

INTERNATIONAL STANDARD

**Programmable controllers –
Part 3: Programming languages**

IEC Standards
(<https://standards.iteh.ai>)
Document Preview

IEC 61131-3:2003

<https://standards.iteh.ai/catalog/standards/iec/34e7370d-d9cf-4237-a2b4-ff7fcd2b8c94/iec-61131-3-2003>

WITHDRAWN



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2003 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester.

If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

IEC Central Office
3, rue de Varembe
CH-1211 Geneva 20
Switzerland
Email: inmail@iec.ch
Web: www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

- Catalogue of IEC publications: www.iec.ch/searchpub

The IEC on-line Catalogue enables you to search by a variety of criteria (reference number, text, technical committee,...). It also gives information on projects, withdrawn and replaced publications.

- IEC Just Published: www.iec.ch/online_news/justpub

Stay up to date on all new IEC publications. Just Published details twice a month all new publications released. Available on-line and also by email.

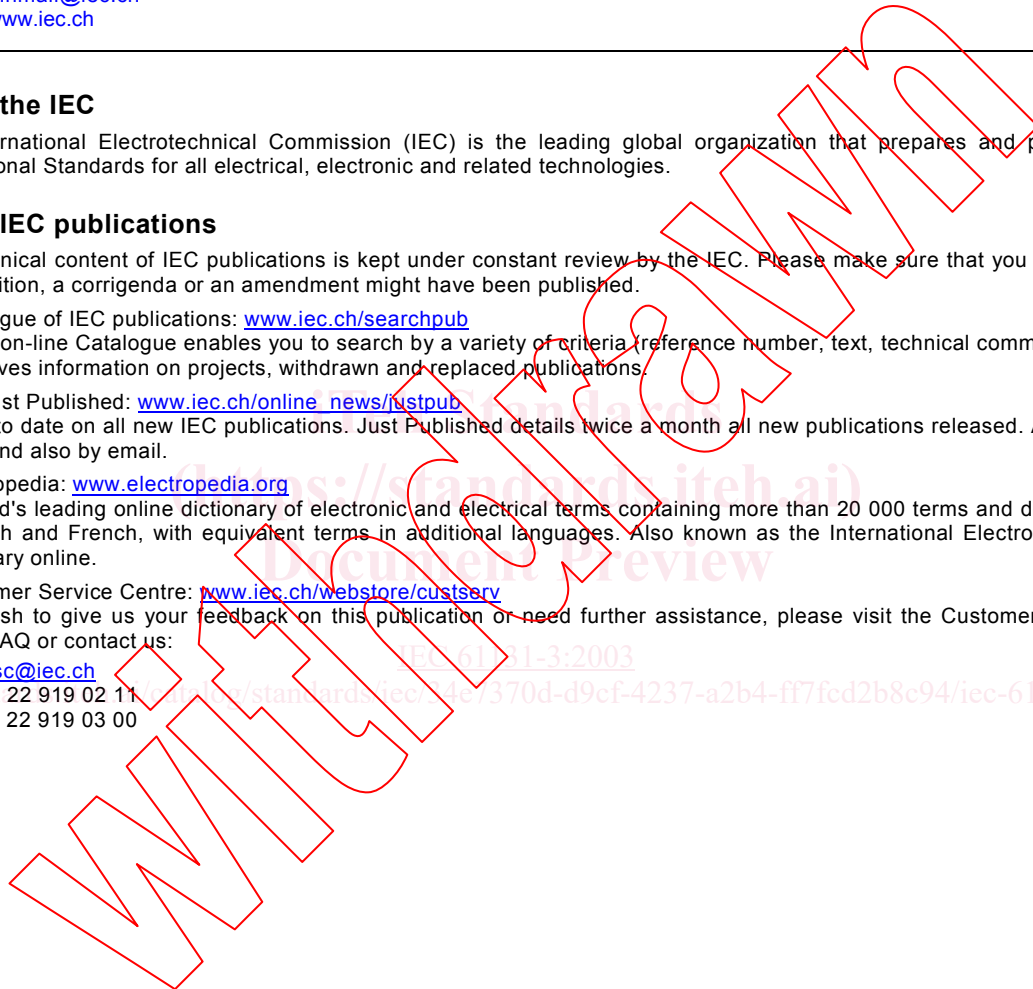
- Electropedia: www.electropedia.org

The world's leading online dictionary of electronic and electrical terms containing more than 20 000 terms and definitions in English and French, with equivalent terms in additional languages. Also known as the International Electrotechnical Vocabulary online.

