Ladder diagram file format (.LAD)

© 2024 Robox SpA

The content of this document is prepared with the utmost care/diligence, and subjected to careful control.
Robox Spa, however, disclaims all liability, direct and indirect, to users and in general to any third party, for any inaccuracies, errors, omissions, damages (direct, indirect, consequential, punishable and sanctionable) resulting from the aforementioned contents

Table of contents

Parte I General index 6

Parte II Minimal requirements 6

Parte III File format (version 1.2) 7

1 File format (version 1.2) ... 7
2 File header ... 7
3 Table of structures ... 10
4 Table of variables ... 11
5 Table of blocks ... 12
6 Table of block's circuits ... 13
7 Table of circuit's items ... 14
8 Table of variable's initializators ... 24
9 Table of strings ... 24
10 Table of compressed strings ... 25
11 Table of live changes ... 25
12 Predefined data types ... 25
13 Predefined variables ... 29

Parte IV File format (version 1.3) 30

1 File format (version 1.3) ... 30
2 File header ... 30
3 Table of structures ... 33
4 Table of variables ... 34
5 Table of blocks ... 37
6 Table of block's circuits ... 38
7 Table of circuit's items ... 38
8 Table of OB classes ... 49
9 Table of OB methods ... 50
10 Table of strings ... 52
11 Table of compressed strings ... 53
12 Table of live changes ... 53
13 Predefined data types ... 53
14 Predefined variables ... 57
15 Source identifiers ... 57

Parte V File format (version 1.4) 58

1 File format (version 1.4) ... 58

© 2024 Robox SpA

2 File header ... 58
3 Table of structures ... 61
4 Table of variables ... 63
5 Table of blocks ... 65
6 Table of block's circuits ... 66
7 Table of circuit's items ... 67
8 Table of OB classes ... 78
9 Table of OB methods ... 79
10 Table of strings ... 81
11 Table of compressed strings ... 81
12 Table of live changes ... 82
13 Predefined data types ... 82
14 Predefined variables ... 85
15 Source indentifiers ... 86

Parte VI File format (version 1.5) 87

1 File format (version 1.5) ... 87
2 File header ... 87
3 Table of structures ... 90
4 Table of variables ... 92
5 Table of blocks ... 94
6 Table of block's circuits ... 95
7 Table of circuit's items ... 96
8 Table of OB classes ... 107
9 Table of OB methods ... 108
10 Table of include files ... 110
11 Table of strings ... 111
12 Table of compressed strings ... 111
13 Table of live changes ... 112
14 Predefined data types ... 112
15 Predefined variables ... 115
16 Source indentifiers ... 116

Parte VII Live changes format (version 1.1) 117

1 Live changes format (version 1.1) ... 117
2 Live changes header ... 117
3 Live changes command table ... 118

Parte VIII Live changes format (version 1.3) 122

1 Live changes format (version 1.3) ... 122

© 2024 Robox SpA

2 Live changes header 123
3 Live changes command table 124

Parte IX Live changes format (version 1.4) 129

1 Live changes format (version 1.4) 129
2 Live changes header 130
3 Live changes command table 131

Parte X Live changes format (version 1.5) 137

1 Live changes format (version 1.5) 137
2 Live changes header 137
3 Live changes command table 138

Index 145

© 2024 Robox SpA

General index

File format:

- File format (version 1.5) 87
- File header 87
- Table of blocks 94
- Table of block's circuits 95
- Table of circuit's items 96
- Table of external files 110
- Table of structures 90
- Table of variables 92
- Table of OB classes 107
- Table of live changes 112
- Table of strings 111
- Table of compressed strings 111

Format of live changes:

- Live changes format (version 1.5) 137
- Live changes header 137
- Live changes commands table 138

Other useful information:

- Minimum requirements 6
- Default data types 112
- Variable defaults 112

Version 1.2 formats:

- File format (version 1.2) 7
- Live changes format (version 1.1) 117

Version 1.3 formats:

