

BS EN 61804-3:2007



BSI Standards Publication

Function blocks (FB) for process control —

Part 3: Electronic Device Description
Language (EDDL)

bsi.

...making excellence a habit.
AVA Documents™
Document Sharing Hub

National foreword

This British Standard is the UK implementation of EN 61804-3:2007. It is identical to IEC 61804-3:2006. Together with BS EN 61804-2:2007, it supersedes BS EN 61804-2:2004 which is withdrawn.

The UK participation in its preparation was entrusted to Technical Committee AMT/7, Industrial communications: process measurement and control, including fieldbus.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© BSI 2010

ISBN 978 0 580 70027 9

ICS 25.040.40; 35.240.50

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 28 February 2010

Amendments issued since publication

Amd. No.	Date	Text affected
----------	------	---------------

EUROPEAN STANDARD
NORME EUROPÉENNE
EUROPÄISCHE NORM

EN 61804-3

June 2007

ICS 25.040.40; 35.240.50

Partially supersedes EN 61804-2:2004

English version

Function blocks (FB) for process control - Part 3: Electronic Device Description Language (EDDL) (IEC 61804-3:2006)

Blocs fonction
pour les processus industriels -
Partie 3: Langage de description
électronique du produit (EDDL)
(CEI 61804-3:2006)

Funktionsbausteine
für die Prozessautomation -
Teil 3: Elektronische
Gerätebeschreibungssprache (EDDL)
(IEC 61804-3:2006)

This European Standard was approved by CENELEC on 2007-05-01. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the Central Secretariat or to any CENELEC member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CENELEC member into its own language and notified to the Central Secretariat has the same status as the official versions.

CENELEC members are the national electrotechnical committees of Austria, Belgium, Bulgaria, Cyprus, the Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, the Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland and the United Kingdom.

CENELEC

European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für Elektrotechnische Normung

Central Secretariat: rue de Stassart 35, B - 1050 Brussels

© 2007 CENELEC - All rights of exploitation in any form and by any means reserved worldwide for CENELEC members.

Ref. No. EN 61804-3:2007 E

 **Documents**
Document Sharing Hub

Foreword

The text of the International Standard IEC 61804-3:2006, prepared by SC 65C, Industrial networks, of IEC TC 65, Industrial-process measurement and control, was submitted to the formal vote and was approved by CENELEC as EN 61804-3 on 2007-05-01 without any modification.

This European Standard supersedes the EDDL specification given in EN 61804-2:2004.

The following dates were fixed:

- latest date by which the EN has to be implemented
at national level by publication of an identical
national standard or by endorsement (dop) 2008-05-01
- latest date by which the national standards conflicting
with the EN have to be withdrawn (dow) 2010-05-01

Annex ZA has been added by CENELEC.

Endorsement notice

The text of the International Standard IEC 61804-3:2006 was approved by CENELEC as a European Standard without any modification.

In the official version, for Bibliography, the following note has to be added for the standard indicated:

IEC 61131-3	NOTE	Harmonized as EN 61131-3:2003 (not modified).
-------------	------	---

Annex ZA (normative)

Normative references to international publications with their corresponding European publications

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.

NOTE Where an International Publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
IEC 61499-1	2005	Function blocks – Part 1: Architecture	EN 61499-1	2005
IEC/TS 61804-1	2003	Function blocks (FB) for process control – Part 1: Overview of system aspects	-	-
ISO/IEC 2022	- ¹⁾	Information technology - Character code structure and extension techniques	-	-
ISO/IEC 2375	2003	Information technology - Procedure for registration of escape sequences and coded character sets	-	-
ISO/IEC 7498-1	- ¹⁾	Information technology - Open systems interconnection - Basic reference model – Part 1: The basic model	EN ISO/IEC 7498-1	1995 ²⁾
ISO/IEC 8859-1	1998	Information technology - 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No.1	-	-
ISO/IEC 9899	- ¹⁾	Programming languages - C	-	-
ISO/IEC 10646-1	2000	Information technology - Universal Multiple- Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane	-	-
ISO 639	Series	Codes for the representation of names of languages	-	-
ISO 3166	Series	Codes for the representation of names of countries and their subdivisions	-	-
IEEE 754	1985	Binary Floating-Point Arithmetic (R1990)	-	-

¹⁾ Undated reference.

²⁾ Valid edition at date of issue.

CONTENTS

INTRODUCTION	18
1 Scope	19
2 Normative references	19
3 Terms, definitions, abbreviated terms and acronyms	20
4 Conformance statement	26
5 Conventions for lexical structures	27
6 EDD and EDDL model	28
6.1 Overview of EDD and EDDL	28
6.2 EDD architecture	28
6.3 Concepts of EDD	28
6.4 Principles of the EDD development process	29
6.5 Interrelations between the lexical structure and formal definitions	30
6.6 Builtins	30
6.7 Profiles	30
7 Electronic Device Description Language	30
7.1 Overview	30
7.2 EDD identification information	40
7.3 AXIS	43
7.4 BLOCK	44
7.5 CHART	53
7.6 COLLECTION	56
7.7 COMMAND	57
7.8 CONNECTION	63
7.9 DOMAIN	64
7.10 EDIT_DISPLAY	65
7.11 FILE	67
7.12 GRAPH	67
7.13 GRID	69
7.14 IMAGE	72
7.15 IMPORT	74
7.16 LIKE	85
7.17 LIST	86
7.18 MENU	87
7.19 METHOD	98
7.20 PROGRAM	100
7.21 RECORD	101
7.22 REFERENCE_ARRAY	101
7.23 Relations	102
7.24 RESPONSE_CODES	103
7.25 SOURCE	104
7.26 VALUE_ARRAY	107
7.27 VARIABLE	108
7.28 VARIABLE_LIST	123
7.29 WAVEFORM	124

7.30 Common attributes	131
7.31 Output redirection (OPEN and CLOSE)	134
7.32 Conditional expression	134
7.33 Referencing	135
7.34 Strings	141
7.35 Expression	143
7.36 Text dictionary	149
Annex A (normative) EDDL formal definition	150
Annex B (normative) EDDL Builtin Library	224
Annex C (informative) EDD Example	337
Annex D (normative) Profiles of EDDL and Builtins	350
Annex E (informative) Historical background	376
Bibliography	377
Figure 1 – Position of the IEC 61804 series related to other standards and products	18
Figure 2 – EDD generation process	29
Figure 3 – BLOCK_A	32
Figure 4 – CHART	32
Figure 5 – COLLECTION	33
Figure 6 – COMMAND	33
Figure 7 – DOMAIN	33
Figure 8 – EDIT_DISPLAY	34
Figure 9 – FILE	34
Figure 10 – GRAPH	34
Figure 11 – GRID	34
Figure 12 – IMAGE	35
Figure 13 – LIKE	35
Figure 14 – LIST	35
Figure 15 – MENU	36
Figure 16 – PROGRAM	36
Figure 17 – RECORD	36
Figure 18 – REFERENCE_ARRAY	37
Figure 19 – REFRESH	37
Figure 20 – UNIT	37
Figure 21 – WRITE_AS_ONE	37
Figure 22 – SOURCE	38
Figure 23 – VALUE_ARRAY	38
Figure 24 – VARIABLE	38
Figure 25 – VARIABLE_LIST	38
Figure 26 – WAVEFORM	39
Figure 27 – EDDL import mechanisms	74
Figure 28 – MENU activation (ACCESS OFFLINE)	94

Figure 29 – Action performed after a new value is entered	95
Figure 30 – Action performed after all VARIABLE inputs of the MENU are accepted (ACCESS OFFLINE)	95
Figure 31 – Method execution	95
Figure 32 – MENU activation (ACCESS ONLINE)	96
Figure 33 – Cyclic reading of dynamic VARIABLES (ACCESS ONLINE)	97
Figure 34 – Action performed after all VARIABLE inputs of the MENU are accepted (ACCESS ONLINE)	97
Figure 35 – Time for read-and-write operation	122
Figure C.1 – Example of an operator screen using EDD.....	337
Table 1 – Field attribute descriptions	27
Table 2 – DD_REVISION attribute	40
Table 3 – DEVICE_REVISION attribute	41
Table 4 – DEVICE_TYPE attribute.....	41
Table 5 – EDD_PROFILE attribute	41
Table 6 – EDD_VERSION attribute.....	42
Table 7 – MANUFACTURER attribute.....	42
Table 8 – MANUFACTURER_EXT attribute	42
Table 9 – AXIS attributes	43
Table 10 – MAX_VALUE, MIN_VALUE attribute.....	43
Table 11 – SCALING attribute	44
Table 12 – BLOCK_A attributes.....	45
Table 13 – CHARACTERISTIC attribute	45
Table 14 – PARAMETER attributes	46
Table 15 – AXIS_ITEMS attribute	46
Table 16 – CHART_ITEMS attribute	46
Table 17 – COLLECTION_ITEMS attribute	47
Table 18 – EDIT_DISPLAY_ITEMS attribute.....	47
Table 19 – FILE_ITEMS attribute.....	47
Table 20 – GRAPH_ITEMS attribute.....	48
Table 21 – GRID_ITEMS attribute	48
Table 22 – IMAGE_ITEMS attribute	48
Table 23 – LIST_ITEMS attribute.....	48
Table 24 – MENU_ITEMS attribute	49
Table 25 – METHOD_ITEMS attribute	49
Table 26 – PARAMETER_LISTS attributes	50
Table 27 – REFERENCE_ARRAY_ITEMS attribute.....	50
Table 28 – REFRESH_ITEMS attribute.....	50
Table 29 – SOURCE_ITEMS attribute	51
Table 30 – UNIT_ITEMS attribute.....	51
Table 31 – WAVEFORM_ITEMS attribute.....	51
Table 32 – WRITE_AS_ONE_ITEMS attribute	51
Table 33 – BLOCK_B attributes.....	

Table 34 – NUMBER attributes	52
Table 35 – TYPE attributes	53
Table 36 – CHART attributes	53
Table 37 – CYCLE_TIME attribute	54
Table 38 – HEIGHT/WIDTH attribute	54
Table 39 – LENGTH attribute	55
Table 40 – TYPE attribute	55
Table 41 – COLLECTION attributes	56
Table 42 – item-type	56
Table 43 – COMMAND attributes	58
Table 44 – OPERATION attribute	58
Table 45 – TRANSACTION attributes	59
Table 46 – REPLY and REQUEST attributes	60
Table 47 – INDEX attribute	61
Table 48 – BLOCK_B attribute	61
Table 49 – NUMBER attribute	61
Table 50 – SLOT attribute	62
Table 51 – CONNECTION attribute	62
Table 52 – HEADER attribute	62
Table 53 – MODULE attribute	63
Table 54 – CONNECTION attribute	63
Table 55 – APPINSTANCE attribute	63
Table 56 – DOMAIN attributes	64
Table 57 – HANDLING attribute	64
Table 58 – EDIT_DISPLAY attributes	65
Table 59 – EDIT_ITEMS attribute	65
Table 60 – DISPLAY_ITEM attributes	66
Table 61 – POST_EDIT_ACTIONS, PRE_EDIT_ACTIONS attribute	67
Table 62 – FILE attributes	67
Table 63 – GRAPH attributes	68
Table 64 – HEIGHT/WIDTH attribute	68
Table 65 – CYCLE_TIME attribute	69
Table 66 – X_AXIS attribute	69
Table 67 – GRID attributes	70
Table 68 – VECTORS attribute	70
Table 69 – HANDLING attribute	71
Table 70 – HEIGHT/WIDTH attribute	71
Table 71 – ORIENTATION attribute	72
Table 72 – VALIDITY attributes	72
Table 73 – IMAGE attributes	72
Table 74 – PATH attribute	73
Table 75 – LINK attribute	73
Table 76 – VALIDITY attributes	73

