual bits in a matrix. Only one bit can be altered per scan; it can be set to one, or cleared to zero.



FUNCTION BLOCK

- The top node is the pointer that controls which bit is modified. It can be a 30XXX input register reference, a 4XXXX holding register reference, or a constant, up to 999 in a Level I 584L PC or up to 9600 in a Level 2 584L PC. If a 4XXXX holding register is used, the pointer value can be increased by control of the bottom input.

- The middle node is the source and destination node; the revised data replaces the original data in the matrix. It can be either a 0XXXX logic coil reference or a 4XXXX holding register reference. If a logic coil is used it can only be used once, in this function block. The logic coil cannot be used in another function block or in the eleventh column of a network; it can be used as a relay contact.

WARNING

The MBIT function overrides the disable state of a coil used in the destination node of the function block. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, because the coil's state can change as a result of the MBIT function.

- The bottom node contains the symbol MBIT and the numerical value that specifies the matrix length. This constant can range from 1 to 255 registers or groups of discretes (16 to 4080 bits) in a Level I 584L PC or from 1 to 600 registers (16 to 9600 bits) in a Level 2 584L PC.

INPUTS

- The top input controls the operation. When it receives power, the bit specified in the pointer register is either set to one or cleared to zero.

- The middle input controls whether the bit is set to one or cleared to zero. If this input receives power, the bit is set to one; if no power is received, the bit is cleared to zero.

- The bottom input, when receiving power with the top input, increases the pointer value. This is possible only if a 4XXXX reference is in the top node.

NOTES

If the pointer value is increased beyond the matrix size, the controller resets the pointer value to one before performing the function.

If a pointer value is inserted which is greater than the matrix size, the pointer value is not reset, the function is not performed, and the bottom output passes power.

OUTPUTS

- The top output passes power when the top input receives power.

- The middle output passes power when the bit being modified is set to one. It passes power when the middle input receives power.

- The bottom output passes power if the pointer value is greater than the matrix size. In this case, no operation is performed and the pointer value is set to the matrix size.

EXAMPLE

The following paragraph provides a detailed explanation of the logic used in Figure 4-7.

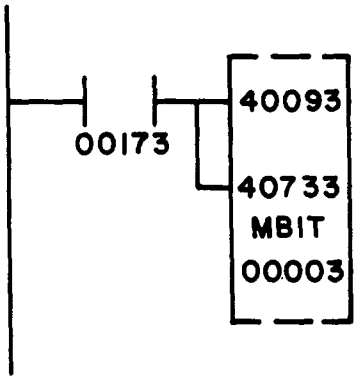


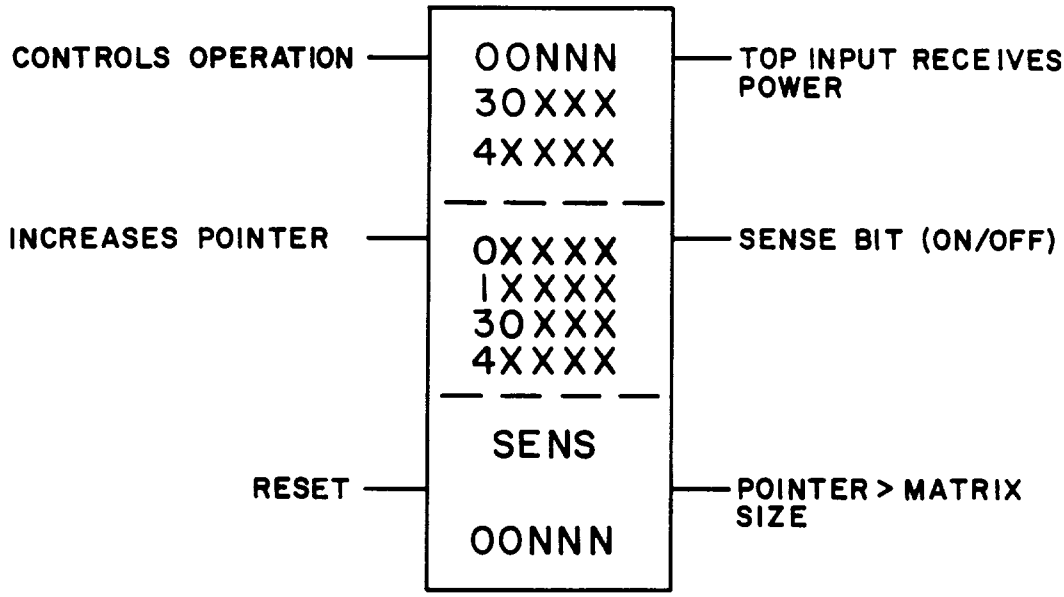
Figure 4-7. Logical Bit Modify

When contact 00173 is energized, the bit in matrix 40733-40735 at the location indicated by the pointer value (register 40093), is set to one, regardless of what it was before. To clear the bit to zero, remove the vertical connection between the top and middle inputs.

4.7 LOGICAL BIT SENSE (SENS)

FUNCTION

The Logical Bit Sense function examines and reports the state of individual bits in a matrix. Only one bit can be examined per scan.



FUNCTION BLOCK

- The top node holds the pointer value that controls which bit is examined. It can be a 30XXX input register reference, a 4XXXX holding register reference, or a constant, up to 999 in a Level I 584L PC or up to 9600 in a Level 2 584L PC. If the reference is a 4XXXX holding register, the pointer value can be increased by control of the middle input.

- The middle node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source matrix is a group of registers or discretes.

- The bottom node contains the symbol SENS and the numerical value that specifies the source matrix length. This constant can range from 1 to 255 registers or groups of discretes (16 to 4080 bits) in a Level I 584L PC or from 1 to 600 registers or groups of discretes (16 to 9600 bits) in a Level 2 584L PC.

INPUTS

- The top input controls the operation. When it receives power, one bit in a matrix is examined and its status is reported.

- The middle input, when receiving power with the top input, increases the pointer value. This is possible only if a 4XXXX reference is in the top node.

NOTES

If the pointer value is increased beyond the matrix size, the controller resets the pointer value to one before performing the function.

If a pointer value is inserted which is greater than the matrix size, the pointer value is not reset, the function is not performed, and the bottom output passes power.

- The bottom input, when receiving power, resets the pointer value to one. This is only possible if the pointer is a 4XXXX reference.

OUTPUTS

- The top output passes power when the top input receives power.
- The middle output passes power when the bit being examined is a one bit. This output does not pass power when the bit is a zero bit.
- The bottom output passes power if the pointer value is greater than the matrix size. In this case, no operation is performed.

EXAMPLE

The following paragraphs provide a detailed explanation of the logic used in Figure 4-6.

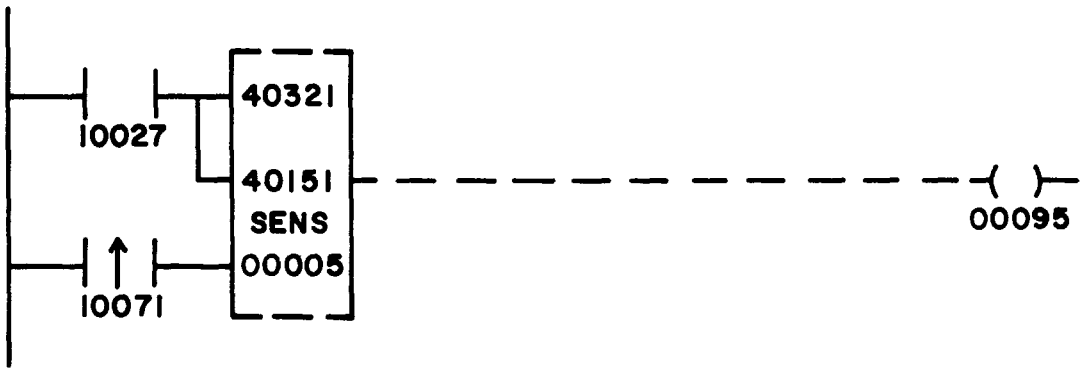


Figure 4-8. Logical Bit Sense

When input 10027 is energized, the top and middle inputs receive power and the bit, in matrix 40151-40155 at the location indicated by the pointer value (register 40321), is examined. Only one bit is examined per scan.

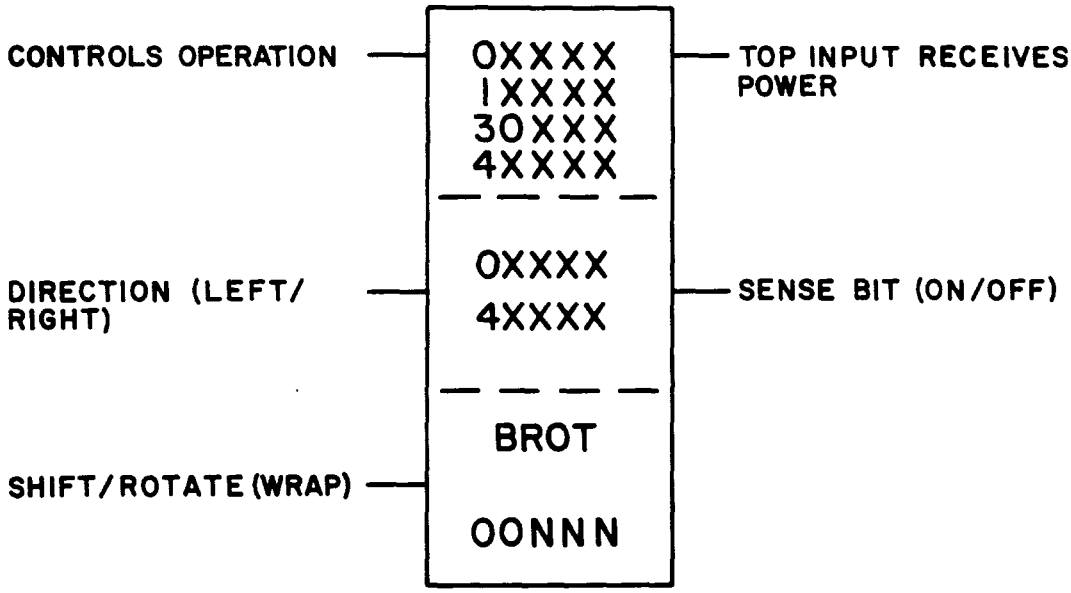
If the bit is a one bit, the middle output passes power and energizes coil 00095. If the bit is a zero bit, no power is passed. Since the middle input receives power by way of the vertical connection between the top and middle inputs, the pointer increases by one during each scan that input 10027 is energized.

To reset the pointer, input 10071 must be energized. A transitional contact is used to ensure that the pointer is reset only once, no matter how many scans input 10071 remains energized.

4.8 LOGICAL BIT ROTATE (BROT)

FUNCTION

The Logical Bit Rotate function shifts or rotates bits in a matrix. The bits can be rotated to the left or to the right, and if bits are moved beyond the boundaries of the matrix, they can be wrapped around to the vacated bits at the start of the matrix. If the bits shifted out of the matrix are not wrapped around, the vacated bits are filled with zeros.



FUNCTION BLOCK

- The top node is the source node. It can be one of the following references: a 0XXXX logic coil, a 1XXXX discrete input, a 30XXX input register, or a 4XXXX holding register. The source matrix is a group of registers or discretes. Its content or status is not altered.

- The middle node is the destination node. It can be either a 0XXXX logic coil reference or a 4XXXX holding register reference. The destination matrix is the same size as the source. If a logic coil is used, it can only be used once, in this function block. The logic coil cannot be used in another function block or in the eleventh column of a network; it can be used as a relay contact.

WARNING

The BROT function overrides the disable state of a coil used in the destination node of the function block. This can cause personal injury if the user assumes a coil has disabled an operation and repairs are being made, because the coil's state can change as a result of the BROT function.

- The bottom node contains the symbol BROT and the numerical value that specifies the length of both matrices in registers or discretes. This constant can range from 1 to 100 registers or groups of discretes (16 to 1600 bits).

INPUTS

- The top input controls the operation. When it receives power, all the bits in a matrix are rotated or shifted one position per scan.

- The middle input controls the direction of the shift. If this input receives power, the bits are shifted toward the left (i.e., bit 17 into 16, bit 16 into 15, ... bit 3 into 2, bit 2 into 1, bit 1 out of the matrix). If this input does not receive power, the bits are shifted toward the right (i.e., bit 1 into 2, bit 2 into 3, ... bit 15 into 16, bit 16 into 17, etc.). The last bit is shifted out of the matrix.

- The bottom input controls the wrap-around of the bits. If it receives power, the bits shifted out of the matrix are carried around unchanged and entered into the opposite end of the matrix — wrapped around. If this input does not receive power, the bits shifted out of the matrix are discarded. The vacant bit positions at the opposite end of the matrix are filled with zeros.

OUTPUTS

- The top output passes power when the top input receives power.

- The middle output passes power when the bit being shifted out of the matrix is a one bit, regardless of whether the bit is being wrapped around or discarded.

EXAMPLE

The following paragraphs provide a detailed explanation of the logic used in Figure 4-9.

