The operator's panel contains two sets of thumbwheels; in this example one set of two digits (0-99) wired to input register 3001 and the other of four digits (0-9999) wired to input register 3002. A single BCD display of four digits is wired to output register 4003, and a keylock switch to provide security is wired to discrete input 1001. If more than 99 registers are to be monitored, the thumbwheels connected to 3001 can be increased to three digits (0-999).

In this example, numerical data is placed in registers 4031-4110. If these registers contained binary status used to drive outputs, or with matrices, input register 3002 should be coded binary and connected to 16 separate toggle switches; output register 4003 should also be coded binary and connected to 16 separate status lamps.

Since the A element of line 121 is referenced to a coil that is never energized (i.e., null data relay line, input that is not wired up, etc.), line 121 always obtains the contents of 3001, subtracts 1 from it, and forces the pointer register 4002 to that value. Line 122, every scan that the A element is closed, obtains from the table of length 80 (starting with 4031), the element referred to by the pointer and places it in 4003 for display. Line 122 is inhibited from progressing through the table at one element per scan as it normally would, since line 121 is always forcing the pointer back to its required value. However, every time a new value is entered into 3001, the corresponding element is automatically extracted from the table; no action is required by the operator, other than to change the value on the thumbwheel connected to 3001. If the thumbwheels are set to zero, no action is desired since there are no elements in the table with the number or address of zero. Line 121 will still take the zero on 3001, subtract one from it, and place the result (a 1) into 4002. However, since this is a *negative* one, its coil does not come ON and no move is performed by line 122; whatever value was in the display remains there. If the value on 3001 exceeds 80 (the length of the table which the operator is allowed to monitor), no moves are performed since the pointer will be forced to 80 or greater, and line 122 is limited by the DX code to 80 registers.

Altering the data is performed in a similar manner. The operator enters on 3001 the element he would like to change, views its current contents on 4003, enters the new value on 3002, and closes the keylock switch. If he does not have a key, he cannot make changes. When the keylock switch (input 1001) is closed, line 123 forces the pointer and line 124 moves the data from 3002 into the table; the display will automatically verify entry of the new data. Again, entry of data into element zero is prevented by coil 123 used in the A element of line 124 and entry of data into registers beyond 4067 (table length 37) is inhibited by the DX function code.

This monitoring capability requires only four logic lines and provides monitoring/altering capability for up to 99 consecutive registers. If more than 99 registers are required, additional capacity can be added at the rate of five logic lines per additional 99 registers or fraction thereof. This monitoring scheme does not in any way affect the use of these registers as presets (C element) in timers/counters, set points (C element) of calculate lines, current times or counts (D element), etc. They can and should be used by other logic lines in the program as required. To be fully confident in this example, select some values between 1 and 80 and test the effects of these logic lines.

Compare the efficiency of this method using Data Transfer with Example III where calculate lines are utilized. These four logic lines using DX provide monitoring and altering of up to 99 registers; 99 calculate lines provide monitoring only of 99 registers.

3.5.2 Matrix Handling (Group 2YXX)

These codes provide the capability to manipulate data as binary bits, either in large segments or as individual bits. The following functions are available with this group:

Basic		Improved	
20XX	Logical AND	25XX	Complement
21XX	Logical OR (Inclusive)	26XX	Logical OR (Exclusive)
22XX	Matrix Compare	27XX	Rotate Left
23XX	Clear Bit or Sense Bit	28XX	Rotate Right
24XX	Set Bit or Sense Bit	29XX	Sequencer Move

NOTE

All 184 Controllers with matrix provide the basic capability; only those executives specifically noted in Table 12 as having the improved capability provide codes 25XX-28XX. All 384 Controllers are provided with both Basic and Improved Matrix capabilities regardless of TEF selected. Only 384A and 384B Controllers can utilize code 29XX, unless specifically noted in Table 12.

Matrices are defined as sequential registers, each of 16 bits, up to a maximum of 99 registers (1584 bits). The size of the matrix in registers is defined by the XX digits of the functional codes. The individual bits of a matrix are numbered 1 through 1584 depending on their location in the matrix. The numbering begins at the high-order bit of the first register and continues left to right, as one would read lines of a page in a book, until the low-order bit of the last register is reached. The last bit of each matrix will be evenly divisible by 16.

For example, Figure 76 illustrates the numbering of a 5-register (80-bit) matrix. The quantity of bits in any matrix is always mutiples of 16; the smallest matrix is one register (16 bits) in length. A bit can be described as ON, ENERGIZED, VALID, SET, or TRUE if its numberical value is one (1); OFF, DE-ENERGIZE, INVALID, CLEAR, or FALSE can be used to describe a bit whose numerical value is zero (0).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	FIRST REGISTER
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	=2 REGISTER
33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	=3 REGISTER
49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	=4 REGISTER
65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	=5 REGISTER

Figure 76. Typical Matrix Bit Numbering

When receiving binary data via input registers (30XX), including analog inputs, these registers must be coded in the I/O Allocation Table (part of the executive) as binary registers. If they are not coded as binary register, the data obtained from the input modules will first have its bits interpreted as a BCD number. See 4.1.4 for additional details. A similar modification must be made to provide binary (NOT BCD) data to the output modules from holding registers (40XX). Figure 77 provides two examples of outputting binary data from register 4208 via a BCD coded register (4006).

Note that if the resultant magnitude of the 16 bits in any register is greater than 9999, an incorrect BCD display will occur. Also viewing a binary register from the Programming Panel results in a binary-to-BCD conversion. Table 13 summarizes the resultant BCD display for various bit configuration. If output register 4006 of Figure 77 was coded as binary in the I/O Allocation Table, the bit pattern in 4208 would be copied into 4006 without change.

All Matrix operations are accomplished in their entirety every scan the A element is closed (passing power), regardless of the length of the matrices.

Table 13. P112 Hexadecimal Symbols

Bit Configuration	Display	
0000	0	
0001	1	
0010	2	
0011	3	
0100	4	
0101	5	
0110	6	
0111	7	
1000	8	
1001	9	
1010	⊏	(10)
1011	⊐	(11)
1100	⊔	(12)
1101	⊑	(13)
1110	⊢	(14)
1111	(blank)	(15)

20XX — Logical AND of Two Matrices

NOTE

The Matrix AND and OR operations are useful to construct masks within the Controller as well as to move blocks of data in one scan.

This code, when the A element is closed, causes the contents of the matrix referred to in the B element to be logically ANDed with the contents of the matrix referred to in the D element and the result stored in the D element matrix. This operation is accomplished on a bit-by-bit basis and is done every scan as long as the A element is passing power; the entire AND operation is done once each scan. The contents of the B element matrix is retained and the previous content of the D element matrix is destroyed and replaced with the result of the AND operation.

For a one (1) to appear in the D element matrix after the AND operation, a one (1) must appear in both the B element matrix and the previous D element matrix; in all other cases, a zero (0) will appear in the resulting D element matrix. The coil has no significance, and is OFF in all cases. As an example, refer to Figure 78.

