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

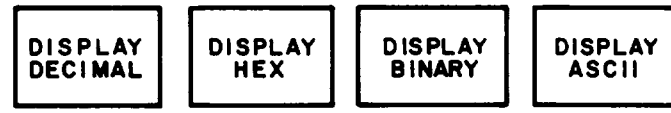
Each bit has a value of one or zero. Registers are generally displayed in decimal format.

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).

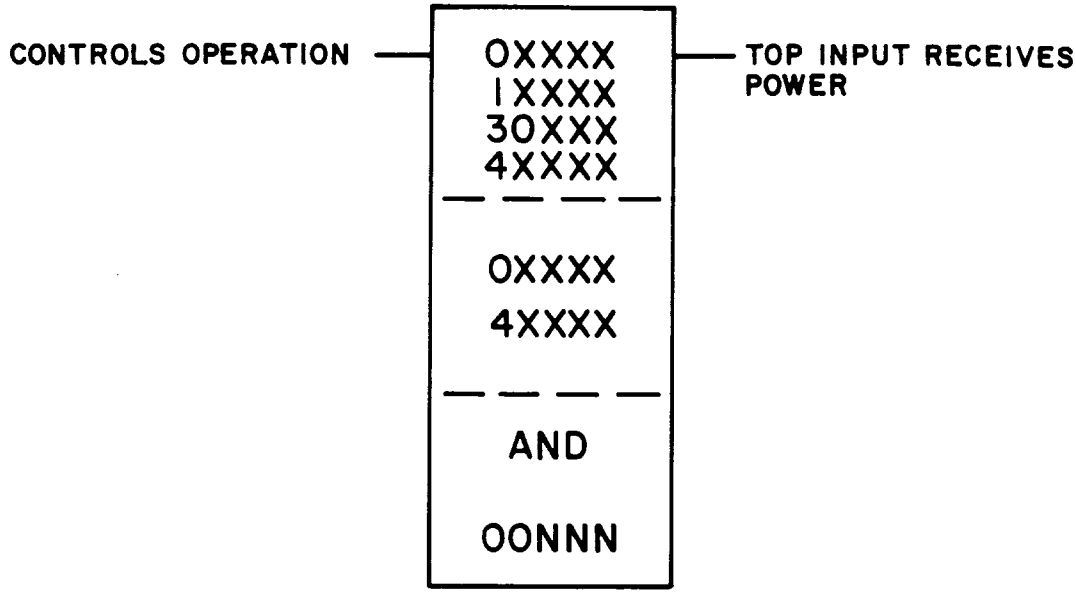
The DX MATRIX functions can revise data, shift data, or examine data in a matrix with or without altering the source.

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.

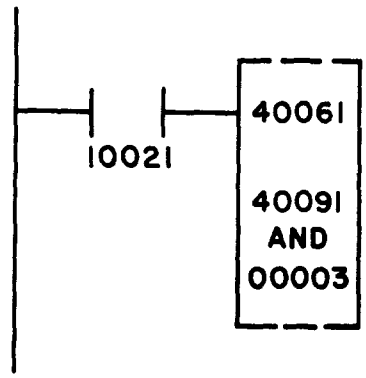
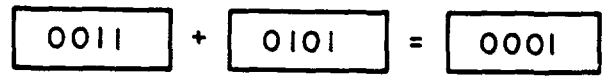