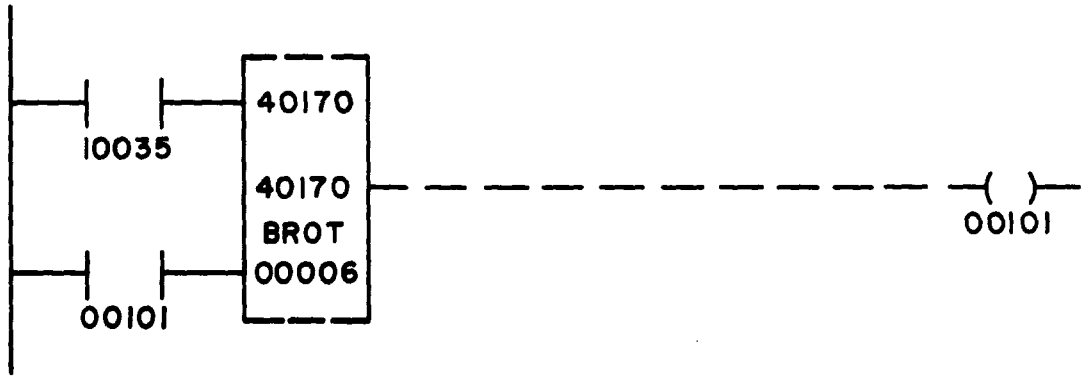


Figure 4-9. Logical Bit Rotate

When input 10035 is energized, the top input receives power and all the bits in matrix 40170-40175 are shifted one bit position to the right. The last bit, bit 96, is shifted out of the matrix.

If this bit is a one bit (ON), the middle output passes power and energizes coil 00101. Since the bottom input is not receiving power, bit 1 is filled with a zero. If the bottom input had been receiving power, the bit shifted out of the matrix would have been wrapped around into the first bit position in the matrix.

4.9 SUMMARY OF MATRIX FUNCTIONS

	Source	Destination	Max. Length	Top Control	Middle Input	Bottom Input	Top Output	Middle Output	Bottom Output
AND	Any Reference	0XXXX, or 4XXXX	100	Control	Not Used	Not Used	Top Input Receives Power	Not Used	Not Used
OR	Any Reference	0XXXX, or 4XXXX	100	Control	Not Used	Not Used	Top Input Receives Power	Not Used	Not Used
XOR	Any Reference	0XXXX, or 4XXXX	100	Control	Not Used	Not Used	Top Input Receives Power	Not Used	Not Used
COMP	Any Reference	0XXXX, or 4XXXX	100	Control	Not Used	Not Used	Top Input Receives Power	Not Used	Not Used
CMPR	Any Reference	4XXXX, Pointer	100	Control	Reset Pointer	Not Used	Top Input Receives Power	Miscompare	ON/OFF State of Miscompare
MBIT	00NNN, 30XXX, or 4XXXX Pointer	0XXXX, or 4XXXX	255/600	Control	Set to one/ Clear to zero	Increase Pointer	Top Input Receives Power	Copy Middle Input	Pointer Matrix Size
SENS	00NNN, 30XXX, or 4XXXX Pointer	Any Reference	255/600	Control	Increase Pointer	Reset Pointer	Top Input Receives Power	Copy Middle Input	Pointer Matrix Size
BROT	Any Reference	0XXXX, or 4XXXX	100	Control	Direction Left/Right	Shift/ Rotate	Top Input Receives Power	ON/OFF State of Bit	Not Used

4.10 LOGIC EXAMPLES

The following are examples of applications using the logic discussed in Section 4.

4.10.1 Truth Table for AND, OR, and XOR Functions

The following truth table illustrates the results of performing AND, OR, and XOR functions on every possible bit combination.

Table 4-1. Truth Table for AND, OR, and XOR Functions

	Bit A	+	Bit B	=	Bit C
AND	0		0		0
	0		1		0
	1		0		0
	1		1		1
OR	0		0		0
	0		1		1
	1		0		1
	1		1		1
XOR	0		0		0
	0		1		1
	1		0		1
	1		1		0

4.10.2 Table Averaging

The following paragraphs provide a detailed explanation of the logic illustrated in Figure 4-10.

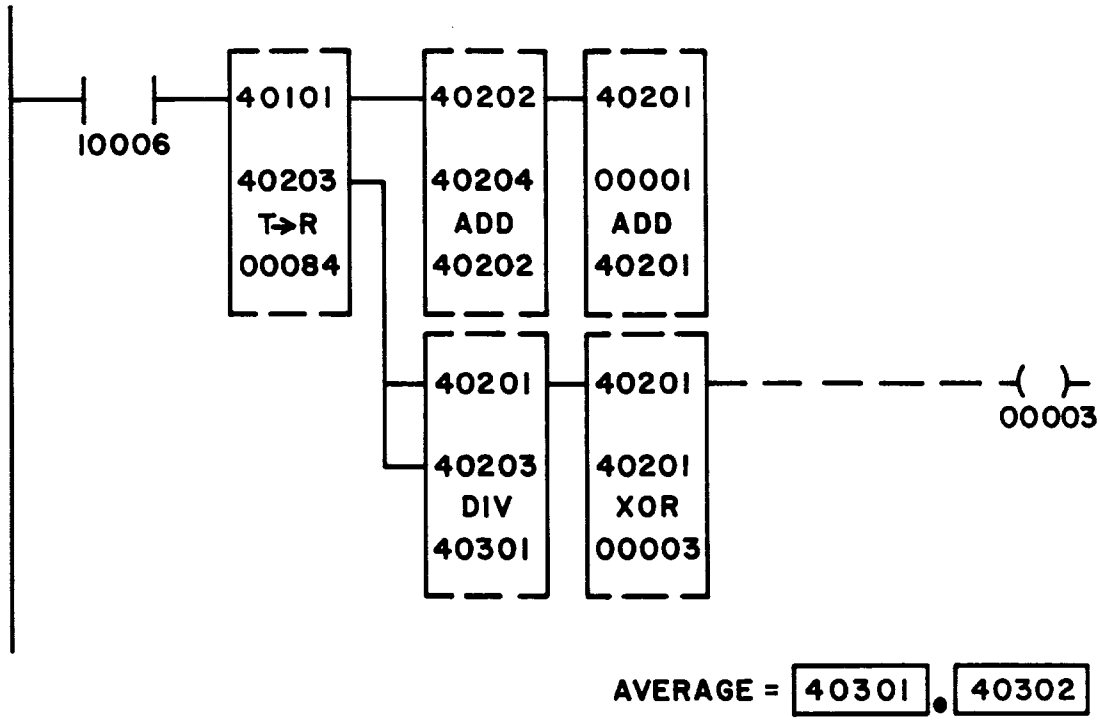


Figure 4-10. Table Averaging

When input 10006 receives power, the top input of the T→R function block receives power and the content of the first register in table 40101 to 40184, 40101, is copied into register 40204. Register 40203 holds the pointer value. Since the top output of the T→R function block passes power when the top input receives power, the first ADD block receives power. The value moved from the table is added to register 40202; register 40202 equals zero to start.

The above functions continue until the pointer value in register 40203 equals the table length, 84. When this happens, the middle output passes power and the DIV block receives power. At this point, the values in registers 40201 and 40202 are divided by the value in 40203 (84). The result is placed in 40301 and the decimal remainder in 40302. The remainder is a decimal because the middle input is receiving power.

The top output of the DIV block passes power and the XOR function is performed. With the XOR function, if both bits (one from each matrix) are ones or if both bits are zeros, the result is a zero. By using the XOR function to compare matrix 40201-40203 with itself, the matrix is cleared to zero.

The top output of the XOR function passes power to coil 00003. This indicates that the operation is complete and will start over again.

4.10.3 Running Table Averaging

The following paragraphs provide a detailed explanation of the logic illustrated in Figure 4-11 function.

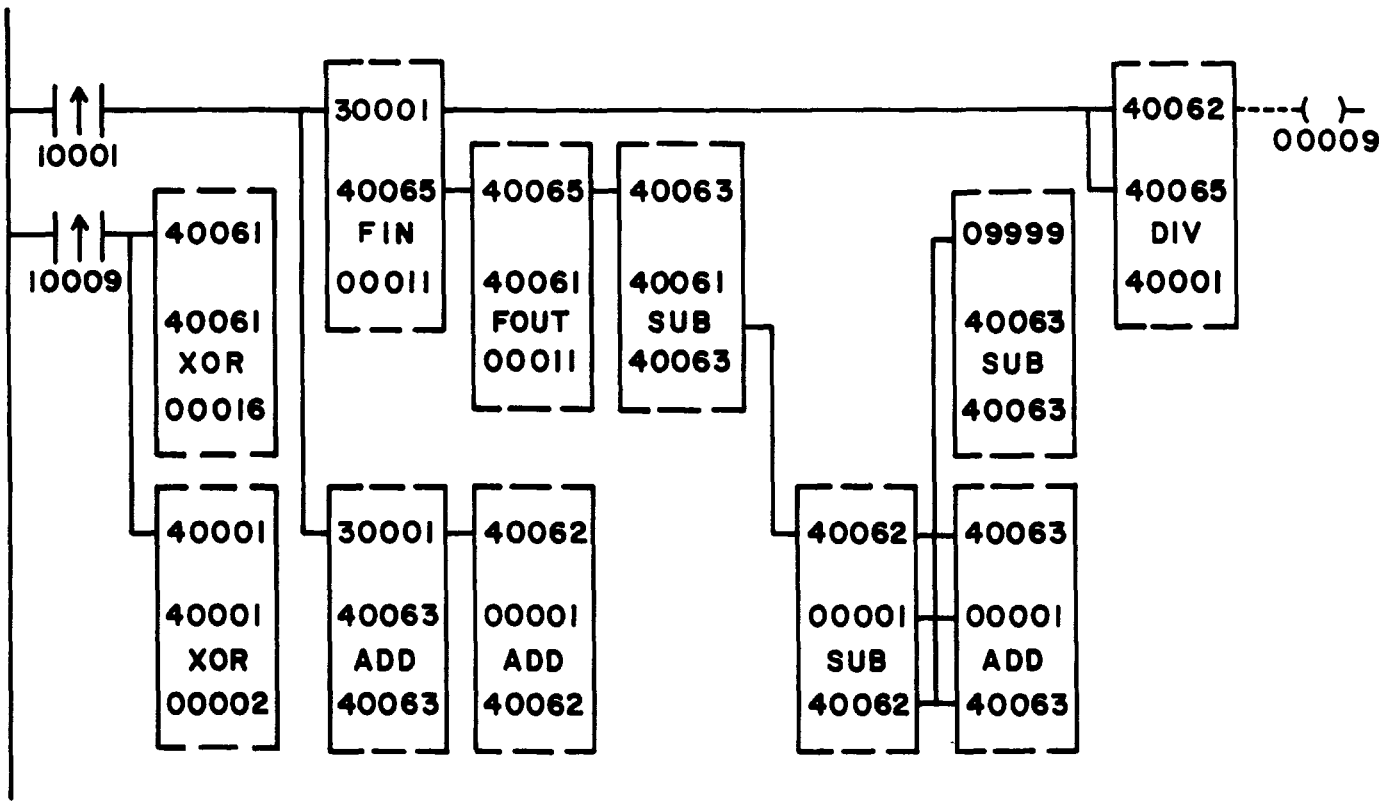


Figure 4-11. Running Table Averaging

When input 10001 receives power, the value in input register 30001 is added to the value in 40063. Register 40063 holds the total of all the registers currently in the queue. If the value overflows, greater than 9999, register 40062 is increased by one.

The top input of the FIN block receives power and the value in 30001 is moved into the first register in the queue (register 40066) as long as the queue is not full (see paragraph below). The top output of the FIN block passes power and the top and middle inputs of the DIV block receive power.

The value in 40062 and 40063 is divided by the total in 40065. Register 40065 holds the total number of registers in the queue. The average of the queue is placed in register 40001 and the decimal remainder is placed in register 40002. The top output of the DIV block passes power and energizes coil 00009.

When input 10009 transitions from OFF to ON, the top inputs of both XOR blocks receive power. These functions clear the data in registers 40061 to 40076, 40001, and 40002. The data is cleared because the XOR is being performed on two identical matrices (the same matrix twice); if both bits, one from each matrix, are ones or if both are zeros, the result is a zero with the XOR function.

If the queue is full when the FIN function block receives power, the value in register 30001 cannot be entered into the queue. To allow a register to be entered into the queue on the next scan, the oldest data in the queue must be removed. The middle output of the FIN block passes power since the queue is full.

The top input of the FOUT block receives power and the oldest data in the queue is moved into register 40061. The value in register 40061 is subtracted from the total in 40063; the value in 40061 is no longer in the queue.

If the value in 40063 is less than the value in 40061, the bottom output of the SUB block passes power and the next SUB block receives power. The second SUB block is used because the value in 40063 could be less than the value in 40061 if register 40063 had previously overflowed into register 40062. This is because the value in 40063, after the subtract, represents a negative number.

