Advance Instruction Set Of PLC

Different Comparison Instructions

Table 4.1 Different Comparison Instructions

Mnemonic	Name	Purpose	Address
EQU	Equal to	To test whether one value is equal to another value.	Source A must be a Word Address and Source B either constant or a Word Address. Negative integers are stored in 2's complement form.
NEQ	Not equal	To test whether one value is not equal to a second value.	Source A must be a Word Address and Source B either constant or a Word Address. Negative integers are stored in 2's complement form.
LES	Less than	To test whether one value is less than a second value.	Source A must be a Word Address and Source B either constant or a Word Address.
LEQ	Less than or equal to	To test whether one value is less than or equal to a second value.	Source A must be a Word Address and Source B either constant or a Word Address.
GRT	Greater than	To test whether one value is greater than another	Source A must be a Word Address and Source B either constant or a Word Address.
GEQ	Greater than or equal to	To test whether one value is greater than or equal to a second value.	Source A must be a Word Address and Source B either constant or a Word Address.
MEQ	Masked comparison for equal	To test a portion of two values to see whether they are equal. Compare a 16-bit data of a source address to 16 bit data at a reference address through a mask.	Source is the address of the value th user wants to compare. Mask is the address of the mask through which the instruction moves data. Mask ca be a Hex value i.e. constant. Compare is an integer value or address of reference.
LIM	limit test	To test whether one value is within the limit range of	Low limit, test and high limit values can be Word Address or constants restricted to the following

4.3 Discussions on Comparison Instructions

4.3.1 "EQUAL" or "EQU" Instruction

The EQU instruction is used to test whether two values are equal. If source A and source B are equal, the instruction is logically true. If these values are not equal, the instruction is logically false. Source A must be a Word Address. Source B can be either a Word Address or a constant. Negative integers are stored in 2's complement form. The functional block of the instruction is shown in Fig. 4.1.

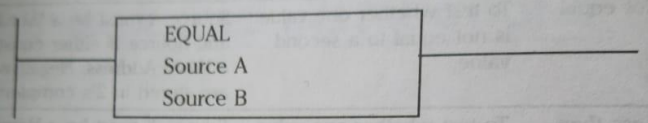

Fig. 4.1 Function Block of EQU Instruction

4.3.2 "NOT EQUAL" or "NEQ" Instruction

The NEQ instruction is used to test whether two values are not equal. If source A and source B are not equal, the instruction is logically true. If the two values are equal, this instruction is logically false. The functional block of the instruction is shown in Fig. 4.2.

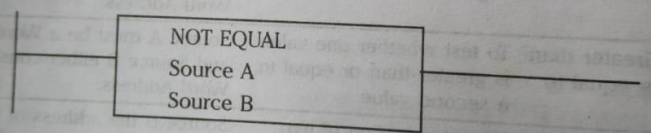

Fig. 4.2 Function Block of NEQ Instruction

4.3.3 "LESS THAN" or "LES" Instruction

The LES instruction is used to test whether the value of source A is less than the value of source B. If this condition is fulfilled, the instruction is logically true, if the value at source A is greater than or equal to the value of source B, the instruction is logically false. The functional block of the instruction is shown in Fig. 4.3.

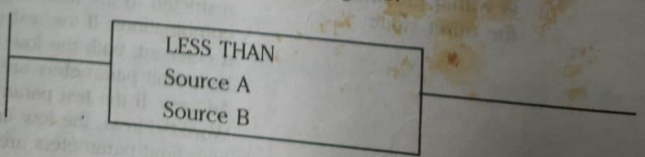

Fig. 4.3 Function Block of LES Instruction

4.3.7 "MASKED COMPARISON FOR EQUAL" or "MEQ" Instruction

The MEQ instruction is used to compare data of the source address to data of reference address. By using this instruction, portions of the data can be masked by a separ word.

The functional block of the instruction is shown in Fig. 4.7.

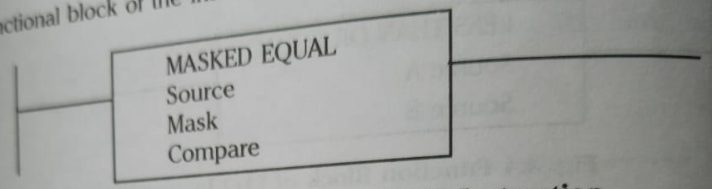

Fig. 4.7 Function Block of MEQ Instruction

4.3.8 "LIMIT TEST" or "LIM" Instruction

The LIM instruction is used to test whether certain values are within or outside a spec limit. The functional block of the instruction is shown in Fig. 4.8.