Figure 4-2. Logical AND

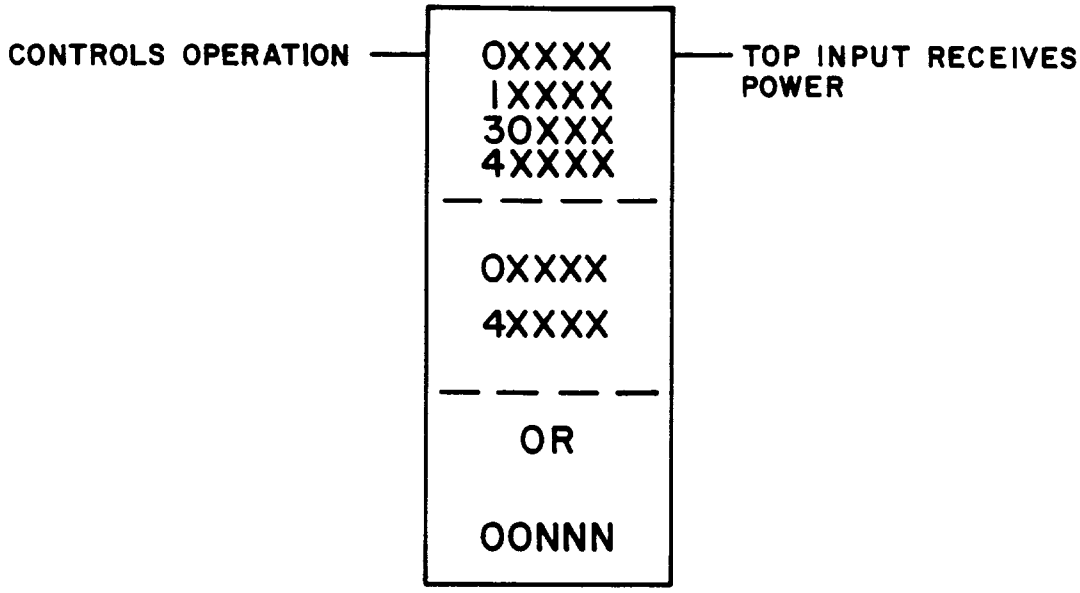
When input 10021 is energized, the top input receives power. The bit pattern in registers 40061 — 40063 is logically ANDed with the current bit pattern in registers 40091-40093. The resulting bit pattern is placed in registers 40091-40093, thereby replacing the previous bit pattern. The source registers (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.

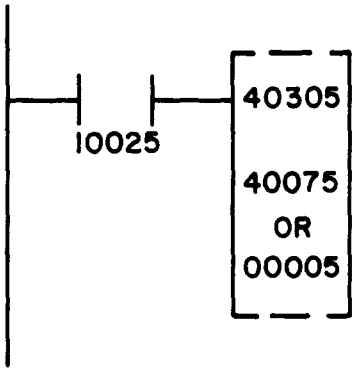


Figure 4-3. Logical Inclusive OR

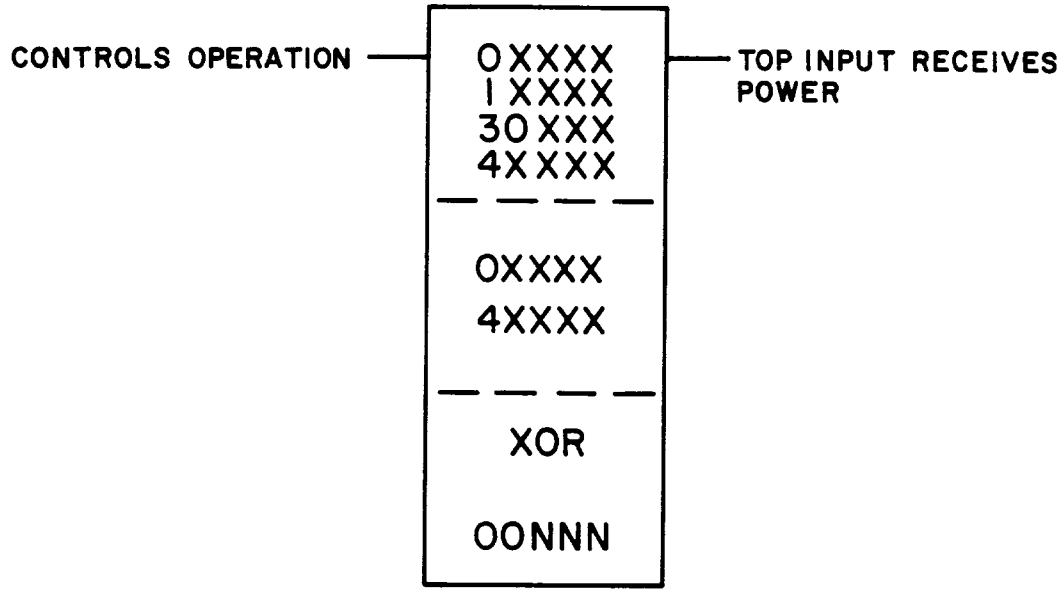
When input 10025 is energized, the bit pattern in registers 40305-40309 is logically ORed with the current bit pattern in registers 40075-40079. The resulting bit pattern is placed in registers 40075-40079, thereby replacing the previous bit pattern. The source registers (40305-40309) are not altered.

