4. Graphic languages (version 1996)

The graphic languages defined in this standard are LD (Ladder Diagram) and FBD (Function Block Diagram). The sequential function chart (SFC) elements defined in 2.6 can be used in conjunction with either of these languages.

4.1 *Common elements*

The elements defined in this clause apply to both the graphic languages in the Standard, that is, LD (Ladder Diagram) and FBD (Function Block Diagram), and to the graphic representation of sequential function chart (SFC) elements.

4.1.1 *Representation of lines and blocks*

The graphic language elements defined in this clause are drawn with line elements using characters from the ISO/IEC 646 character set, or using graphic or semigraphic elements, as shown in table 57.
Lines can be extended by the use of *connectors* as shown in table 57. No storage of data or association with data elements shall be associated with the use of connectors; hence, to avoid ambiguity, it shall be an *error* if the identifier used as a connector label is the same as the name of another named element within the same program organization unit.

4.1.2 *Direction of flow in networks*

A *network* is defined as a maximal set of interconnected graphic elements, excluding the left and right rails in the case of networks in the LD language defined in 4.2. Provision shall be made to associate with each network or group of networks in a graphic language a *network label* delimited on the right by a colon (:). This label shall have the form of an identifier or an unsigned decimal integer as defined in clause 2 of this Part. The *scope* of a network and its label shall be *local* to the program organization unit in which the network is located. Examples of networks and network labels are shown in annex F.
Graphic languages are used to represent the flow of a conceptual quantity through one or more networks representing a control plan, that is:

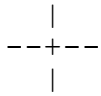
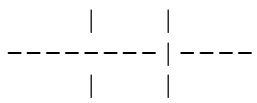
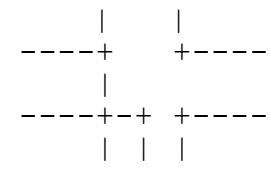
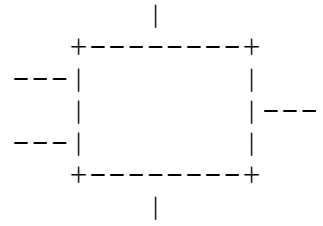
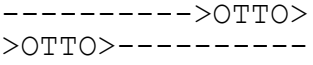
- "Power flow", analogous to the flow of electric power in an electromechanical relay system, typically used in relay ladder diagrams;
- "Signal flow", analogous to the flow of signals between elements of a signal processing system, typically used in function block diagrams;
- "Activity flow", analogous to the flow of control between elements of an organization, or between the steps of an electromechanical sequencer, typically used in sequential function charts.

The appropriate conceptual quantity shall flow along lines between elements of a network according to the following rules:

1) Power flow in the LD language shall be from left to right.
2) Signal flow in the FBD language shall be from the output (right-hand) side of a function or function block to the input (left-hand) side of the function or function block(s) so connected.
3) Activity flow between the SFC elements defined in 2.6 shall be from the bottom of a step through the appropriate transition to the top of the corresponding successor step(s).

Table 57 - Representation of lines and blocks

No.	Feature	Example
	Horizontal lines:	
1	ISO / IEC 646 "minus" character	-----
2	Graphic or semigraphic	

3 4	**Vertical lines:** ISO / IEC 646 "vertical line" character Graphic or semigraphic	
5 6	**Horizontal/vertical connection:** ISO / IEC 646 "plus" character Graphic or semigraphic	
7 8	**Line crossings without connection:** ISO / IEC 646 characters Graphic or semigraphic	
9 10	**Connected and non-connected corners:** ISO / IEC 646 characters Graphic or semigraphic	
11 12	**Blocks with connecting lines**: ISO / IEC 646 characters Graphic or semigraphic	
13 14	**Connectors using ISO / IEC 646 characters:** Connector Continuation of a connected line Graphic or semigraphic connectors	

4.1.3 *Evaluation of networks*

The order in which networks and their elements are evaluated is not necessarily the same as the order in which they are labeled or displayed. Similarly, it is not necessary that all networks be evaluated before the evaluation of a given network can be repeated. However, when the body of a program organization unit consists of one or more networks, the results of network evaluation within said body shall be functionally equivalent to the observance of the following rules:

1) No element of a network shall be evaluated until the states of all of its inputs have been evaluated.

2) The evaluation of a network element shall not be complete until the states of all of its outputs have been evaluated.

3) The evaluation of a network is not complete until the outputs of all of its elements have been evaluated, even if the network contains one of the execution control elements defined in 4.1.4.

(4) The order in which networks are evaluated shall conform to the provisions of 4.2.6 for the LD language and 4.3.3 for the FBD language.

