5.6 Function blocks

The term *function block diagram (FBD)* is used for PLC programs described in terms of graphical blocks. It is described as being a graphical language for depicting signal and data flows through blocks, these being reusable software elements. A function block is a program instruction unit which, when executed, yields one or more output values. Thus a block is represented in the manner shown in Figure 5.23 with the function name written in the box.

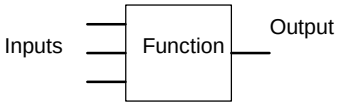

Figure 5.23 *Function block*

The IEC 113-3 standard for drawing such blocks is shown in Figure 5.24. A function block is depicted as a rectangular block with inputs entering from the left and outputs emerging from the right. The function block type name is shown in the block, e.g. AND, with the name of the function block in the system shown above it, Timer1. Names of function block inputs are shown within the block at the appropriate input and output points. Cross diagram connectors are used to indicate where graphical lines would be difficult to draw without cluttering up or complicating a diagram and show where an output at one point is used as an input at another.

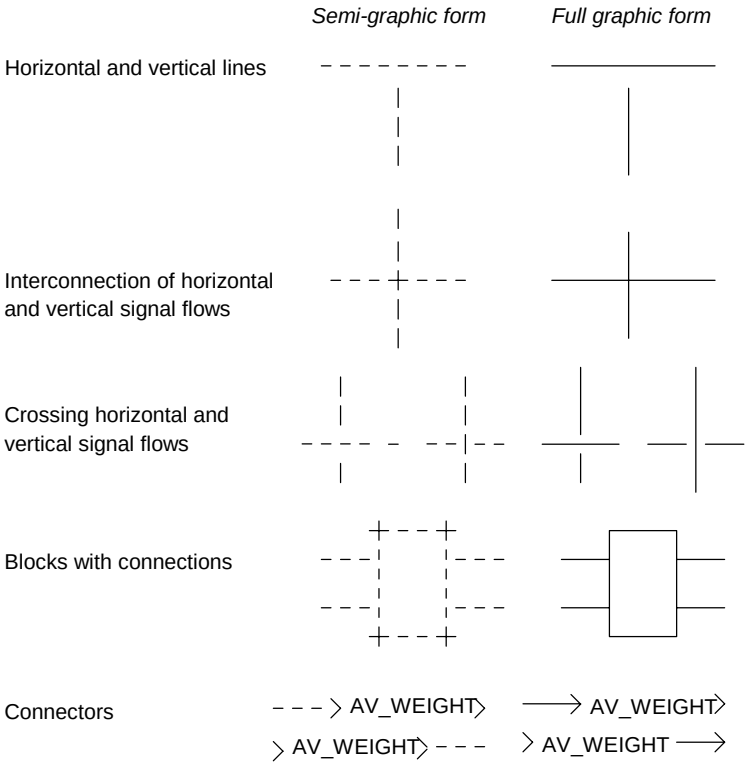

Figure 5.24 *Function block diagrams representation*

Function blocks can have standard functions, such as those of the logic gates or counters or times, or have functions defined by the user, e.g. a block to obtain an average value of inputs.

5.6.1 Logic gates

Programs are often concerned with logic gates. Two forms of standard circuit symbols are used for logic gates, one having originated in the United States and the other being an international standard form (IEEE/ANSI) which uses a rectangle with the logic function written inside it. The 1 in a box indicates that there is an output when the input is 1. The OR function is given by ≥1, this is because there is an output if an input is greater than or equal to 1. A negated input is represented by a small circle on the input, a negative output by a small circle on the output (Figure 5.25). Figure 5.26 shows the symbols. In FBD diagrams the notation used in the IEEE/ANSI form is often encountered.

Figure 5.27 shows the effect of such functional blocks in PLC programs.