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.

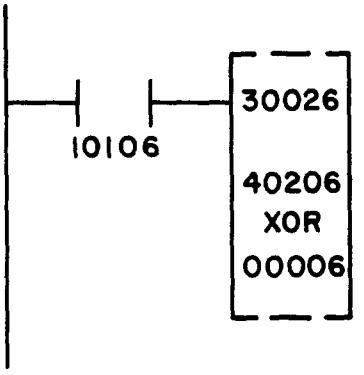
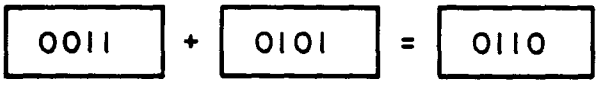


Figure 4-4. Logical Exclusive OR (XOR)

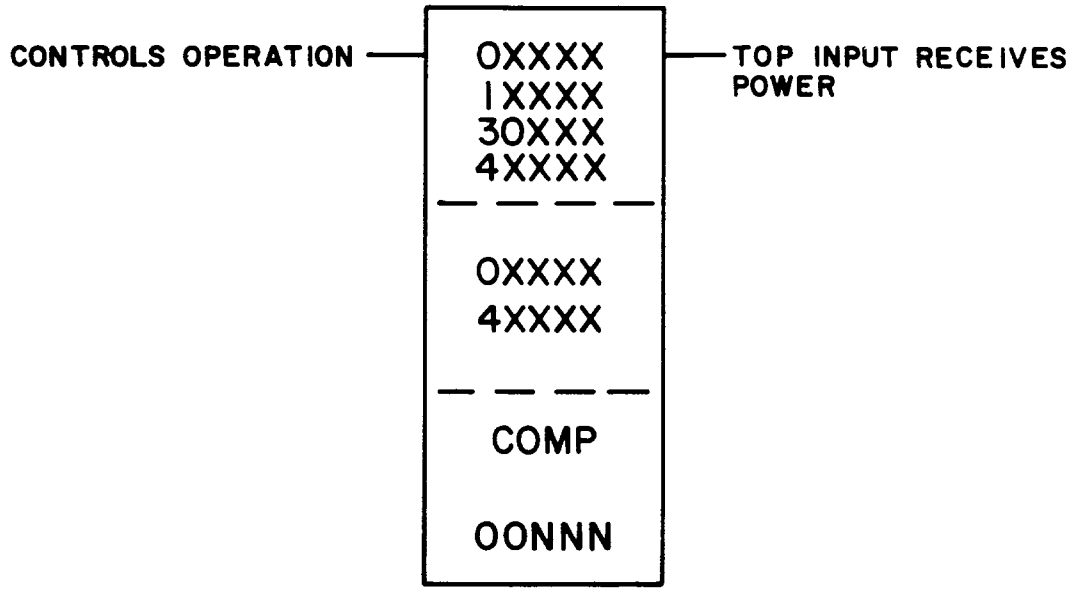
When input 10106 is energized, the top input receives power. A Logical Exclusive Or is performed between the bit patterns in registers 30026-30031 and registers 40206-40211. The resulting bit pattern is placed in registers 40206-40211, thereby replacing the previous bit pattern. The source registers (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.