A *feedback path* is said to exist in a network when the output of a function or function block is used as the input to a function or function block which precedes it in the network; the associated variable is called a *feedback variable*. For instance, the Boolean variable RUN is the feedback variable in the example shown in figure 23. A feedback variable can also be an output element of a function block data structure as defined in 2.5.2.

Feedback paths can be utilized in the graphic languages defined in 4.2 and 4.3, subject to the following rules:

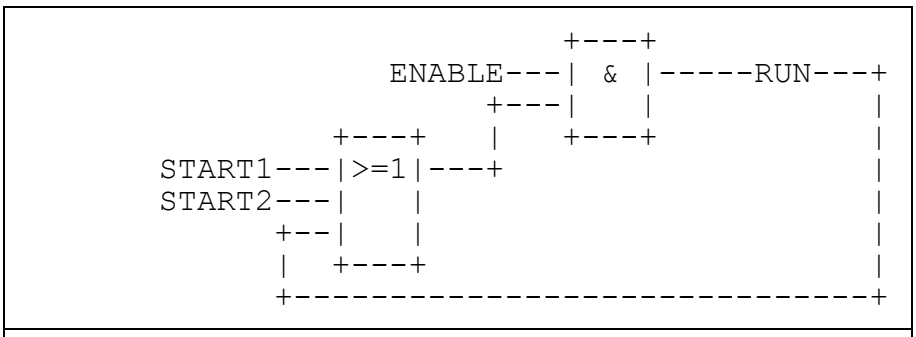
1) Explicit loops such as the one shown in 23a shall only appear in the FBD language defined in 4.3.

2) It shall be possible for the user to define the order of execution of the elements in an explicit loop, for instance by selection of feedback variables to form an implicit loop as shown in figure 23b.

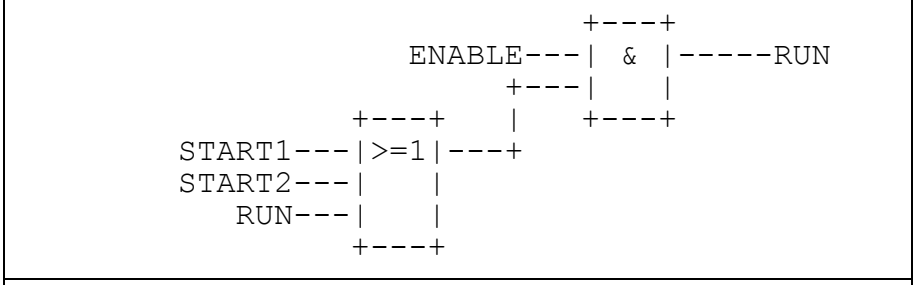
3) Feedback variables shall be initialized by one of the mechanisms defined in clause 2. The initial value shall be used during the first evaluation of the network.

4) Once the element with a feedback variable as output has been evaluated, the new value of the feedback variable shall be used until the next evaluation of the element.

a)



b)



c)

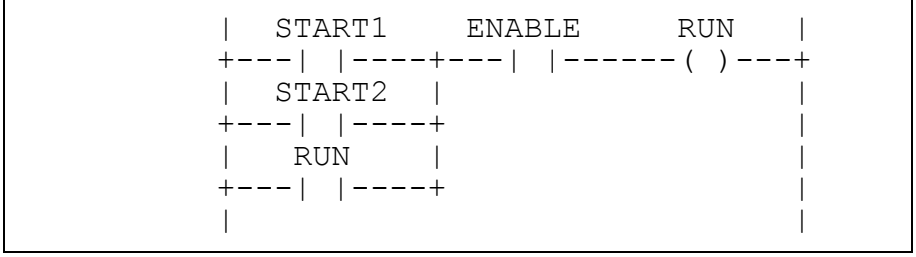


Figure 23 - Feedback path example
a) Explicit loop
b) Implicit loop
c) LD language equivalent

4.1.4 *Execution control elements*

Transfer of program control in the LD and FBD languages shall be represented by the graphical elements shown in table 58.
Jumps shall be shown by a Boolean signal line terminated in a double arrowhead. The signal line for a jump condition shall originate at a Boolean variable, at a Boolean output of a function or function block, or on the power flow line of a ladder diagram. A transfer of program control to the designated network label shall occur when the Boolean value of the signal line is 1 (TRUE); thus, the unconditional jump is a special case of the conditional jump.
The target of a jump shall be a network label within the program organization unit within which the jump occurs. If the jump occurs within an ACTION...END_ACTION construct, the target of the jump shall be within the same construct. Conditional returnns_XE "return2018XE "conditional: return"