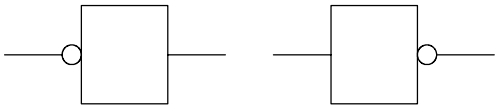

Figure 5.25 *(a) Negated input, (b) negated output*

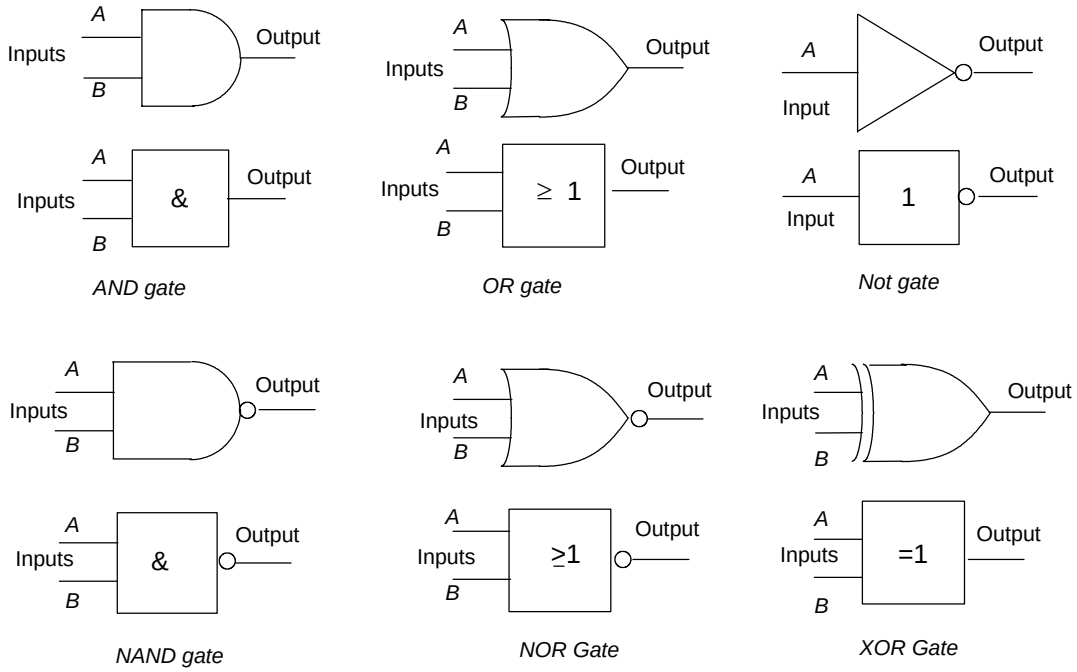

Figure 5.26 *Logic gate symbols*

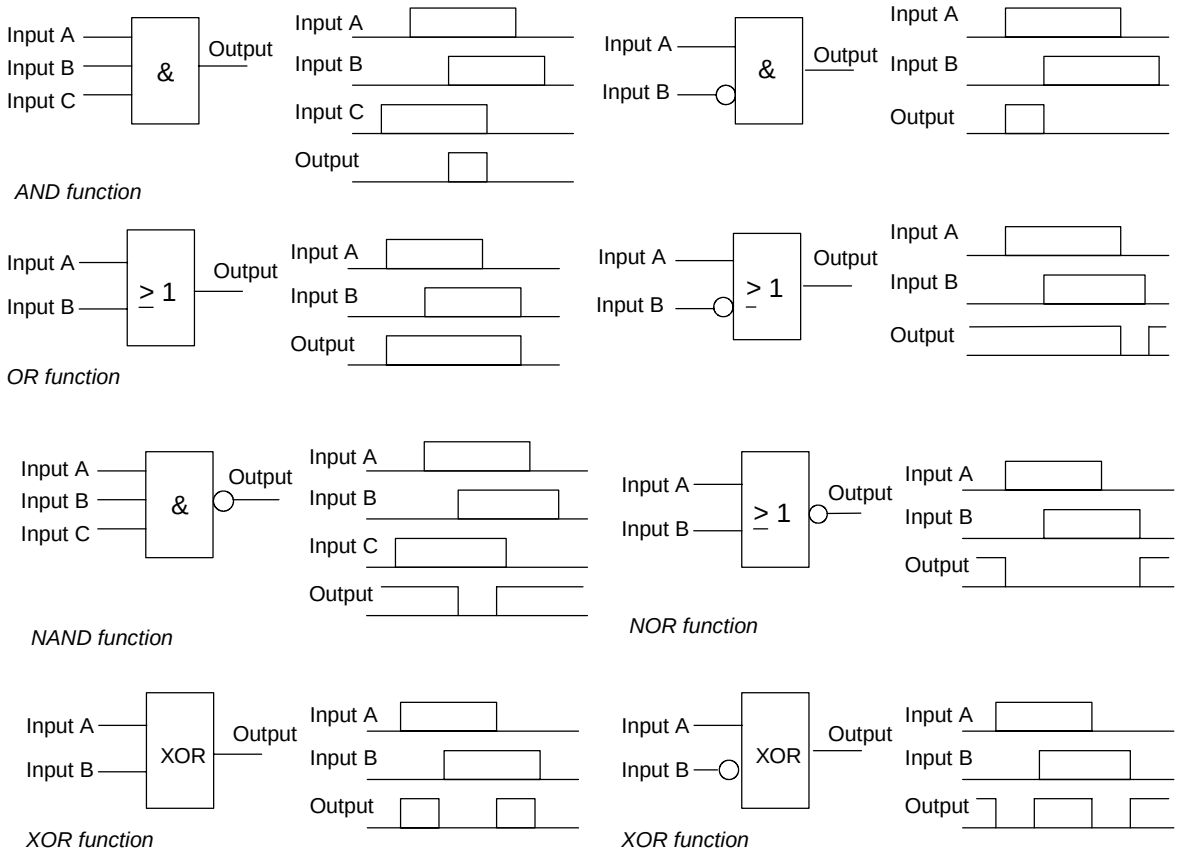

Figure 5.27 *Functional blocks*

To illustrate the form of such a diagram and its relationship to the ladder diagram, Figure 5.28 shows an OR gate. When A or B inputs are 1 then there is an output.

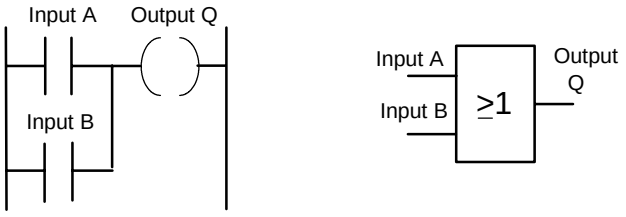

Figure 5.28 *Ladder diagram and equivalent functional block diagram*

Figure 5.29 shows a ladder diagram and its function block equivalent in Siemens notation. The = block is used to indicate an output from the system.

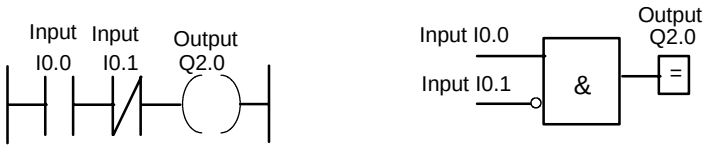

Figure 5.29 *Ladder diagram and equivalent function block diagram*

Figure 5.30 shows a ladder diagram involving the output having contacts acting as an input. The function block diagram equivalent can be shown as a feedback loop.

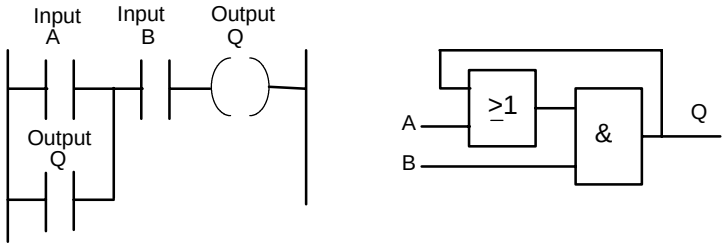

Figure 5.30 *Ladder diagram and equivalent function block diagram*

Consider the development of a function block diagram and ladder diagram for an application in which a pump is required to be activated and pump liquid into a tank when the start switch is closed, the level of liquid in