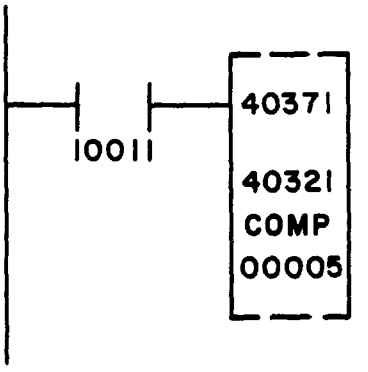


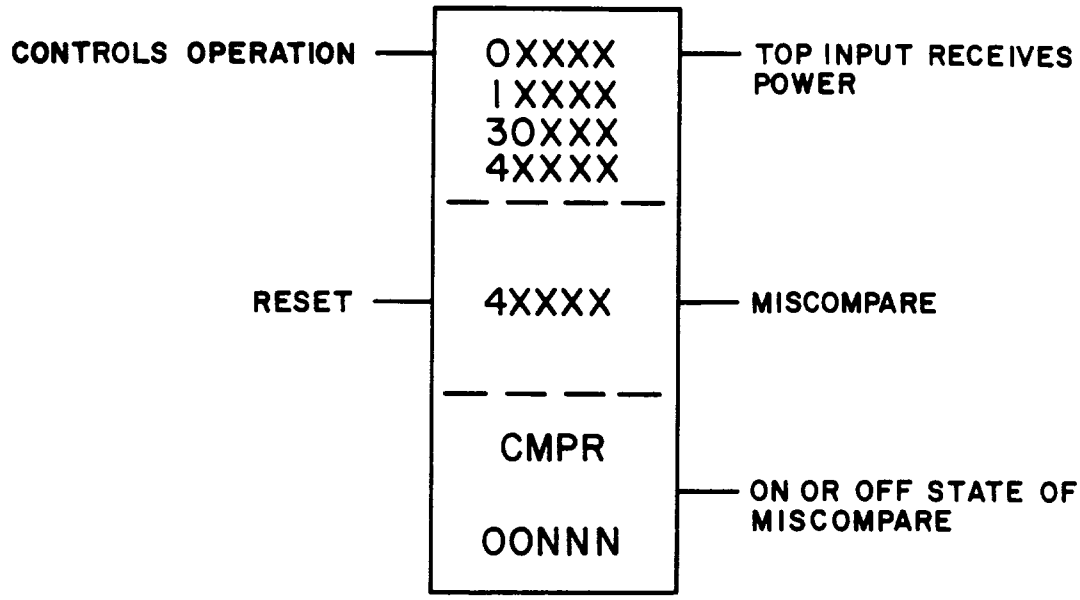
Figure 4-5. Logical Complement

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.

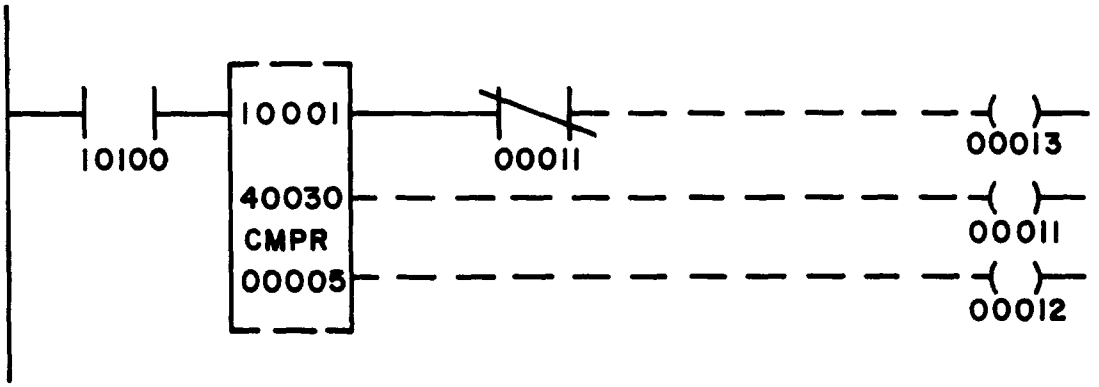


Figure 4-6. Logical Compare

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.