- Customer Service Centre: www.iec.ch/webstore/custserv

If you wish to give us your feedback on this publication or need further assistance, please visit the Customer Service Centre FAQ or contact us:

Email: csc@iec.ch
Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00



<https://standards.iteh.ai/catalog/standards/iec/4e7370d-d9cf-4237-a2b4-ff7fcd2b8c94/iec-61131-3-2003>

INTERNATIONAL STANDARD

**Programmable controllers –
Part 2: Equipment requirements and tests**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

PRICE CODE
CODE PRIX

XH

ICS 25.040; 35.240.50

ISBN 2-8318-6653-7

CONTENTS

FOREWORD	8
1 General	9
1.1 Scope	9
1.2 Normative references	9
1.3 Definitions	9
1.4 Overview and general requirements	14
1.4.1 Software model	14
1.4.2 Communication model	16
1.4.3 Programming model	18
1.5 Compliance	19
1.5.1 System compliance	20
1.5.2 Program compliance	22
2 Common elements	23
2.1 Use of printed characters	23
2.1.1 Character set	23
2.1.2 Identifiers	23
2.1.3 Keywords	24
2.1.4 Use of white space	24
2.1.5 Comments	24
2.1.6 Pragmas	25
2.2 External representation of data	25
2.2.1 Numeric literals	25
2.2.2 Character string literals	26
2.2.3 Time literals	28
2.2.3.1 Duration	28
2.2.3.2 Time of day and date	29
2.3 Data types	29
2.3.1 Elementary data types	30
2.3.2 Generic data types	31
2.3.3 Derived data types	32
2.3.3.1 Declaration	32
2.3.3.2 Initialization	33
2.3.3.3 Usage	35
2.4 Variables	36
2.4.1 Representation	36
2.4.1.1 Single-element variables	36
2.4.1.2 Multi-element variables	38
2.4.2 Initialization	38
2.4.3 Declaration	39
2.4.3.1 Type assignment	41
2.4.3.2 Initial value assignment	42
2.5 Program organization units	45
2.5.1 Functions	45
2.5.1.1 Representation	46
2.5.1.2 Execution control	49
2.5.1.3 Declaration	50
2.5.1.4 Typing, overloading, and type conversion	52
2.5.1.5 Standard functions	55
2.5.1.5.1 Type conversion functions	55
2.5.1.5.2 Numerical functions	56
2.5.1.5.3 Bit string functions	59
2.5.1.5.4 Selection and comparison functions	59
2.5.1.5.5 Character string functions	62
2.5.1.5.6 Functions of time data types	64
2.5.1.5.7 Functions of enumerated data types	66