Table 77 – Importing Device Description.....	75
Table 78 – Redefinition attributes	76
Table 79 – Redefinition rules for AXIS attributes.....	76
Table 80 – Redefinition rules for BLOCK_A attributes	77
Table 81 – Redefinition rules for BLOCK_B attributes	77
Table 82 – Redefinition rules for CHART attributes	78
Table 83 – Redefinition rules for COLLECTION attributes.....	78
Table 84 – Redefinition rules for COMMAND attributes.....	78
Table 85 – Redefinition rules for CONNECTION attributes.....	79
Table 86 – Redefinition rules for DOMAIN attributes	79
Table 87 – Redefinition rules for EDIT_DISPLAY attributes.....	79
Table 88 – Redefinition rules for FILE attributes	79
Table 89 – Redefinition rules for GRAPH attributes.....	80
Table 90 – Redefinition rules for GRID attributes	80
Table 91 – Redefinition rules for IMAGE attributes.....	80
Table 92 – Redefinition rules for LIST attributes	81
Table 93 – Redefinition rules for MENU attributes.....	81
Table 94 – Redefinition rules for METHOD attributes	82
Table 95 – Redefinition rules for PROGRAM attributes	82
Table 96 – Redefinition rules for RECORD attributes	82
Table 97 – Redefinition rules for REFERENCE_ARRAY attributes	82
Table 98 – Redefinition rules for RESPONSE_CODES attributes	83
Table 99 – Redefinition rules for SOURCE attributes	83
Table 100 – Redefinition rules for VALUE_ARRAY attributes	83
Table 101 – Redefinition rules for VARIABLE attributes	84
Table 102 – Redefinition rules for VARIABLE_LIST attributes	85
Table 103 – Redefinition rules for WAVEFORM attributes.....	85
Table 104 – LIKE attributes	86
Table 105 – LIST attributes	86
Table 106 – TYPE attribute	86
Table 107 – CAPACITY, COUNT attribute	87
Table 108 – MENU attribute	88
Table 109 – ITEMS attribute.....	89
Table 110 – ACCESS attribute	89
Table 111 – ENTRY attribute.....	90
Table 112 – POST_EDIT_ACTIONS, PRE_EDIT_ACTIONS, POST_READ_ACTIONS, PRE_READ_ACTIONS, POST_WRITE_ACTIONS, PRE_WRITE_ACTIONS attributes.....	90
Table 113 – PURPOSE attribute.....	92
Table 114 – ROLE attribute	93
Table 115 – STYLE attribute	93
Table 116 – VALIDITY attributes	94
Table 117 – METHOD attributes.....	98
Table 118 – ACCESS attributes.....	98
Table 119 – TYPE attributes.....	99

Table 120 – VALIDITY attributes	99
Table 121 – PROGRAM attributes	100
Table 122 – ARGUMENT attribute	100
Table 123 – RECORD attributes	101
Table 124 – REFERENCE_ARRAY attribute	101
Table 125 – ELEMENTS attribute	102
Table 126 – REFRESH attributes	102
Table 127 – UNIT attributes	103
Table 128 – WRITE_AS_ONE attribute	103
Table 129 – RESPONSE_CODES attributes	104
Table 130 – SOURCE attributes	104
Table 131 – EMPHASIS attribute	105
Table 132 – LINE_COLOR attribute	105
Table 133 – LINE_TYPE attribute	106
Table 134 – Y_AXIS attribute	106
Table 135 – VALUE_ARRAY attributes	107
Table 136 – NUMBER_OF_ELEMENTS attribute	108
Table 137 – TYPE attribute	108
Table 138 – VARIABLE attributes	108
Table 139 – CLASS attributes	109
Table 140 – TYPE attributes	110
Table 141 – DOUBLE, FLOAT, INTEGER, UNSIGNED_INTEGER attributes	112
Table 142 – BIT_ENUMERATED attributes	114
Table 143 – status-class attributes	115
Table 144 – ALL, AO, DV, TV attributes	116
Table 145 – Enumerated types attributes	116
Table 146 – Index type attributes	117
Table 147 – Object reference type attribute	117
Table 148 – DEFAULT_REFERENCE attributes	117
Table 149 – String types attributes	119
Table 150 – CONSTANT_UNIT attribute	119
Table 151 – DEFAULT_VALUE attribute	119
Table 152 – HANDLING attribute	120
Table 153 – INITIAL_VALUE attribute	120
Table 154 – POST_EDIT_ACTIONS, PRE_EDIT_ACTIONS, POST_READ_ACTIONS, PRE_READ_ACTIONS, POST_WRITE_ACTIONS, PRE_WRITE_ACTIONS, REFRESH_ACTIONS attributes	121
Table 155 – READ/WRITE_TIMEOUT attributes	123
Table 156 – STYLE attribute	123
Table 157 – VALIDITY attributes	123
Table 158 – VARIABLE_LIST attributes	124
Table 159 – WAVEFORM attributes	124
Table 160 – TYPE attribute	125
Table 161 – XY attribute	125

Table 162 – YT attribute	126
Table 163 – HORIZONTAL attribute	127
Table 164 – VERTICAL attribute	127
Table 165 – EMPHASIS attribute	127
Table 166 – HANDLING attribute	128
Table 167 – EXIT_ACTIONS, INIT_ACTIONS, REFRESH_ACTIONS attribute	128
Table 168 – KEY_POINTS attribute	129
Table 169 – X_VALUES, Y_VALUES attribute	129
Table 170 – LINE_COLOR attribute	130
Table 171 – LINE_TYPE attribute	130
Table 172 – Y_AXIS attribute	131
Table 173 – DEFINITION attributes	131
Table 174 – HELP attribute	132
Table 175 – LABEL attribute	132
Table 176 – MEMBERS attributes	133
Table 177 – RESPONSE_CODES attribute	133
Table 178 – OPEN and CLOSE attributes	134
Table 179 – IF, SELECT conditional	135
Table 180 – Referencing an EDD instance	136
Table 181 – Referencing elements of VARIABLE	136
Table 182 – Referencing elements of RECORD	136
Table 183 – Referencing elements of VALUE_ARRAY	137
Table 184 – Referencing members of COLLECTION	137
Table 185 – Referencing members of REFERENCE_ARRAY	137
Table 186 – Referencing members of VARIABLE_LISTS	138
Table 187 – Referencing members of a BLOCK_A PARAMETERS	138
Table 188 – Referencing members of BLOCK_A PARAMETER_LISTS	138
Table 189 – Referencing members of BLOCK_A LOCAL_PARAMETER	138
Table 190 – Referencing BLOCK_A CHARACTERISTICS	139
Table 191 – Referencing members of FILE	139
Table 192 – Referencing elements of LIST	139
Table 193 – Referencing members of CHART	140
Table 194 – Referencing members of GRAPH	140
Table 195 – Referencing members of SOURCE	140
Table 196 – Referencing AXIS of a GRAPH. SOURCE, WAVEFORM	141
Table 197 – String as a string literal	141
Table 198 – String as a string variable	141
Table 199 – String as an enumeration value	142
Table 200 – String as a dictionary reference	142
Table 201 – Referencing HELP and LABEL attributes of EDD instances	142
Table 202 – String operation	143
Table 203 – Format specifier	143
Table 204 – Primary expressions	144

Table 205 – Attribute values of VARIABLES	144
Table 206 – AXIS Attribute Values	145
Table 207 – LIST Attribute Values	145
Table 208 – Unary expressions	145
Table 209 – Multiplicative operators	146
Table 210 – Additive operators	146
Table 211 – Shift operators	147
Table 212 – Relational operators	147
Table 213 – Equality operators	147
Table 214 – Text dictionary attributes	149
Table A.1 – Conventions for integer constants	154
Table A.2 – Using escape sequences in string literals	155
Table A.3 – Using language codes in string literals	155
Table A.4 – EDDL operators	156
Table A.5 – EDDL keywords	156
Table B.1 – Format for the Builtins lexical element tables	224
Table B.2 – Contents of the lexical element table	224
Table B.3 – Builtin abort	225
Table B.4 – Builtin abort_on_all_comm_errors	225
Table B.5 – Builtin ABORT_ON_ALL_COMM_STATUS	226
Table B.6 – Builtin ABORT_ON_ALL_DEVICE_STATUS	226
Table B.7 – Builtin ABORT_ON_ALL_RESPONSE_CODES	227
Table B.8 – Builtin abort_on_all_response_codes	227
Table B.9 – Builtin abort_on_comm_error	227
Table B.10 – Builtin ABORT_ON_COMM_ERROR	228
Table B.11 – Builtin ABORT_ON_COMM_STATUS	228
Table B.12 – Builtin ABORT_ON_DEVICE_STATUS	229
Table B.13 – Builtin ABORT_ON_NO_DEVICE	229
Table B.14 – Builtin ABORT_ON_RESPONSE_CODE	230
Table B.15 – Builtin abort_on_response_code	231
Table B.16 – Builtin abs	231
Table B.17 – Builtin ACKNOWLEDGE	232
Table B.18 – Builtin acknowledge	232
Table B.19 – Builtin acos	232
Table B.20 – Builtin add_abort_method	233
Table B.21 – Builtin add_abort_method	233
Table B.22 – Builtin asin	234
Table B.23 – Builtin assign	234
Table B.24 – Builtin assign_double	235
Table B.25 – Builtin assign_float	235
Table B.26 – Builtin assign_int	235
Table B.27 – Builtin assign_var	236
Table B.28 – Builtin atan	236

Table B.29 – Builtin ATOF	236
Table B.30 – Builtin atof	237
Table B.31 – Builtin ATOL	237
Table B.32 – Builtin atoi	237
Table B.33 – Builtin BUILD_MESSAGE	238
Table B.34 – Builtin cbrt	238
Table B.35 – Builtin ceil	238
Table B.36 – Builtin cos	239
Table B.37 – Builtin cosh	239
Table B.38 – Builtin dassign	239
Table B.39 – Builtin Date_to_DayOfMonth	240
Table B.40 – Builtin Date_to_Month	240
Table B.41 – Builtin Date_to_Year	240
Table B.42 – Builtin DELAY	241
Table B.43 – Builtin delay	241
Table B.44 – Builtin DELAY_TIME	242
Table B.45 – Builtin delayfor	242
Table B.46 – Builtin DICT_ID	243
Table B.47 – Builtin discard_on_exit	243
Table B.48 – Builtin display	244
Table B.49 – Builtin display_builtin_error	244
Table B.50 – Builtin display_comm_error	245
Table B.51 – Builtin display_comm_status	245
Table B.52 – Builtin display_device_status	245
Table B.53 – Builtin display_dynamics	246
Table B.54 – Builtin display_message	246
Table B.55 – Builtin display_response_code	247
Table B.56 – Builtin display_response_status	248
Table B.57 – Builtin display_xmtr_status	248
Table B.58 – Builtin edit_device_value	249
Table B.59 – Builtin edit_local_value	250
Table B.60 – Builtin exp	250
Table B.61 – Builtin ext_send_command	251
Table B.62 – Builtin ext_send_command_trans	251
Table B.63 – Builtin fail_on_all_comm_errors	252
Table B.64 – Builtin fail_on_all_response_codes	252
Table B.65 – Builtin fail_on_comm_error	253
Table B.66 – Builtin fail_on_response_code	253
Table B.67 – Builtin fassign	254
Table B.68 – Builtin fgetval	254
Table B.69 – Builtin float_value	254
Table B.70 – Builtin floor	255
Table B.71 – Builtin fmod	255

Table B.72 – Builtin fsetval	256
Table B.73 – Builtin ftoa	256
Table B.74 – Builtin fvar_value	256
Table B.75 – Builtin get_acknowledgement	257
Table B.76 – Builtin get_comm_error	258
Table B.77 – Builtin get_comm_error_string	258
Table B.78 – Builtin get_date	259
Table B.79 – Builtin get_date_value	259
Table B.80 – Builtin get_dds_error	260
Table B.81 – Builtin GET_DEV_VAR_VALUE	260
Table B.82 – Builtin get_dev_var_value	261
Table B.83 – Builtin get_dictionary_string	262
Table B.84 – Builtin get_double	262
Table B.85 – Builtin get_double_value	263
Table B.86 – Builtin get_enum_string	263
Table B.87 – Builtin get_float	264
Table B.88 – Builtin get_float_value	264
Table B.89 – Builtin GET_LOCAL_VAR_VALUE	265
Table B.90 – Builtin get_local_var_value	265
Table B.91 – Builtin get_more_status	266
Table B.92 – Builtin get_resolve_status	266
Table B.93 – Builtin get_response_code	267
Table B.94 – Builtin get_response_code_string	268
Table B.95 – Builtin get_signed	268
Table B.96 – Builtin get_signed_value	269
Table B.97 – Builtin get_status_code_string	269
Table B.98 – Builtin get_status_string	270
Table B.99 – Builtin get_stddict_string	270
Table B.100 – Builtin get_string	271
Table B.101 – Builtin get_string_value	271
Table B.102 – Builtin GET_TICK_COUNT	272
Table B.103 – Builtin get_unsigned	272
Table B.104 – Builtin get_unsigned_value	273
Table B.105 – Builtin iassign	273
Table B.106 – Builtin igetval	274
Table B.107 – Builtin IGNORE_ALL_COMM_STATUS	274
Table B.108 – Builtin IGNORE_ALL_DEVICE_STATUS	275
Table B.109 – Builtin IGNORE_ALL_RESPONSE_CODES	275
Table B.110 – Builtin IGNORE_COMM_ERROR	276
Table B.111 – Builtin IGNORE_COMM_STATUS	276
Table B.112 – Builtin IGNORE_DEVICE_STATUS	277
Table B.113 – Builtin IGNORE_NO_DEVICE	277
Table B.114 – Builtin IGNORE_RESPONSE_CODE	278