- File format (version 1.3) 30
- Live changes format (version 1.3) 122

Version 1.4 formats:

- File format (version 1.4) 58
- Live changes format (version 1.4) 129

Minimal requirements

The minimum requirements for using the ladder diagram (.LAD) file are as follows, depending on the version of the file (otherwise called the compatibility level):

Version	Requirements
v1.5	RTE version 34.19.10 (or higher) RDE version 3.51.1 (or higher)
v1.4	For target RTE (PowerPC G2): - RTE version 33.17.0 (or higher) - RDE version 3.7.0 (or higher) For RTE destination (PowerPC P2020): - RTE version 34.0.0 (or higher) - RDE version 3.10.0 (or higher)
v1.3	RTE version 33.14.0 (or higher) RDE3 version 3.6.0 (or higher)
v1.2	RTE version 33.8.0 (or higher)

© 2024 Robox SpA

Version	Requirements
	RDE3 version 3.5.1 (or higher)
v1.1	Not usable (to be converted to v1.2)
v1.0	Not usable (to be converted to v1.2)

File format (version 1.2)

File format (version 1.2)

CAUTION: this information refers to an earlier version of the file format (version 1.2).

A ladder/diagram programming language program is a single file (type .LAD) that contains both editing (source) information and execution information (typically on RTE or compatible firmware).

A ladder program consists primarily of a header 7, then several sections of which the main ones are:

- table of structures 10, which are the user-defined data types.
- table of variables 11, i.e., the general variables of the program.
- table of blocks 12, i.e., the sub-routines in other programming languages;
- table of live changes 25, i.e., any embedded live changes files;
- table of strings 24;
- table of compressed strings 25.

In the block list, there is always a main block called MAIN and possibly additional blocks (comparable to sub-routines in other programming languages): each block contains a list of circuits 13 that constitute the actual ladder program: each circuit in turn is a logically structured list of elements 14.

NOTE: all numeric data stored in this file are to be considered in little endian format (standard x86/x64).

File header

CAUTION: this information refers to an earlier version of the file format (version 1.2).

At the beginning of the file (offset 0) is a general header (HEADER) that contains access information for all remaining components and sub-sections of the file.

Name	Offset	Type	Description
SIGN	+0	U32	File signature ('LADF').
VERSION	+4	U32	Version of the file (1.2.0).
GENERATOR	+8	U32	Name and version of the file generator (offset relative to string table).
TYPE	+12	U16	Execution type:

© 2024 Robox SpA

Name	Offset	Type	Description
			0 = Synchronous execution (SY). 1 = High priority execution (HP). 2 = Normal priority execution (NP). 3 = Low priority execution (LP).
FREQUENCY	+14	U16	Execution frequency in [hz], 1hz to 3000hz, default 200hz .
FLAGS	+16	U32	General settings: 0x000001 = file contains formal verification errors.
BUILDYEAR	+20	U16	Year last generation (4 digits).
BUILDMONTH	+22	U8	Month last generation (1-12).
BUILDDAY	+23	U8	Day last generation (1-31).
BUILDHOUR	+24	U8	Last hour generation (0-23).
BUILDMINUTE	+25	U8	Last minute compilation (0-59).
BUILDSECOND	+26	U8	Second last compilation (0-59).
TARGETID	+27	U8	Runtime destination ID: 0x00 = Firmware RTE (PowerPC G2) 0x01 = RTE firmware (PowerPC P2020)
VID	+28	U32	File version ID.
PLVID	+32	U32	Version ID of the file pre live edits (if present, otherwise worth 0).
	+36	U8[92]	(reserved)
BLOCKSOFF	+128	U32	Offset of the block table 12 (relative to the file).