The value in 40062 is 0001 and the value in 40063 is 2170. However, when the subtract is performed, the 1 in register 40062 which represents 10,000, is not considerd. Therefore, the value placed in 40063 is presently incorrect. To obtain the correct answer:

1. The value 1 is subtracted from register 40062.

2. The value in 40063 must be subtracted from 9999 and a 1 must be added to 40063.

3. The value in 40063 is subtracted from 9999 because in the original SUB block the value was not subtracted from the whole number; it was only subtracted from the last 4 digits of it.

4. Add a 1 to 40063. The correct total of the queue is in register 40063.

The following is an example of the function with specific values. It starts with the first SUB block.

40063		40061		40063
2170	-	5000	=	2830

The bottom output passes power and the next subtract is performed.

40062		40062		40062
0001	-	0001	=	0000

All outputs pass power and the next subtract and an add are performed.

		40063		40063
9999	-	2830	=	7169

40063				40063
7169	+	0001	=	7170

4.10.4 Reporting Status Information

The following paragraphs provide a detailed explanation of the logic illustrated in Figure 4-12 to obtain status information and report it.

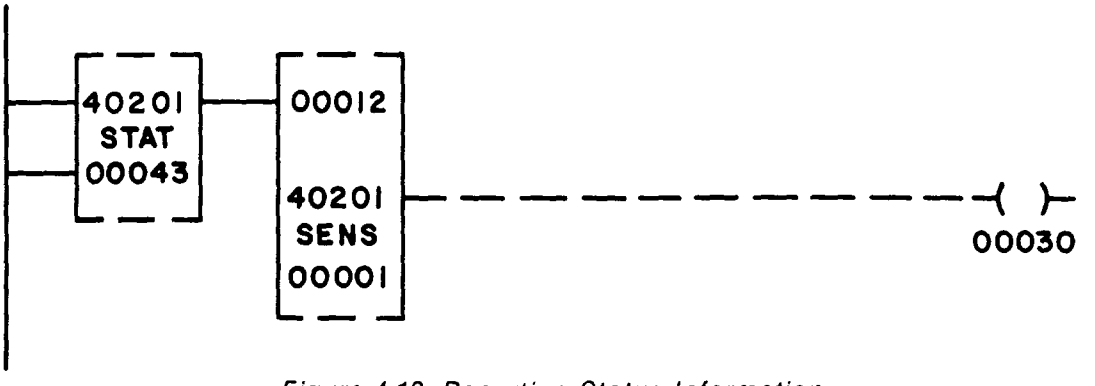


Figure 4-12. Reporting Status Information

The top input of the STAT block receives power every scan because it is attached to the power rail. Status information is recorded in registers 40201-40243. Register 40201 holds the 584L controller status. This is valuable information but it must be interpreted for the user.

Since each bit's (ON/OFF) state represents different information, a BIT SENSE function block can be used to report the status of a particular bit. By connecting the top input of the SENS block to the top output of the STAT block, the bit status is checked and reported every scan.

For example, bit 12, when ON, indicates that the battery is OK.

If bit 12 is a one bit, the middle output passes power and energizes coil 00401. Therefore, if coil 00401 is ON, the battery is OK.

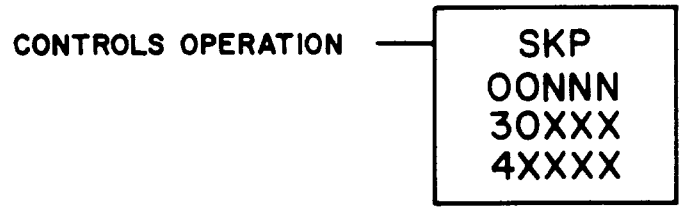
A bit sense function block can be used to determine the state of any bit in any matrix referenced to in the STAT block.

SECTION 5
ADDITIONAL FUNCTIONS

This section describes the additional functions that can be used in programming the 584L Programmable Controller.

5.1 SKIP (SKP)

FUNCTION
The SKIP function allows logic in a group of networks to be skipped, and thus not solved in order to reduce scan time. The SKIP function can be used to bypass seldom used program sequences or to create subroutines. See Section 5.5.1 on Subroutines.



FUNCTION BLOCK
This function block only occupies one node in a network. It contains the symbol SKP and either a constant or a register reference. It can be a constant, up to 999 in a Level 1 or 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference. If a 4XXXX reference is used it should be unique to this function block to avoid mishaps (i.e., using the same register to hold a counter value, etc.). The value specifies the number of networks to be skipped. To skip the remainder of the networks in the current segment, a value of zero is entered into the function block.

NOTES

If a 30XXX input register is used and the input is coming from a thumbwheel, the data read by the controller can be incorrect (i.e., if it was read while the number was still being entered). In order to use a thumbwheel and ensure a correct number, use a 4XXXX reference in the SKIP block and use a Subtract block to subtract zero from the input register to load it into the holding register.

The SKIP function cannot pass the boundary of a segment. Regardless of how many networks were programmed to be skipped, the function stops when it reaches the end of a segment. Also, skips within skips cause the controller to shut down; this should be avoided.

INPUT
The top and only input, when it receives power, causes the remainder of the current network and the specified number of networks to be skipped over by the controller's scan.

OUTPUT
There is no output with this function.

NOTE

Power flow shown on the P190 screen for skipped networks is invalid. To prevent misinterpretation of the data, the message "POWER FLOW INVALID" appears on the P190 screen.

EXAMPLE
The following paragraph provides a detailed explanation of the logic used in Figure 5-1.

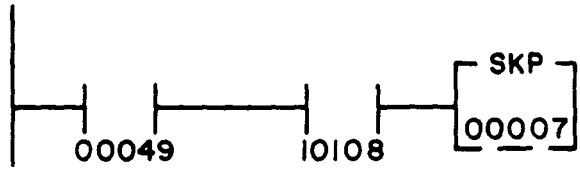


Figure 5-1. Skip

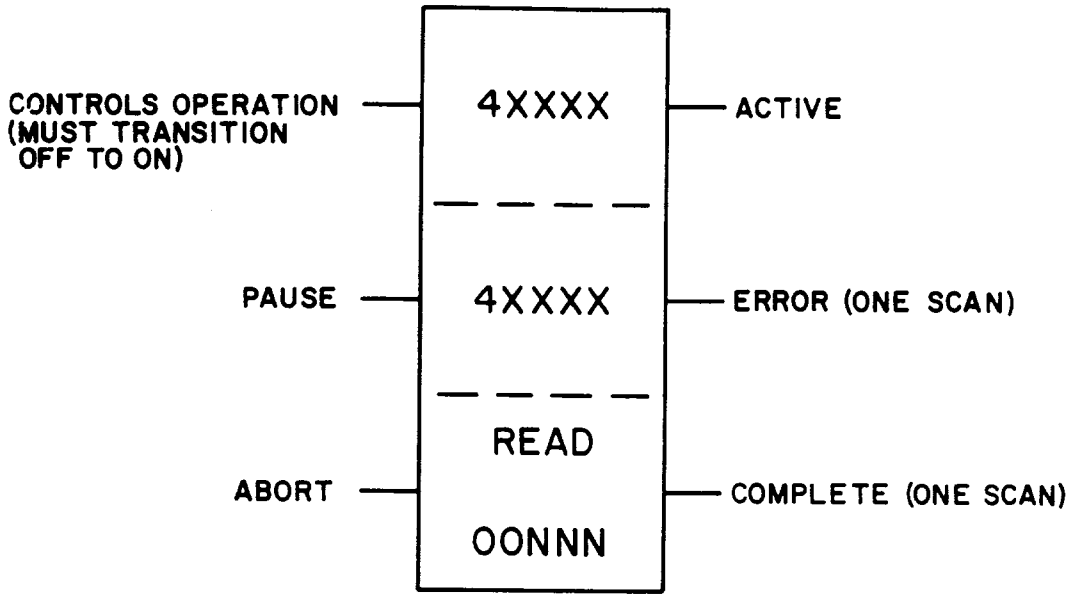
When the SKIP function block's input receives power, the remainder of the network containing the block (if any) is skipped and the next six networks are skipped. If network 17 contains the SKIP function, the remainder of 17 and all of 18-24 are skipped. If there are only five networks left in the segment, the operation stops after the fifth network.

NOTE

A skip node in any location of the network, except the bottom rung, can cause the 584L to stop.

5.2 READ (READ)

FUNCTION
The Read function allows the 584L PC to read data from an ASCII device through a P453 and an RS-232-C port on the 584L PC. The data is stored in a table of registers.



FUNCTION BLOCK

- The top node is the source node. It is a 4XXXX holding register reference. It refers to a table of seven registers. The table starts with the reference in the top node; the next six registers are implied. These registers must be unique to this function block.

  The following are register assignments:

  4XXXX = port number and error codes
  4XXXX + 1 = message number
  4XXXX + 2 = status word
  4XXXX + 3 = number of registers required
  4XXXX + 4 = number of registers received
  4XXXX + 5 = number of registers needed
  4XXXX + 6 = checksum

- The middle node is the destination node. It is a 4XXXX holding register reference. It is the reference to the first register in a table of registers. The information read by the controller is stored in these registers.

- The bottom node contains the symbol READ and the numerical value that specifies the destination table length. This constant can range from 1 to 255.

INPUTS

- The top input controls the operation. When this input transitions from OFF to ON the READ function is performed.

- The middle input, when receiving power, stops the READ function. When this input loses power, the READ function resumes where it left off.

- The bottom input, when it receives power, aborts/stops the operation. The top input must be cycled for the function to begin again; it does not resume where it left off.

OUTPUTS

- The top output, when passing power, indicates that the READ function is communicating with the specified port (FUNCTION ACTIVE).

- The middle output passes power for one scan when an error is detected. The error code is placed in either the 4 most significant bits of the source's first register or the next six bits (first four — 584L error; next six — P453 Remote I/O Interface error). For a list of the error codes, see the ASCII Programming Guide.

- The bottom output passes power for one scan when the operation is complete.

EXAMPLE

The following paragraph provides a detailed explanation of the logic used in Figure 5-2.

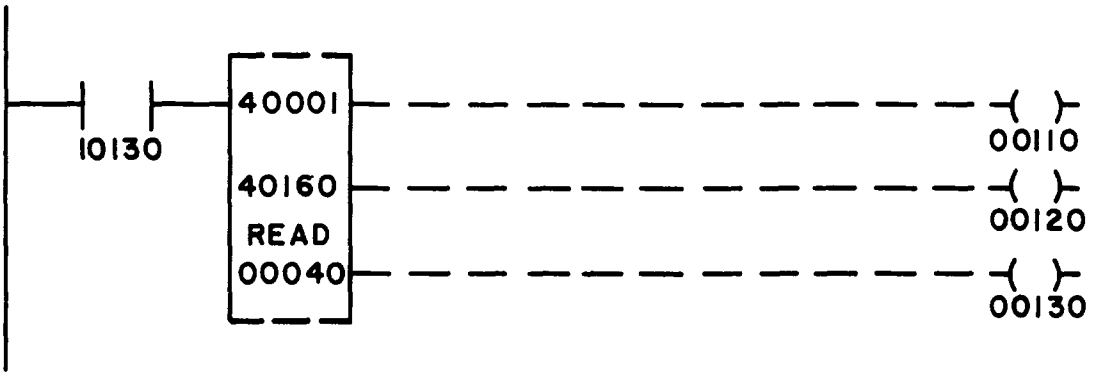


Figure 5-2. Read

When input 10130 transitions from OFF to ON, the top input receives power and starts to read information from an ASCII device. The information is stored in registers 40160 to 40199. The top output passes power and energizes coil 00110 while the read is taking place. If an error is found, the middle output passes power for one scan, energizing coil 00120 for one scan. When the read is complete, the bottom output passes power and energizes coil 00130.

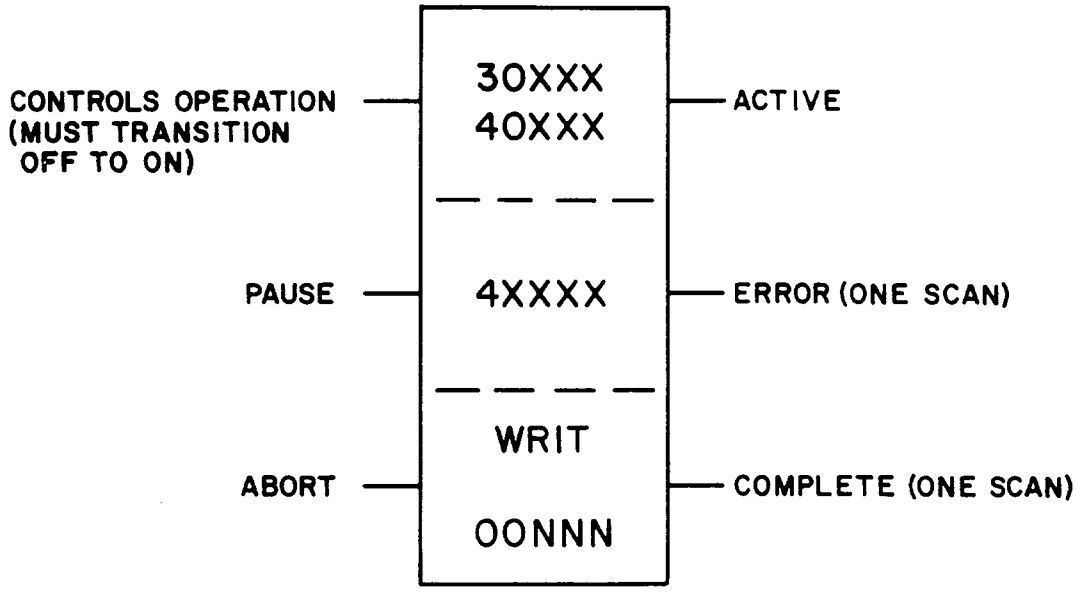
NOTE

An error code goes into the status word for a carriage return on the read block.