Table B.115 – Builtin int_value	278
Table B.116 – Builtin is_NaN	279
Table B.117 – Builtin isetval	279
Table B.118 – Builtin ITEM_ID	280
Table B.119 – Builtin ITOA	280
Table B.120 – Builtin itoa	280
Table B.121 – Builtin ivar_value	281
Table B.122 – Builtin lassign	281
Table B.123 – Builtin lgetval	281
Table B.124 – Builtin ListDeleteElementAt	282
Table B.125 – Builtin ListInsert	282
Table B.126 – Builtin log	283
Table B.127 – Builtin log10	283
Table B.128 – Builtin log2	284
Table B.129 – Lexical elements of Builtin LOG_MESSAGE	284
Table B.130 – Builtin long_value	284
Table B.131 – Builtin lsetval	285
Table B.132 – Builtin lvar_value	285
Table B.133 – Builtin MEMBER_ID	286
Table B.134 – Builtin MenuDisplay	286
Table B.135 – Builtin method_abort	287
Table B.136 – Builtin process_abort	287
Table B.137 – Builtin pow	287
Table B.138 – Builtin process_abort	288
Table B.139 – Builtin put_date	288
Table B.140 – Builtin put_date_value	289
Table B.141 – Builtin put_double	289
Table B.142 – Builtin put_double_value	290
Table B.143 – Builtin put_float	290
Table B.144 – Builtin put_float_value	291
Table B.145 – Builtin PUT_MESSAGE	291
Table B.146 – Builtin put_message	292
Table B.147 – Builtin put_signed	292
Table B.148 – Builtin put_signed_value	293
Table B.149 – Builtin put_string	294
Table B.150 – Builtin put_string_value	294
Table B.151 – Builtin put_unsigned	295
Table B.152 – Builtin put_unsigned_value	296
Table B.153 – Lexical elements of Builtin READ_COMMAND	296
Table B.154 – Builtin read_value	297
Table B.155 – Builtin remove_abort_method	297
Table B.156 – Builtin remove_abort_method	298
Table B.157 – Builtin remove_all_abort_methods	298

Table B.158 – Builtin resolve_array_ref	299
Table B.159 – Builtin resolve_block_ref	299
Table B.160 – Builtin resolve_param_list_ref	300
Table B.161 – Builtin resolve_parm_ref	300
Table B.162 – Builtin resolve_record_ref	301
Table B.163 – Builtin retry_on_all_comm_errors	301
Table B.164 – Builtin RETRY_ON_ALL_COMM_STATUS	302
Table B.165 – Builtin RETRY_ON_ALL_DEVICE_STATUS	302
Table B.166 – Builtin RETRY_ON_ALL_RESPONSE_CODES	303
Table B.167 – Builtin retry_on_all_response_codes	303
Table B.168 – Builtin RETRY_ON_COMM_ERROR	304
Table B.169 – Builtin retry_on_comm_error	304
Table B.170 – Builtin RETRY_ON_COMM_STATUS	305
Table B.171 – Builtin RETRY_ON_DEVICE_STATUS	305
Table B.172 – Builtin RETRY_ON_NO_DEVICE	306
Table B.173 – Builtin RETRY_ON_RESPONSE_CODE	307
Table B.174 – Builtin retry_on_response_code	307
Table B.175 – Builtin round	307
Table B.176 – Builtin rspcode_string	308
Table B.177 – Builtin save_on_exit	309
Table B.178 – Builtin save_values	309
Table B.179 – Builtin SELECT_FROM_LIST	309
Table B.180 – Builtin select_from_list	310
Table B.181 – Builtin select_from_menu	311
Table B.182 – Builtin send	311
Table B.183 – Builtin send_all_values	312
Table B.184 – Builtin send_command	312
Table B.185 – Builtin send_command_trans	313
Table B.186 – Builtin send_on_exit	314
Table B.187 – Builtin send_trans	314
Table B.188 – Builtin send_value	315
Table B.189 – Builtin SET_NUMBER_OF_RETRIES	315
Table B.190 – Builtin sin	315
Table B.191 – Builtin sinh	316
Table B.192 – Builtin sqrt	316
Table B.193 – Builtin strcmp	316
Table B.194 – Builtin strlen	317
Table B.195 – Builtin strlwr	317
Table B.196 – Builtin strmid	318
Table B.197 – Builtin strstr	318
Table B.198 – Builtin strtrim	318
Table B.199 – Builtin strupr	319
Table B.200 – Builtin tan	319

Table B.201 – Builtin tanh	319
Table B.202 – Builtin To_Date_and_Time	320
Table B.203 – Builtin trunc	320
Table B.204 – Builtin VARID	321
Table B.205 – Builtin vassign	321
Table B.206 – Builtin WRITE_COMMAND	321
Table B.207 – Builtin XMTR_ABORT_ON_ALL_COMM_STATUS	322
Table B.208 – Builtin XMTR_ABORT_ON_ALL_DEVICE_STATUS	323
Table B.209 – Builtin XMTR_ABORT_ON_ALL_RESPONSE_CODES	323
Table B.210 – Builtin XMTR_ABORT_ON_COMM_ERROR	324
Table B.211 – Builtin XMTR_ABORT_ON_COMM_STATUS	324
Table B.212 – Builtin XMTR_ABORT_ON_DATA	325
Table B.213 – Builtin XMTR_ABORT_ON_DEVICE_STATUS	325
Table B.214 – Builtin XMTR_ABORT_ON_NO_DEVICE	326
Table B.215 – Builtin XMTR_ABORT_ON_RESPONSE_CODE	326
Table B.216 – Builtin XMTR_IGNORE_ALL_COMM_STATUS	327
Table B.217 – Builtin XMTR_IGNORE_ALL_DEVICE_STATUS	327
Table B.218 – Builtin XMTR_IGNORE_ALL_RESPONSE_CODES	328
Table B.219 – Builtin XMTR_IGNORE_COMM_ERROR	328
Table B.220 – Builtin XMTR_IGNORE_COMM_STATUS	329
Table B.221 – Builtin XMTR_IGNORE_DEVICE_STATUS	329
Table B.222 – Builtin XMTR_IGNORE_NO_DEVICE	330
Table B.223 – Builtin XMTR_IGNORE_RESPONSE_CODE	330
Table B.224 – Builtin XMTR_RETRY_ON_ALL_DEVICE_STATUS	331
Table B.225 – Builtin XMTR_RETRY_ON_ALL_RESPONSE_CODE	332
Table B.226 – Builtin XMTR_RETRY_ON_ALL_RESPONSE_CODES	332
Table B.227 – Builtin XMTR_RETRY_ON_COMM_ERROR	333
Table B.228 – Builtin XMTR_RETRY_ON_COMM_STATUS	333
Table B.229 – Builtin XMTR_RETRY_ON_DATA	334
Table B.230 – Builtin XMTR_RETRY_ON_DEVICE_STATUS	334
Table B.231 – Builtin XMTR_RETRY_ON_NO_DEVICE	335
Table B.232 – Builtin XMTR_RETRY_ON_RESPONSE_CODE	335
Table B.233 – Builtin YearMonthDay_to_Date	336
Table B.234 – Contents of the return codes description table	336
Table B.235 – Return Code Description	336
Table D.1 – Profile selection tables	350
Table D.2 – EDDL Formal Definition profile tables	350
Table D.3 – Contents of selection tables	350
Table D.4 – EDDL element selection for PROFIBUS	351
Table D.5 – Builtin profile for PROFIBUS	354
Table D.6 – EDDL element selection for Fieldbus Foundation	357
Table D.7 – Builtin profile for Fieldbus Foundation	361
Table D.8 – EDDL element selection for HCF	365

Table D.9 – Builtin profile for HCF	368
Table D.10 – METHOD DEFINITIONS data types	372
Table D.11 – VARIABLE TYPES.....	372
Table D.12 – DATE coding	373
Table D.13 – DATE_AND_TIME coding	373
Table D.14 – DURATION coding	374
Table D.15 – TIME coding	374
Table D.16 – TIME_VALUE coding	374
Table D.17 – PACKED_ASCII coding.....	375

INTRODUCTION

The EDDL fills the gap between the conceptual FB specification of IEC 61804-2 and a product implementation. It allows the manufacturers to use the same description method for devices based on different technologies and platforms. Figure 1 shows these aspects.

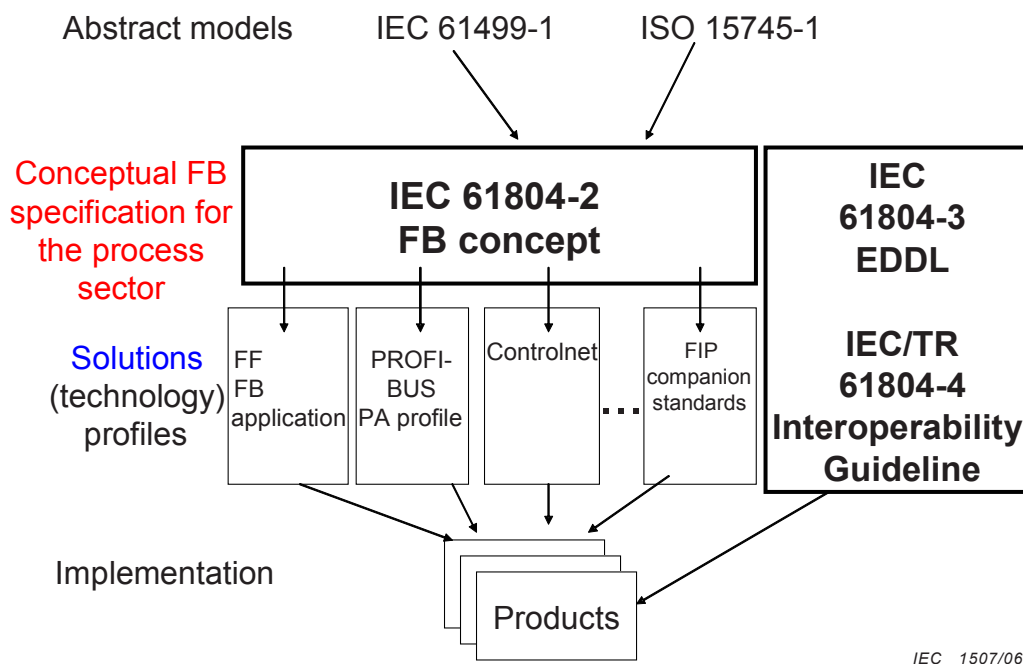


Figure 1 – Position of the IEC 61804 series related to other standards and products

FUNCTION BLOCKS (FB) FOR PROCESS CONTROL –

Part 3: Electronic Device Description Language (EDDL)

1 Scope

This part of IEC 61804 specifies the Electronic Device Description Language (EDDL) technology, which enables the integration of real product details using the tools of the engineering life cycle.

This standard specifies EDDL as a generic language for describing the properties of automation system components. EDDL is capable of describing

- device parameters and their dependencies;
- device functions, for example, simulation mode, calibration;
- graphical representations, for example, menus;
- interactions with control devices
- graphical representations
 - enhanced user interface
 - graphing system
- persistent data store.

EDDL is to be used to create Electronic Device Description (EDD). This EDD is used with appropriate tools to generate interpretative code to support parameter handling, operation, and monitoring of automation system components such as remote I/Os, controllers, sensors, and programmable controllers. Tool implementation is outside the scope of this standard.