The B element refers to three consecutive registers (4166-4168) as does the D element (4009-4011) since the functional code has defined each matrix as three registers in length. In all cases, matrix AND will be accomplished on matrices of equal length. For this example, on the first scan that input 1021 is energized, a bit-by-bit AND will be performed between registers 4166-4168 and registers 4009-4011, and the result stored

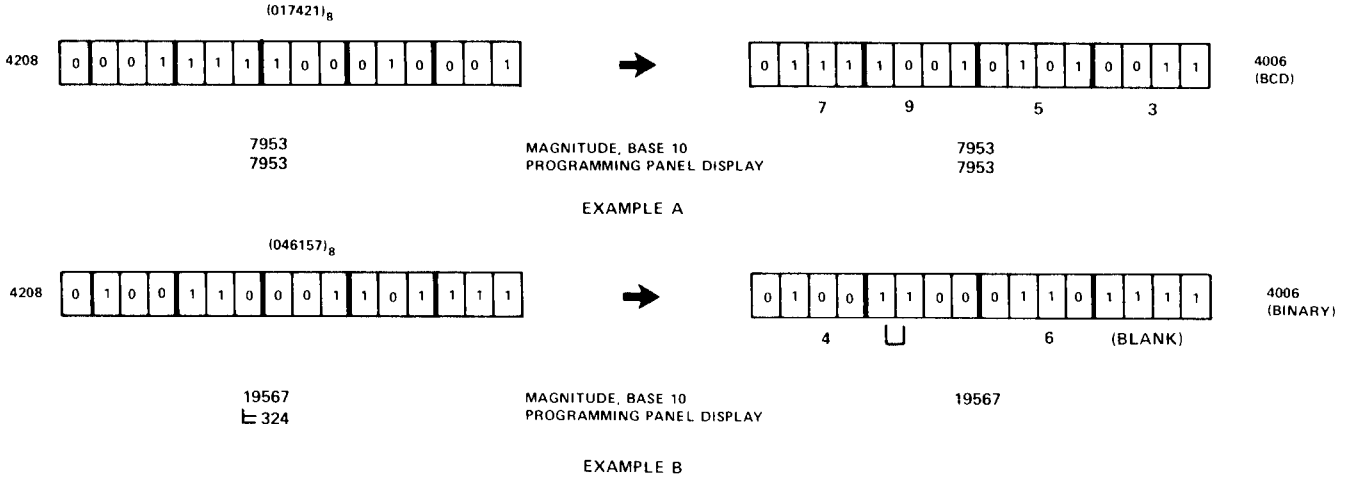

Figure 77. Programming Panel Display of Binary Data

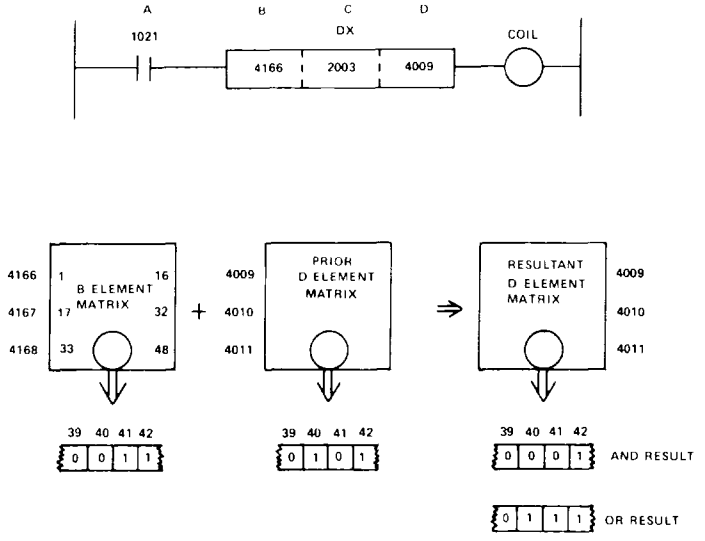

Figure 78. Sample Matrix AND and OR

in registers 4009-4011 assuming these output registers are coded for binary data. All bits in the matrices will be ANDed each scan the A element is passing power.

On subsequent scans, an AND operation is performed between the B element matrix and the resultant D element matrix; as long as the B element matrix does not change, the resultant D element matrix will not be altered.

If a one (1) in the B element matrix is changed to zero (0), the D element matrix will have a zero (0) in that location, even if the bit in the B element matrix is changed back to a one (1). Specific examples are provided by observing the operation of bits 39-42. Since there is a zero (0) in the B or previous D element matrices (or both) for bits 39-41, the resultant D element matrix contains a zero (0) for these bits. Only bit 42 has a one (1) in both matrices and thus has a one (1) in the resultant matrix. At no time will the coil be energized.

21XX — Logical OR (Inclusive) of Two Matrices

This code, when the A element is closed, causes the contents of the matrix referred to in the B element to be logically ORed with the contents of the matrix referred to in the D element, and the result stored in the D element matrix. This function operates similar to code 20XX discussed previously, except that a logical OR is performed between the two matrices. As an example, assume that the functional code of Figure 78 was 2103. The only difference in operation would be the specific results; bits 39-42 of the resultant D element matrix would contain a 0, 1, 1, 1, respectively.

Since only bit 39 has a zero (0) in both the B element matrix and the prior D element matrix, it is the only bit to contain a zero in the resultant matrix; all other bits are set to one (1). Thus, the logical OR will result in a one (1) if either matrix contains a one (1); it will result in a zero (0) only if *both* matrices contains a zero (0). Note that since this is an *inclusive* OR, a bit that is one (1) in both matrices will be a one (1) in the resultant matrix. Again the coil will be OFF in all cases. On subsequent scans, if the B element matrix has a zero changed to a one, the D element matrix will have a one in that location, even if the bit in the B element matrix is changed back to a zero.

22XX — Matrix Compare of Two Matrices

NOTE

The matrix compare is useful to monitor the status of large numbers of inputs or states as well as to detect changes in bit status. It is a very useful function.

This code, when the A element is closed, causes the contents of two matrices to be compared on a bit-by-bit basis. The compare operation will continue until a miscompare is observed, which will cause the coil to be energized and the compare terminated, or until the end of the matrices is reached. If the matrices compare exactly, the coil remains OFF and the next line of logic is performed.

The B element register refers to the first register of one matrix; the remaining registers of this matrix must follow this reference in ascending order. The functional code in the C element defines both the operation and the length in registers (XX) of each matrix. The D element register refers to a pointer register where the bit number that is currently being compared is stored; registers for the second matrix of the compare must follow this pointer register in consecutive order. Thus the B element register refers to XX registers and the D element register refers to XX+1 registers.

The contents of the pointer register is incremented by one before a compare is accomplished and will not be changed when a miscompare is encountered. Thus the bit number causing the miscompare is available from this pointer register after completion of a compare with coil ON; if no miscompare was detected (coil OFF), the pointer register will contain the bit number of the last bit in the matrix plus one.