2.5.2 Function blocks.....	66
2.5.2.1 Representation	67
2.5.2.1a) Use of EN and ENO in function blocks	68
2.5.2.2 Declaration	69
2.5.2.3 Standard function blocks	77
2.5.2.3.1 Bistable elements	77
2.5.2.3.2 Edge detection.....	78
2.5.2.3.3 Counters	78
2.5.2.3.4 Timers.....	81
2.5.2.3.5 Communication function blocks.....	83
2.5.3 Programs	83
2.6 Sequential Function Chart (SFC) elements.....	84
2.6.1 General.....	84
2.6.2 Steps.....	84
2.6.3 Transitions	86
2.6.4 Actions	89
2.6.4.1 Declaration	89
2.6.4.2 Association with steps	91
2.6.4.3 Action blocks	92
2.6.4.4 Action qualifiers	93
2.6.4.5 Action control.....	94
2.6.5 Rules of evolution	99
2.6.6 Compatibility of SFC elements	107
2.6.7 SFC Compliance requirements	108
2.7 Configuration elements.....	108
2.7.1 Configurations, resources, and access paths.....	110
2.7.2 Tasks	114
3 Textual languages	123
3.1 Common elements	123
3.2 Instruction list (IL)	123
3.2.1 Instructions	123
3.2.2 Operators, modifiers and operands.....	124
3.2.3 Functions and function blocks	126
3.3 Structured Text (ST).....	129
3.3.1 Expressions.....	129
3.3.2 Statements	131
3.3.2.1 Assignment statements	133
3.3.2.2 Function and function block control statements	133
3.3.2.3 Selection statements.....	133
3.3.2.4 Iteration statements	134
4 Graphic languages	135
4.1 Common elements	135
4.1.1 Representation of lines and blocks	135
4.1.2 Direction of flow in networks.....	135
4.1.3 Evaluation of networks	136
4.1.4 Execution control elements	138
4.2 Ladder diagram (LD)	139
4.2.1 Power rails.....	139
4.2.2 Link elements and states.....	139
4.2.3 Contacts	140
4.2.4 Coils.....	140
4.2.5 Functions and function blocks.....	140
4.2.6 Order of network evaluation	141
4.3 Function Block Diagram (FBD).....	143
4.3.1 General.....	143
4.3.2 Combination of elements.....	143
4.3.3 Order of network evaluation	143

ANNEX A (normative) Specification method for textual languages	144
A.1 Syntax	144
A.1.1 Terminal symbols	144
A.1.2 Non-terminal symbols	144
A.1.3 Production rules	145
A.2 Semantics	145
ANNEX B (normative) Formal specifications of language elements	146
B.0 Programming model	146
B.1 Common elements	146
B.1.1 Letters, digits and identifiers	146
B.1.2 Constants	147
B.1.2.1 Numeric literals	147
B.1.2.2 Character strings	147
B.1.2.3 Time literals	148
B.1.2.3.1 Duration	148
B.1.2.3.2 Time of day and date	148
B.1.3 Data types	149
B.1.3.1 Elementary data types	149
B.1.3.2 Generic data types	149
B.1.3.3 Derived data types	149
B.1.4 Variables	151
B.1.4.1 Directly represented variables	151
B.1.4.2 Multi-element variables	151
B.1.4.3 Declaration and initialization	152
B.1.5 Program organization units	154
B.1.5.1 Functions	154
B.1.5.2 Function blocks	155
B.1.5.3 Programs	156
B.1.6 Sequential function chart elements	156
B.1.7 Configuration elements	157
B.2 Language IL (Instruction List)	159
B.2.1 Instructions and operands	159
B.2.2 Operators	159
B.3 Language ST (Structured Text)	160
B.3.1 Expressions	160
B.3.2 Statements	160
B.3.2.1 Assignment statements	161
B.3.2.2 Subprogram control statements	161
B.3.2.3 Selection statements	161
B.3.2.4 Iteration statements	161
ANNEX C (normative) Delimiters and keywords	162
ANNEX D (normative) Implementation-dependent parameters	165
ANNEX E (normative) Error conditions	167
ANNEX F (informative) Examples	169
F.1 Function WEIGH	169
F.2 Function block CMD_MONITOR	170
F.3 Function block FWD_REV_MON	173
F.4 Function block STACK_INT	178
F.5 Function block MIX_2_BRIX	183
F.6 Analog signal processing	186
F.6.1 Function block LAG1	187
F.6.2 Function block DELAY	187