This standard specifies the semantic and lexical structure in a syntax-independent manner. A specific syntax is defined in Annex A, but it is possible to use the semantic model also with different syntaxes.

NOTE 1 The EDDL may also be used for the description of product properties in other domains.

The EDDL and the device-related EDD is applicable to industrial automation.

NOTE 2 Industrial automation may include devices such as generic digital and analog input/output modules, motion controllers, human machine interfaces, sensors, closed-loop controllers, encoders, hydraulic valves, and programmable controllers.

This International Standard satisfies the requirements of Clause 9 of IEC 61804-1.

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 61499-1:2005, *Function blocks – Part 1: Architecture*

IEC 61804-1:2003, *Function blocks (FB) for process control – Part 1: Overview of system aspects*

ISO/IEC 2022, *Information technology – Character code structure and extension techniques*

ISO/IEC 2375:2003, *Information technology – Procedure for registration of escape sequences and coded character sets*

ISO/IEC 7498-1, *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model*

ISO/IEC 8859-1:1998, *Information technology – 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No. 1*

ISO/IEC 9899, *Programming languages – C*

ISO/IEC 10646-1:2000, *Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane*

ISO 639, *Codes for the representation of names of languages*

ISO 3166, *Codes for the representation of names of countries and their subdivisions*

IEEE 754:1985 (R1990), *Binary Floating-Point Arithmetic*

3 Terms, definitions, abbreviated terms and acronyms

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions, some of which have been compiled from the referenced documents, apply.

NOTE The terms in IEC 60050-351 apply partially.

3.2

algorithm

finite set of well-defined rules for the solution of a problem in a finite number of operations

3.3

attribute

description of an externally visible characteristic or feature of an object

NOTE The attributes of an object contain information about variable portions of an object. Typically, they provide status information or govern the operation of an object. Attributes may also affect the behaviour of an object. Attributes are divided into class attributes and instance attributes.

3.4

Builtin

predefined subroutine for communication and display executed by the EDD application

3.5

configuration (of a system or device)

step in system design: selecting functional units, assigning their locations and defining their interconnections

[IEC 61499-1]

3.6**data**

representation of facts, concepts or instructions in a formalized manner suitable for communication, interpretation or processing by human beings or by automatic means

[ISO/AFNOR Dictionary of Computer Science]

3.7**data connection**

association established between functional units for conveyance of data

[IEC 61499-1]

3.8**data input**

interface of an FB which receives data from a data connection

[IEC 61499-1]

3.9**data output**

interface of an FB, which supplies data to a data connection

[IEC 61499-1]

3.10**data type**

set of values together with a set of permitted operations

[ISO 2382 series]

3.11**device**

independent physical entity capable of performing one or more specified functions in a particular context and delimited by its interfaces

[IEC 61499-1]

3.12**EDD application**

program using the EDD, or any translated form, which offers functionality like communication representation, data representation, graphical representation, etc.

3.13**EDDL processor**

processor or program which translates the EDD into an executable form that can be processed by an EDD application

3.14**EDDL profile**

selection of the supported elements of the EDDL lexical structure including the syntax definitions for a number of specific consortia

3.15**Electronic Device Description Language****EDDL**

methodology for describing parameter(s) of an automation system component

3.16**Electronic Device Description****EDD**

data collection containing the device parameter(s), their dependencies, their graphical representation and a description of the data sets which are transferred.

NOTE The Electronic Device Description is created using the Electronic Device Description Language (EDDL)

3.17**Electronic Device Description Source****EDDS**

ASCII file containing a specific device description

3.18**Electronic Device Description Technology****EDDT**

technology which includes the EDD development process, the EDD usage and the involved tool chain

3.19**Electronic Device Description Checker**

test tool which checks the Lexical Structure and partly the semantic of EDD sources to guarantee compliance of EDD sources with the EDD language

3.20**Electronic Device Description Compiler**

compiler which translates the EDD source in an internal format that is used by the EDD Interpreter

3.21**Electronic Device Description Interpreter****EDDI**

Interpreter which uses the EDD source or an internal format that is given by the EDDL Compiler to provide the EDD information to the EDD user.

3.22**entity**

particular thing, such as a person, place, process, object, concept, association, or event

[IEC 61499-1]

3.23**event**

instantaneous occurrence that is significant to scheduling the execution of an algorithm

[IEC 61499-1]

NOTE The execution of an algorithm may make use of variables associated with an event.

3.24**exception**

event that causes suspension of normal execution

[IEC 61499-1]

3.25

function

specific purpose of an entity or its characteristic action

[IEC 61499-1]

3.26

functional unit

entity of hardware or software, or both, capable of accomplishing a specified purpose

[ISO/AFNOR Dictionary of Computer Science]

3.27

function block (function block instance)

software functional unit comprising an individual, named copy of a data structure and associated operations specified by a corresponding FB type

[IEC 61499-1]

NOTE Typical operations of a FB include modification of the values of the data in its associated data structure.

3.28

hardware

physical equipment, as opposed to programs, procedures, rules and associated documentation

[ISO/AFNOR Dictionary of Computer Science]

3.29

implementation

development phase in which the hardware and software of a system become operational

[IEC 61499-1]

3.30

input variable

variable whose value is supplied by a data input, and which may be used in one or more operations of a FB

NOTE An input parameter of an FB, as defined in IEC 61131-3, is an input variable.

[IEC 61499-1]

3.31

instance

functional unit comprising an individual, named entity with the attributes of a defined type

[IEC 61499-1]

3.32

instance name

identifier associated with, and designating, an instance

[IEC 61499-1]

3.33

instantiation

creation of an instance of a specified type

[IEC 61499-1]

3.34**interface**

shared boundary between two functional units, defined by functional characteristics, signal characteristics, or other characteristics as appropriate

[IEC 60050-351:1998, 11-19]

3.35**internal variable**

variable whose value is used or modified by one or more operations of an FB but is not supplied by a data input or to a data output

[IEC 61499-1]

3.36**invocation**

process of initiating the execution of the sequence of operations specified in an algorithm

[IEC 61499-1]

3.37**mapping**

set of values having defined correspondence with the quantities or values of another set

[ISO/AFNOR Dictionary of Computer Science]

3.38**model**

representation of a real world process, device, or concept

[IEC 61499-1]

3.39**operation**

well-defined action that, when applied to any permissible combination of known entities, produces a new entity

[ISO/AFNOR Dictionary of Computer Science]

3.40**output variable**

variable whose value is established by one or more operations of a FB, and is supplied to a data output

NOTE An output parameter of an FB, as defined in IEC 61131-3, is an output variable.

[IEC 61499-1]

3.41**parameter**

variable that is given a constant value for a specified application and that may denote the application

[ISO/AFNOR Dictionary of Computer Science]

3.42**preprocessor**

part of the run-time environment that transforms the information given in an EDD

3.43**preprocessor directives**

description of conditions for filtering the EDD code before compilation or interpretation

NOTE For example, a preprocessor directive providing the facility to define names for constants or to write macros to make code easier to read

3.44**resource**

functional unit contained within a device which has independent control of its operation, and which provides various services to applications, including the scheduling and execution of algorithms

NOTE 1 The RESOURCE defined in IEC 61131-3 is a programming language element corresponding to the resource defined above.

NOTE 2 A device contains one or more resources.

3.45**resource management application**

application whose primary function is the management of a single resource

[IEC 61499-1]

3.46**service**

functional capability of a resource, which can be modelled by a sequence of service primitives

[IEC 61499-1]

3.47**software**

intellectual creation comprising the programs, procedures, rules and any associated documentation pertaining to the operation of a system

[IEC 61499-1]

3.48**system**

set of interrelated elements considered in a defined context as a whole and separated from its environment

[IEV 350-11-01:1998]

NOTE 1 Such elements may be both material objects and concepts as well as the results thereof (for example, forms of organization, mathematical methods, and programming languages).

NOTE 2 The system is considered to be separated from the environment and other external systems by an imaginary surface, which can cut the links between them and the considered system.

3.49**text dictionary**

collection of multilingual or other texts within the EDD

NOTE References within an EDD are used to select an appropriate text dictionary.

3.50**type**

software element, which specifies the common attributes shared by all instances of the type

[IEC 61499-1]

3.51**type name**

identifier associated with, and designating, a type

[IEC 61499-1]

3.52**variable**

software entity that may take different values, one at a time

NOTE 1 The values of a variable are usually restricted to a certain data type.

NOTE 2 Variables are described as input variables, output variables, and internal variables.

[IEC 61499-1]

3.53 Abbreviated terms and acronyms

ANSI	American National Standard Institute
ANSI C	American National Standard Institute for the programming language C (see ISO/IEC 9899)
ASCII	American Standard Code for Information Interchange (see ISO/IEC 10646-1)
ASN.1	Abstract Lexical Structure Notation 1
DD	Device Description
EDD	Electronic Device Description
EDDI	Electronic Device Description Interpreter
EDDL	Electronic Device Description Language
EUC	Extended Unit Code (see ISO/IEC 2022)
FB	Function Block
HMI	Human Machine Interface
HTML	Hypertext Mark-up Language
I/O	Input/Output
ID	Identifier
OSI	Open Systems Interconnection
wao	Write as one

4 Conformance statement

A statement of conformance to a profile of this International Standard shall be stated² as either

- compliance to IEC 61804-3:2006 Annex D.n <Type>
- or
- compliance to IEC 61804-3/Ed1.0 Annex D.n <Type>

where the Type within the angle brackets < > is optional and the angle brackets are not to be included.

NOTE The Type may be, for example, the name of the communication protocol used like PROFIBUS, Foundation Fieldbus, or HART.

² In accordance with the ISO/IEC Directives.

Product standards shall not include any conformity assessment aspects (including QM provisions), neither normative nor informative, other than provisions for product testing (evaluation and examination).

5 Conventions for lexical structures

The EDDL is generally described using lexical structures in which the elements and the presence of fields are specified. A general form of the lexical structures is shown below.

ABC field1, field2

ABC is a lexical element. This element shall be coded in a concrete syntax. It is not required to code this element with the name “ABC”. It is also possible to code this element, for example, with a tag number.

Field1 and field2 are fields of the lexical element ABC. Each field is mandatory and may have more than one attribute. If a field has attributes, the presence of the attributes is specified in a table. A comma separates Field1 and 2. The comma is a lexical element and is not coded explicitly.

If a field has additional attributes, the attributes are defined in a table. The table layout and the possible usage qualifiers are shown in Table 1.

Table 1 – Field attribute descriptions

Usage	Attribute	Description
m	yyy	The presence of this attribute is mandatory
o	xxx	The presence of this attribute is optional
s	z1	The presence of this attribute is selectable with other attributes, which are also marked with “s” in the usage column). Only one of the selectable attributes (z1 or z2) is present
s	z2	The presence of this attribute is selectable with other attributes, which are also marked with “s” in the usage column). Only one of the selectable attributes (z1 or z2) is present
c	uuu	The presence of this attribute is conditional and it is only present if the condition is true

The characters in the Usage column have the following meanings.

- m: this attribute is mandatory and shall be present.
- o: this attribute is optional and need not be present.
- s: this attribute is a selection. One, and only one, of the fields marked with s (Z1 or Z2) shall be present.
- c: this attribute is conditional. The condition is described in the Description column.

In the Attribute column, if more than one attribute exists, which have the same usage, the attributes are sorted in alphabetical order.

ABC field1+

The plus (+) behind field1 is used to indicate that field1 is used at least once. It may be used more than once.

ABC field2*

The asterisk (*) behind field2 is used to indicate that field2 is optional and need not be used. If it is used it may be used more than once.

ABC [field1, field2]+

The elements field1 and field2 in the square brackets [] are an unsorted list. The plus behind the closing bracket indicates that field1 and field2 are used at least once and may be used more than once as group.

ABC field1, (field2, field3)<exp>

<exp> indicates that field2 and field3 are used in conjunction with the conditional expression, (conditional constructs are specified in 7.32). The expression <exp> refers only to the fields within the preceding curved brackets (). Usage of conditional expressions is optional.

6 EDD and EDDL model

6.1 Overview of EDD and EDDL

An Electronic Device Description (EDD) contains all device parameter(s) of an automation system component. The technology used to describe an EDD is called Electronic Device Description Language (EDDL). The EDDL provides a set of scalable basic language elements to handle simple, complex or modular devices. The EDDL is a descriptive language based on an ASCII format with clear separation between data and program.