the tank is below the required level and there is liquid in the reservoir from which it is to be pumped. What is required is an AND logic situation between the start switch input and a sensor input which is on when the liquid in the tank is below the required level. We might have a switch which is on until the liquid is at the required level. These two elements are then in an AND logic situation with a switch indicating that there is liquid in the reservoir. Suppose this switch gives an input when there is liquid. The function block diagram, and the equivalent ladder diagram, is then of the form shown in Figure 5.31.

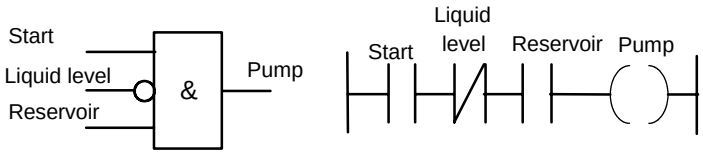

Figure 5.31 *Pump application*

5.6.2 Boolean algebra

Ladder programs can be derived from Boolean expressions since we are concerned with a mathematical system of logic. In Boolean algebra there are just two digits, 0 and 1. When we have an AND operation for inputs A and B then we can write:

$$A \cdot B = Q$$

where Q is the output. Thus Q is equal to 1 only when A = 1 and B = 1. The OR operation for inputs A and B is written as:

$$A + B = Q$$

Thus Q is equal to 1 only when A = 1 or B = 1. The NOT operation for an input A is written as:

$$\overline{A} = Q$$

Thus when A is not 1 there is an output.