NOTE

If more than one miscompare occurs in a matrix, the coil will remain ON for successive scans until the end of the matrix is reached, and the pointer register will contain (for one scan) each successive bit number that miscompares.

The compare is always begun at the bit referred to by the contents of the pointer register *plus* one (1), unless this value exceeds the number of bits in the matrix, in which case the comparison begins with the first bit of each matrix. The contents of the pointer register will exceed the number of bits in the matrix if the compare is restarted after a successful compare (i.e., next scan, if A element contact remains closed), since the pointer register would contain the bit number of the last bit in the matrix plus one, and that value would be incremented by one at the start of the next compare; or the pointer register could be forced to any value by other logic lines. The contents of either matrix are *not* altered by this function; only the pointer and coil status change.

As an example, refer to Figure 79 and assume register 4372 contains a zero. When input 1056 is turned ON, the comparison will begin with bit one (1) and continue until either a miscompare or the end of the matrix is encountered. Assume that at bit 23, a miscompare occurs; either a one (1) is in the B element matrix and a zero (0) in the D element matrix or vice-versa. The coil will be energized and the scan continued with the next line of logic; the pointer register will retain the value 23.

If input 1056 remains ON, and the contents of register 4372 has not been altered, on the next scan the comparison will be restarted from where it was last terminated, by comparing bit 24. If the second miscompare was encountered at bit 56, for example, the compare will be terminated a second time, the coil will remain energized, and register 4372 will contain the value 56.

Note that the location of the first miscompare is lost unless it is copied to another location (e.g., register-to-table move, continuous, controlled by the compare coil) before the line is solved again. If no miscompares are located, register 4372 will contain the value 81 and the coil will be OFF.

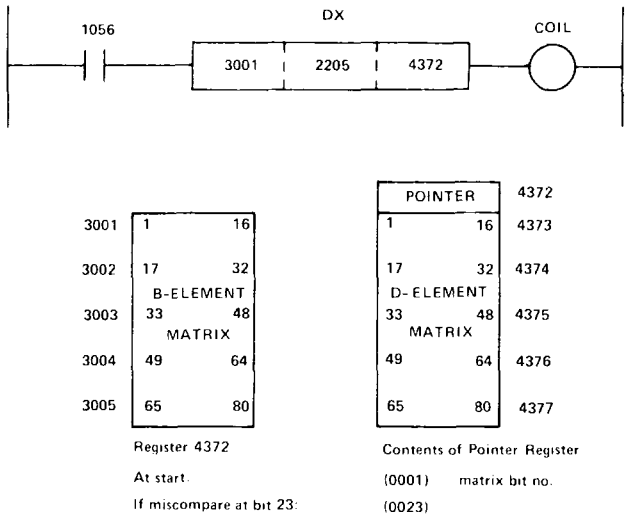

Figure 79. Sample Matrix Compare

On the next scan, assuming input 1056 is still energized, register 4372 will be incremented to 82 and, since this exceeds the size of the matrix as defined by the functional code, it will be reset to one (1) and a comparison will begin again at the first bit.

Note that, in general, either matrix of the compare can be referred to by either the B or D element; however, since the holding register whose value this code alters is associated with the D element, input registers must be referred to only by the B element.

If the last bit of the matrices miscompares, the next compare will be accomplished starting at the beginning of the matrices. Thus the end of the matrices (coil OFF) will not be detected between compares. Using the previous example, if only bit 80 miscompares, the holding register will always contain an 80 and the coil will be ON; the coil will not oscillate between ON (miscompare) and OFF (end of matrix). A calculate line can be used to detect when the pointer is equal to or greater than the number of bits in the matrix; this indicates the compare is at the end of the matrix.

If all bits agree, the entire matrices will be compared (monitored) every scan that the A element is closed regardless of the length of the matrices (maximum 1584 points).

23XX — Clear Bit or Sense Bit

NOTE

The matrix clear and set operations are useful to build matrices within the Controller as well as examine individual bits of any matrix and provide a coil reference for use as a control element in relay symbology.

This code, when the A element is closed, causes a single bit in a matrix to be cleared to zero (0) regardless of its previous value (either a 1 or a 0). The bit number is contained in the B element register, the matrix length in registers is defined by XX of the functional code, and the specific matrix location starts with the D element register.

The operation is performed continuously upon closure of the A element contact; a series of bits can be cleared by changing the value in the B element register with a table-to-register move or by using an input register in the B element. If the A element contact is NOT closed, this line will sense the state of the bit referred to by the contents of the B element register, but not alter it. The coil will be ON if it is a one (1) and OFF it it is a zero (0); the coil will *always* reflect the status of the B element bit, regardless of the state of the A element contact.

The coil status prior to closure of the A element contact will be that of the bit to be operated on and, after closure of the A element contact, the coil will be OFF (since the bit is to be cleared). When utilized as a sense line, an input register can be placed in the D element, since no alteration of this register is required. If an input register is utilized in the D element, closure of the A element contact will have no affect on either the coil status or the matrix contents. The B element register refers just to itself, and the D element register to XX registers — the size of the matrix. As an example, refer to Figure 80.

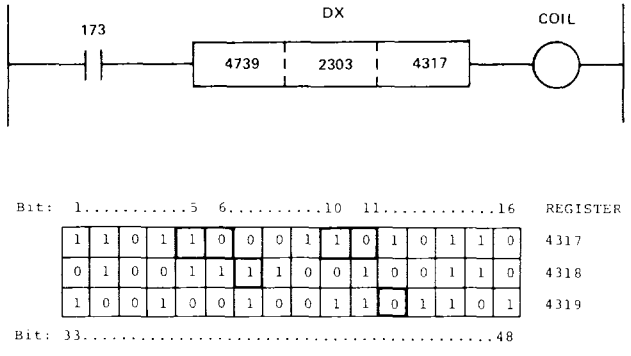

Figure 80. Sample Matrix Clear Line

Assume that line 173 is OFF and register 4739 contains the value 5. The coil will be ON, sensing the one (1) in the matrix at bit 5. When line 173 is ON, this bit in register 4317 will be cleared to zero (0) and the coil will be OFF. If the value in register 4739 is changed to 10, the tenth bit will be cleared to zero (0).

CAUTION

If an input register is to be used in the B element and is connected to thumbwheel switches, intermediate bits between 5 and 10 could be cleared if the A element contact remains closed while the input register is changed.

If the A element contact is referenced to a null data line, an input that is not utilized, or to an AND or OR logic line (i.e., some reference that will never be energized), this function can be used as a sense line. If line 173 of Figure 80 is an OR logic line (DX line with a 21XX functional code), and register 4739 contains a 23, the coil will be ON and no clearing operation is possible; if register 4739 contains a 44, the coil will be OFF.

If the contents of the B element register exceeds the number of bits in the matrix as defined by the functional code (or is zero), no operation is performed and the coil will be OFF regardless of the state of the A element contact.