5.3 WRITE (WRIT)

FUNCTION

The Write function transmits data from the 584L PC through an RS-232-C port and a P453 to an ASCII device.



FUNCTION BLOCK

- The top node is the source node. It can be either a 30XXX input register or a 4XXXX holding register reference. It is the reference to the first register in a table of registers.

- The middle node is the destination node. It is a 4XXXX holding register reference. It refers to a table of seven registers. The table starts with the reference in the top node; the next six registers are implied. These registers must be unique to this function block.

  The following are register assignments:

  4XXXX = port number and error codes
  4XXXX + 1 = message number
  4XXXX + 2 = status word
  4XXXX + 3 = number of registers required
  4XXXX + 4 = number of registers received
  4XXXX + 5 = number of registers needed
  4XXXX + 6 = checksum

- The bottom node contains the symbol WRIT and the numerical value that specifies the source table length. This constant can range from 1 to 255.

INPUTS

- The top input controls the operation. When this input transitions from OFF to ON the WRITE function is performed.

- The middle input, when receiving power, stops the WRITE function. When this input loses power, the WRITE function resumes where it left off.

- The bottom input, when it receives power, aborts/stops the operation. The top input must be cycled for the function to begin again; it does not resume where it left off.

OUTPUTS

- The top output, when passing power indicates that the WRITE function is communicating with the specified port (function active).

- The middle output passes power for one scan when an error is detected. The error code is placed in either the four most significant bits of the source's first register or the next six bits (first four — 584L error; next six — P453 Remote I/O Interface error). For a list of the error codes, see the ASCII User's Guide.

- The bottom output passes power for one scan when the transfer of data from the 584L PC to the P453 is complete.

EXAMPLE

The following paragraph provides a detailed explanation of the logic used in Figure 5-3.

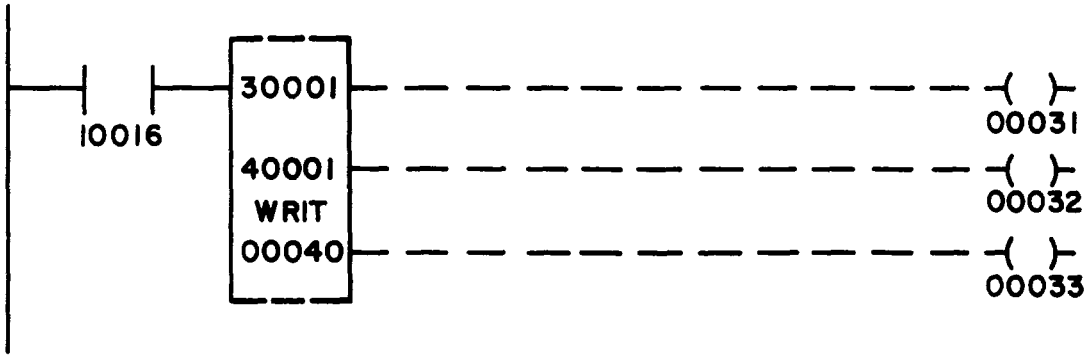


Figure 5-3. Write

When input 10016 transitions from OFF to ON, the top input receives power and starts to write information to an ASCII device. The information is taken from registers 30001 - 30040. The top output passes power and energizes coil 00031 while the write is taking place. If an error is found, the middle output passes power for one scan, energizing coil 00032. When the write is complete, the bottom output passes power and energizes coil 00033.

NOTES

1. A write to any "reserved" ASCII holding register will not set the error output.

2. A write to a busy ASCII port does not set the busy flag.

3. Write Block issues a done flag when the ASCII message is "sent" to the P453, instead of when it is actually "sent".

5.4 SWEEP FUNCTIONS

The SWEEP FUNCTIONS in the 584L PC allow the user's program to be scanned at a fixed interval. This interval can be a fixed time between scans or a fixed number of scans. These functions do not allow the controller to solve logic faster or to end a scan prematurely.

The CONSTANT SWEEP function allows the user to set target scan times from 10 to 200 milliseconds in multiples of 10. Target scan time is the time between the start of one scan and the start of the next scan, not the time between the end of one scan and the start of the next scan. This function is useful in applications in which data must be sampled at constant time intervals. The constant sweep value is recorded when a dump tape is made and can be transferred to another controller in this way.

The SINGLE SWEEP function allows the controller to execute a fixed number of scans, ranging from I to I5, and then to STOP solving logic but continue to service I/O. This function is useful to debug programs in which logic is complicated, or multiple scan logic is used.

When the SWEEP FUNCTIONS software label key is pressed, the following software labels appear on the screen:

INVOKE CONSTANT	CANCEL CONSTANT	INVOKE SINGLE	TRIGGER SINGLE	CANCEL SINGLE

5.4.1 Constant Sweep

The CONSTANT SWEEP function allows target scan times of 10 to 200 milliseconds. Target scan time is the time between the start of one scan and the start of the next scan, not the time between the end of one scan and the start of the next scan. The target scan time can be in multiples of 10 milliseconds and is not restricted to any two scans.

If a constant sweep is invoked with a time lapse smaller than the actual scan time, the time lapse is ignored and the scan is completed before the next scan starts. The constant sweep specifies the time between starts, not the time between the end of one scan and the start of another scan. (For example, if a sweep specifies 200 milliseconds as a desired scan time and the actual scan time is 90 milliseconds, the logic is solved in 90 milliseconds then the controller waits 110 milliseconds before starting the next scan.) To invoke a constant sweep:

1. Press the INVOKE CONSTANT software label key.

Four software labels appear on the screen: COMMENCE, TIME 10-200, HLDG REG 4XXXX, and PREVIOUS MENU.

2. Enter the desired scan time in milliseconds (10-200 in multiples of 10) into the AR.

3. Press the TIME 10-200 software label key to enter the value.

4. Select a holding register (4XXXX reference) and enter the reference number into the AR.

5. Press the HLDG REG 4XXXX software label key.

6. Press the COMMENCE software label key to start the CONSTANT SWEEP.

The constant scan time desired is placed in the previously specified holding register. The actual scan time is placed in the next consecutive holding register. To stop the constant sweep, press the PREVIOUS MENU software label key then the CANCEL CONSTANT software label key. A message appears on the P190 screen indicating that the sweep is cancelled.

NOTES

1. The constant sweep target scan time includes logic solving, I/O servicing, and system diagnostics. It does not include Modbus service time. If a target scan time of 40 milliseconds is set, and logic solving, I/O servicing, and systems diagnostics take only 30 milliseconds, the 584L PC waits 10 milliseconds. The 584L PC checks to see if either or both Modbus ports require service. If the ports require service, an additional 3 milliseconds is automatically added by the controller, to the 40 millisecond target scan time.

2. The constant sweep function should not be used if redundancy (J2II/584L) is active.

5.4.2 Single Sweep

The SINGLE SWEEP function allows the controller to complete between 1 and 15 scans and then to stop scanning. This is very useful for diagnostic work; it allows solved logic, moved data, or performed calculations to be examined for errors.

To invoke a single sweep:

1. Press the INVOKE SINGLE software label key.

Three software labels appear on the screen: COMMENCE, TIME 10-200, and PREVIOUS MENU.

2. Enter the scan time in milliseconds (10-200 in multiples of 10) into the AR.

3. Press the TIME 10-200 software label key to enter the value.

4. Press the COMMENCE software label key.

5. Press the PREVIOUS MENU software label key.

6. Press the TRIGGER SINGLE software label key.

Three software labels appear on the screen: COMMENCE, #SWEEPS 1-15, and PREVIOUS MENU.

7. Enter the number of scans desired (1-15) into the AR.

8. Press the #SWEEPS 1-15 software label key.

9. Press the COMMENCE software label key.

The controller will scan the specified number of times then stop.

To continue single sweeps:

1. Press the TRIGGER SINGLE software label key.
2. Enter the number of sweeps into the AR.
3. Press the #SWEEPS 1-15 software label key.
4. Press the COMMENCE software label key.

To return to normal scanning, press the PREVIOUS MENU software label key then the CANCEL SINGLE software label key.

CAUTION

The SINGLE SWEEP function should not be used to debug controls on machine tools, processes, or material handling systems when they are active. Once the specified number of scans has been solved, all outputs are frozen in their last state. Since no logic solving is taking place, all input information is ignored. This can result in unsafe, hazardous, and destructive operation of the machine or process connected to the 584L PC.

5.5 SUBROUTINE EXAMPLE

The SKIP function can be used to create subroutines within a program. This is useful when a routine needs to be performed every third, or fourth scan, etc. It is also useful to perform a certain routine several times within a function. The SKIP functions allow the routine's logic to be written once and performed as many times as necessary.

The following example uses the SKIP function to allow a different routine to take place each scan for a limited sequence.

On three consecutive scans, three different functions can be performed, one function each scan.

Network 3 contains a skip block with a holding register, not a constant,

Network 5 contains a function, the first of three,

Network 6 contains a function, the second of three,

and Network 7 contains a function, the third of three.

The first network (i.e., 10) in the next segment contains an ADD block which adds a one to the holding register (i.e., 40001) used in network 3's skip block.

On the first scan, at network 3 the controller skips to the beginning of the next segment because 40001 contains zero. In network 10, a one is placed in 40001. On the next scan at network 3, the controller skips to network 5. The first function is

performed. At the end of network 5 is a skip block containing zero. The controller skips to the beginning of the next segment, and adds a one to 40001, increasing it to two.

On the next scan at network 3, the controller skips to network 6. The second function is performed. At the end of network 6 is a skip block containing zero. The controller skips to the beginning of the next segment and adds a one to register 40001, increasing it to three.

On the next scan at network 3, the controller skips to network 7. The third function is performed. At the end of network 7 is an ADD block which places a zero in register 40001. The controller solves the rest of the logic in the network. At network 10 a one is placed in register 40001.

On the next scan, the process starts all over again. The only difference is that the controller Skips to network 3 on this scan. It does not have to Skip to the beginning of the next segment first because there is already a one in register 40001.

APPENDIX A
CONFIGURATION

Before the 584L PC can be programmed, it must be configured for the user's system requirements. Configuring the 584L PC sets maximum values for inputs, outputs, registers, I/O Channels, RS-232-C ports, etc. This organizes the user's programming elements for greater efficiency.

To configure the 584L PC, a P190 Programmer and a T584-004 Configurator Tape are required. Insert the tape into the P190 tape drive and press the red INIT and INIT LOCK keys on the P190 panel simultaneously to load the tape. Once the tape is loaded, the user can start configuring the controller. Figure A-1 is a flow chart of all the software labels available with the Configurator Tape.

When the P190 has finished loading the configurator tape, the user is presented with the option of selecting "584 Config" or "584L Config". When using a 584L, press the 584L Config. The software labels 584L CONFIG and ATTACH UNIT # are displayed on the P190 Screen. Before either one is pressed, a unit number within the range of 1 to 247, must be entered into the assembly register (AR). The unit number refers to the location of the 584L in the data line communicating with the P190. Press the 584L CONFIG software label key to start entering information. To attach the P190 to the 584L PC, press the ATTACH UNIT # software label key; the RELEASE 584L software label appears on the screen. This software label key is pressed to detach the 584L PC.