NOTE Data in a text field, which is marked with a country code like Japanese, may use multi-byte code.

6.2 EDD architecture

From the viewpoint of the ISO/OSI model (ISO/IEC 7498-1), an EDD is above Layer 7. However, the EDD application uses the communication system to transfer its information. An EDD contains constructs that support mapping to a supporting communication system.

The device manufacturer defines the objects, which are reflected by the logical representation of the objects within an EDD application. For that reason, EDDL has language elements, which map the EDD data to the data representation of the communication system, so that the user of an EDD does not need to know the physical location (address) of a device object.

EDD describes the management of information to be displayed to the user. The specific representation of such visualization is not part of an EDD or EDDL definitions.

6.3 Concepts of EDD

The manufacturer of a device or of an automation system component describes the properties of the device by using the EDDL. The resulting EDD contains information such as:

- description of the device parameters;
- description of parameter dependencies;
- logical grouping of the device parameters;
- selection and execution of supported device functions;
- description of the transferred data sets.

Depending on the required usage, the EDD may be physically located

- in a device;
- in an external data storage medium such as a compact disk, floppy or a server;
- partially distributed in the device and an external storage medium.

EDD supports text strings (common terms, phrases etc.) in more than one language (English, German, French, etc.). Text strings may be stored in separate dictionaries. There may be more than one dictionary for one EDD.

An EDD implementation includes sufficient information about the target device, for example, manufacturer, device type, revision, etc. This is used to match a specific EDD to a specific device.

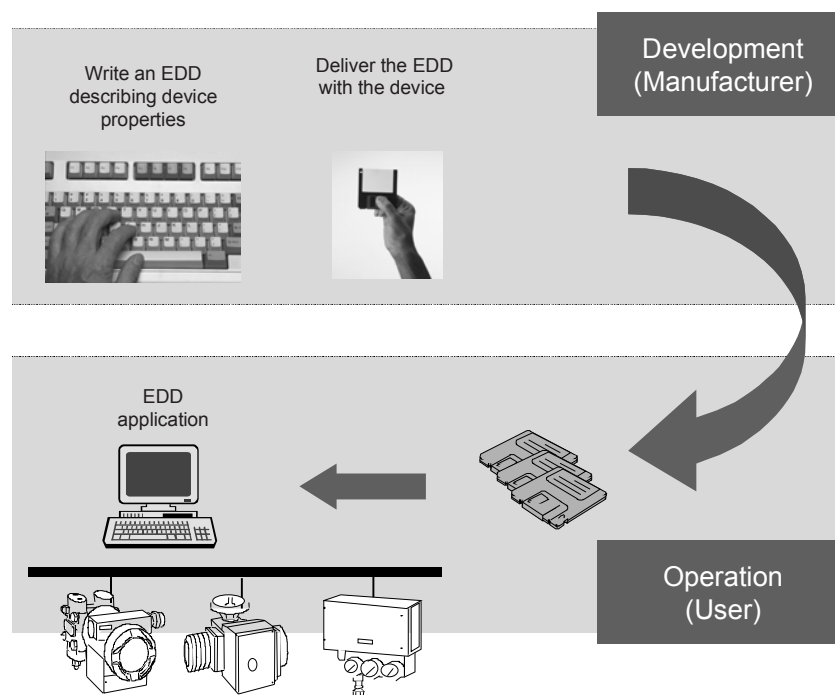
6.4 Principles of the EDD development process

6.4.1 General

The creation of an EDD is a three-stage process: EDD source generation, EDD preprocessing and EDD compilation.

6.4.2 EDD source generation

The EDD generation process is shown in Figure 2.



IEC 1511/06

Figure 2 – EDD generation process

The device manufacturer writes an appropriate EDD for his device and delivers both (EDD and device). If the automation system supports the EDD method, a system integration can be made by the user.

The EDDs for devices may be embedded in device memory or delivered using separate storage media or downloaded from an appropriate network server. The EDD is 'interpreted' or 'browsed' by an EDD-application. EDDs are normally stored as source files or preprocessed files.

6.4.3 EDD preprocessing

In the preprocessing stage an EDD preprocessor generates a consistent EDD representation suitable for final compilation.

Preprocessing supports, for example, substitution of definitions and inclusion of external text. The output of the preprocessor is a complete EDD without any preprocessor directive.

6.4.4 EDD compilation

The EDD compilation stage produces an EDD application internal representation from a preprocessed EDD to be used in the EDD application.

6.5 Interrelations between the lexical structure and formal definitions

The lexical structure of EDDL and its elements are described in Clause 7. Formal definitions and syntax for each EDDL element are given in Annex A. The lexical structure and its formal description use the same name.

NOTE Instead of the specified formal definitions and syntax in Annex A, another specification may be developed as an additional option.

6.6 Builtins

Builtins are predefined subroutines which are executed in the EDD application.

EXAMPLE A hand-held terminal is a simple device having a small display and limited cursor functions. For this type of device, a Builtin could be specified to provide display entry using only up/down right/left cursor actions.

The library of Builtins is defined in Annex B.

6.7 Profiles

EDDL is a harmonized specification of existing legacy EDD concepts. Concrete EDD applications use a subset of the EDDL specification. Selection of EDDL elements and Builtins is made from the profiles defined in Annex D.

In addition to EDD profiles, implementing consortia also publish “Device Profiles”, which are used to support interchangeability of compliant devices. These Device Profiles may be described using the EDDL specified in this document.

7 Electronic Device Description Language

7.1 Overview

7.1.1 EDDL features

The EDDL is a structured and interpretative language for describing device properties. Also the interactions between the devices and the EDD run-time environment are incorporated in the EDDL. The EDDL provides a set of language elements for these purposes.

For a specific EDD implementation it is not necessary to use all of the elements provided by the language.

Compatible subsets of EDDL are permitted and may be specified using profiles (for example, choice of constructs, number of recursions, and selection of options). Profiles supported by some industrial consortia are specified in Annex D. EDD developers are required to identify within each device details as to which profile has been used.

7.1.2 Syntax representation

The specification of the EDDL language in this clause of the standard uses an abstract syntax. The abstract lexical structure in this section is converted into specific syntaxes specified in Annex A by the element name (for example, VARIABLE in the lexical structure equates to the keyword “VARIABLE” in the annex).

NOTE Beside the defined syntax in this standard, it is possible to use other syntax definitions, which may be added in future to allow other features and representations for future progress.

7.1.3 EDD language elements

The language is structured in the following language elements:

- identification element;
- basic construction elements;
- special elements.

Frequently used text strings, for example, help text and multi-lingual lists of labels should be separated in a text dictionary (see 7.36).

Identification information shall be the first entry in every EDD file and shall appear only once. The identification information uniquely identifies the version of EDDL used, together with the specific device type, model codes and revision details covered by the EDD file.

7.1.4 Basic construction elements

These basic constructs have been specified to support descriptions of devices used within industrial control applications, together with their properties and functionality.

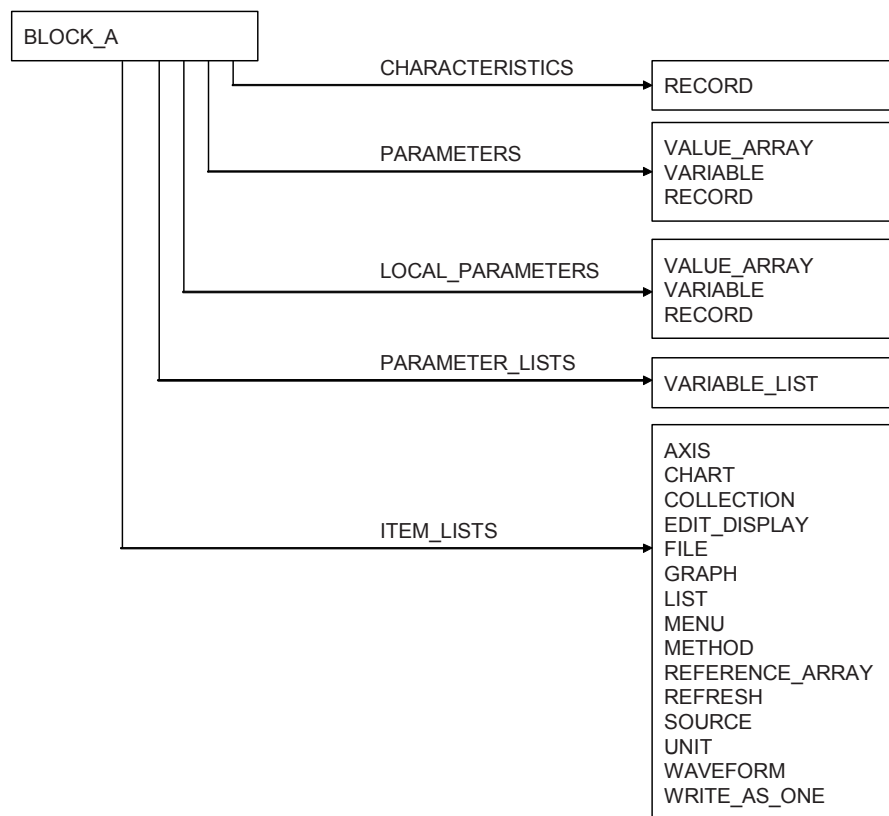
Some constructs have similar names and functions while differing in their detailed specification. This additional variety has been included to ensure compatibility with several existing description languages. Appropriate mapping cross-references are given by profiles.

Each of the basic constructs has a set of attributes associated with it. Attributes can also have subattributes, which refine the definition of the attribute and hence the definition of the construct itself.

The definition of an attribute may be static or dynamic. A static attribute definition never changes, while a dynamic attribute definition may change in order to accommodate parameter value changes.

Some basic constructs refer to other basic constructs. In the following list, the basic constructs as well as the relationships between the basic constructs are shown:

- **AXIS** describes the axis of a CHART or GRAPH (see 7.3).
- **BLOCK_A** is a logical grouping of CHARACTERISTICS, PARAMETERS, PARAMETER_LISTS, and ITEM_LISTS, see Figure 3. To access one item of BLOCK_A, the instance of the block should be used (see 7.4.1 and Figure 3).



IEC 1512/06

Figure 3 – BLOCK_A

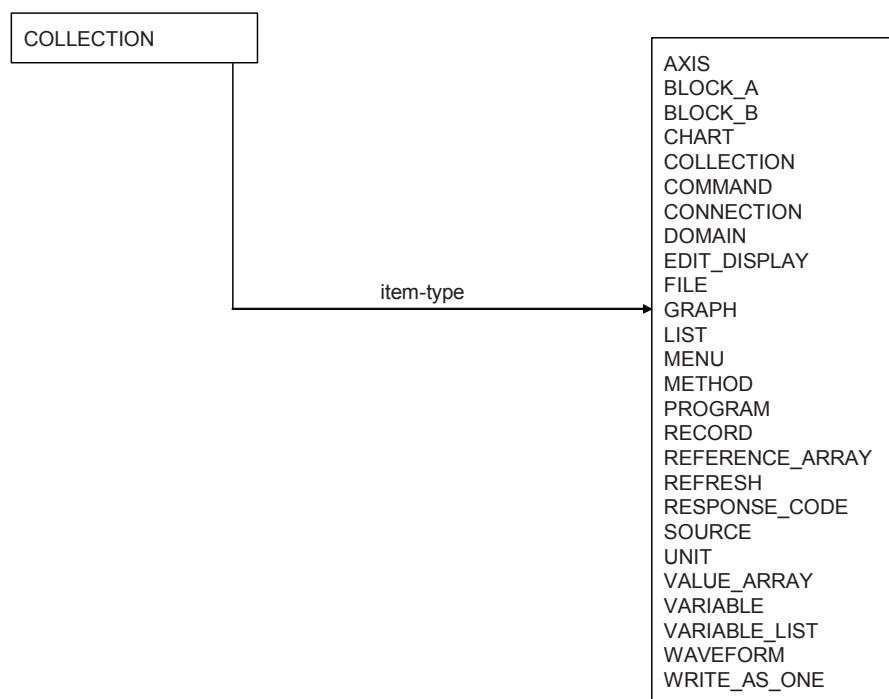
- **BLOCK_B** is used to structure variables in instances of logical block types. For accessing a **VARIABLE** within a **BLOCK_B** construct, the basic construct **COMMAND** is used to provide the relative addressing (see 7.4.2).
- **CHART** describes a chart used to display data from a device in the EDD application (see 7.5 and Figure 4).