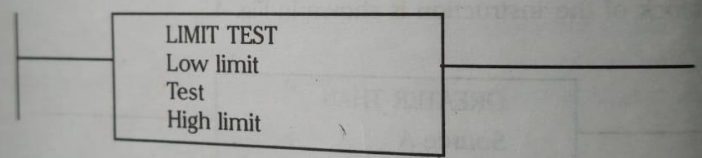

Fig. 4.8 Function Block of LIM Instruction

If the test value is equal to or greater than the low-limit and equal to or less than the limit, the instruction is true. The instruction is false if the test value is less than the low lim higher than the high limit. The condition is demonstrated in Fig. 4.9.

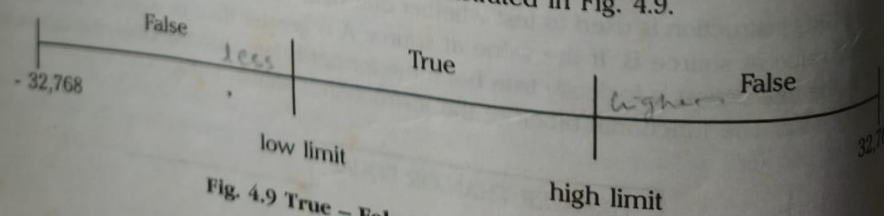

Fig. 4.9 True – False Status of Instruction

Problem 4.1: If the low limit is 5 and the high limit is 8, i.e., low limit < high limit, tabulate the instruction status.

The instruction status has been shown in Table 4.2.

Table 4.2 Instruction Status

Low limit	High limit	Instruction is true when the test value is between	Instruction is false when test value is
5	8	5 to 8	-32768 through 4 and 9 to 32767

Problem 4.2: If the low limit is 10 and the high limit is 6, i.e., low limit > high limit, tabulate the instruction status.

The condition is shown in Fig. 4.10 and tabulated in Table 4.3.

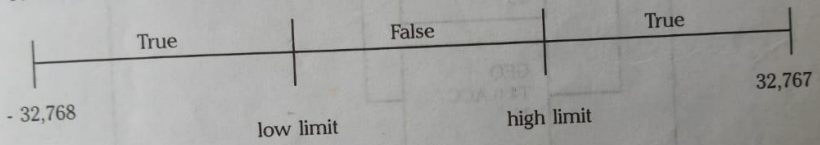

Fig. 4.10 True-False Status of Instruction

Table 4.3 Instruction Status

Low limit	High limit	Instruction is true when the test value is between	Instruction is false when test value is
10	6	-32768 through 10 6 through +32767	9, 8 and 7

Problem 4.3: Motor 1 (M1) starts as soon as the PLC starts. After 10 seconds, motor 1 goes off and motor 2 starts. After 5 seconds, M2 goes off and M3 starts. After another 10 seconds, M2 restarts and after 5 seconds it stops, and M1 starts, and the cycle is repeated. Prepare the logic diagram for the process.

To develop the ladder diagram, the following inputs and outputs are considered:

INPUT	OUTPUT
Start/stop = I:0/1	M1 = O:0/1; M2 = O/0/2; M3 = O:0/3

Based on the above inputs and outputs, the ladder diagram has been developed as shown in Fig. 4.11.

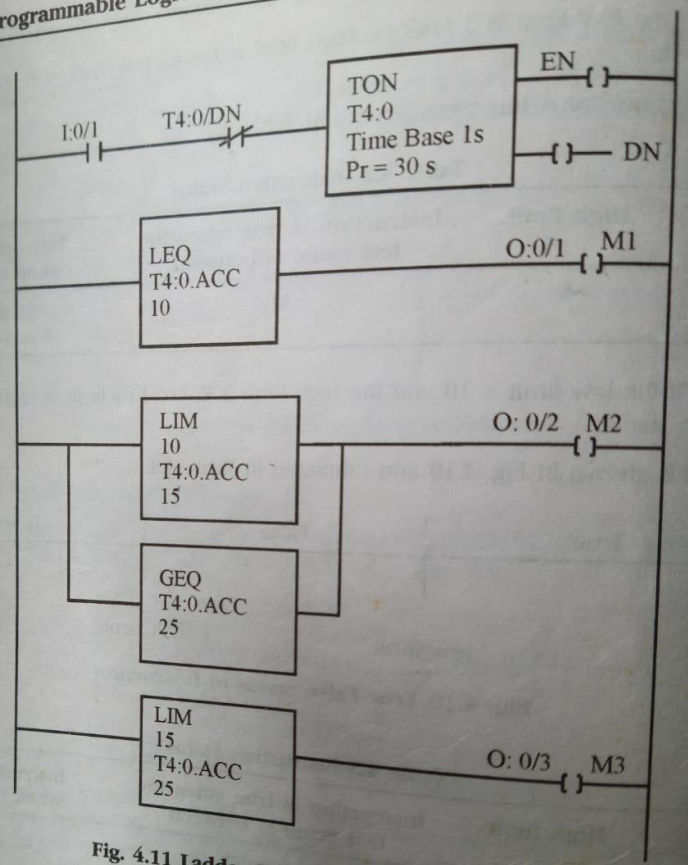

Fig. 4.11 Ladder Diagram for Example 4.3

4.4 Addressing Data Files

To deal with a data file, it is necessary to address data files for their proper identi
For the purpose of addressing, each data file type is identified by a letter called an "id
and a file number. The identifier and file numbers for different types of files are sh
Table 4.4.