When the 584L CONFIG software label key is pressed, the following software labels are displayed:

NO SKIPS ↓ SKIPS ↓	SET SIZE	PORT 1	PORT 2	ASCII	SPECIALS	WRITE CONFIG	NEXT MENU

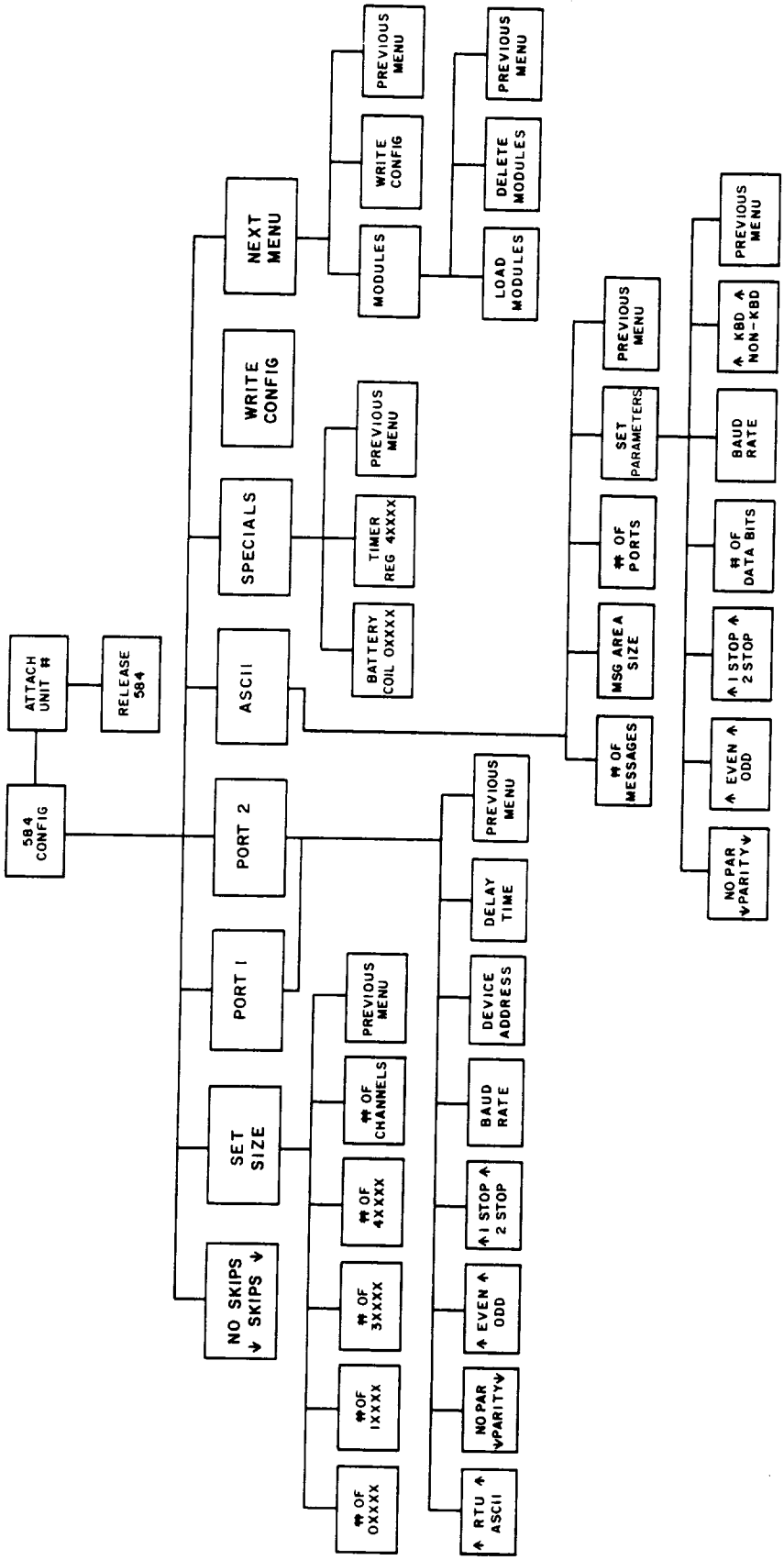


Figure A-1. Flow Chart of Configuration Software Labels

A.1 NO SKIPS/SKIPS

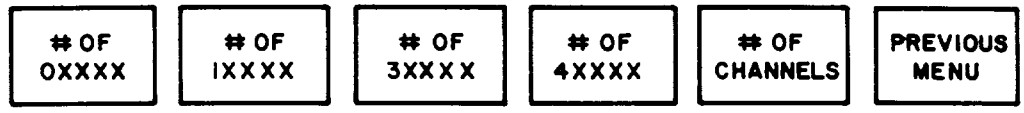
Toggle this software label key to select SKIPS or NO SKIPS. The selection is indicated by the UP or DOWN arrows in the software label.

The choice of SKIPS or NO SKIPS is dependent on whether or not the SKIP function will be used in the program.

On the initial configuration the controller defaults to SKIPS.

A.2 SET SIZE

When this software label key is pressed, the following software labels are displayed:



These software label keys are used to insert the maximum number of references and I/O channels available for the user's program.

A.2.1 # of 0XXXX

Enter into the AR the total number of logic coils that will be available for the program, a value divisible by 16, and press this software label key.

On the initial configuration, the controller defaults to the value 16.

A.2.2 # of 1XXXX

Enter into the AR the total number of discrete inputs that will be available for the program, a value divisible by 16, and press this software label key.

On the initial configuration, the controller defaults to the value 16.

A.2.3 # of 30XXX

Enter into the AR the total number of input registers that will be available for the program, a maximum of three digits, and press this software label key.

On the initial configuration of the 584L, the controller defaults to the value 1.

A.2.4 # of 4XXXX

Enter into the AR the total number of output/holding registers that will be available for the program and press this software label key.

On the initial configuration, the controller defaults to the value 1.

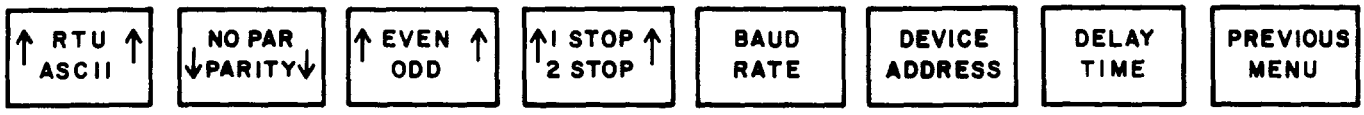
A.2.5 # of CHANNELS

Enter into the AR the number of I/O channels that will be available for the program, a maximum value of 16, and press this software label key. This must be an even number between 2 and 32 which is determined by taking the number of coils divided by 128, or by taking the number of discrete inputs divided by 128 plus the number of input registers divided by 8. The larger of the two values is the number of I/O channels.

On the initial configuration the controller defaults to 2 I/O channels. Remember, 584L offers a choice of all remote (32) or 4 local and 28 remote I/O channels.

A.3 PORT 1

Press this software label key to set the parameters for PORT 1 on the 584L PC. When this key is pressed, the following software labels appear on the screen:



This selection is made by pressing the "specials" soft label key on the PI90.

A.3.1 RTU/ASCII

Toggle this software label key to select the Modbus communication mode — RTU or ASCII. The selection is indicated by UP or DOWN arrows in the software label.

RTU (Remote Terminal Unit) mode is generally used to communicate between the 584L PC and the P190 programmer. It can also be used for Modbus communications.

ASCII (American Standard Code for Information Interchange) mode is used for Modbus communications, it is more complex than RTU but it is easier to implement.

On the initial configuration, the controller defaults to RTU.

A.3.2 No Par/Parity

Toggle this software label key to select PARITY or NO PARITY. The selection is indicated by UP or DOWN arrows in the software label.

On the initial configuration, the controller defaults to even PARITY.

A.3.3 Even/Odd

This software label key is only used if PARITY is selected. Toggle the EVEN/ODD software label key to select EVEN parity or ODD parity. The selection is indicated by UP or DOWN arrows in the software label.

When communicating between the P190 and the 584L PC, EVEN parity is selected. On the initial configuration, the controller defaults to EVEN parity.

A.3.4 1 Stop/2 Stop

Toggle this software label key to select 1 STOP bit or 2 STOP bits. The selection is indicated by UP or DOWN arrows in the software label.

One stop bit is selected if the 584L PC is communicating with other Modicon equipment. On the initial configuration, the controller defaults to 1 STOP BIT.

A.3.5 Baud Rate

Enter one of the following baud rates into the AR:

50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, or 19200.

When communicating between the P190 and the 584L PC, the baud rate is 9600. Press the BAUD RATE software label key to enter the value.

On the initial configuration, the controller defaults to a baud rate of 9600.

A.3.6 Device Address

Enter into the AR the address of the 584L PC which is currently communicating with the P190, a maximum value of 247, and press this software label key.

The P190 is capable of communicating, through a modem, with 247 devices in a line. The address of the device is determined by its location in the line. If only one device, a 584L PC, is communicating with the P190, enter a device address of 1.

On the initial configuration, the controller defaults to a device address of 001.

A.3.7 Delay Time

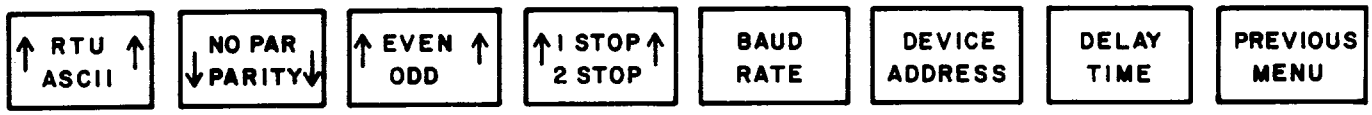
Enter the desired delay time into the AR, a maximum value of 20, and press this software label key. This value represents one tenth of the desired delay time (e.g., a value of 20 is 200 milliseconds).

The delay time is the time lapse between sending a message and receiving an answer. If there is no time lapse between sending and receiving, it is possible that the answer to a message will be lost because a device is not ready to receive.

The delay time for communications between Modicon equipment is 10 milliseconds; enter a 1 for delay time. On initial configuration, the controller defaults to a delay time of 01.

A.4 PORT 2

Press this software label key to set the parameters for PORT 2 on the 584L PC. When this key is pressed, it brings up the same set of software labels as the PORT 1 software label key:



See sections A.3.1 through A.3.7 for descriptions of these software lables. The function of each software label is the same for PORT 1 or PORT 2.

A.5 ASCII

Press this software label key to set the limits and parameters for ASCII functions. When this key is pressed, the following software labels appear on the screen:

NOTE

There is no error reported for an ASCII write to a disconnected ASCII port.

A.5.1 # of Messages

Enter into the AR the number of ASCII messages, format statements, which will be stored in memory. The maximum is 9999 messages.

Each 584L PC message has a code number ranging from 1 to 9999. The value entered for # OF MESSAGES reflects the total number of messages and not this code number.

On the initial configuration, the controller defaults to zero.

A.5.2 MSG Area Size

Enter a value into the AR (maximum 9999) and press the MSG AREA SIZE software label key to enter the total words of memory (TOTAL MESSAGE WORDS) to be set aside for the storage of ASCII messages.

One word of memory equals two ASCII characters. A message area of at least ten words is needed to adequately cover the average message length of seventeen characters.

On the initial configuration, the controller defaults to zero.

A.5.3 # of Ports

Enter into the AR the number of RS-232-C ports included in the system for ASCII communications, a maximum value of 32, and press this software label key.

The system consists of a 584L PC communicating, through a P453 Remote I/O Interface, with up to thirty-two devices.

On the initial configuration, the controller defaults to zero.

A.5.4 Set Parameters

Press this software label key to set the ASCII port parameters. When this software label key is pressed, the following set of software labels appear on the screen:

PORT	PARITY	STOPS	#BIT	BAUD RATE	KBD	PORT	PARITY	STOPS	#BIT	BAUD RATE	KBD
01	EVEN	1	8	01200	Y	02	ODD	1	8	01200	Y
03	NONE	2	7	09600	N						

Figure A-2. ASCII Parameter Table

Figure A-2 is an example parameter table for three ASCII ports. To change the parameters for any of the ports, position the cursor to the left of the port number and press the appropriate software label keys. Move the cursor with the arrow keys on the P190 panel. The software label keys are explained in the next six sections (A.5.4.1 through A.5.4.6).

A.5.4.1 No Par/Parity

Toggle this software label key to select PARITY or NO PARITY. The selection is indicated by UP or DOWN arrows in the software label.

On the initial configuration, the controller defaults to even PARITY.

A.5.4.2 Even/Odd

Toggle this software label key to select EVEN parity or ODD parity. The selection is indicated by UP or DOWN arrows in the software label.

When communicating between the P190 and the 584L PC, EVEN parity is selected. On the initial configuration, the controller defaults to EVEN parity.

A.5.4.3 1 Stop/2 Stop

Toggle this software label key to select 1 STOP bit or 2 STOP bits. The selection is indicated by UP or DOWN arrows in the software label.

One stop bit is selected if the 584L PC is communicating with other Modicon equipment. On the initial configuration, the controller defaults to 1 STOP bit.

A.5.4.4 # of Data Bits

Enter into the AR the number of data bits to be passed through a specific port (a value of 5, 6, 7, or 8), and press the # OF DATA BITS software label key.

ASCII communications require seven data bits, and RTU communications require eight data bits. Five or six data bits are rarely used.

On the initial configuration, the controller defaults to 8 data bits.

A.5.4.5 Baud Rate

Enter one of the following baud rates into the AR:

50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, or 19200.

Press the BAUD RATE software label key to enter this value. When communicating between the P190 and the 584L PC, the baud rate is 9600.

On the initial configuration, there is no 584L that defaults to I200; the 584L PC defaults to a baud rate of 9600.

A.5.4.6 KBD/NON-KBD

Toggle this software label key to select keyboard (KBD) or no keyboard (NON-KBD). The selection is indicated by UP or DOWN arrows in the software label.

The choice of KBD or NON-KBD is dependent on the type of device connected to the ASCII port. If the device does not have a keyboard (i.e., a bar code reader, or a microprocessor), NON-KBD is selected. If the device is a CRT or printer terminal, KBD is usually selected although NON-KBD is allowed.