IEC 1513/06

Figure 4 – CHART

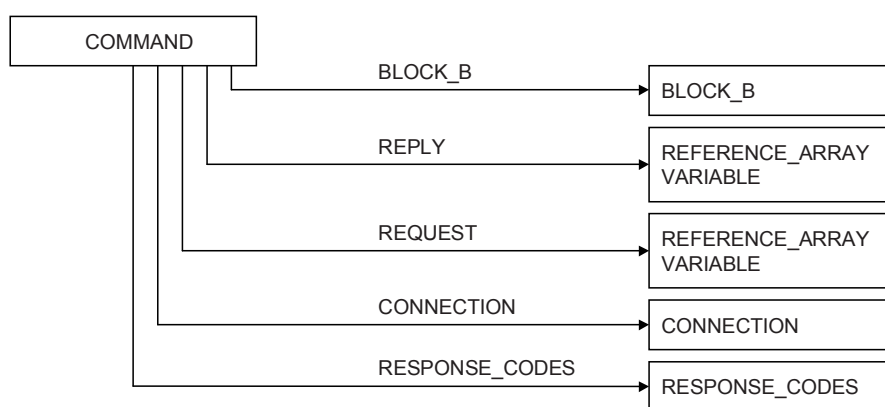
- **COLLECTION** describes logical groupings of data (see 7.6 and Figure 5).



IEC 1514/06

Figure 5 – COLLECTION

- **COMMAND** describes the structure and the addressing of the variables in the device (see 7.7 and Figure 6).



IEC 1515/06

Figure 6 – COMMAND

- **CONNECTION** defines multiple applications in a device (see 7.8).
- **DOMAIN** can be used to download or upload moderately large amounts of data to or from a device (see 7.9 and Figure 7).



IEC 1516/06

Figure 7 – DOMAIN

- **EDIT_DISPLAY** describes how the data will be represented to a user by a display device (see 7.10 and Figure 8).

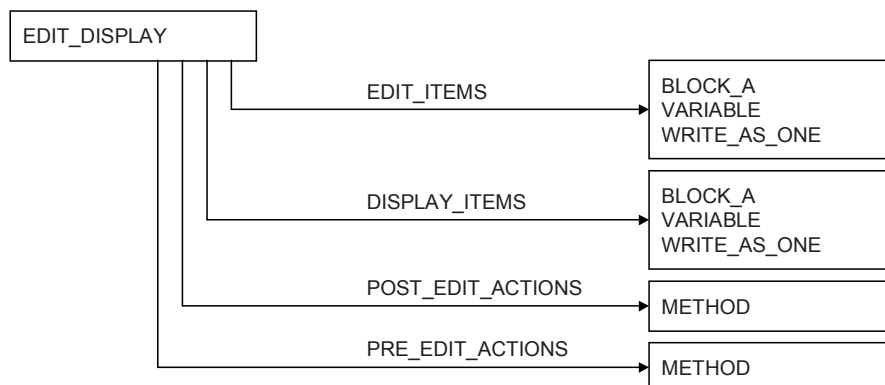


Figure 8 – EDIT_DISPLAY

IEC 1517/06

- **FILE** describes an area of persistent storage (see 7.11 and Figure 9).



Figure 9 – FILE

IEC 1518/06

- **GRAPH** describes a graph used to display data from a device in the EDD application (see 7.12 and Figure 10).

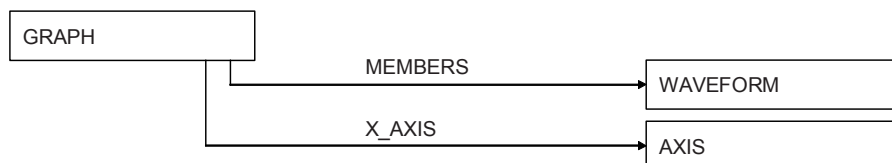
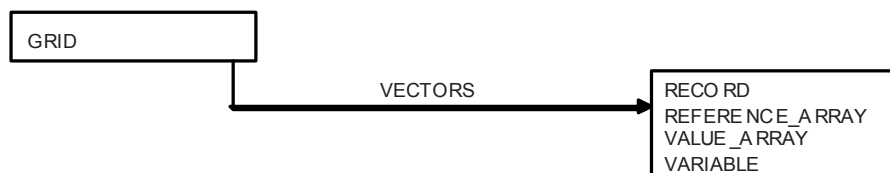


Figure 10 – GRAPH

IEC 1519/06

- **GRID** describes a matrix used to display data from a device in the EDD application (see 7.13 and Figure 11).



IEC 1520/06

Figure 11 – GRID

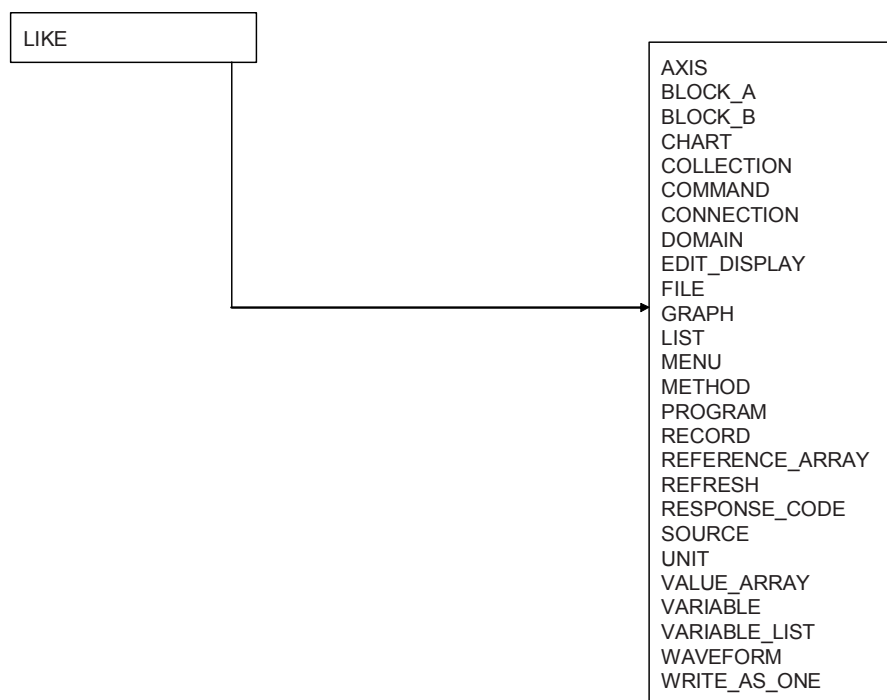
- **IMAGE** describes a visual image (see 7.13 and Figure 12).



IEC 1521/06

Figure 12 – IMAGE

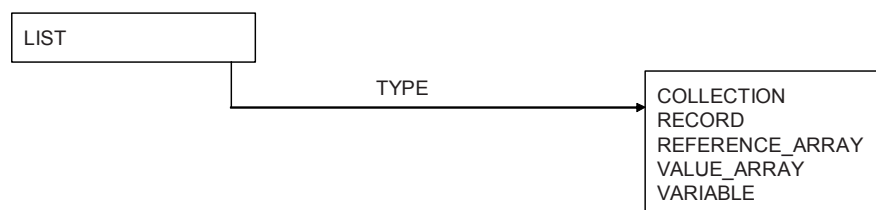
- **IMPORT** is used to import an EDD and to modify its existing definitions (see 7.13).
- **LIKE** creates a new instance of an existing instance of a basic construct. The attributes of the new instance may be redefined, added or deleted (see 7.16 and Figure 13).



IEC 1522/06

Figure 13 – LIKE

- **LIST** describes a sequence of elements (see 7.17 and Figure 14).



IEC 1523/06

Figure 14 – LIST

- **MENU** describes how the data will be presented to a user by an EDD application (see 7.18 and Figure 15).

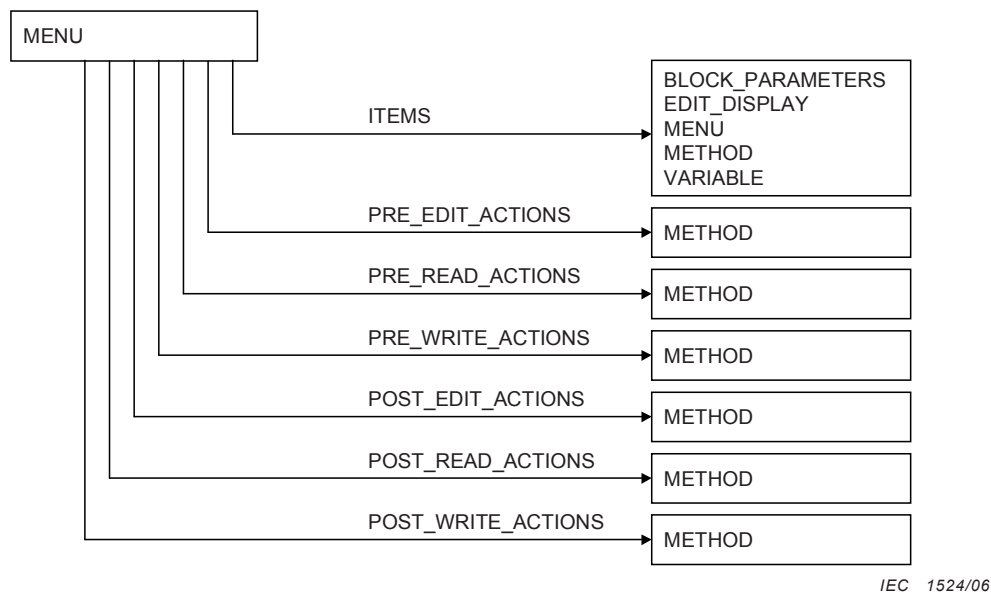


Figure 15 – MENU

- **METHOD** describes the execution of complex sequence of event interactions that shall take place between system devices such as display units, configurators and field devices (see 7.19).
- **PROGRAM** specifies how device executable code can be initiated by an appropriate agent (see 7.20 and Figure 16).



Figure 16 – PROGRAM

- **RECORD** describes the data contained in the device (see 7.21 and Figure 17).

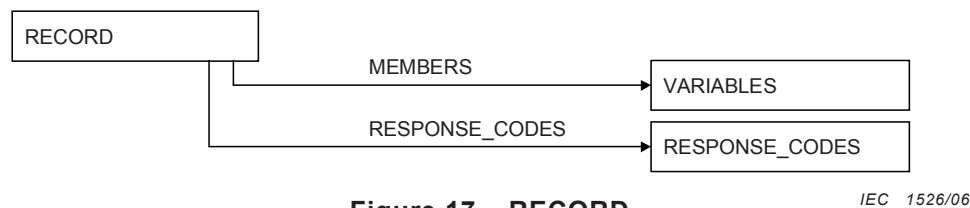
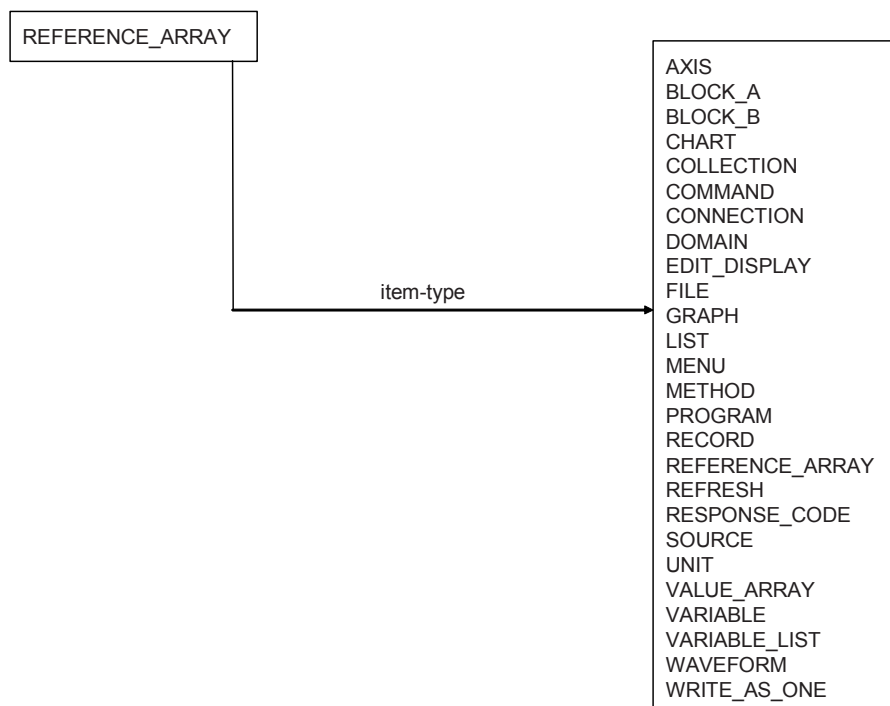


Figure 17 – RECORD

- **REFERENCE_ARRAY** is a set of EDD items of the same EDDL item type (for example, VARIABLE or MENU). An item can be referenced in the EDD via the REFERENCE_ARRAY name and the index associated with the item (see 7.22 and Figure 18).



