Programmable Logic Controllers

Comparison and Math Operations

Comparison Instructions

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mnemonic</strong></td>
<td><strong>Name</strong></td>
</tr>
<tr>
<td>EQU</td>
<td>Equal</td>
</tr>
<tr>
<td>NEQ</td>
<td>Not Equal</td>
</tr>
<tr>
<td>LES</td>
<td>Less Than</td>
</tr>
<tr>
<td>LEQ</td>
<td>Less Than or Equal</td>
</tr>
<tr>
<td>GRT</td>
<td>Greater Than</td>
</tr>
<tr>
<td>GEQ</td>
<td>Greater Than or Equal</td>
</tr>
<tr>
<td>MEQ</td>
<td>Masked Comparison for Equal</td>
</tr>
<tr>
<td>LIM</td>
<td>Limit Test</td>
</tr>
</tbody>
</table>
**About the Comparison Instructions**

- Comparison instructions are used to test pairs of values to condition the logical continuity of a rung
  - Thus, comparison instructions would seldom, if ever, be the last instruction on a rung
- As an example, suppose a LES instruction is presented with two values
  - If the first value is less than the second, then the comparison instruction is true

---

**Equal (EQU) & Not Equal (NEQ)**

| EQU  | Use the EQU instruction to test whether two values are equal.
|------|-----------------------------------------------
| EQU  | 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 | Use the NEQ instruction to test whether two values are not equal.
| Source B | If source A and source B are not equal, the instruction is logically true.

- Source A must be an address.
- Source B can be either a program constant or an address.
- Values are stored in two’s complementary form.
Less Than (LES) & Less Than or Equal (LEQ)

- Use the LES instruction to test whether one value (source A) is less than another (source B).
  - If source A is less than the value at source B, the instruction is logically true.
- Use the LEQ instruction to test whether one value (source A) is less than or equal to another (source B).
  - If the value at source A is less than or equal to the value at source B, the instruction is logically true.
- Source A must be an address. Source B can be either a program constant or an address.
- Values are stored in two's complementary form.

Greater Than (GRT) & Greater Than Or Equal (GEQ)

- Use the GRT instruction to test whether one value (source A) is greater than another (source B).
  - If the value at source A is greater than the value at source B, the instruction is logically true.
- Use the GEQ instruction to test whether one value (source A) is greater than or equal to another (source B).
  - If the value at source A is greater than or equal to the value at source B, the instruction is logically true.
Masked Comparison for Equal (MEQ)

- **Source** is the address of the value you want to compare.
- **Mask** is the address of the mask through which the instruction moves data
  - The mask can be a hexadecimal value.
- **Compare** is an integer value or the address of the reference.
  - If the 16 bits of data at the source address are equal to the 16 bits of data at the compare address (less masked bits), the instruction is true.
  - The instruction becomes false as soon as it detects a mismatch.

Use the MEQ instruction to compare data at a source address with data at a compare address.

Use of this instruction allows portions of the data to be masked by a separate word.

Limit Test (LIM)

- Use the LIM instruction to test for values within or outside a specified range, depending on how you set the limits.
- The Low Limit, Test, and High Limit values can be word addresses or constants, restricted to the following combinations:
  - If the Test parameter is a program constant, both the Low Limit and High Limit parameters must be word addresses.
  - If the Test parameter is a word address, the Low Limit and High Limit parameters can be either a program constant or a word address.
True/False Status of the LIM Instruction

- If the Low Limit has a value equal to or less than the High Limit, the instruction is true when the Test value is between the limits or is equal to either limit.

<table>
<thead>
<tr>
<th>Low Limit</th>
<th>High Limit</th>
<th>Instruction is True when Test value is</th>
<th>Instruction is False when Test value is</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>8</td>
<td>5 through 8</td>
<td>-32768 through 4 and 9 through 32767</td>
</tr>
</tbody>
</table>

Example, low limit less than high limit:

<table>
<thead>
<tr>
<th>Low Limit</th>
<th>High Limit</th>
<th>Instruction is True when Test value is</th>
<th>Instruction is False when Test value is</th>
</tr>
</thead>
<tbody>
<tr>
<td>-32768</td>
<td>+32767</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

True/False Status of the LIM Instruction (continued)

- If the Low Limit has a value greater than the High Limit, the instruction is false when the Test value is between the limits.

<table>
<thead>
<tr>
<th>Low Limit</th>
<th>High Limit</th>
<th>Instruction is True when Test value is</th>
<th>Instruction is False when Test value is</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>5</td>
<td>-32768 through 5 and 8 through 32767</td>
<td>6 through 7</td>
</tr>
</tbody>
</table>

Example, high limit less than low limit:
Math Instructions

• **Math Instructions Overview**

• **Source** is the address(es) of the value(s) on which the mathematical, logical, or move operation is to be performed.
  - This can be word addresses or program constants.
  - An instruction that has two source operands does not accept program constants in both operands.

• **Destination** is the address of the result of the operation. Signed integers are stored in two’s complementary form and apply to both source and destination parameters.

• Location of math instructions in ladder logic determine to operation performed (i.e. if the operation A(B+C) is desired, the ADD operation must appear before the MUL).

• There are notable differences between the simulator implementation of these instructions and the actual PLC hardware.

---

Available Math Instructions

• **ADD**: Add two values Dest=(A+B)

• **SUB**: Subtract one value from another Dest=(A-B)

• **MUL**: Multiply two values Dest=(A*B)

• **DIV**: Divide one value by another Dest=(A/B)

• **SQR**: Take the square root of a value

• **NEG**: Negate a value (2’s complement)

• **TOD**: Convert from integer to BCD

• **FRD**: Convert from BCD to integer

• All integer operations only

• *Other operations available with the Micrologix PLC*
Updates to Arithmetic Status Bits

- The arithmetic status bits are found in Word 0, bits 0–3 in the controller status file.
- After an instruction is executed, the arithmetic status bits in the status file are updated:

<table>
<thead>
<tr>
<th>With this bit:</th>
<th>The Controller:</th>
</tr>
</thead>
<tbody>
<tr>
<td>S:0/0 Carry (C)</td>
<td>sets if carry is generated; otherwise cleared</td>
</tr>
<tr>
<td>S:0/1 Overflow (V)</td>
<td>indicates that the actual result of a math instruction does not fit in the designated destination</td>
</tr>
<tr>
<td>S:0/2 Zero (Z)</td>
<td>indicates a 0 value after a math, move, or logic instruction</td>
</tr>
<tr>
<td>S:0/3 Sign (S)</td>
<td>indicates a negative (less than 0) value after a math, move, or logic instruction</td>
</tr>
</tbody>
</table>