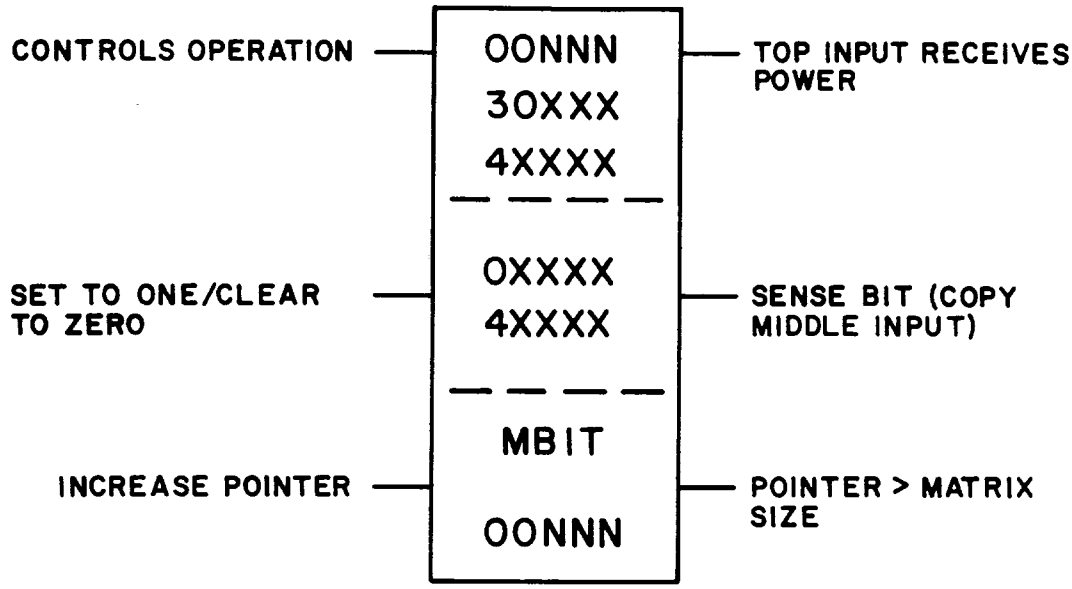
If a normal relay contact is used, the compare function continues after finding a miscompare so the pointer value changes. If the location of the miscompare is bit 5, the compare continues at bit 6.

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.

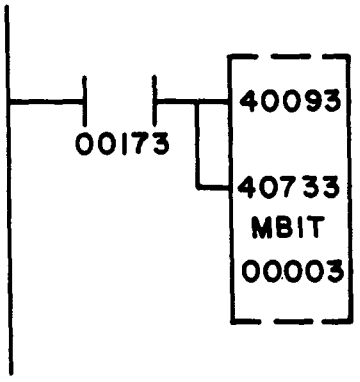


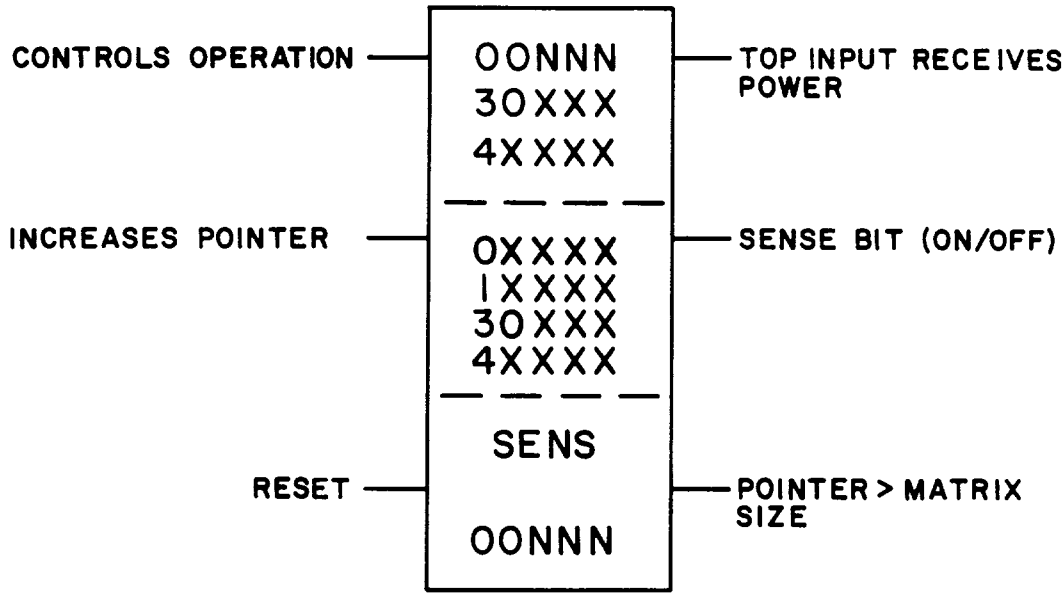
Figure 4-7. Logical Bit Modify

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.