F.6.3 Function block AVERAGE	188
F.6.4 Function block INTEGRAL.....	188
F.6.5 Function block DERIVATIVE	189
F.6.6 Function block HYSTERESIS	189
F.6.7 Function block LIMITS_ALARM.....	190
F.6.8 Structure ANALOG_LIMITS.....	190
F.6.9 Function block ANALOG_MONITOR.....	191
F.6.10 Function block PID.....	192
F.6.11 Function block DIFFEQ	193
F.6.12 Function block RAMP	194
F.6.13 Function block TRANSFER.....	195
F.7 Program GRAVEL.....	195
F.8 Program AGV	203
F.9 Use of enumerated data types	206
F.10 Function block RTC (Real Time Clock).....	206
F.11 Function block ALRM_INT.....	206
ANNEX G (informative) Reference character set.....	208
Index	210
Table 1 - Character set features.....	23
Table 2 - Identifier features.....	24
Table 3 - Comment feature.....	25
Table 3a - Pragma feature.....	25
Table 4 - Numeric literals.....	26
Table 5 - Character string literal features.....	27
Table 6 - Two-character combinations in character strings	28
Table 7 - Duration literal features.....	29
Table 8 - Date and time of day literals.....	29
Table 9 - Examples of date and time of day literals	29
Table 10 - Elementary data types	30
Table 11 - Hierarchy of generic data types	32
Table 12 - Data type declaration features	33
Table 13 - Default initial values of elementary data types.....	34
Table 14 - Data type initial value declaration features	35
Table 15 - Location and size prefix features for directly represented variables.....	37
Table 16a - Variable declaration keywords	39
Table 16b - Usages of VAR_GLOBAL, VAR_EXTERNAL and CONSTANT declarations	41
Table 17 - Variable type assignment features.....	41
Table 18 - Variable initial value assignment features.....	43
Table 19 - Graphical negation of Boolean signals	47
Table 19a - Textual invocation of functions for formal and non-formal argument list	49
Table 20 - Use of EN input and ENO output	50
Table 20a - Function features.....	51
Table 21 - Typed and overloaded functions	53
Table 22 - Type conversion function features	55
Table 23 - Standard functions of one numeric variable.....	57
Table 24 - Standard arithmetic functions.....	58

Table 25 - Standard bit shift functions	59
Table 26 - Standard bitwise Boolean functions	60
Table 27 - Standard selection functions ^d	61
Table 28 - Standard comparison functions	62
Table 29 - Standard character string functions	63
Table 30 - Functions of time data types	64
Table 31 - Functions of enumerated data types	66
Table 32 - Examples of function block I/O variable usage	68
Table 33 - Function block declaration and usage features	71
Table 34 - Standard bistable function blocks ^a	77
Table 35 - Standard edge detection function blocks	78
Table 36 - Standard counter function blocks	79
Table 37 - Standard timer function blocks	81
Table 38 - Standard timer function blocks - timing diagrams	82
Table 39 - Program declaration features	83
Table 40 - Step features	85
Table 41 - Transitions and transition conditions	87
Table 42 - Declaration of actions ^{a,b}	90
Table 43 - Step/action association	92
Table 44 - Action block features	93
Table 45 - Action qualifiers	94
Table 45a - Action control features	98
Table 46 - Sequence evolution	101
Table 47 - Compatible SFC features	108
Table 48 - SFC minimal compliance requirements	108
Table 49 - Configuration and resource declaration features	112
Table 50 - Task features	116
Table 51a - Examples of instruction fields	124
Table 51b - Parenthesized expression features for IL language	125
Table 52 - Instruction List operators	125
Table 53 - Function Block invocation and Function invocation features for IL language	127
Table 54 - Standard Function Block input operators for IL language	129
Table 55 - Operators of the ST language	131
Table 56 - ST language statements	132
Table 57 - Representation of lines and blocks	136
Table 58 - Graphic execution control elements	138
Table 59 - Power rails	139
Table 60 - Link elements	140
Table 61 - Contacts ^a	141
Table 62 - Coils	142
Table C.1 - Delimiters	162
Table C.2 - Keywords	163
Table D.1 - Implementation-dependent parameters	165
Table E.1 - Error conditions	167