A selection of KBD allows backspace and rubout keys to be used if a mistake is made in entering data. The field terminator for a data entry is an escape (ESC) key. The end of message indicator is a carriage return.

On the initial configuration, the controller defaults to KBD.

A.6 SPECIALS

Press this software label key to reach software labels BATTERY COIL 0XXXX and TIMER REG 4XXXX. Battery coil and timer reg do not have to be configured for the 584L PC to run properly; they are available if the user needs them. The user can now select the desired remote I/O configuration by selecting the desired number of channels.

A.6.1 Battery Coil 0XXXX

Enter a 0XXXX reference number into the AR and press the BATTERY COIL 0XXXX software label key to enter the coil number. When this coil is used in the user's program, it reflects the status of the battery back-up system.

On the initial configuration, the controller defaults to zero; no battery coil is available.

NOTES

1. Changing certain configuration options, ASCII parameters, skip, 584 baud rates, does not clear 584 memory, but does affect coil and register tables (history disable, state).

2. Once a battery coil has been configured during the configuration process, it cannot be released, even if another coil is configured as the battery coil. Immediately after configuring the battery coil (before the 584 has started) the coil can be programmed in user logic.

A.6.2 Timer Reg 4XXXX

Enter a 4XXXX reference number into the AR. Press the TIMER REG 4XXXX software label key to enter the holding register number. This holding register is set aside to hold the number of 10 millisecond clock cycles.

On the initial configuration, the controller defaults to zero; no timer register is available.

A.7 WRITE CONFIG

When all the configuration information has been entered into the configuration table, press this software label key to insert the table into the 584L PC.

If changes are made to the table and the WRITE CONFIG software label key is not pressed, the changes to the table are not acknowledged by the 584L PC. The 584L PC must be STOPPED in order to implement the WRITE CONFIG function.

A.8 NEXT MENU

Press this software label key to enter module information. The software labels MODULES, WRITE CONFIG, and PREVIOUS MENU appear on the screen.

A.8.1 Modules

When this software label key is pressed, the software labels LOAD MODULES, DELETE MODULES, and PREVIOUS MENU are displayed on the screen.

A.8.1.1 Load Modules

Press this software label key in order to load a 584L Modules Program Tape. When this software label key is pressed, the software labels PROCEED and CANCEL appear on the screen along with the message "Insert 584L Modules Program Tape". The 584L can have multiple loadable modules. The 584L Modules Program Tapes are not part of the Configurator Tape package.

A.8.1.2 Delete Modules

Press this software label key in order to delete part or all of a 584L Modules Program Tape from the 584L memory. The software labels DELETE ALL, DELETE PROGRAM, and PREVIOUS MENU appear on the screen. The heading "584L Controller Directory" appears with a list of the modules in the 584L PC.

To delete all the modules in the 584L, press DELETE ALL. To delete individual modules, enter the module/program name beside "Enter Program Name:" on the screen and press DELETE PROGRAM.

A.8.2 Write Config

This function is explained in Section A.7.

A.9 MOVE SEGMENT AND TRAFFIC COP

When the Configurator Tape (T584-004) is loaded into the P190, the software labels 584L CONFIG and ATTACH UNIT # appear on the screen. To reach the MOVE SEGMENT and TRAFFIC COP functions, do the following:

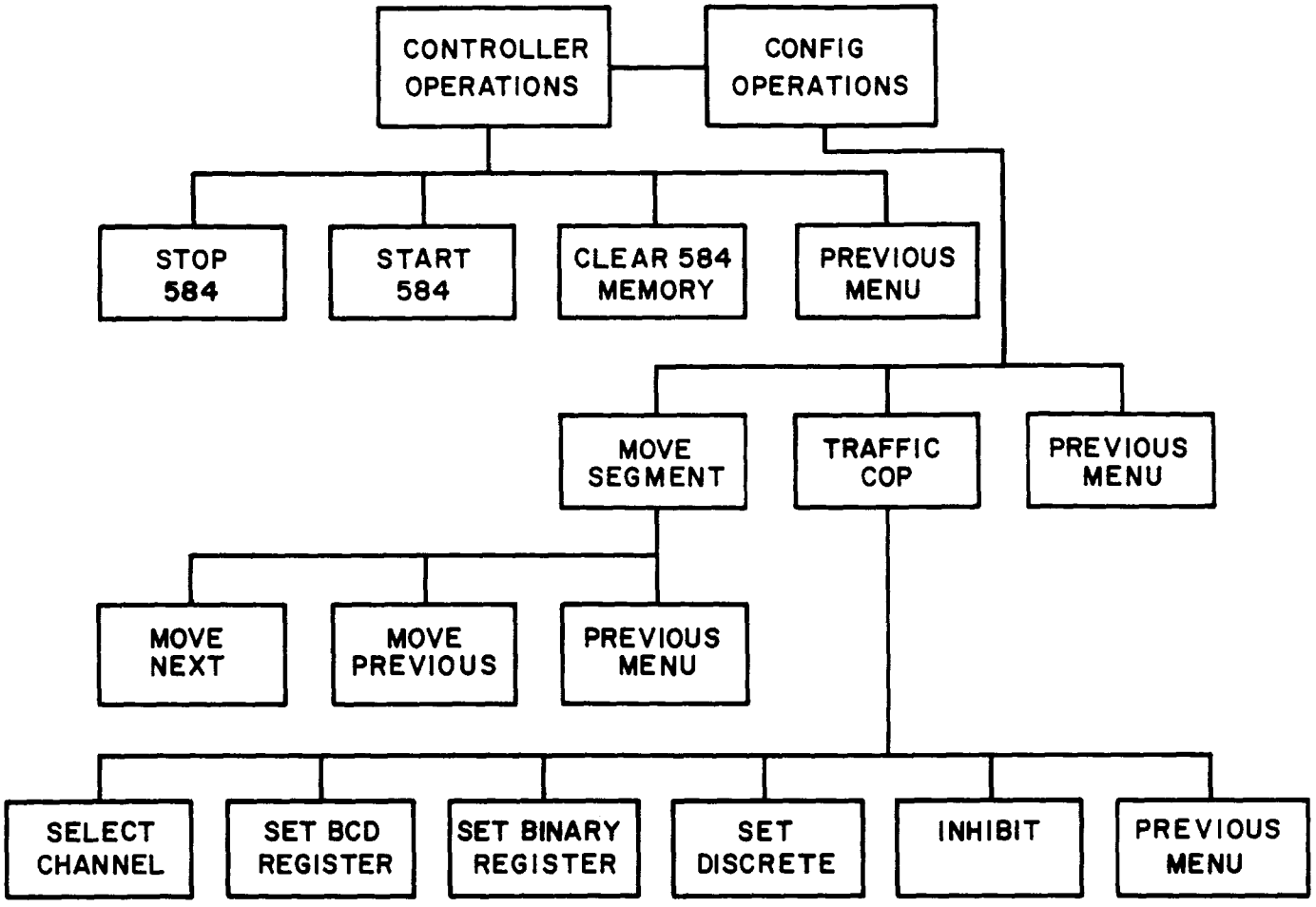


Figure A-3. Flow Chart of Traffic Cop Software Labels

1. Enter a unit number into AR (e.g., 00001).

2. Press the ATTACH UNIT # software label key.

   The RELEASE 584L software label appears on the screen.

3. Press the RESET/EXIT key on the P190 panel.
   The software labels CONTROLLER OPERATIONS and CONFIG OPERATIONS appear on the screen.

4. Stop the controller if it is running. To do this:

   1) Press the CONTROLLER OPERATIONS software label key.
   2) Press the STOP 584L software label key.
   3) Press the PROCEED software label key.

5. Press the CONFIG OPERATIONS software label key.

The software labels MOVE SEGMENT, TRAFFIC COP, and PREVIOUS MENU appear on the screen.

A.9.1 Move Segment

The MOVE SEGMENT functions allow whole or partial segments to be moved into adjacent segments.

When this software label key is pressed, the following software labels are displayed on the screen:

MOVE NEXT	MOVE PREVIOUS	PREVIOUS MENU

A.9.1.1 Move Next

Enter a network number into the AR (e.g., 28) and press this software label key. All the networks, starting at the number entered (e.g., 28) and continuing to the end of the segment (e.g., segment 2), are moved into the next segment (e.g., segment 3).

NOTE

When moving a portion of a segment to the next segment, remember that only the networks from a specific network to the end of the segment can be moved. A group of networks in the middle of a segment or from the beginning to middle of a segment cannot be moved using this software label key.

A.9.1.2 Move Previous

Enter a network number into the AR (e.g., 35) and press this software label key. All the networks, from the beginning of the segment (e.g., segment 2) to the network number entered minus one (e.g., 34), are moved to the previous segment (e.g., segment 1).

NOTE

When moving a portion of a segment to the previous segment, remember that only the networks from the beginning of a segment to a specific network in the segment can be moved. A group of networks in the middle of a segment or from the middle to the end of a segment cannot be moved using this software label key. Also, the last network in a segment cannot be moved.

A.9.2 Traffic Cop

The Traffic Cop is used to direct the flow of data between the various I/O modules and the logic program. It is the tie between the references used in the logic program and the I/O module connection points.

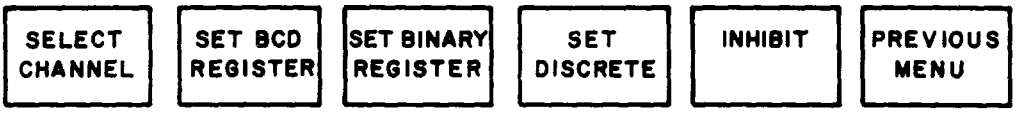
The Traffic Cop table is constructed by specifying the references to be associated with I/O modules mounted in particular channels and slots, and specifying the type of data — Discrete, BCD, or Binary. Figure A-4 is a typical Traffic Cop for Channel 1.

CHANNEL: 01

SLOT	INPUT REF#	TYPE	OUTPUT REF#	TYPE
1	30001	BCD	40001	BIN
2	10001	DIS	40002	BCD
3	30002	BIN	00001	DIS
4	30003	BIN	00033	DIS
5	10017	DIS	00801	DIS
6	10033	DIS	40105	BIN
7	10065	DIS	40106	BIN
8	10801	DIS	40108	BCD

Figure A-4. Typical Traffic Cop

When the TRAFFIC COP software label key is pressed, the following software labels are displayed on the screen:



Position the cursor to the left of the REF# column in the row desired to enter information. The cursor position is changed using the arrow keys on the P190 panel.

The SLOT numbers refer to the address on the I/O Rack. There are REF# and TYPE columns for both INPUT and OUTPUT. INPUT includes 1XXXX discrete input references and 30XXX input register references. OUTPUT includes 0XXXX logic coil references and 4XXXX holding/output register references. The following is a summary of the options for each reference:

Reference	Options
0XXXX	OUTPUT; DISCRETE
1XXXX	INPUT; DISCRETE
30XXX	INPUT; BCD REGISTER or BINARY REGISTER
4XXXX	OUTPUT; BCD REGISTER or BINARY REGISTER

A.9.2.1 Select Channel

Enter a channel number into the AR, a maximum value of 32, and press this software label key. The TRAFFIC COP table is displayed for the specified channel.

On the initial configuration of the Traffic Cop, the controller defaults to the Traffic Cop Table for Channel 1.

A.9.2.2 Set BCD Register

Enter a 30XXX or 4XXXX reference into the AR and press this software label key. This register is entered into the Traffic Cop as a Binary Coded Decimal (BCD) register and the cursor moves down one position.

NOTE

I/O modules wired to numerical devices (i.e., thumbwheel switches, digital LED displays, etc.) should be defined as BCD registers.

A.9.2.3 Set Binary Register

Enter a 30XXX or 4XXXX reference into the AR and press this software label key. This register is entered into the Traffic Cop as a Binary register and the cursor moves down one position.

NOTE

I/O Modules wired to analog transducers, and module slots which use the 584L PC's matrix logic capabilities, should be defined as Binary registers.

A.9.2.4 Set Discrete

Enter a 0XXXX or IXXXX reference into the AR, a value divisible by 16 with a remainder of 1 (i.e., 00017, 00033, 10049, 10065, etc.), and press this software label key. The reference represents 16 discretes (i.e., 00017-00032, 10065-10080). The cursor moves down one position after this software label key is pressed.

A.9.2.5 Inhibit

Position the cursor to the left of the reference to be inhibited and press this software label key. This clears the reference number and its type. The word INHIBIT is placed in the TYPE column and the cursor moves down one position.

On the initial configuration of the Traffic Cop, the controller defaults to INHIBIT for all the slots.