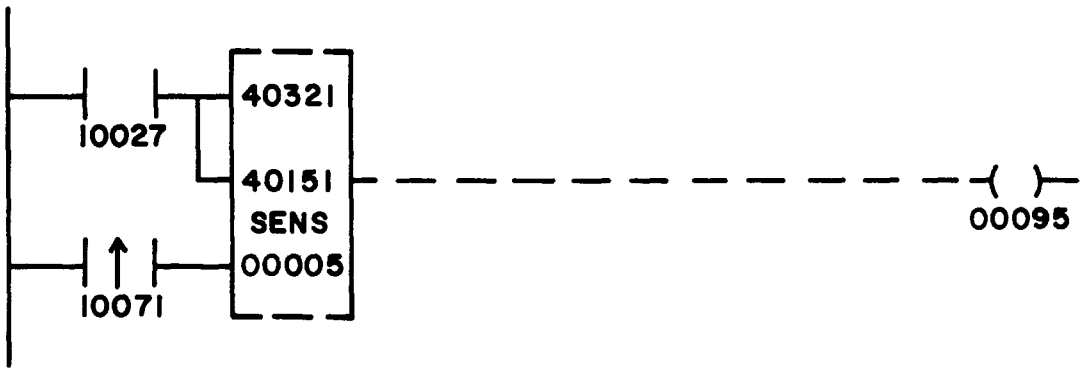


Figure 4-8. Logical Bit Sense

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.

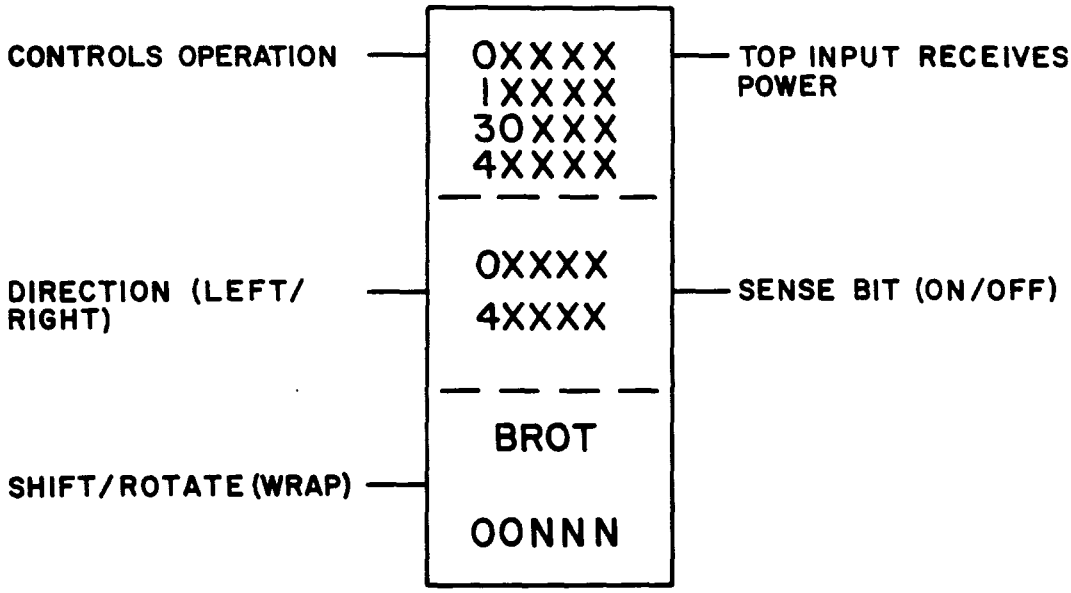
If the bit is a one bit, the middle output passes power and energizes coil 00095. If the bit is a zero bit, no power is passed. Since the middle input receives power by way of the vertical connection between the top and middle inputs, the pointer increases by one during each scan that input 10027 is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