As an illustration of how we can relate Boolean expressions with ladder diagrams, consider the expression:

$$A + B \cdot C = Q$$

This tells us that we have A or the term B and C giving the output Q. Figure 5.32 shows the ladder and functional block diagrams. Written in terms of Mitsubishi notation, the above expression might be:

$$X400 + X401 \cdot X402 = Y430$$

In Siemens notation it might be:

$$I0.0 + I0.1 \cdot I0.2 = Q2.0$$

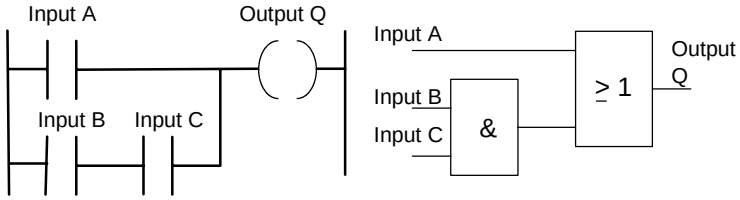

Figure 5.32 *Ladder diagram*

As a further illustration, consider the Boolean expression:

$$A + \overline{B} = Q$$

Figure 5.33 shows the ladder and functional block diagrams.

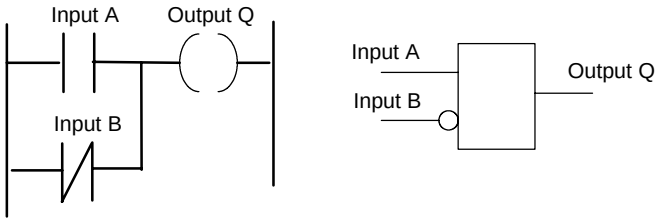

Figure 5.33 *Ladder diagram*

Written in terms of Mitsubishi notation, the expression might be:

$$X400 + \overline{X401} = Y430$$

and in Siemens notation:

$$I0.0 + \overline{I0.1} = Q2.0$$

Consider the exclusive-OR gate and its assembly from NOT, AND and OR gates, as shown in Figure 5.34.

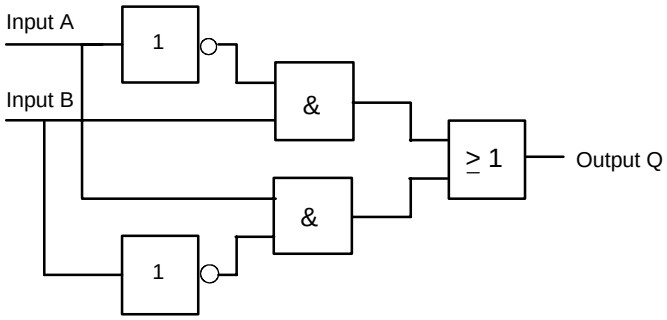

Figure 5.34 *XOR gate*

The input to the bottom AND gate is:

$$A \text{ and } \overline{B}$$

and so its output is:

$$A \cdot \overline{B}$$

The input to the top AND gate is:

$$\overline{A} \text{ and } B$$

and so its output is:

$$\overline{A} \cdot B$$

Thus the Boolean expression for the output from the OR gate is:

$$A \cdot \overline{B} + \overline{A} \cdot B = Q$$

Consider a logic diagram with many inputs, as shown in Figure 5.35, and its representation by a Boolean expression and a ladder rung.

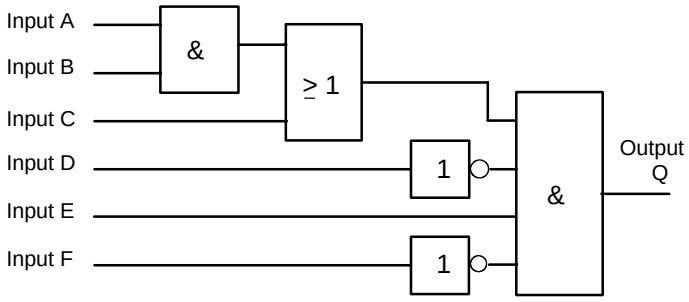

Figure 5.35 *Logic diagram*

For inputs A and B we obtain an output from the upper AND gate of A·B. From the OR gate we obtain an output of A·B + C. From the lower AND gate we obtain an output Q of:

$$(A \cdot B + C) \cdot \overline{D} \cdot E \cdot \overline{F} = Q$$

The ladder diagram to represent this is shown in Figure 5.36.