24XX — Sense Bit or Set Bit

This code, when the A element is closed, causes a single bit in a matrix to be set to one (1) regardless of its prior value. Its operation is very similar to the 23XX clear function discussed above, except that the bits are set to one (1) in lieu of zero (0). Either functional codes, 23XX or 24XX, provide exactly the same sense functions.

For example, assume the functional code of Figure 80 is 2403, line 173 is OFF, and register 4739 contains the value 6. Initially, the coil will be OFF reflecting the status of the referenced bit. When line 173 becomes valid, bit 6 in register 4317 will be set to one (1) and the coil energized, again reflecting the status of the bit. If the value in register 4739 is changed to 11, bit 11 in register 4317 will also be set to one (1).

If the A element is referenced to a line or input that will not be energized, and register 4739 contains a 23, the coil will be ON and no setting operation is possible; if register 4739 contains a 44, the coil will be OFF.

25XX — Matrix Complement

NOTE

The complement capability is useful to prepare data for comparison when the data is exactly reversed from the source data (e.g., output matrix driving valves, compared to limitswitch inputs from valves, which close when the valve is de-energized).

This code, when the A element is closed, causes the contents of the matrix referred to in the B element to be complemented and the result placed in the D element matrix. Every bit in the B element matrix has its value reversed, zeros replaced with ones and ones replaced with zeros, and the result placed in the D element matrix. The entire matrix is complemented every scan that the A element is closed (passing power). The contents of the B element matrix is retained and the previous contents of the D element matrix is destroyed and replaced with the result of the complement operation. The coil has no significance, and is OFF in all cases.

It is possible to specify the same matrix in both B and D elements, resulting in the complement replacing the source matrix. However, if this is desired, the A element should be referenced to a one-shot to prevent the matrix from being complemented a second time during the next scan. As an example, refer to Figure 81.

The B element refers to three consecutive registers (4134-4136) as does the D element (4479-4481), since the function code has defined each

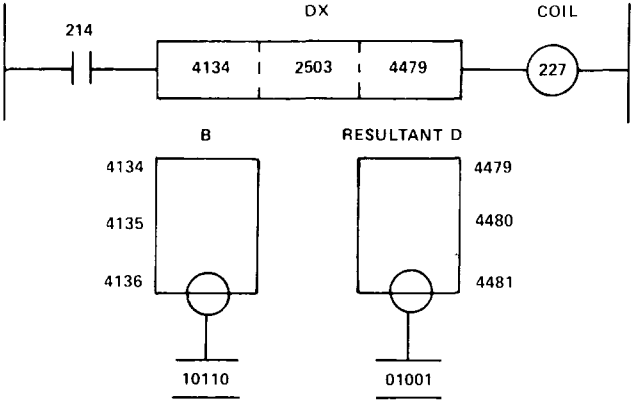

Figure 81. Sample Matrix Complement

matrix as three registers in length. In all cases, matrix COMPLEMENT will be accomplished on matrices of equal length. For this example, on the first scan that line 214 is energized, all bits in registers 4134-4136 will be complemented, and the result placed in registers 4479-4481. The contents in registers 4134-4136 will not be altered and the previous contents in 4479-4481 will be destroyed and replaced with the result of the complement operation. On subsequent scans, the complement operation will be performed as long as the A element passes power. As long as the B element matrix is not altered, the results in the D element matrix will not be altered from its initial value. Note the illustrated effect of five bits in register 4136 and their corresponding results in register 4481.

If the D element reference was changed to 4134 (same matrix referred to in both the B and D matrices), on the first scan 214 was energized, the 10110 in register 4136 will be replaced with 01001. On the next scan, assuming 214 remains energized, 4136 is again complemented, and 10110 (the original value) is placed in register 4136. Thus, every scan, the bits in each register will oscillate between ones and zeros; the result when 214 is de-energized cannot be guaranteed. To prevent this oscillation when using the same matrix in the B and D elements, the A element should be referred to a one-shot that is valid for exactly one scan. In all cases, the coil will not be energized.

26XX — Logical Exclusive OR of Two Matrices

NOTE

The exclusive OR is useful to compare two matrices in one scan and allow specific identification of those bits that miscompared at some future time. After an exclusive OR is performed, a one indicates a miscompare and a zero a compare. These miscompares can be located by comparing (22XX code) the result with a zero matrix (a matrix containing all zeros).

This code, when the A element is closed, causes the contents of the matrix referred to in the B element to be logically ORed (exclusively) with the contents of the matrix referred to in the D element, and the result stored in the D element matrix. This function operates similar to code 20XX and 21XX discussed previously, except the logical Exclusive OR is performed between the two matrices. Since an Exclusive OR is accomplished every scan the A element remains closed, oscillations can occur wherever the B matrix bits are ones; thus the A element references should *always* be to a one-shot. As an example, refer to Figure 82.

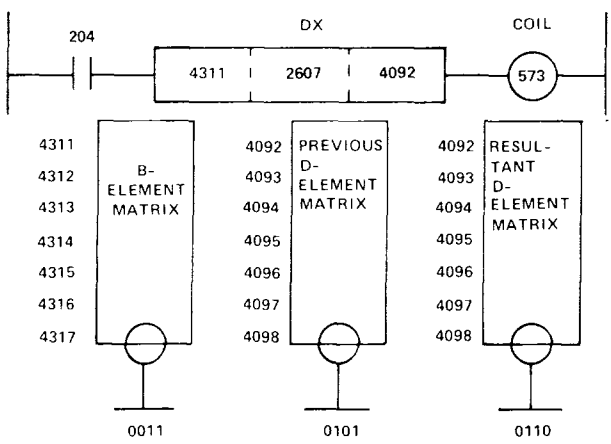

Figure 82. Sample Matrix Exclusive OR

The B element refers to seven consecutive registers (4311-4317) as does the D element (4092-4098), since the function code has defined each matrix as seven registers in length. In all cases, matrix Exclusive OR will be accomplished on matrices of equal length. For this example, on the first scan that line 204 is energized, a bit-by-bit Exclusive OR will be performed between registers 4311-4317 and registers 4092-4098. Note the specific example of four bits provided in registers 4317 and 4098. The result is a one only if there is a one in either the B element matrix or the D element matrix, but not (excluding) if they are both ones.

If line 204 remains energized on the next scan, another Exclusive OR is performed between the retained (not changed) contents of the B element matrix and the new D element matrix. The four bits in register 4097 will become 0101, the original values; to prevent this, line 204 should be a one-shot valid for exactly one scan. The coil on line 573 has no significance and is never energized.

27XX — Rotate Left

NOTE

The Rotate Left and Rotate Right are both useful to form single-bit retentive shift registers or to shift data as required for reformatting.

This code, when the A element is closed, causes the contents of the matrix referred to by the B element to be rotated one position to the left and the result placed in the D element matrix. All bits are shifted down one position, e.g., the status of bit 30 is placed in bit 29, 29 in 28, 17 in 16, 2 in 1, etc. The status of bit 1 is carried around (true rotate not just shift) and placed in the last bit of the matrix. The coil reflects the status of this bit carried around; the coil will be ON if the bit is a one, and OFF if the bit is a zero. For example, refer to Figure 83.