Table 4.5 Addressing Format for Micrologic System

File type	Identifier	File number	Element number
Output	O	0	0-15 } for a p
Input	I	1	0-15
Status	S	2	0-32
Binary	B	3	0-31
Timer	T	4	0-39
Counter	C	5	0-31
Control	R	6	0-15
Integer	N	7	0-104

4.7 Different Addressing Types

Logical addresses can be assigned to instructions from the highest level (el lowest level (bit). The different addressing types have been discussed in the f sections.

4.7.1 Word within an Integer File

An example of a Word Address within an integer file is shown in Fig. 4.13

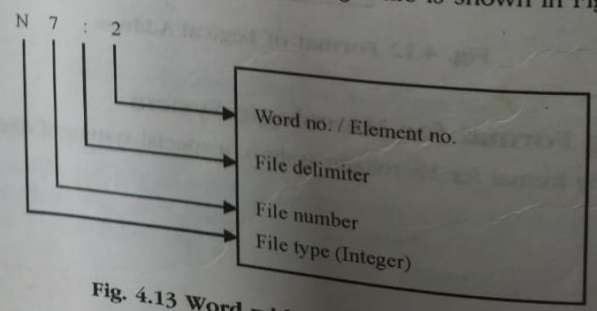

Fig. 4.13 Word within an Integer File

4.7.2 Bit within an Integer File

An example of a Bit

Table 4.7 Main Features of Logical Instructions

Instruction	Operation	Purpose	Address
And	AND	Value at Source A is ANDed bit by bit with that at Source B and the result is stored in the destination.	Source A and B can either be Word Address or constants but both cannot be constants simultaneously. The destination must be a Word Address.
Or	OR	Value at Source A is ORed bit by bit with that at Source B and the result is stored in the destination.	Source A and B can either be a Word Address or constants but both cannot be constants simultaneously. The destination must be a Word Address.
Exclusive Or	XOR	Value at Source A is XORed bit by bit with that at Source B and the result is stored in the destination.	Source A and B can either be Word Address or constants but both cannot be constants simultaneously. The destination must be a Word Address.
Not	NOT	Value at a Source is NOTed bit by bit and the result is stored in the destination.	Source and destination must be Word Address
Negate	NEG	This is used to change the sign of a data. Thus the output at destination is a 2's complement of the input at source.	Source and destination must be a Word Address.

4.9.2 Logic AND Instruction

The functional block diagram of the instruction is shown in Fig. 4.17, the truth table is given in Table 4.8, and the updates of the arithmetic status bits are presented in Table 4.9.

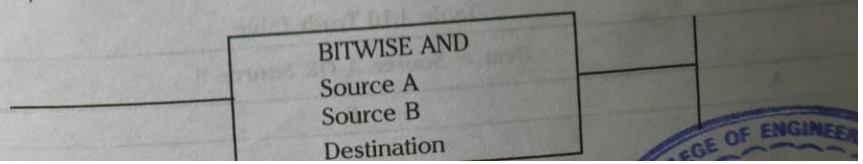

Table 4.18 Main Features of Mathematical Instructions

Instruction	Operation	Purpose	Address
ADD	Addition	One value (Source A) is added to another value (Source B) and the result is stored in the destination.	Source A and B can eithe Word Address or constan
SUB	Subtract	One value (Source A) is subtracted from another value (Source B) and the result is stored in the destination.	Source A and B can eithe Word Address or constan
MUL	Multiply	One value (Source A) is multiplied by another value (Source B) and the result is stored in Destination	Source A and B can eithe Word Address or constan
DIV	Division	One value (Source A) is divided by another value (Source B) and the result is stored in the destination.	Source A and B can eithe Word Address or constan
DDV	Double division	32-bit content of math register is divided by 16-bit value at source and the rounded quotient is placed in the destination. If the remainder is 0.5 or greater, the destination is rounded off.	Source A and B can eithe Word Address or constan
CLR	Clear	This is used to set the destination to zero. All status bits are also reset.	Source can either be Wor Address or constants.
SQR	Square root	The square root of the absolute value of the source is calculated when the instruction is true. The result is rounded off and kept in the destination.	Source can either be Wor Address or constants.