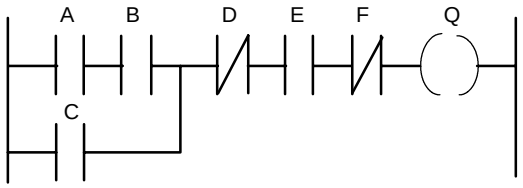

Figure 5.36 *Ladder diagram for Figure 5.35*

5.7 Program examples

The following tasks are intended to illustrate the application of the programming techniques given in this chapter.

A signal lamp is required to be switched on if a pump is running and the pressure is satisfactory, or if the lamp test switch is closed. For the inputs from the pump and the pressure sensors we have an AND logic situation since both are required if there is to be an output from the lamp. We, however, have an OR logic situation with the test switch in that it is required to give an output of lamp on regardless of whether there is a signal from the AND system. The function block diagram and the ladder diagram are thus of the form shown in Figure 5.37. Note that with the ladder diagram we tell the PLC when it has reached the end of the program by the use of the END or RET instruction.

As another example, consider a valve which is to be operated to lift a load when a pump is running and either the lift switch is operated or a switch operated indicating that the load has not already been lifted and is at the bottom of its lift channel. We have an OR situation for the two switches and an AND situation involving the two switches and the pump. Figure 5.38 shows a possible program.

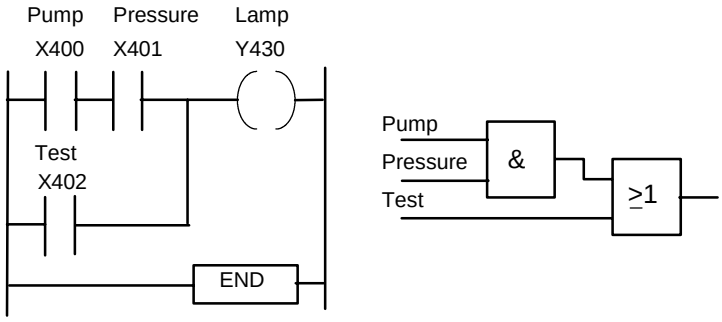

Figure 5.37 *Signal lamp task*

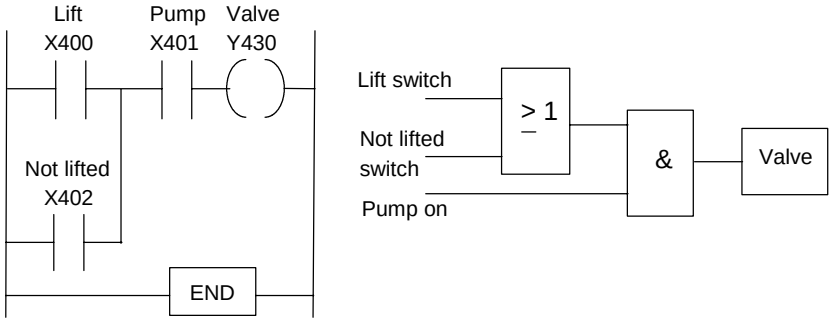

Figure 5.38 *Valve operation program*

As another example, consider a system where there has to be no output when any one of four sensors gives an output, otherwise there is to be an output. One way we could write a program for this is for each sensor to have contacts that are normally closed so there is an output. When there is an input to the sensor the contacts open and the output stops. We have an AND logic situation. Figure 5.39 shows the functional block diagram and the ladder diagram of a system that might be used.

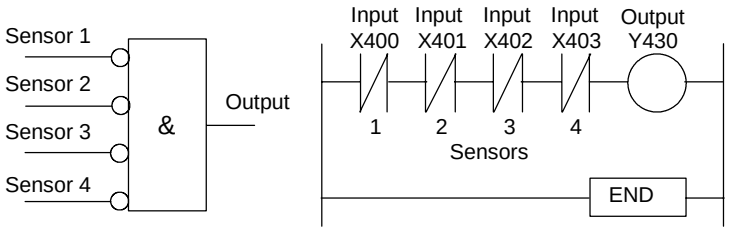

Figure 5.39 *Output switched off by any one of four sensors being activated*