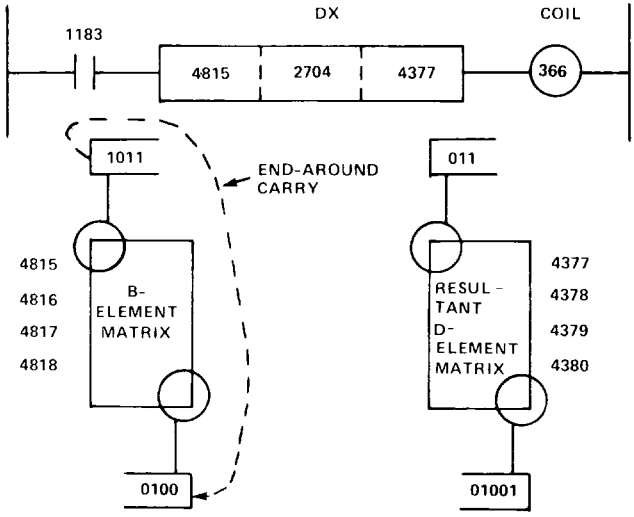

Figure 83. Sample Matrix Rotate Left

The B element refers to four consecutive registers (4815-4818) as does the D element (4377-4380), since the function code has defined each matrix as four registers in length. In all cases, matrix Rotate will utilize matrices of equal length. For this example, during the first scan input 1183 is energized, the contents of registers 4815-4818 will be rotated one position to the left and the result placed in registers 4377-4380; the contents of the B element matrix is not altered and the previous contents of the D element matrix is destroyed and replaced by the result of the rotate. The coil will be ON since the end-around carry bit was a one.

As long as input 1183 remains closed, the rotate will be performed and the coil status not altered, since on every scan the rotate goes back to the B matrix for its initial data. Note the specific status of bits 1-4 and 61-64 of the B element matrix. Bits 61-64 are shifted down to 60-63 and placed in the D element matrix; bit 1 of the B element matrix is placed in bit 64 of the D element matrix; bits 2-4 of the B element matrix are placed in bits 1-3 of the D element matrix.

28XX — Rotate Right

This code, when the A element is closed, causes the contents of the matrix referred to in the B element to be rotated one position to the right and the result placed in the D element matrix. This function operates similar to code 27XX except that the bits are shifted up, status of bit 1 into bit 2, 2 into 3, 16 into 17, 25 into 26, 32 into 33, etc., and the last bit end-around carried to bit 1. The coil is still set to the status of this end-around carried bit.

As an example, assume the function code in Figure 83 is 2804; the only difference in operation would be the specific results. Bits 2-5 of the resultant D element matrix would be 1011; bits 62-64 would be 010; bit 1 would be zero (obtained from bit 64 of the B element matrix); and the coil would be OFF.

A continuous rotate can also be obtained if the same matrix is referred to in the B and D elements of a 27XX or 28XX code. A calculate line can be used to control how long the rotate is performed, and thus how many bits are rotated.

NOTE

Function Code 29XX (384A and 384B only) is discussed in Section 3.7.

Additional 30XX References

Some of the executives that provide matrix capabilities also provide a capability referred to as Additional 30XX References. Input registers normally start at channel III and up to 16 are provided with any executive (3001-3016). However, it is possible to develop and executive that provides input registers in all four channels, maximum 32 input registers (3001-3032). Thus these additional references start at 3033, which has no possible meaning relative to the hardware I/O.

The contents of 3033 are controlled by the coil status of logic lines 1-16. If the coil of line one is ON, bit one in register 3033 will be a 1; if coil one is OFF, bit one will be a 0. The same technique is used to determine that status of bits 2-16 from coils 2-16. These additional references continue at 16 lines per register, until the last logic line is used. For example, if the MOPS provides 608 lines, the additional references used for coil status will be 3033-3070; bit 16 of register 3070 is controlled by line 608, the WDT line.

With additional references, the logic to drive discrete outputs (e.g., lines 1-256) can be built with simple relay logic; but their status is also available in registers 3033-3048 for monitoring by matrix compare lines. The additional references bridge the gap between discretes and registers.

The next additional reference after the logic lines (e.g., 3071) reflects the status of inputs 1001-1016. If input 1001 is energized, bit one in register 3071 is a 1; if 1001 is de-energized, bit one is a 0. These references continue at 16 inputs per reference until the limit of discrete inputs is reached

(e.g., 3071-3086 if 256 inputs are provided). Discrete inputs are thus available as discrete references and as a bit in a register for matrix operations. For an example of additional references, see Table 14 which assumes 608 lines and 256 discrete inputs.

EXAMPLE V — Monitoring of Discrete Inputs Versus Outputs

Assume discrete outputs 81-160 are used to drive 80 solenoid valves with standard relay logic. On each valve is a limitswitch that closes when the valve goes to the energized position. A monitoring system is required that constantly compares the outputs against the limitswitches to ensure all energized valves go to their proper position and that, when de-energized, the valves do not remain in the energized positions.

Table 14. Example Additional 30XX References (608 Lines, 256 Discrete Inputs)

Register	Contents Controlled by Lines	Register	Contents Controlled by Lines	Register	Contents Controlled by Inputs
3033	1-16	3052	305-320	3071	1001-1016
3034	17-32	3053	321-336	3072	1017-1032
3035	33-48	3054	337-352	3073	1033-1048
3036	49-64	3055	353-368	3074	1049-1064
3037	65-80	3056	369-384	3075	1065-1080
3038	81-96	3057	385-400	3076	1081-1096
3039	97-112	3058	401-416	3077	1097-1112
3040	113-128	3059	417-432	3078	1113-1128
3041	129-144	3060	433-448	3079	1129-1144
3042	145-160	3061	449-464	3080	1145-1160
3043	161-176	3062	465-480	3081	1161-1176
3044	177-192	3063	481-496	3082	1177-1192
3045	193-208	3064	497-512	3083	1193-1208
3046	209-224	3065	513-528	3084	1209-1224
3047	225-240	3066	529-544	3085	1225-1240
3048	241-256	3067	545-560	3086	1241-1256
3049	257-272	3068	561-576		
3050	273-288	3069	577-592		
3051	289-304	3070	593-608		

Assuming the limitswitches are connected to discrete inputs 1129-1208 for use in the relay logic, Figure 84 is an example of the logic that will perform this monitoring. Referring to Table 14, discrete outputs 81-160 control the contents of registers 3038-3042 and discrete inputs 1129-1208 control registers 3079-3083. Line 325 clears all five registers in the compare matrix (4487-4491) to zero every scan by ANDing them with zero. The zero matrix is loaded initially with zero, and never altered; thus it always contains known zeros unless action is taken to alter its contents. Line 326 moves the contents of registers 3079-3083 (inputs 1129-1208) into the compare matrix by ORing them with a known zero placed in the matrix by line 325. This is required since an input register (3079-3038) cannot be placed in the D element of a DX compare line. Line 327 does the actual comparing between the outputs (3038-3042) and the inputs (loaded into 4487-4491). Coil 327 will be ON if any miscompares are detected and the contents of 4486 will be the bit number (valve) that caused the miscompare.