Table G.1 - Character representations	208
Table G.2 - Character encodings	209
Figure 1 - Software model	15
Figure 2 a) - Data flow connection within a program	16
Figure 2 b) - Communication via GLOBAL variables	16
Figure 2 c) - Communication function blocks	17
Figure 2 d) - Communication via access paths	17
Figure 3 - Combination of programmable controller language elements	19
Figure 4 - Examples of function usage	45
Figure 5 - Use of formal argument names	48
Figure 6 - Examples of function declarations and usage	52
Figure 7 - Examples of explicit type conversion with overloaded functions	54
Figure 8 - Examples of explicit type conversion with typed functions	54
Figure 9 - Function block instantiation examples	67
Figure 10 - Examples of function block declarations	70
Figure 11 a) - Graphical use of a function block name as an input variable	73
Figure 11 b) - Graphical use of a function block name as an in-out variable	74
Figure 11 c) - Graphical use of a function block name as an external variable	75
Figure 12 - Declaration and usage of in-out variables in function blocks	76
Figure 14 - ACTION_CONTROL function block - External interface (Not visible to the user)	95
Figure 15 a) - ACTION_CONTROL function block body with "final scan" logic	96
Figure 15 b) - ACTION_CONTROL function block body without "final scan" logic	97
Figure 16 a) - Action control example - SFC representation	98
Figure 16 b) - Action control example - functional equivalent	99
Figure 17 - Examples of SFC evolution rules	105
Figure 18 a) - Examples of SFC errors: an "unsafe" SFC	106
Figure 18 b) - Examples of SFC errors: an "unreachable" SFC	107
Figure 19 a) - Graphical example of a configuration	109
Figure 19 b) - Skeleton function block and program declarations for configuration example	110
Figure 20 - Examples of CONFIGURATION and RESOURCE declaration features	113
Figure 21 a) - Synchronization of function blocks with explicit task associations	120
Figure 21 b) - Synchronization of function blocks with implicit task associations	121
Figure 21 c) - Explicit task associations equivalent to figure 21 b)	122
Figure 22 - EXIT statement example	134
Figure 23 - Feedback path example	137
Figure 24 - Boolean OR examples	143

INTERNATIONAL ELECTROTECHNICAL COMMISSION

PROGRAMMABLE CONTROLLERS –

Part 3: Programming languages

FOREWORD

- 1) The IEC (International Electrotechnical Commission) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of the IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, the IEC publishes International Standards. Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. The IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of the IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested National Committees.
- 3) The documents produced have the form of recommendations for international use and are published in the form of standards, technical specifications, technical reports or guides and they are accepted by the National Committees in that sense.
- 4) In order to promote international unification, IEC National Committees undertake to apply IEC International Standards transparently to the maximum extent possible in their national and regional standards. Any divergence between the IEC Standard and the corresponding national or regional standard shall be clearly indicated in the latter.
- 5) The IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with one of its standards.
- 6) Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. The IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61131-3 has been prepared by subcommittee 65B: Devices, of IEC technical committee 65: Industrial-process measurement and control.

<https://standards.iteh.ai/catalog/standards/iec/4e7370d-d9cf-4237-a2b4-f7fcd2b8c94/iec-61131-3-2003>

The text of this standard is based on the following documents:

FDIS	Report on voting
65B/456/FDIS	65B/465/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This second edition of IEC 61131-3 cancels and replaces the first edition, published in 1993, and constitutes a technical revision.

This International Standard has been reproduced without significant modification to its original contents or drafting.

The committee has decided that the contents of this publication will remain unchanged until 2007. At this date, the publication will be

- reconfirmed;
- withdrawn;
- replaced by a revised edition, or
- amended.

PROGRAMMABLE CONTROLLERS –

Part 3: Programming languages

1 General

1.1 Scope

This part of IEC 61131 specifies syntax and semantics of programming languages for *programmable controllers* as defined in part 1 of IEC 61131.