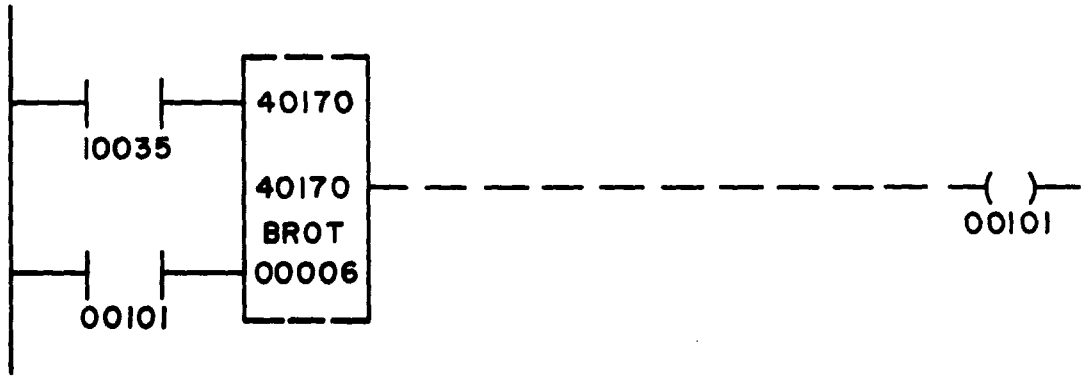


Figure 4-9. Logical Bit Rotate

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.

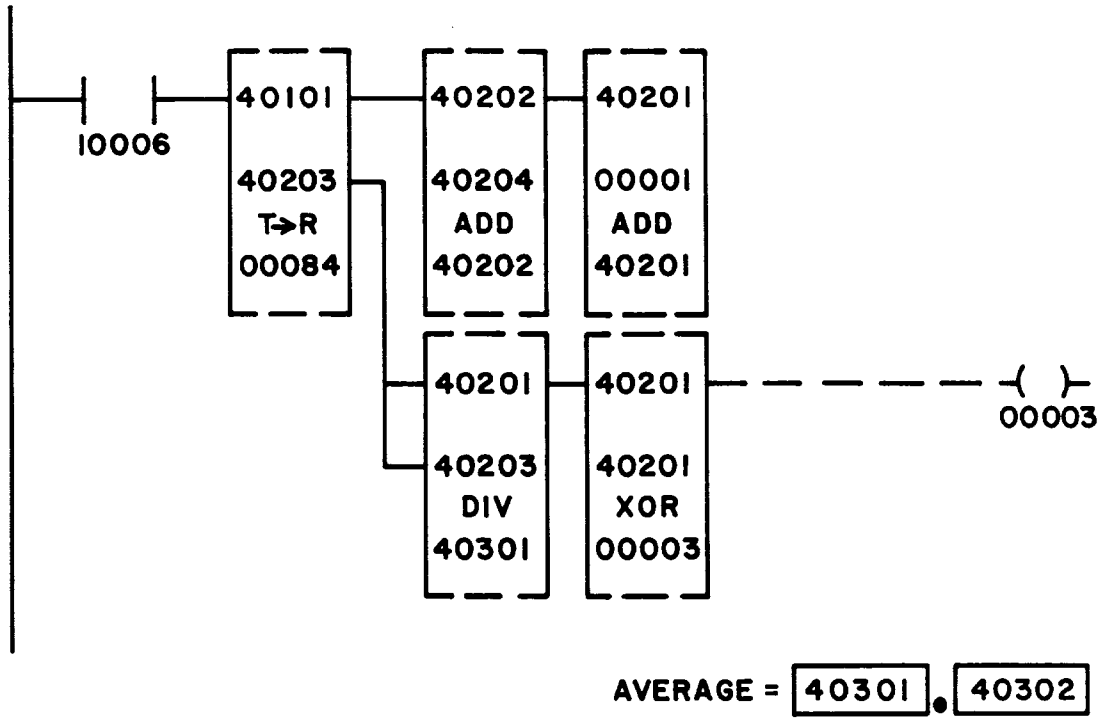


Figure 4-10. Table Averaging

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.