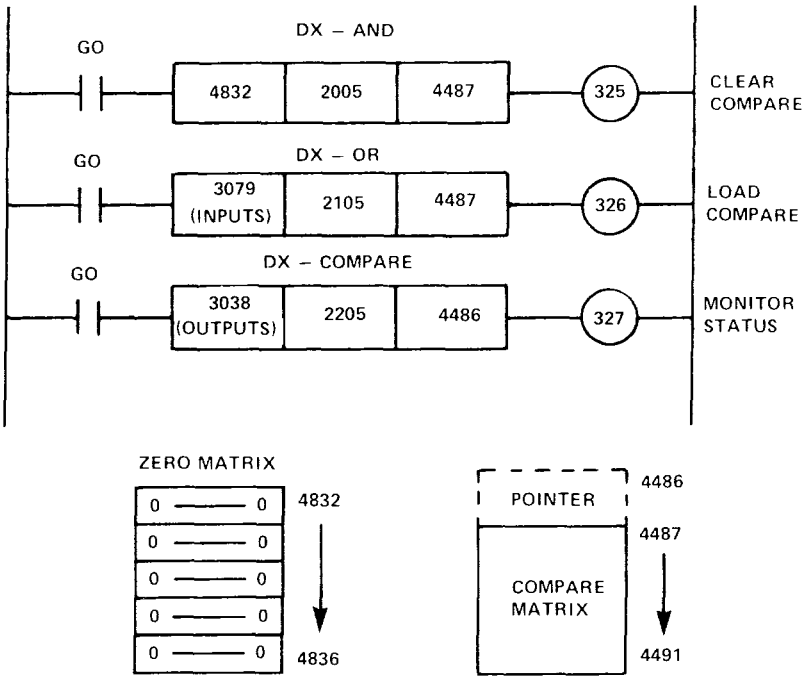

Figure 84. Example: Monitoring of Discretes

Since the matrix operations are accomplished every scan in their entirety, all inputs are compared once a scan until a miscompare is detected; one miscompare is detected every scan. The GO contact controls when the compare is enabled and can be a normally-closed contact reference to null data if constant comparisons are required. Note that the comparison is not affected by whether the output is energized or de-energized; if it does not compare (i.e., energized output and open limitswitch, or de-energized output and closed limitswitch) coil 327 will be ON. Timers can be incorporated such that a miscompare must be detected for a continuous time period (e.g., 0.5 seconds, 2.0 seconds, etc.) before action is taken. If more or less valves or other devices are to be monitored, the only action required is to change the DX function codes to adjust the size of the matrices; these three logic lines (9 words of core memory) perform the monitoring function for one to 99 registers of data (16 to 1584 devices).

If a second limitswitch is incorporated on each valve that is closed when the solenoid valve is de-energized, and open when it is energized, another comparison can be made to detect valves that "hang up" between limits. Lines 325 and 326 are duplicated with a single complement line (25XX) that will take these limitswitch inputs, complement their status, and place the result into a COMPARE matrix. A complement is required so that direct comparisons can be made to the output lines. For example, if an output is energized, a one is placed into the matrix starting at 3038; however, since these new limitswitches are open when the valve is energized (opposite of first limitswitches), a zero is placed into their input matrix. Direct comparisons will result in a miscompare whenever the valves are operating properly. To correct this, the inputs from these new limitswitches are first complemented, then direct comparisons are made and miscompares only occur when malfunctions are detected.

Various actions are possible when a miscompare is detected, as with a fault diagnostic system. The malfunction can be printed on the P500 Printer; operator alerted by lights, bells, displays, etc.; process shut down or next step not allowed; machine forced to a safe condition; etc. Which action is selected depends upon the specific application and system design requirements. Two items are available to support whatever action is taken; the compare coil (e.g., line 327) will be energized and the pointer (e.g., register 4486) will contain the bit number (related to the specific valve) that caused the miscompare.

EXAMPLE VI — Matrix Retentive Shift Register

The matrix rotate capability can be utilized to develop a retentive shift register, driven by basically three logic lines (9 words of core memory), up to 1584 stages. Either rotate function (27XX-left or 28XX-right) can be used with equal efficiency; Figure 85 illustrates a shift register built using the 2803 DX line to form a 48-station shift register. Line 497 shifts the contents of the three-register matrix, which starts at register 4757, one position to the right and places the result back into the same registers. To ensure that only one shift is made, the shift input should be a one-shot. Since the rotate function also takes the status of the last bit (e.g., 48th bit) and places it into the first bit, line 498 will always clear the first bit whenever the end-around carry is a one (coil 497 energized). Line 499 places one in the first stage of the shift register whenever commanded by the input signal. If a one is not placed in this stage by the time the next shift is performed, a zero is shifted into the shift register.

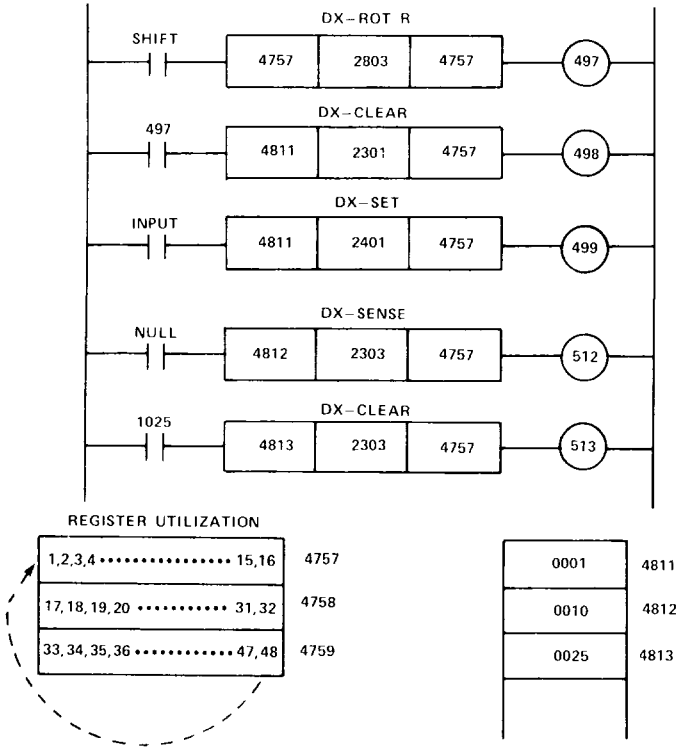

Figure 85. Example: Matrix Shift Register

Coil 497 can represent the output of the shift register if this output is required for only one scan. Otherwise, a sense line similar to line 512 can be utilized for continuous output status. In addition, a sense line can be provided to obtain the status of any stage of the shift register. Line 512 currently provides the status of the tenth stage since register 4812 contains a ten; if register 4812 contents are altered, another stage's status is obtained. Coil 512 is ON for a one in the stage and OFF for a zero. Parallel entry of any stage can also be provided with clear and set lines similar to line 513. This line is currently designed to clear stage number 25 when input 1025 is energized, since register 4813 contains a 25; a similar line with a 2403 code can be utilized to set any stage to a one. It is to be noted that lines 497-499 are the basic drive for this shift register; lines similar to 512 and 513 can be added if the application requires. If more than 48 stages are required, the DX code in line 497 is increased to provide 16 stages per additional register; a DX code of 2899 provides 1584 stages. Since the shift register is built in holding registers, it is retentive upon power failure.