The functions of program entry, testing, monitoring, operating system, etc., are specified in Part 1 of IEC 61131.

1.2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 60050 (all parts): *International Electrotechnical Vocabulary (IEV)*

IEC 60559:1989, *Binary floating-point arithmetic for microprocessors systems*

IEC 60617-12:1997, *Graphical symbols for diagrams – Part 12: Binary logic elements*

IEC 60617-13:1993, *Graphical symbols for diagrams – Part 13: Analogue elements*

IEC 60848:2002, *GRAFCET specification language for sequential function charts*

IEC 61131-1, *Programmable controllers – Part 1: General information*

IEC 61131-5, *Programmable controllers – Part 5: Communications*

ISO/AFNOR: 1989, *Dictionary of computer science – The standardised vocabulary*

ISO/IEC 10646-1:1993, *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane*

1.3 Definitions

For the purposes of this part of IEC 61131, the following definitions apply. Definitions applying to all parts of IEC 61131 are given in part 1.

NOTE 1 Terms defined in this subclause are *italicized* where they appear in the bodies of definitions.

NOTE 2 The notation “(ISO)” following a definition indicates that the definition is taken from the ISO/AFNOR Dictionary of computer science.

NOTE 3 The ISO/AFNOR Dictionary of computer science and the IEC 60050 should be consulted for terms not defined in this standard.

- 1.3.1 absolute time:** the combination of time of day and date information.
- 1.3.2 access path:** the association of a symbolic name with a variable for the purpose of open communication.
- 1.3.3 action:** Boolean variable, or a collection of operations to be performed, together with an associated control structure, as specified in 2.6.4.
- 1.3.4 action block:** graphical language element which utilizes a Boolean input variable to determine the value of a Boolean output variable or the enabling condition for an *action*, according to a predetermined control structure as defined in 2.6.4.5.
- 1.3.5 aggregate:** structured collection of data objects forming a *data type*. (ISO)
- 1.3.6 argument:** synonymous with *input variable*, *output variable* or *in-out variable*.
- 1.3.7 array:** *aggregate* that consists of data objects, with identical attributes, each of which may be uniquely referenced by *subscripting*. (ISO)
- 1.3.8 assignment:** mechanism to give a value to a variable or to an *aggregate*. (ISO)
- 1.3.9 based number:** number represented in a specified base other than ten.
- 1.3.10 bistable function block:** *function block* with two stable states controlled by one or more inputs.
- 1.3.11 bit string:** data element consisting of one or more bits.
- 1.3.12 body:** that portion of a *program organization unit* which specifies the operations to be performed on the declared *operands* of the program organization unit when its execution is *invoked*.
- 1.3.13 call:** language construct for *invoking* the execution of a *function* or *function block*.
- 1.3.14 character string:** *aggregate* that consists of an ordered sequence of characters.
- 1.3.15 comment:** language construct for the inclusion of text in a program and having no impact on the execution of the program. (ISO)
- 1.3.16 compile:** to translate a *program organization unit* or *data type* specification into its machine language equivalent or an intermediate form.
- 1.3.17 configuration:** language element corresponding to a *programmable controller system* as defined in IEC 61131-1.
- 1.3.18 counter function block:** *function block* which accumulates a value for the number of changes sensed at one or more specified *inputs*.
- 1.3.19 data type:** set of values together with a set of permitted operations. (ISO)
- 1.3.20 date and time:** the date within the year and the time of day represented as a single language element.
- 1.3.21 declaration:** the mechanism for establishing the definition of a *language element*. A declaration normally involves attaching an identifier to the language element, and allocating attributes such as *data types* and algorithms to it.

1.3.22 delimiter: character or combination of characters used to separate program *language elements*.

1.3.23 direct representation: means of representing a variable in a programmable controller program from which a manufacturer-specified correspondence to a physical or *logical location* may be determined directly.

1.3.24 double word: data element containing 32 bits.