IEC 1527/06

Figure 18 – REFERENCE_ARRAY

- **Relation:** three types of relations are specified.
 - REFRESH describes relationships between VARIABLES, RECORDs, and VALUE_ARRAYs (see 7.23.2 and Figure 19).



IEC 1528/06

Figure 19 – REFRESH

- UNIT describes relationships between VARIABLES, RECORDs, and VALUE_ARRAYs (see 7.23.3 and Figure 20).



IEC 1529/06

Figure 20 – UNIT

- WRITE_AS_ONE RELATION describes relationships between VARIABLES, RECORDs, and VALUE_ARRAYs (see 7.23.4 and Figure 21).



IEC 1530/06

Figure 21 – WRITE_AS_ONE

- **RESPONSE_CODES** specifies the application specific response codes for use when managing a VARIABLE, RECORD, VALUE_ARRAY, VARIABLE_LIST, PROGRAM, or DOMAIN (see 7.24).
- **SOURCE** (see 7.25 and Figure 22).

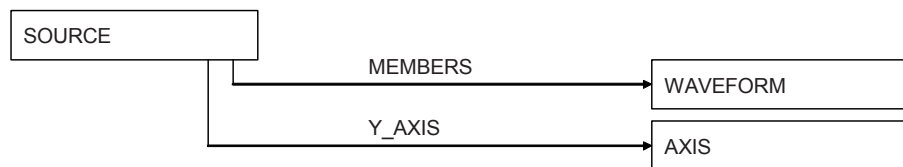


Figure 22 – SOURCE

IEC 1531/06

- **VALUE_ARRAY** describes logical groups of EDDL variables (see 7.26 and Figure 23).

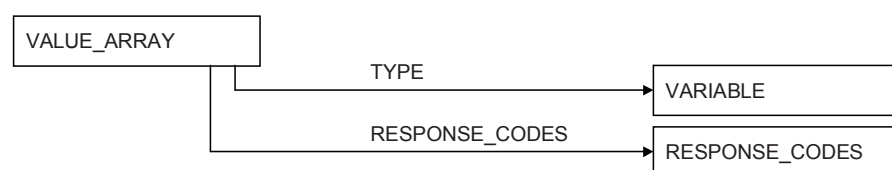


Figure 23 – VALUE_ARRAY

IEC 1532/06

- **VARIABLE** describes the data contained in the device (see 7.27 and Figure 24).

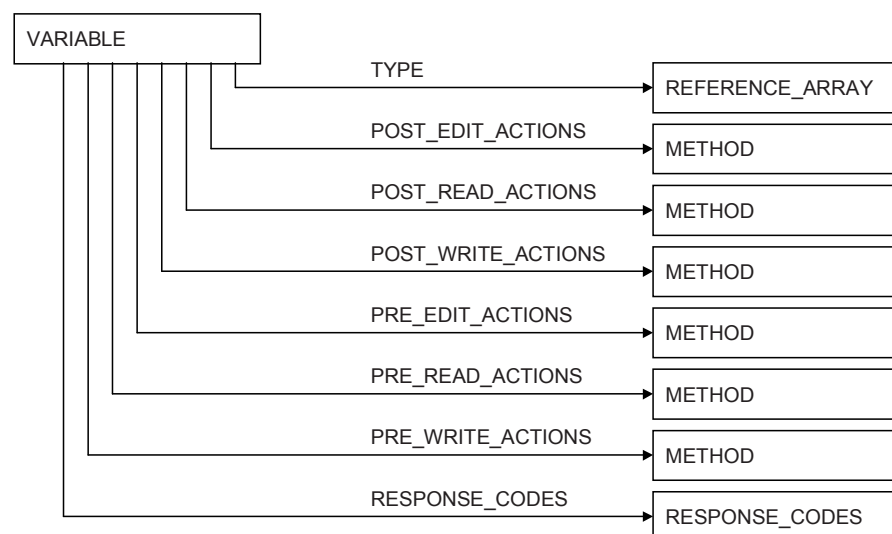


Figure 24 – VARIABLE

IEC 1533/06

- **VARIABLE_LIST** describes logical groupings of data contained in the device that may be communicated as a list (see 7.28 and Figure 25).

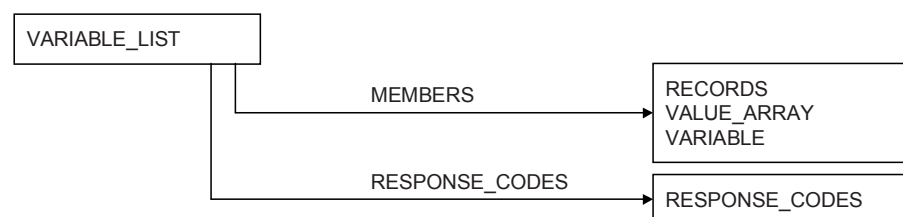


Figure 25 – VARIABLE_LIST

IEC 1534/06

- **WAVEFORM** (see 7.29 and Figure 26).

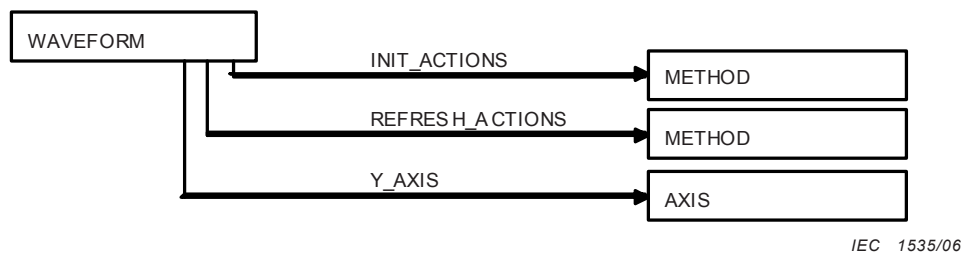


Figure 26 – WAVEFORM

7.1.5 Common attributes

The following common attributes are used in many basic constructs:

- DEFINITION
- MEMBERS
- LABEL
- HELP
- RESPONSE_CODES

7.1.6 Special elements

The special elements are additional EDDL mechanisms to support file handling, multiple instances and modification of EDDL basic elements, such as:

- **Output Redirection**
 - **OPEN** is used to create a separate output file for an EDD.
 - **CLOSE** is used to close an opened output file for an EDD.
- **Conditional Expressions** specify attributes values, which are dependent on the values of variables at run-time
- **Reference** is used throughout a device description to refer to other items within the same EDDL
- **Expression** is a logical or mathematical operation on variable values in order to modify attribute values of a basic construct at run-time

7.1.7 Rules for instances

Each instance of a basic construct shall be identified by an identifier of type string. This identifier shall be unique within the complete EDD. All EDDL constructs represent actual instances of device information. If a device contains two or more pieces of information which have the same structure the appropriate construct shall be repeated in the EDD with different identifiers. Alternatively the LIKE construct may be used to create a copy of the construct with a different identifier. The identifier in the lexical definition part is not described within each construct again.

7.1.8 Rules for list of VARIABLES

A list of VARIABLES may contain VARIABLES, elements of composed types or composed types. Composed types are:

- BLOCK_A PARAMETERS
- COLLECTIONS

- RECORDS
- REFERENCE_ARRAY
- VALUE_ARRAY
- VARIABLE_LISTS

Instead of referencing every single member of a composed type, a reference can be made to the type instance.

7.2 EDD identification information

7.2.1 General structure

Purpose

EDD identification information uniquely identifies the device description of a specific device type from a device manufacturer. It also specifies the EDDL version and EDDL profile used.

This identification information shall be the first entry in every EDD and shall appear only once.

Lexical structure

DD_REVISION, DEVICE_REVISION, DEVICE_TYPE, EDD_PROFILE, EDD_VERSION,
MANUFACTURER, MANUFACTURER_EXT

NOTE 1 EDD identification information is not structured in the same way as other EDDL constructs.

NOTE 2 Strict order of the keywords of the EDD identification information is not necessary.

7.2.2 Specific attributes

7.2.2.1 DD_REVISION

Purpose

This attribute specifies a unique code for the EDD revision of this device, as defined by the manufacturer.

NOTE 1 The manufacturer should change the revision number each time a new device description is issued for a device.

NOTE 2 There can be multiple EDDs for a particular revision of a particular device, in which case the DD_REVISION distinguishes them.

Lexical structure

DD_REVISION integer

The attribute of DD_REVISION is specified in Table 2.

Table 2 – DD_REVISION attribute

Usage	Attribute	Description
m	integer	is a two-octet unsigned integer containing the code for a specific EDD file revision

7.2.2.2 DEVICE_REVISION

Purpose

This attribute specifies a unique code for the device revision of the field device, as defined by the manufacturer.

Lexical structure

DEVICE_REVISION integer

The attribute of DEVICE_REVISION is specified in Table 3.

Table 3 – DEVICE_REVISION attribute

Usage	Attribute	Description
m	integer	is a two-octet unsigned integer containing the code for a specific device revision

7.2.2.3 DEVICE_TYPE**Purpose**

This attribute specifies an identifier for the device type, as defined by the manufacturer of the device. The identifier should be unique for each type.

NOTE The device type may specify either a category of devices or a specific product.

Lexical structure

DEVICE_TYPE (integer, identifier)

The attribute of DEVICE_TYPE is specified in Table 4.

Table 4 – DEVICE_TYPE attribute

Usage	Attribute	Description
s	integer	is a two-octet unsigned integer containing the code for a specific device type or group of device types
s	identifier	is a reference to a two-octet integer which contains the code for a specific device type or group of device types
NOTE The mapping of the identifier to an integer constant may be stored in an extra file named "devices.cfg". This file contains a list of identifiers for devices or groups of device types.		

7.2.2.4 EDD_PROFILE**Purpose**

This attribute specifies the selected profile of the EDDL as defined in Annex D.

Lexical structure

EDD_PROFILE integer

The attribute of EDD_PROFILE is specified in Table 5.

Table 5 – EDD_PROFILE attribute

Usage	Attribute	Description
o	integer	is a two-octet unsigned integer containing the code for the specific profiles selected from Annex D. The following codes are currently defined: 0x01 Profile for HART Communication Foundation (see Clause D.4) 0x02 Profile for Fieldbus Foundation (see Clause D.3) 0x03 Profile for PROFIBUS (see Clause D.2)

7.2.2.5 EDD_VERSION**Purpose**

This keyword specifies a unique code for the EDD version, which has been used by the manufacturer in constructing the EDD. This attribute shall be included with all device descriptions conforming to this standard.

Lexical structure

EDD_VERSION integer

The attribute of EDD_VERSION is specified in Table 6.

Table 6 – EDD_VERSION attribute

Usage	Attribute	Description
o	integer	is a two-octet unsigned integer containing the number 1 for this first edition of the standard

7.2.2.6 MANUFACTURER**Purpose**

This attribute identifies the manufacturer. The code allocation should be managed by the specific organizations which are responsible for the different EDDL profiles. The combination of EDD_PROFILE and MANUFACTURER shall be unique.

Lexical structure

MANUFACTURER [(long integer, identifier)<exp>]

The attribute of MANUFACTURER is specified in Table 7.