APPENDIX B
SOLVE TIME SPECIFICATIONS

Function	Level 1 or 2 584L Minimum Solve Time	Level 1 or 2 584L Maximum Solve Time
UCTR	13.09 microseconds	13.97 microseconds
DCTR	13.09 microseconds	13.97 microseconds
TIMER	8.05 microseconds	9.56 microseconds
ADD	9.59 microseconds	10.85 microseconds
SUB	9.59 microseconds	11.10 microseconds
MUL	9.59 microseconds	63.58 microseconds
DIV	9.59 microseconds	116.50 microseconds
R→T	12.50 microseconds	15.75 microseconds
T→R	13.00 microseconds	16.50 microseconds
T→T	13.25 microseconds	17.00 microseconds
BLKM	9.75 microseconds	137.50 microseconds
FIN	14.25 microseconds	19.25 microseconds
FOUT	16.00 microseconds	17.25 microseconds
SRCH	13.25 microseconds	163.50 microseconds
STAT	10.75 microseconds	241.50 microseconds
AND	9.75 microseconds	210.00 microseconds
OR	10.25 microseconds	210.50 microseconds
XOR	12.00 microseconds	212.25 microseconds
COMP	11.75 microseconds	212.00 microseconds
CMPR	10.75 microseconds	284.00 microseconds

SOLVE TIME SPECIFICATIONS

Function	Level 1 or 2 584L Minimum Solve Time	Level 1 or 2 584L Maximum Solve Time
MBIT	10.50 microseconds	31.50 microseconds
SENS	13.00 microseconds	25.00 microseconds
BROT	11.75 microseconds	216.25 microseconds (right) 215.75 microseconds (left)
READ	3 milliseconds	6 milliseconds
WRIT	3 milliseconds	6 milliseconds

APPENDIX C
INFORMATION AND ERROR MESSAGES

Information messages and error messages appear on the error line of the P190 screen. The message can either be supplying instructions or information to the user, or alerting the user to errors in programming or operation. The messages in this appendix are listed alphabetically along with a description of the message. When a certain action should be taken, the action is listed in the far right, SUGGESTED ACTION, column.

Most messages can be cleared by pressing the CLEAR ERROR key. The AR is cleared by pressing the CLEAR AR key.

Message	Description	Suggested Action
Address limit	The number of I/O addresses assigned exceed the available space.	Reassess I/O allocations. Reduce the number of I/O addresses.
AR not decimal	The information being entered must be decimal, as in the case of reference numbers.	Clear the AR and enter decimal data.
Bad length received	Communications error in message received from 584L PC.	CLEAR ERROR, RESET, and reattempt operation.
Cannot login-unit has programmer attached	Only one P190 programmer at a time may be attached to a 584L PC as a programmer (i.e., with P190 keylock unlocked).	Attach to 584L PC as a monitor (i.e., with keylock locked) or wait until other programmer logs out or detaches.
Coil not disabled	The coil can not be forced because it has not been disabled.	Disable the coil. It can then be forced.
Coil not in a network	The requested coil has not yet been used.	
Coil used	The requested coil has already been programmed and may not be used again.	Select a new coil number.
Coils not allowed here	A coil may not be placed in this position, such as between two contacts.	Move cursor to appropriate area and coil.

INFORMATION AND ERROR MESSAGES

Message	Description	Suggested Action
Compress not allowed	There are no nodes beneath the cursor to compress horizontally or vertically.	
Controller in dim awareness, attach not allowed.	The controller has not been configured and cannot fully communicate with the P190 programmer.	Configure controller using 584L Configuration Tape and retry. Make sure the correct controller has been addressed.
Controller running	The attempted action (i.e., trying to clear memory) cannot be performed because the controller is running.	Stop controller, then perform action.
CRC failure	Indicates a communications error picked up by the error checking. CRC = cyclical redundancy check.	CLEAR ERROR, RESET and reattempt operation.
End of logic memory	There are no more networks started or programmed into logic memory.	If another network is desired, press START NETWORK.
Expand not allowed	Network expansion, either vertical or horizontal is not allowed due to space.	
Fatal I/O error must initiate reset sequence	Communications error message has been cleared from screen. RESET must be pressed to be able to reinstate communications.	RESET, and reattempt operation.
Function not allowed	The requested function may not be performed at this time.	
Illegal baud rate	Occurs when configuring a 584L PC. The baud rate selected (i.e., 5000) is not allowed.	Select a legal baud rate.

Message	Description	Suggested Action
Illegal channel number	Occurs during traffic cop operations. The number entered is not legal number for a channel (1-4 local, 5-31 odd numbers only for remote).	Clear the AR and enter legal channel number.
Illegal configuration	Displayed when attempt is made to program a 584L PC when the configuration has not been properly set.	Check configuration and assure that it is complete and accurate.
Illegal device address	The device address given is less than 1 or greater than 255 decimal. This message appears during configuration.	Select legal device address (from 1 to 255 inclusive).
Illegal replacement	This message is displayed if the user attempts to replace one type of node with another type which is not legal at that position. An example would be attempting to replace a contact with a coil.	Select legal replacement.
Illegal 584L memory configuration	The 584L configuration, at time of start-up is not valid.	Reload tape. If error reoccurs obtain new dump tape and reload.
Invalid address	Communications error. Message not received correctly due to invalid field.	CLEAR ERROR, and reattempt operation.
Invalid command		
Invalid date		
Invalid network number	The network number given does not correspond to any network in the controller.	

INFORMATION AND ERROR MESSAGES

Message	Description	Suggested Action
Invalid node	Communications error in the area indicated.	CLEAR ERROR, and reattempt operation.
Invalid parameter		
Invalid reference number	The reference number (e.g., 0XXXX, 30XXX, 4XXXX) is not valid for the type of node or operation used. (Such as trying to place a 1XXXX reference in the bottom of a timer.)	Enter valid reference number: For example, 0XXXX for a coil, 0XXXX or 1XXXX for a contact, or 4XXXX for a DX destination.
Invalid unit number	The unit number (device address) is not in the valid range, 1 through 246 inclusive.	Enter a unit number from 1 to 246.
Memory full	The limit for user logic space has been reached.	Review program for best use of memory.
Memory protect ON	The desired action may not be performed because the 584L memory protect is ON.	Memory protect must be unlocked to perform desired action.
Network not found highest # - - - - -	The requested network was not found in user logic. The last network number found is given.	
Network not found	The 584L PC user logic has been altered by another device.	Logout and reattach.
No. DX's in this controller	The 584L PC has "Basic Level" capabilities rather than "Advanced." There are no data transfer functions available.	
No element at cursor	To perform the operation requested, the cursor must be placed on an element.	Place cursor on appropriate element and perform operation.

Message	Description	Suggested Action
No element to compress	The compress function vertical or horizontal, cannot be performed because the cursor is not on any elements.	Place cursor on appropriate element and perform operation.
No empty spaces	Occurs during TRACE/ RETRCE function. Indicates that there is no more room for displaying traced references in the reference area.	ERASE displayed references to make room, or CHANGE SCREEN to allow display on alternate screen.
No function attached to key	The selected key does not have a valid function at this stage of operation. For example, pressing a key with no software label has no valid function.	Select another key.
No network in controller	The controller has no networks programmed into user logic.	
No network on screen	The selected operation cannot be performed because no network is displayed.	
No other disabled coil	Appears during search procedures when all disabled coils have been found and another search is given.	
No reference present	Occurs when ERASE, GET PREVIOUS, or GET NEXT is pressed but no reference is present at the cursor.	Reposition cursor to be on desired reference.
No search parameters	This message appears when SEARCH is pressed when no parameters (type of node, reference number) have been set.	Move cursor to set search area of CRT screen and choose parameters.

INFORMATION AND ERROR MESSAGES

Message	Description	Suggested Action
Not attached to the controller	The selected operation cannot be performed because the P190 programmer is not attached to the controller.	Attach to controller, and perform operation.
Not enough memory	This message appears during configuration, indicating that the memory size of the machine cannot support the configuration parameter chosen.	Choose parameter within system capabilities.
Not enough room	Occurs when attempt is made to insert a two or three node function block into the last row of a network.	Redesign network or go to next network.
Not enough room to compress	This message indicates that the network cannot be compressed, vertically or horizontally, due to lack of space.	
Not in program mode	The selected function or key (i.e., START NEXT, DELETE, ENTER) is only operative when the P190 is acting as a programmer, P190 keylock unlocked.	Unlock P190 keylock and perform operation.
Not logged in	The selected function or key is not operative because the P190 Programmer is not attached to the 584L PC.	Attach P190 Programmer to 584L PC and perform operation.
Not logic screen	The function selected (i.e., START NEXT) can only be performed on the logic screen.	Access logic screen.
Only decimal or hexidecimal characters allowed in AR	Special characters (i.e., /, ?, ;) are not allowed in the AR. Only decimal (0-9) and hexidecimal (0-9, A-F) are allowed.	Re-enter valid data.

Message	Description	Suggested Action
Port empty or unattached	Communication is not possible through the selected port because there is no connection, either to a printer or a 584L PC.	Check connections at P190 and at peripheral device. Check to assure that correct cable is used.
Port 2 not connected	Peripheral port 2 (used to communicate with the 584L PC) is not connected; communications are not possible.	Check connections at P190 port 2 and at peripheral device.
Port 2 transmit timeout	Occurs during print operations. Indicates that communications have been interrupted.	CLEAR ERROR, retry operation.
Port 2 UART status error	Communications error.	CLEAR ERROR, retry operation.
Power display invalid — network skipped	The network displayed has been bypassed using the SKIP function. Disregard the power flow display.	
Programming going on	This message indicates to a P190 being used as a monitor that changes have been made to memory by another P190 programmer.	To make sure display of memory is current. LOGOUT and then reattach.
P190 UART status error	Communications error.	CLEAR ERROR, RESET.
Reference on alternate screen	This message alerts the user that a reference retrieved by the TRACE function is on the alternate screen. This message only appears if the reference area is full.	Change to alternate screen, if desired.
Running 584L	The 584L PC attached to the P190 programmer is running.	

Message	Description	Suggested Action
Search failed	No node and/or reference number as specified in the search parameters was found in the data base.	
Segment boundary crossed	This message informs the user that a segment boundary has been crossed, either forwards of backwards.	
Start of logic memory	The first network is shown. GET PREVIOUS is invalid.	
Stopped 584L	The 584L PC connected to the P190 programmer is not running.	
System Error: XXXX	Four digit number may indicate a single error or the hexadecimal, no-carry sum of several errors. For example Peripheral port "Stop" 8000 CPU diagnostic failure 0020 Invalid node type 0008 Logic checksum error 0004 System error: 802C	Error codes may appear after a tape is loaded and before the 584L PC is configured. In most cases, these are caused by loading and may be ignored. But if communications appear to be interrupted, reload the tape.
System Error Code Number	Since there is no carry, each set of errors produces a unique system error code.	
0001	Illegal configuration.	
0002	Backup checksum error.	
0004	Logic checksum error.	
0008	Invalid note type.	
0010	Invalid traffic cop type.	
0020	CPU diagnostic failure.	
0040	Realtime clock failure.	

Message	Description	Suggested Action
0080	Watchdog timer (WDT) failure.	
0100	No end-of-logic (EOL) or bad number of segments.	
0200	State RAM test failure.	
0400	Start-of-network (SON) Node did not start segment.	
0800	Bad multi-rate table.	
1000	Illegal panel or host CPU intervention.	
2000	Illegal mini-code instruction.	
4000	(Reserved).	
8000	Peripheral port "Stop."	
Timeout error-communications down	Communications error in message 584L PC.	CLEAR ERROR, RESET.
Too many DX's-loading	All data transfer functions (DX's) cannot be loaded due to lack of available memory.	Reload tape. Reconfigure if necessary.
Total # of segments = 0 or	584L PC appears to have illegal number of 16	Check configuration, reload tape. segments.
Trace stack empty	The user has traced back to the original network in the operation.	
Verticals not allowed	Appears when attempt is made to insert vertical in row 7.	

Message	Description	Suggested Action
9999 Overflow	Value in register is too large to display in decimal with four digits. This message appears in the reference area, next to the reference number.	
# of coils must be multiples of 16 accepted	Coils may be specified only in multiples of 16. No other numbers are	Enter correct number.

APPENDIX D
GLOSSARY

A

Address:

A numeric value used to identify a specific I/O channel and/or module.

AND (Logical):

A mathematical operation between two bits. The result of the logical AND will be a one (ON) bit only if both bits are one bits; otherwise, the result will be a zero (OFF) bit. This operation can be performed between bits with each pair of bits, one from each, examined by their relative location within each.

Arithmetic Function:

A type of logic used to add, subtract, multiply, or divide two numeric values. The status of the output is governed by the result of the arithmetic operation: additional overflow; comparisons (greater than, equal to, or less than); and illegal division.

ASCII:

A 7-bit digital coding of standard alphanumeric characters as established by the American National Standards Institute. ASCII stands for the American Standard Code for Information Interchange.

B

Baud:

A unit of data transmission speed equal to the number of code elements (bits) per second.

BCD (Binary-Coded Decimal):

A system of numbers representing decimal digits (0-9) using four binary digits (1 or 0). BCD is a recognized industrial standard; BCD input (e.g., thumbwheels) and output (e.g., numerical displays) are readily available.

Binary:

A numeric system wherein values are represented only by numbers 1 and 0 (ON/OFF). Also called "base 2". This system is commonly employed in modern electronic hardware since circuits can be economically designed for ON/OFF status.

Bit:

Contraction of binary digit. A single number whose value can be either a ONE or a ZERO. The smallest division of a PC word.