1.3.25 evaluation: the process of establishing a value for an expression or a *function*, or for the *outputs* of a network or *function block*, during program execution.

1.3.26 execution control element: A *language element* which controls the flow of program execution.

1.3.27 falling edge: the change from 1 to 0 of a Boolean variable.

1.3.28 function (procedure): *program organization unit* which, when executed, yields exactly one data element and possibly additional *output variables* (which may be multi-valued, for example, an *array* or *structure*), and whose *invocation* can be used in textual languages as an *operand* in an expression.

1.3.29 function block instance (function block): *instance* of a *function block type*.

1.3.30 function block type: programmable controller programming *language element* consisting of:

- 1) the definition of a data structure partitioned into input, output, and internal variables; and
- 2) a set of operations to be performed upon the elements of the data structure when an *instance* of the function block type is *invoked*.

1.3.31 function block diagram: *network* in which the nodes are *function block instances*, graphically represented *functions (procedures), variables, literals, and labels*.

1.3.32 generic data type: *data type* which represents more than one type of data, as specified in 2.3.2.

1.3.33 global scope: scope of a declaration applying to all program organization units within a *resource* or *configuration*.

1.3.34 global variable: variable whose *scope* is *global*.

1.3.35 hierarchical addressing: the *direct representation* of a data element as a member of a physical or logical hierarchy, for example, a point within a module which is contained in a rack, which in turn is contained in a cubicle, etc.

1.3.36 identifier: combination of letters, numbers, and underline characters, as specified in 2.1.2, which begins with a letter or underline and which names a *language element*.

1.3.37 in-out variable: *variable* that is declared in a VAR_IN_OUT . . . END_VAR block.

1.3.38 initial value: the value assigned to a variable at system start-up.

1.3.39 input variable (input): variable which is used to supply an argument to a *program organization unit*.

1.3.40 instance: individual, named copy of the data structure associated with a *function block type* or *program type*, which persists from one *invocation* of the associated operations to the next.

1.3.41 instance name: *identifier* associated with a specific *instance*.

1.3.42 instantiation: the creation of an *instance*.

1.3.43 integer literal: *literal* which directly represents a value of type SINT, INT, DINT, LINT, BOOL, BYTE, WORD, DWORD, or LWORD, as defined in 2.3.1.

1.3.44 invocation: the process of initiating the execution of the operations specified in a *program organization unit*.

1.3.45 keyword: lexical unit that characterizes a *language element*, for example, "IF".

1.3.46 label: language construction naming an instruction, network, or group of networks, and including an *identifier*.

1.3.47 language element: any item identified by a symbol on the left-hand side of a production rule in the formal specification given in annex B of this standard.

1.3.48 literal: lexical unit that directly represents a value. (ISO)

1.3.49 local scope: the *scope* of a *declaration* or *label* applying only to the *program organization unit* in which the declaration or label appears.

1.3.50 logical location: the location of a *hierarchically addressed* variable in a schema which may or may not bear any relation to the physical structure of the programmable controller's inputs, outputs, and memory.

1.3.51 long real: real number represented in a *long word*.

1.3.52 long word: 64-bit data element.

1.3.53 memory (user data storage): functional unit to which the user program can store data and from which it can retrieve the stored data.

1.3.54 named element: element of a *structure* which is named by its associated *identifier*.

1.3.55 network: arrangement of nodes and interconnecting branches.

1.3.56 off-delay (on-delay) timer function block: *function block* which delays the *falling (rising) edge* of a Boolean *input* by a specified duration.

1.3.57 operand: *language element* on which an operation is performed.

1.3.58 operator: symbol that represents the action to be performed in an operation.

1.3.59 output variable (output): *variable* which is used to return the result(s) of the *evaluation* of a *program organization unit*.

1.3.60 overloaded: with respect to an operation or *function*, capable of operating on data of different types, as specified in 2.5.1.4.

1.3.61 power flow: the symbolic flow of electrical power in a ladder diagram, used to denote the progression of a logic solving algorithm.