3.5.3 Extended Arithmetic (Group 3YXX)

These codes provide the extended arithmetic capability so that multiplication, division, and other special functions can be accomplished. The resultant product of a multiply operation and the dividend prior to a divide operation are stored and referred to within memory as double-precision. This implies that the magnitude may exceed the capability of one register and thus two registers are allotted to store this value; one register contains the four high-order digits (tens of millions through tens of thousands) and the next register in sequence contains the four low-order digits (thousands through units). This concept is important since the result of a multiply must be evaluated from the contents of both registers, and preparation for division must include loading two registers.

When either a multiply or divide operation is performed, the coil will come ON to indicate successful accomplishment. The following are valid functional codes currently available with the extended arithmetic capability. All 184 executives with this capability are provided with the four multiply/divide codes; only selected executives also provide the special functions (see Table 12). All 384 controllers are provided with the four multiply/divide codes; the 384A and 384B are provided with all eight codes (Basic plus Special).

Basic		Special	
3000	General Multiply	34XX	PID (Seconds)
3100	General Divide	35XX	PID (Minutes)
32XX	Multiply by XX	36XX	SORT (Ascending)
33XX	Divide by XX	37XX	SORT (Descending)

The multiply/divide functions are performed only once on closure of the A element contact and are completed prior to commencing operation on the next line of logic.

3000 — General Multiply

This code causes the contents of the B element register to be multiplied by the contents of the register immediately following the B element register, and the resultant product deposited as a double-precision number in the D element registers. The multiply is accomplished only on transition of the A element from OFF to ON; the coil will come on at the completion of this line. Both the B element and the D element refer to two consecutive registers; their contents are entered and read as four BCD digits.

For example, refer to Figure 86 and assume input 1035 is not energized.

If the input registers contain the values 976 and 42 as illustrated, on the first scan that input 1035 is energized, the product (40,992) of 976 and 42

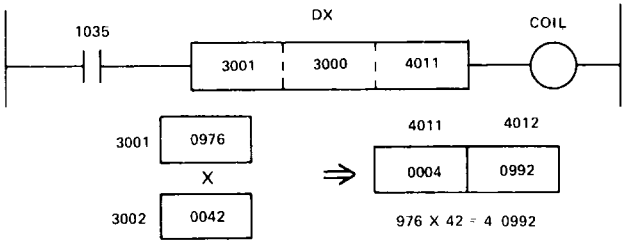

Figure 86. Sample Multiply

will be deposited into registers 4011 and 4012 as a double-precision number, and the coil energized. Note that the resulting product is separated with the four low-order digits being placed in the low-order product register (D element reference plus one) and the high-order digits (with leading zeros) placed in the high-order product register (D element reference). If the input registers change their values, input 1035 must be cycled ON-OFF-ON to produce a new product. The coil will be ON after the product is available and will remain ON until input 1035 is de-energized. Either or both B element registers can be zero.

NOTE

There are no conditions under which the coil of a multiply will not come ON as long as the A element contact is closed (passing power).

3100 — General Divide

This code causes the contents of the B element registers (double-precision) to be divided by the contents of the register referred to by the B element plus 2. The resultant quotient is stored in the D element register, with any remainder stored in the D element register plus 1. The divide operation is performed only when the A element contact is closed; its results are available prior to commencing the following line of logic.

The B element register refers to three consecutive registers and the D element register to two consecutive registers; their contents are entered and displayed as four BCD digits. The coil will come ON only if the operation is successful. Division by zero (0) or a resultant quotient too large (greater than 9999) to be stored in one register are both valid reasons for the coil to remain OFF; if the coil is not ON, the contents of any register will not be altered.

For example, refer to Figure 87 and assume line 413 is not ON.

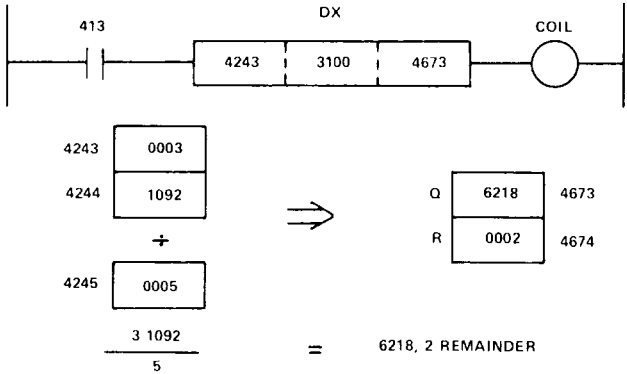

Figure 87. Sample Divide

If registers 4243 through 4245 are preloaded as indicated, on the first scan that line 413 is on, the double-precision number (31,092) in registers 4243 and 4244 will be divided by the contents (5) in register 4245. The resultant quotient (6218) will be stored in register 4673 and any remainder (2) in register 4674; the coil will be ON until line 413 is turned OFF.

If the divisor in register 4245 was a 3, the resultant quotient would be 10,364 which is too large to fit into one register; thus the coil would be OFF and the contents of all registers retained. If registers 4243-4245 change their values, a new division will take place only after line 413 is cycled ON-OFF-ON.

32XX — Fixed Multiply

This code causes the contents of the B element register to be multipied by a fixed quantity (XX), which is part of the functional code, and the resultant product is stored as a double-precision number in the registers referred to by the D element reference. The operation of this code is very similar in all respects (including coil status) to the general multiply functional code 3000 discussed previously, except the second B element register is replaced by the last two digits of the functional code.

Two limitations should be clearly understood; the multiplier must be only two digits in magnitude (99 or less excluding zero) and the multiplier cannot be altered by other logic operations. If either of these features are required, the general multiply code (3000) must be used.

For example, if the functional code of Figure 86 was 3242, the same result would be stored in registers 4011 and 4012; register 3002 is not required and can be used for another operation. Note that, for the 32XX code, the B element refers to only one register and the D element to two registers.

33XX — Fixed Divide

This code causes the double-precision contents of the B element registers to be divided by a fixed quantity (XX) which is part of the functional code, and the quotient stored in the D element register, with the remainder in the D element register plus 1. The operation of this code is very similar in all respects (including coil status) to the general divide functional code (3100) discussed above, except that the third B element register is replaced by the last two digits of the functional code.