Table 7 – MANUFACTURER attribute

Usage	Attribute	Description
s	long integer	is an unsigned long integer containing the code for a specific manufacturer. If all bits in the value of the integer are set to one, the lexical structure "MANUFACTURER_EXT" shall be present and identifies the manufacturer
sm	identifier	is a reference to an unsigned long integer which contains the code for a specific manufacturer
NOTE The mapping of the identifier to an integer constant should be stored in an extra file named "devices.cfg". This file contains a list of identifiers for manufacturers.		

7.2.2.7 MANUFACTURER_EXT**Purpose**

This attribute is reserved for future use. It is a placeholder for a string providing global identification for the MANUFACTURER.

NOTE The specification of this attribute will be included when it becomes internationally available.

Lexical structure

MANUFACTURER_EXT string

The attribute of MANUFACTURER_EXT is specified in Table 8.

Table 8 – MANUFACTURER_EXT attribute

Usage	Attribute	Description
c	string	is a string containing the information for a unique worldwide manufacturer identification

7.3 AXIS

7.3.1 General structure

Purpose

AXIS describes an axis of a CHART or GRAPH.

Lexical structure

AXIS identifier [HELP, LABEL, MAX_VALUE, MIN_VALUE, SCALING, CONSTANT_UNIT]

The attributes of AXIS are specified in Table 9.

Table 9 – AXIS attributes

Usage	Attribute	Description
o	HELP	lexical element (see 7.30.2)
o	LABEL	lexical element (see 7.30.3)
o	MAX_VALUE	lexical element (see 7.3.2.1)
o	MIN_VALUE	lexical element (see 7.3.2.2)
o	SCALING	lexical element (see 7.3.2.3)
o	CONSTANT_UNIT	lexical element (see 7.3.2.4)

7.3.2 Specific attributes

7.3.2.1 MAX_VALUE

Purpose

The MAX_VALUE attribute specifies the value associated with the last tick mark of the AXIS. The last tick mark of an X_AXIS and Y_AXIS is the right-most and top-most tick mark, respectively. By default, the value associated with the last tick mark of an AXIS without a MAX_VALUE attribute is inferred from the WAVEFORMs on the GRAPH.

Lexical structure

MAX_VALUE (reference, double, float, integer, unsigned_integer)<exp>

The attributes of MAX_VALUE are specified in Table 10.

Table 10 – MAX_VALUE, MIN_VALUE attribute

Usage	Attribute	Description
o	reference	is a reference to either a VARIABLE instance, a member of a RECORD instance, an element of a VALUE_ARRAY instance or an element of a REFERENCE_ARRAY instance
o	double	is a double-precision floating-point constant
o	float	is a single-precision floating-point constant
o	integer	is an integer constant
o	unsigned_integer	is an unsigned integer constant

7.3.2.2 MIN_VALUE

Purpose

The MIN_VALUE attribute specifies the value associated with the first tick mark of the AXIS. The first tick mark of an X_AXIS and Y_AXIS is the left-most and bottom-most tick mark, respectively. By default, the value associated with the first tick mark of an AXIS without a MIN_VALUE attribute is inferred from the WAVEFORMs on the GRAPH.

Lexical structure

MIN_VALUE (reference, double, float, integer, unsigned_integer)<exp>

The attributes of MIN_VALUE are specified in Table 10.

7.3.2.3 SCALING**Purpose**

The SCALING attribute specifies the way the tick marks of the AXIS should be scaled. By default, an AXIS without a SCALING attribute is scaled linearly.

Lexical structure

SCALING (LINEAR, LOGARITHMIC)<exp>

The attributes of SCALING are specified in Table 11.

Table 11 – SCALING attribute

Usage	Attribute	Description
s	LINEAR	specifies that the tick marks on the axis should be scaled linearly
s	LOGARITHMIC	specifies that the tick marks on the axis should scaled logarithmically (base 10)

7.3.2.4 CONSTANT_UNIT**Purpose**

The CONSTANT_UNIT attribute is used, if an AXIS has a unit code which never changes. The CONSTANT_UNIT is specified as a string that will be displayed along with the value. An AXIS without a constant unit either has no units associated with it or the units are not constant.

Lexical structure

CONSTANT_UNIT (string)<exp>

The attribute of CONSTANT_UNIT is specified in Table 150.

7.4 BLOCK**7.4.1 BLOCK_A****7.4.1.1 General structure****Purpose**

A device may be organized in logical blocks. Each BLOCK_A is a logical grouping of PARAMETERS and additional information for BLOCK_A applications. Types of blocks are described by the CHARACTERISTICS attribute. More than one BLOCK_A from the same type can be instantiated.

NOTE This BLOCK_A approach is optimized for access efficiency.

Lexical structure

BLOCK_A identifier [CHARACTERISTICS, LABEL, PARAMETERS, AXIS_ITEMS, CHART_ITEMS, COLLECTION_ITEMS, EDIT_DISPLAY_ITEMS, FILE_ITEMS, GRAPH_ITEMS, GRID_ITEMS, IMAGE_ITEMS, LIST_ITEMS, HELP, MENU_ITEMS, METHOD_ITEMS, PARAMETER_LISTS, REFERENCE_ARRAY_ITEMS, REFRESH_ITEMS, SOURCE_ITEMS, UNIT_ITEMS, WAVEFORM_ITEMS, WRITE_AS_ONE_ITEMS]

The attributes of BLOCK_A are specified in Table 12.

Table 12 – BLOCK_A attributes

Usage	Attribute	Description
m	CHARACTERISTICS	lexical element (see 7.4.1.2.1)
m	LABEL	lexical element (see 7.30.3)
m	PARAMETERS	lexical element (see 7.4.1.2.2)
o	AXIS_ITEMS	lexical element (see 7.4.1.2.3)
o	CHART_ITEMS	lexical element (see 7.4.1.2.4)
o	COLLECTION_ITEMS	lexical element (see 7.4.1.2.5)
o	EDIT_DISPLAY_ITEMS	lexical element (see 7.4.1.2.6)
o	FILE_ITEMS	lexical element (see 7.4.1.2.7)
o	GRAPH_ITEMS	lexical element (see 7.4.1.2.8)
o	GRID_ITEMS	lexical elements (see 7.4.1.2.9)
o	IMAGE_ITEMS	lexical elements (see 7.4.1.2.10)
o	LIST_ITEMS	lexical element (see 7.4.1.2.11)
o	LOCAL_PARAMETERS	lexical element (see 7.4.1.2.12)
o	HELP	lexical element (see 7.30.2)
o	MENU_ITEMS	lexical element (see 7.4.1.2.13)
o	METHOD_ITEMS	lexical element (see 7.4.1.2.14)
o	PARAMETER_LISTS	lexical element (see 7.4.1.2.15)
o	REFERENCE_ARRAY_ITEMS	lexical element (see 7.4.1.2.16)
o	REFRESH_ITEMS	lexical element (see 7.4.1.2.17)
o	SOURCE_ITEMS	lexical element (see 7.4.1.2.18)
o	UNIT_ITEMS	lexical element (see 7.4.1.2.19)
o	WAVEFORM_ITEMS	lexical element (see 7.4.1.2.20)
o	WRITE_AS_ONE_ITEMS	lexical element (see 7.4.1.2.21)

7.4.1.2 Specific attributes

7.4.1.2.1 CHARACTERISTICS

Purpose

The CHARACTERISTICS attribute specifies the external characteristics for a BLOCK_A. CHARACTERISTICS may include class, function type, block type and execution time. The external characteristics of a BLOCK_A are contained in a RECORD.

NOTE In general the BLOCK_A CHARACTERISTICS fix the type of the BLOCK_A.

Lexical structure

CHARACTERISTICS reference

The attribute of CHARACTERISTICS is specified in Table 13.

Table 13 – CHARACTERISTIC attribute

Usage	Attribute	Description
m	reference	is a reference to a RECORD instance

7.4.1.2.2 PARAMETERS

Purpose

The PARAMETERS attribute contains a list of references to VARIABLE, VALUE_ARRAY or RECORD instances. Each reference shall have an identifier and may have a description and/or a help text.

NOTE The PARAMETERS in a BLOCK_A correspond to individual communication objects, listed in a device object dictionary. The PARAMETERS elements reference the communication definitions of these objects. The object dictionary maps into the specific location for data in a device.

Lexical structure

PARAMETERS [identifier, reference, description, help]+

The attributes of PARAMETERS are specified in Table 14.

Table 14 – PARAMETER attributes

Usage	Attribute	Description
m	identifier	is the identifier of the VARIABLE, VALUE_ARRAY or RECORD. Every element of the PARAMETERS list shall have an identifier to be used in the EDD to refer to it
m	reference	is a reference to a VARIABLE, VALUE_ARRAY or RECORD instance
o	description	is a short description for the reference
o	help	specifies help text for the reference

7.4.1.2.3 AXIS_ITEMS

Purpose

AXIS_ITEMS is a list of the AXIS instances associated with the BLOCK_A.

Lexical structure

AXIS_ITEMS reference+

The attribute of AXIS_ITEMS is specified in Table 15.

Table 15 – AXIS_ITEMS attribute

Usage	Attribute	Description
m	reference	is a reference to a AXIS instance

7.4.1.2.4 CHART_ITEMS

Purpose

CHART_ITEMS is a list of the CHART instances associated with the BLOCK_A.

Lexical structure

CHART_ITEMS reference+

The attribute of CHART_ITEMS is specified in Table 16.

Table 16 – CHART_ITEMS attribute

Usage	Attribute	Description
m	reference	is a reference to a CHART instance

7.4.1.2.5 COLLECTION_ITEMS

Purpose

COLLECTION_ITEMS is a list of the COLLECTION instances associated with the BLOCK_A.

Lexical structure

COLLECTION_ITEMS reference+

The attribute of COLLECTION_ITEMS is specified in Table 17.

Table 17 – COLLECTION_ITEMS attribute

Usage	Attribute	Description
m	reference	is a reference to a COLLECTION instance

7.4.1.2.6 EDIT_DISPLAY_ITEMS

Purpose

EDIT_DISPLAY_ITEMS is a list of the EDIT_DISPLAY instances associated with BLOCK_A.

Lexical structure

EDIT_DISPLAY_ITEMS reference+

The attribute of EDIT_DISPLAY_ITEMS is specified in Table 18.

Table 18 – EDIT_DISPLAY_ITEMS attribute

Usage	Attribute	Description
m	reference	is a reference to an EDIT_DISPLAY instance

7.4.1.2.7 FILE_ITEMS

Purpose

FILE_ITEMS is a list of the FILE instances associated with the BLOCK_A.

Lexical structure

FILE_ITEMS reference+

The attribute of FILE_ITEMS is specified in Table 19.

Table 19 – FILE_ITEMS attribute

Usage	Attribute	Description
m	reference	is a reference to a FILE instance

7.4.1.2.8 GRAPH_ITEMS

Purpose

GRAPH_ITEMS is a list of the GRAPH instances associated with the BLOCK_A.

Lexical structure

GRAPH_ITEMS reference+

The attribute of GRAPH_ITEMS is specified in Table 20.

Table 20 – GRAPH_ITEMS attribute

Usage	Attribute	Description
m	reference	is a reference to a GRAPH instance

7.4.1.2.9 GRID_ITEMS**Purpose**

GRID_ITEMS is a list of the GRID instances associated with the BLOCK_A.

Lexical structure

GRID_ITEMS reference+

The attribute of GRID_ITEMS is specified in Table 21.

Table 21 – GRID_ITEMS attribute

Usage	Attribute	Description
m	reference	is a reference to a GRID instance

7.4.1.2.10 IMAGE_ITEMS**Purpose**

IMAGE_ITEMS is a list of the IMAGE instances associated with BLOCK_A.

Lexical structure

IMAGE_ITEMS reference+

The attribute of IMAGE_ITEMS is specified in Table 22.

Table 22 – IMAGE_ITEMS attribute

Usage	Attribute	Description
m	reference	is a reference to a IMAGE instance

7.4.1.2.11 LIST_ITEMS**Purpose**

LIST_ITEMS is a list of the LIST instances associated with BLOCK_A.

Lexical structure

LIST_ITEMS reference+

The attribute of LIST_ITEMS is specified in Table 23.

Table 23 – LIST_ITEMS attribute

Usage	Attribute	Description
m	reference	is a reference to a LIST instance