Bit Modify Function:

This function allows individual bits in a matrix to be altered. Only one bit per scan can be affected by this function; all other bits retain their state. Bits can be either set to a one (ON) condition or cleared to a zero (OFF) condition.

Bit Rotate Function:

This function allows a series of bits to be rotated or shifted through a matrix. If the function is enabled for a number of scans (e.g., five), all bits in the matrix will be rotated the same number of bit locations (e.g., five). Provisions are made to select the direction of rotation, left or right.

Bit Sense Function:

This function allows individual bits in a matrix to be examined, but not altered. An output is used to indicate a one (ON) bit with power flow and a zero (OFF) bit without power flow. The status of only one bit can be obtained each scan.

C

Cascade Function:

Connecting two or more functions together to control one output. For example, timers and counters can be cascaded to produce results that cannot be achieved by one counter or one timer.

Channel:

A group of I/O modules that are separately connected to the mainframe. For example, a channel of I/O can contain up to 128 input points and 128 output points.

Coil:

A discrete logical conclusion to a series of logical operations performed by the programmable controller. The results can be output to the real world via an output module to activate motor starters, solenoids, relays or pilot lamps. Coils are turned OFF when power is removed from the mainframe. (See Latch.)

Compare Function:

This function causes two matrices to be compared on a bit-by-bit basis to find all the bit locations which differ, and save the result for later use; their contents are not altered, but only examined.

Complement Function:

This function causes the content of one matrix to be complemented (all ones replaced by zeros, and zeros by ones) and placed in another matrix for reference by any other function.

Counter:

A type of logic that is used to simulate the operation of external counters. In relay panel hardware, an electromechanical device which can be wired and preset to control other devices according to the total cycle of one ON or OFF function. In a PC, a counter is internal to the processor, which is to say it is an electronic function controlled by a user programmed instruction.

Cursor:

Visual movable pointer used on a CRT or programming panel by the programmer to indicate where an instruction is to be added to the ladder diagram. The cursor is also used for editing functions.

D

Data Transfer Block:
A PC function block used in data transfer (DX) programming.

Data Transfer (DX) Function:
A technique of moving and manipulating data within the controller under control of DX logic.

Data Transfer Line:
A line of ladder logic containing data transfer (DX) functions.

Digital:
Having discrete states. Digital logic can have up to 16 states. However, most digital logic is binary logic with two states, ON or OFF.

Disable:
The capability to disconnect a logic coil or a discrete input from its normal control, and force it unconditionally ON or OFF. (See Force.)

Discrete Reference:
A reference that can be either ON or OFF. A discrete reference can be an input, output, or internal logic element.

Double Precision Function:
The technique of storing a single numerical value in two consecutive registers. Since each register can store up to four digits (maximum value 9,999), double precision allows magnitudes of up to 99,999,999 to be stored.

E

Edit:
To deliberately modify the user program.

Element:
The basic building block of the PC ladder logic. An element can be a relay contact, horizontal short, vertical short, fixed numeric value, register reference, coil, or function block. Sometimes refered to as a logic element.

Enable:
To reconnect a logic coil or discrete input after it has been disabled. The opposite of "Disable."

Exclusive OR (XOR):
A mathematical operation between two bits. The result of the exclusive OR will be a one (ON) bit only if either bit is a one bit. Only if they are both zeros (OFF) or both ones (ON) will the result be a zero.

F

FIFO Function:
A special DX table that maintains the order of data entered into the table, First In, First Out.

Force:

The function that can be used to change the state of a disabled reference. The reference can be changed from OFF to ON or ON to OFF. This allows the user to energize or de-energize any input or output by means of the program panel independent of the PC program.

H

Hexadecimal:

The numbering system that represents all possible ON/OFF combinations of four bits with sixteen unique digits (0-9 then A-F).

I

Inclusive OR:

A mathematical operation between two bits. The result of the inclusive OR will be a one (ON) bit if either bit is a one bit or both bits are ones; only if both bits are zeros (OFF) will the result be a zero. This operation can be performed between groups of bits with each pair of bits examined by their relative location within each.

Input:

A signal that provides information to the controller; can be either discrete input (pushbutton, relay contacts, limit switches, etc.) or numeric input (thumbwheel, external solid-state device, etc.)

Input Devices:

Devices such as limit switches, pressure switches, push buttons, etc., that supply data to a programmable controller. These discrete inputs can have a common return or an individual return (referred to as isolated inputs). Other inputs include analog devices and digital encoders.

L

Ladder Diagram:

An industry standard for representing control logic relay systems with logic lines representing rungs on a ladder. It expresses the user programmed logic of the controller in relay equivalent symbology.

Latch:

The type of coil that is retentive upon power failure. Can be used similar to a latching relay. Normally, coils are reset to OFF conditions upon powerup; those coils selected by the user as latched (L) will not be altered and thus retain their previous condition (ON or OFF).

Logic:

A systematic interconnection of digital switching functions, circuits, or devices, as in electronic digital computers.

Logic Diagram:

A graphic description of logic functions and conditions. It is used to find the result of an addition of the contents of two registers; a logical compare of two matrices; as well as other arithmetic operations.

Logic Element:

Any one of the elements that can be used in a ladder logic diagram. The elements include relays, coils, shunts, timers, counters, arithmetic functions, and DX functions.

Logic Line:

A line of user logic used to construct the unique logic for the application.

M

Matrix Function:

Matrices are defined as sequential registers, each as 16 bits, up to a maximum of 99 registers (1584 bits). A group of consecutive registers referred to by logic, such that individual bits can be utilized in lieu of numerical values. Bit operations that can be performed include: AND, OR (inclusive), XOR (exclusive), COMPARE, MODIFY, SENSE, COMPLEMENT, and ROTATE.

Memory:

Storage area for binary data and programs.

Memory Protect:

The hardware capability to prevent a portion of the memory from being altered by an external device. This hardware feature is under keylock control.

Move Function:

A DX capability which allows data to be transferred without modification within the controller. Data can be transferred from a register to a table, from a table to a register, from a table to a table, into a FIFO stack, or out of a FIFO stack.

N

Network:

A group of logic elements that are connected together to perform a specific function (e.g., a motor starter control circuit).

Node:

The smallest possible programming increment in a ladder logic diagram. (Most logic elements require only one node, others require two or more nodes.)

O

One-Shot:

A discrete reference, typically a logic coil, that is energized (valid) for exactly one scan of the controller's logic.

Output:

A signal provided from the Controller to the "real world"; can be either discrete output (e.g., solenoid valve, relay, motor starter, indicator lamp, etc.) or numeric output (e.g. display of values stored within the controller).

Output Devices:

Devices such as solenoids, motor starters, etc., that receive signals from the programmable controller.

P

Parity:
Method of verifying the accuracy of recorded data.

Parity Bit:
An additional bit added to a memory word to make the sum of the number of "1's" in a word always "even parity" or "odd parity."

Port:
An I/O connection on a processor or peripheral device.

Preset:
The upper limit specified for a counter or timer function. When the specified preset value is reached, an output is energized indicating the status of a counter or timer.

Programmable Controller (PC):
A solid-state control system which has a user programmable memory for storage of instructions to implement specific functions such as: I/O control logic, timing, counting, arithmetic and data manipulation. A PC consists of a central processor, an input/output interface, memory, and a programming device that typically uses relay equipment symbols. PC is purposely designed as an industrial control system that can perform functions equivalent to a relay panel or a wired solid-state logic control system.

Programming Panel (Programmer):
Device for inserting, monitoring, and editing a program in a PC.

R

Reference Numbers:
Numbers which identify the elements of the relay ladder logic. References can be either discrete (logic coils, inputs, or sequencer steps) or register (input or holding).

Register:
A location within the controller allocated to the storage of numerical values. All holding registers are retentive on power failure. There are three types of registers: input whose contents are controlled by the "real world" outside the controller; holding registers whose contents are controlled from within the controller; and output registers, which are special holding registers since their contents can also be provided to the "real world".

Relay:
An electromagnetic device operated by a variation in conditions of an electric circuit. When so operated, it controls other devices such as switches.

Relay Element:
A logic symbol used to simulate the effect of a relay. Contacts can be normally open, normally closed, or transitional contacts.

S

Scan:

The technique of examining or solving logic networks one at a time in their numeric order. After the last logic network is solved, the next scan begins at network one; logic is always solved in this fixed cyclic process.

Skip Function:

This function allows a group of consecutive networks to be skipped or omitted in the scanned logic solution. The status (ON/OFF) of all coils and the contents of registers controlled by these networks are not altered when they are skipped.

T

Table:

A group of consecutive registers used to store numerical values.

Table Search Operation (SRCH):

This function searches a table of registers for a specified value. The source is not altered, only examined. The SRCH function uses a pointer to indicate the location(s) within the table of registers which contain the desired value. This pointer is the only register whose value is altered by the SRCH function.

Timer:

PC logic used to measure and record the time of an event or sequence of events. Timers can accumulate time in either seconds, tenths of seconds, or hundredths of seconds depending on the PC.

Traffic Cop:

A portion of the PC executive that controls how input and output data is interpreted relative to its channel number and address index position.

MODICON
an AEG company

Publications Comment Form

Document Part Number PI-584L-002 Rev. C

Title 584L Programmable Controller Programming Guide

We are constantly striving to improve the content and usability of our technical documents. You can help us by answering the questions below and mailing this form to us. Also, if you find any errors or have any suggestions for improvement, please let us know.

How do you use this document?

- ☐ Introduction to the product
- ☐ Classroom resource
- ☐ Self-study
- ☐ Programming Procedures
- ☐ Advanced programming techniques
- ☐ Operating instructions
- ☐ Reference
- ☐ Other _____

How did you get this document?

- ☐ Received with equipment
- ☐ Received from Sales or Customer Service Representative
- ☐ Ordered from MODICON
- ☐ Do not know
- ☐ Other _____

Please rate this document.

		Excellent	Very Good	Good	Fair	Poor
Technical Accuracy	- Does the system work the way it is described in the manual?	☐	☐	☐	☐	☐
Readability	- Is the manual easy to read and understand?	☐	☐	☐	☐	☐
Clarity	- Are the instructions easy to follow?	☐	☐	☐	☐	☐
Examples	- Are the examples helpful and realistic?	☐	☐	☐	☐	☐
	Are there enough examples?	☐	☐	☐	☐	☐
Organization	- Is the organization of the manual logical?	☐	☐	☐	☐	☐
	Is it easy to find what you are looking for?	☐	☐	☐	☐	☐
Illustrations	- Are the illustrations clear and useful?	☐	☐	☐	☐	☐
Physical Attractiveness	- What did you think of the layout, printing, binding, etc?	☐	☐	☐	☐	☐

	Y	N
Are there any terms or concepts that are not defined clearly?	☐	☐

If so, what are they? _____

	Y	N
After reading this document, are you able to use the equipment?	☐	☐

What errors did you find in the manual? (Please include page numbers. Attach an extra sheet if necessary.)

Do you have any comments or suggestions? _____

Name _____ Street _____

Title _____ City _____

Dept./Mail Stop _____ State/Country _____

Company _____ Zip Code _____ Telephone _____

Thank you for your help.

CUT ALONG LINE

CUT ALONG LINE

FOLD ALONG LINE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 234 ANDOVER, MA

Postage will be paid by addressee:

MODICON, Inc.
1 High Street
North Andover, MA 01845-9943

Attn: Order Entry 3-2B

FOLD ALONG LINE

MODICON
an AEG company

Publications Order Form

TO ORDER BY PHONE, CALL
1-800-468-5342 and ask
for an Order Entry Coordinator

CUT ALONG LINE

Bill To:	Ship To: (if different):

Customer Contact
Name: _____

Purchase Order No. _____ Date _____ Telephone: _____

Document Part Number	Description	Quantity	Unit Price	Total Price

Sub Total _____
Freight _____
Total _____

Authorized Signature _____ Date _____

Ordering Instructions

1. Provide appropriate billing address and shipping address (if different from billing address).
2. Please provide a contact name and phone number in case we have a question about your order.
3. Indicate your purchase order number and date. (You may prepay by enclosing a check for the full amount.)
4. Give the part number, description, and quantity for each document ordered.

Delivery

Unless otherwise specified, all orders are shipped best way surface, F.O.B. Andover, MA (prepay and add if UPS surface, collect for truck and air). If you specify insurance, you will be billed for these changes. MODICON will not assume any liability in connection with the shipment nor shall the carrier be construed to be an agent of MODICON.

Payment

Terms are net 30 days from date of invoice. Unless otherwise stated, partial shipments will generate partial invoices.

Prices

Prices are subject to change without notice. Individual prices can be found in the Publications Catalog or obtained by calling an Order Entry Coordinator at 1-800-468-5342.

CUT ALONG LINE

FOLD ALONG LINE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 234 ANDOVER, MA

Postage will be paid by addressee:

MODICON, Inc.
1 High Street
North Andover, MA 01845-9943

Attn: Technical Publications 7-2A

FOLD ALONG LINE

Modicon, Inc., Industrial Automation Systems
One High Street, North Andover, MA 01845
(508) 794-0800
24 Hour Support Center 1-800-468-5342

1C 1/92 NP Printed in U.S.A.