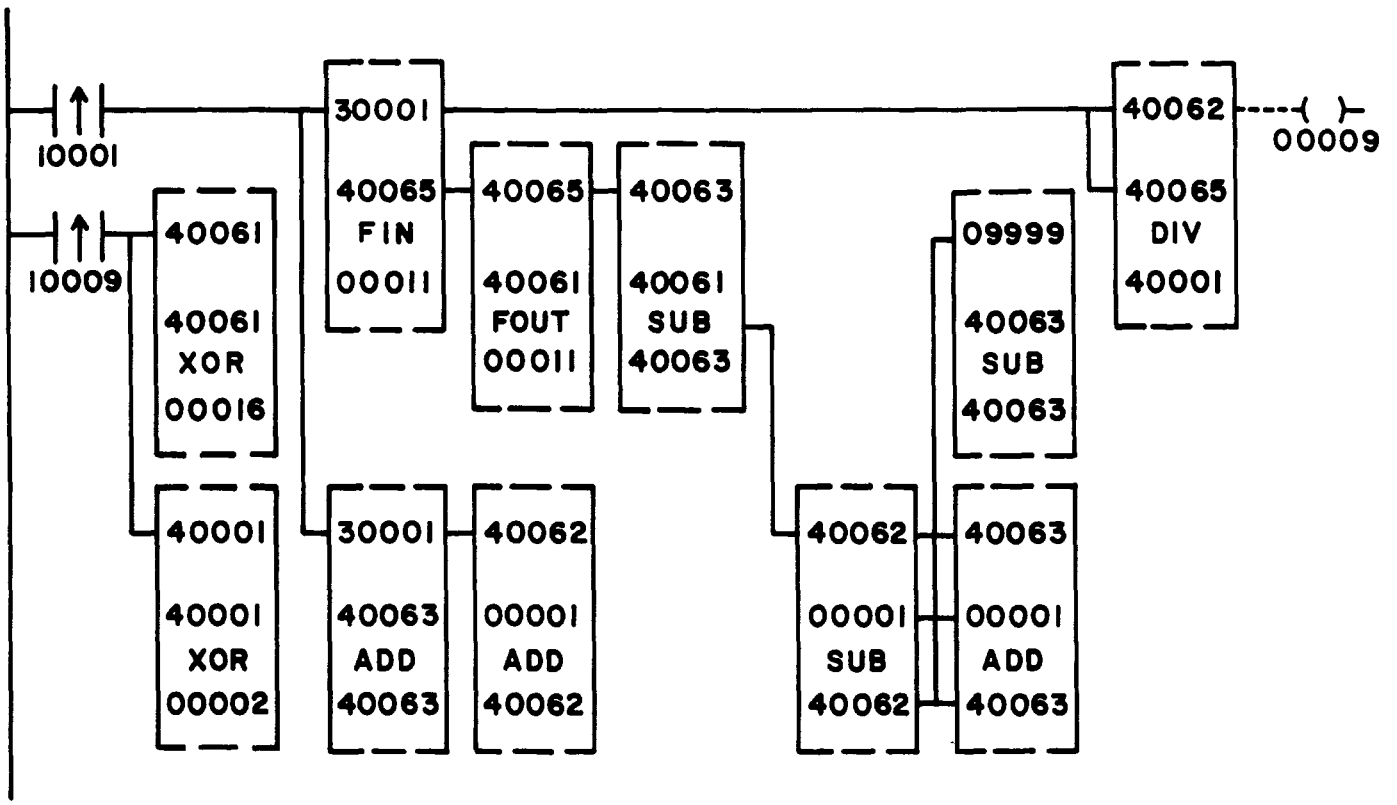


Figure 4-11. Running Table Averaging

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.

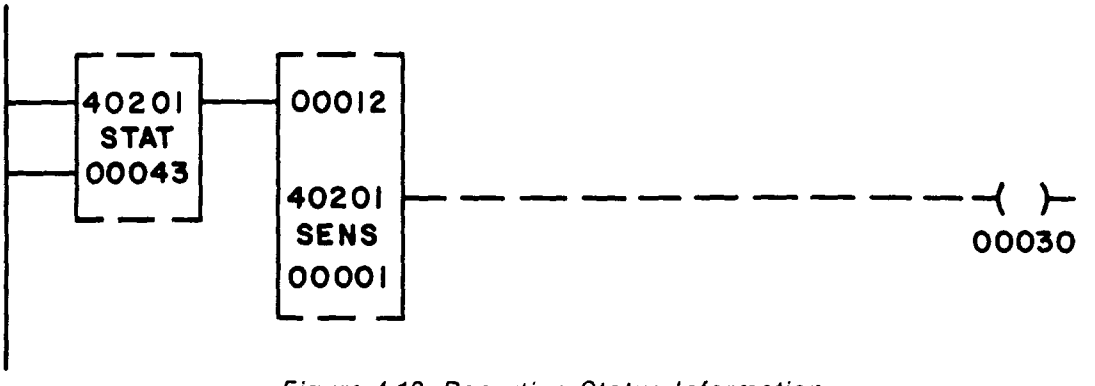


Figure 4-12. Reporting Status Information

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.