There are two limitations. The divisor must be magnitude 99 or less, but not zero, and it cannot be altered by other logic lines. If either or both of these features is required, the general divide instruction (3100) must be used. For example, if the functional code of Figure 87 was 3305, the same results would be stored in registers 4673 and 4674; register 4245 is not required. Note that, for the 33XX code, both the B and D element refer to two registers.

3400 — PID (Proportional, Integral, Derivative) Control

This type of control is normally used for temperature, pressure, pH, etc., and other variables usually controlled by analog controls. In addition, positioning and servo loops are also ideal applications for this type of control.

The controller solves the following equation to provide the control action desired:

$$P = K_1 e + K_2 \int_0^t e\,dt + K_3 \frac{de}{dt} + K_4$$

where,

P = Output

K_1 = Gain (00.01—99.99) — proportional band is equal to $1/K_1$

K_2 = integral rate, or reset rate (0.001—9.999) repeats per *second*

K_3 = derivative time or the rate time (0001—9999) *seconds*

K_4 = Initial starting point

e = error = input minus set point

PROGRAMMING PID CONTROL

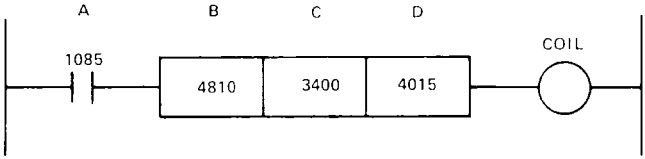

Figure 88. Sample PID Logic Line

A: Starts control when closes.

B: First register in input table (see below for input data).

C: 3400 is function code for PID control.

NOTE

Function codes 3401, 3500, and 3501 (PID-384A and 384B only) are discussed in section 3.7.

D: First register in output table (see below for output data table).

Input Table Description

Register Number (Typical)	Value in Register (Typical)	Description
4810	0517	Input (i.e., could be 517°F)
4811	0525	Set Point (i.e., could be 525°F)
4812	0025	K_1 — Gain
4813	0.050	K_2 — integral rate in repeats per *second*
4814	0000	K_3 — derivative time in *seconds* (derivative action not wanted so zeros used).
4815	0550	High limit of 550°F
4816	0500	Low limit of 500°F

NOTE

Seven sequential registers starting with the register used in the B element are set aside for PID operations when code 3400 is used. Therefore, do not use those register numbers for other functions.

Output Table (6 Registers)

Register 4015 contains the output values which can be used to position control valves or servos, etc. This output is a four-digit numerical quantity that will vary between limits established by 4815 and 4816 in this example. Maximum range of output is 0000 to 9999.

NOTE

Five register numbers following the register used in D (in this example 4016—4020) are used for internal use in PID control in the controller and cannot be used anywhere else in your program.

Additional Application Notes

1. This PID control is *direct acting*. If *reverse acting* is desired, a (B−C) calculate line can be used.
2. *Profile Control* (predetermine a non linear set point program): Can be very easily done by using a table to register DX move, storing up to 99 different set points. Linear ramp control of set points can be done in the same manner.
3. *Cascade Control:* The output of one PID line in the Controller can be the input to another PID line within the Controller by using a (B+C) calculate line to move the output value in the output register (in this example 4015) to the input register in another PID line.
4. *Adaptive Control:* The gain, reset rate and derivative time can each or all be changed at will by (B+C) calculate lines or DX transfer operations. Therefore, adaptive control can be used with no extra hardware by simply using additional programs in the controller.
5. *Feed Forward Control:* Can be accomplished very similar to adaptive control discussed above with additional programmed techniques to do the arithmetics required.
6. *High-Low Limits:* The coil of the PID line will be on when the limits are exceeded.

NOTE

Function codes 36XX and 37XX (384A and 384B only) are discussed in section 3.7.

3.5.4 Entering a Data Transfer Line

1. Set the Line Number on the LINE NUMBER switches.
2. Put the Controller MEMORY PROTECT switch in the OFF position.
3. Press the DATA TRANSFER pushbutton. It will light and all other LINE TYPE lights will go out.
4. Press the A element pushbutton. The A element lamp will light and the B, C, and D lights will be OFF.
5. Set the REFERENCE NUMBER switches to the line number, input number, or latch that is to operate the contact in the A position.
6. Press either the Normally-Open or Normally-Closed Series ELEMENT TYPE pushbutton. The ELEMENT TYPE pushbutton that is pressed will light and the REFERENCE DISPLAY will show the number that has been entered.

NOTE

On 184 controllers, the Data Transfer lines can be programmed in any element order. However, the 384 controller requires the Function Code (C element) to be entered prior to programming either the B or D elements to establish proper error checking. This requirement is for DX lines only.

7. Select the B element position and enter on the REFERENCE NUMBER thumbwheels the register which is the location of the SOURCE data. Press any ELEMENT TYPE pushbutton. The REFERENCE DISPLAY will show the register that has been entered.
8. Select the C element position and enter on the REFERENCE NUMBER thumbwheels the FUNCTION CODE. Press any ELEMENT TYPE pushbutton. The REFERENCE DISPLAY will show the code that has been entered.
9. Select D element position and enter on REFERENCE NUMBER thumbwheels the register which is the DESTINATION of the data.
10. Press any ELEMENT TYPE pushbuttons. The REFERENCE DISPLAY will show the register that has been entered.
11. If the DISABLE pushbutton is lit, and not specifically desired, press it to turn it OFF.

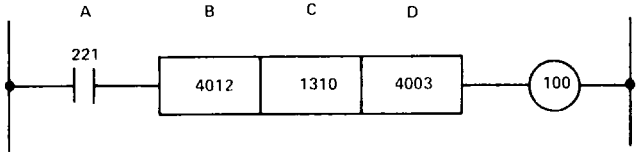

Figure 89. Data Transfer (Move) Line

1. Set Line Number switches to 0100.
2. Press DATA TRANSFER pushbutton.
3. Press A element pushbutton.
4. Set REFERENCE NUMBER switches to 0221.
5. Press Normally-Open Series ELEMENT TYPE pushbutton.
6. Press B element pushbutton.
7. Set REFERENCE NUMBER switches to 4012. Press any ELEMENT TYPE pushbutton.
8. Repeat steps 6 and 7 for the C element (1310), and D element (4003).
9. If the DISABLE Light is lit, and not specifically desired, press it to turn it OFF.

3.6 P500 PRINTER/D285 DISPLAY UNIT

3.6.1 Introduction

The P500 Printer (see Figure 90) is an industrial-environment hardcopy printer designed to provide data on the plant floor. It is capable of printing out management information such as number of parts produced, up times, efficiencies, recipe contents, etc.; or operator information such as error messages, batch completed notation, manual operations required, etc. The P500 Printer is not designed to document the user's program with ladder diagrams; this support is available from the Service Center (see 4.2.2).

The paper used in the P500 is pressure sensitive; thus the printer does not require carbon paper or ink and the associated maintenance problems. The paper is 3½ inches wide, each page approximately 5½ inches long— a convenient size to place in shirt pockets. Each page can contain up to 20 lines, each line 21 characters (see Figure 91). A summary of P500 specifications are provided in Table 15.