4.10.2 Logic ADD Instruction

The functional block diagram of the instruction is shown in Fig. 4.24, and the update the arithmetic status bits are presented in Table 4.19.

4.11 Special Mathematical Instructions

These instructions perform the special mathematical operations as mentioned in Table 4.25.

4.11.1 Main Features of Special Mathematical Instructions

The main features of the special Mathematical Instructions are tabulated in Table 4.25.

Table 4.25 Main Features of the Special Mathematical Instructions

Instruction	Operation	Purpose	Address
SCP	Scale with parameters	This instruction is used to produce a scaled output value that has a linear relationship between the input and scaled values. This instruction supports integer and floating point values.	Input value can be a Word Address or an address of floating point data elements. Input minimum and Input maximum values determine the range of data that appears in the input value parameter. Scaled minimum and scaled maximum values determine the range of data that appears in the scaled output parameter. Input minimum and input maximum value and scaled minimum and scaled maximum value can be a Word Address, an integer constant, floating point data element, or a floating point constant. The scaled output value can be a Word Address or an address of floating point data elements.
SCL	Scale data	When this instruction is true, the value at the source address is multiplied by the rate value. The rounded off result is added to the offset value and placed in the destination.	Source can be either a constant or a Word Address. Rate (or slope) is the positive or negative value. It can be either a constant or a word address. Offset can be either a constant or a Word Address.
ABS	Absolute	This instruction is used to calculate the absolute value of the source, and place the result in the destination.	Source can be a Word Address, an integer constant, floating point data element, or a floating point constant. Destination can only be a Word Address or a floating point data element.

Instruction	Operation	Purpose	Address
CPT	Compute	This instruction performs copy, arithmetic, logical, and conversion operations. The operation is defined in an expression and the result is written in the destination.	The destination can be a word address or the address of a floating-point data element. Expression is zero or more lines, with up to 28 characters per line, up to 255 characters.
SWP	Swap	This is used to swap the low and high bytes of a specified number of words in a bit, integer, ASCII, or string file.	Source can only be an indexed Word Address. Length refers to the number of words to be swapped, regardless of the file type. The address is limited to integer constants. For bit, integer, and ASCII file types, the length ranges is 1 to 128. For the string file type, the length range is 1 to 41.
ASN	Arc Sine	This is used to take the arc sine of a number (source in radians) and store the result (in radians) in the destination.	Source and destination are both numbers.
ACS	Arc Cosine	This is used to get the arc cosine of a number (source in radians) and store the result (in radians) in the destination.	Source and destination are both numbers.
ATN	Arc Tangent	This is used to take the arc tangent of a number (source) and store the result (in radians) in the destination.	Source and destination are both numbers.
COS	Cosine	This is used to take the cosine of a number (source in radians) and store the result in the destination.	Source and destination are both numbers.
LN	Natural Log	This is used to take the natural log of the value in the source and store the result in the destination.	Source and destination are both numbers.
LOG	Log to the base 10	This is used to take the log base 10 of the value in the source and store the result in the destination.	Source and destination are both numbers.

Instruction	Operation	Purpose	Address
SIN	Sine	This is used to take the sine of a number (source in radians) and store the result in the destination.	Source and destination are both numbers.
TAN	Tangent	This is used to take the tangent of a number (source in radians) and store the result in the destination.	Source and destination are both numbers.
XPY	X to the power Y	This can be used to raise a value (source A) to a power (source B) and store the result in the destination. If the value in source A is negative, the exponent (source B) should be a whole number. If it is not a whole number, the overflow bit is set and the absolute value of the base is used in the calculation.	Source and destination are both numbers.

4.11.2 Scale with Parameters or SCP Instruction

The functional block diagram of the instruction is shown in Fig. 4.32, and the updates of the arithmetic status bits are presented in Table 4.26.

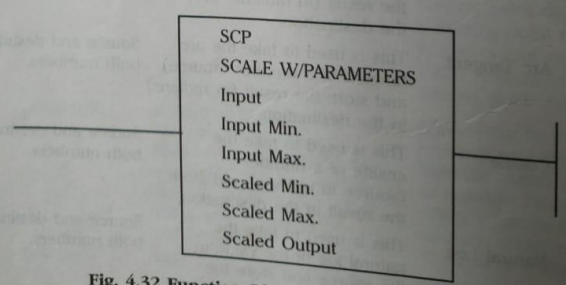

Fig. 4.32 Function Block of SCP Instruction

The source must be greater than or equal to -1 and less than or equal to 1. The resulting value in the destination is always greater than or equal to -π and