© 2024 Robox SpA

Name	Offset	Type	Description
BLOCKSCNT	+132	U32	No. of elements of the block table 12.
STRINGSOFF	+136	U32	Offset of the string table 24 (relative to the file).
STRINGSSIZE	+140	U32	Size of the string table 24.
CSTRINGSOFF	+144	U32	Offset of the compressed string table 25 (relative to the file).
CSTRINGSSIZE	+148	U32	Size of the compressed string table 25.
CSTRINGSCTYPE	+152	U8	Type of compression used for ta compressed string table 25:. 0 = No compression. 1 = Standard Qt 3.3.x compression.
EMBLIVESOFF	+153	U32	Offset of embedded live changes table (relative to file) .
EMBLIVESCNT	+157	U32	No. of elements of the embedded live changes table.
	+161	U8[3]	(reserved)
STRUCTSOFF	+164	U32	Offset of the structures table 10 (relative to the file).
STRUCTSCNT	+168	U32	No. of elements of the structures table 10.
VARSOFF	+172	U32	Offset of variable table 11 (relative to file).
VARSCNT	+176	U32	No. of elements of the variable table 11.
VARSINITOFF	+180	U32	Offset of variable initializer table 24 (relative to file) .
VARSINITCNT	+184	U32	No. of elements of the variable initializer table 24.
LASTBLOCKID	+188	U32	Last block ID used.

© 2024 Robox SpA

Name	Offset	Type	Description
LASTRUNGID	+192	U32	Last circuit ID used.
LASTSTRUCTID	+196	U32	Last structure ID used.
LASTVARID	+200	U32	Last used variable ID.
	+204	U8[308]	(reserved)

Table of structures

CAUTION: this information refers to an earlier version of the file format (version 1.2).

The structures table (STRUCTS) consists of a number of contiguous definitions (STRUCTs), each of which contains all the information about a specific user-defined structure: there are no default data types 25 in this table.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 7.STRUCTSOFF and HEADER 7.STRUCTSSIZE fields, respectively.

Structure definition

A structure (STRUCT) consists of the following information:

Name	Offset	Type	Description
TYPEID	+0	U32	Structure type ID Note: only values 0x00010000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
STATUS	+4	U16	(reserved, runtime use)
FLAGS	+6	U16	Structure settings: 0x0001 Uncompressed TEXT field (NCF0).
NAME	+8	U32	Offset structure name (relative to strings 24).
TEXT	+12	U32	Offset structure description (relative to compressed 25 strings, or strings 24 if NCF0).
VARSOFF	+16	U32	Variable table 11 offset (relative to file) (#1)
VARSCNT	+20	U32	Variable table 11 size
SIZE	+24	U32	Structure size [bytes]
EXTRA	+28	U32	Offset extra information (0=none) (relative to

© 2024 Robox SpA

Name	Offset	Type	Description
			file)

(#1) The structure definition uses a normal variable definition table to define its elements. In this case, variable IDs are meaningless and should always be 0.

Table of variables

CAUTION: This information refers to an earlier version of the file format (version 1.2).

The variable table (VARS) consists of a number of contiguous definitions (VARs), each of which contains all the information about a specific user-defined variable: there are no default variables 29 in this table.

The location (relative to the file) and number of elements in the program variable table are specified in the following fields, respectively, depending on where used

- by the HEADER 7.VARSOFF and HEADER 7.VARSSIZE fields if used in the file header (HEADER);
- by the fields STRUCT 10.VARSOFF and STRUCT 10.VARSCNT if used in user structure definition (STRUCT);
- by the LIVECMD 118.VARSOFF and LIVECMD 118.VARSCNT fields if used in live command definition (LIVECMD).

Variable definition

A variable (VAR) consists of the following information:

Name	Offset	Type	Description
VARID	+0	U32	Variable ID (0 if variable in user structure 10 definition) Note: only values 0x00010000-0xFFFFFE; value 0xFFFFFF is a special meaning ID; values 0x000001-0x0000FFFF are reserved for default variables.
TYPEID	+4	U32	Variable type ID Note: This can be a default 25 or user-defined 10 type 25.
STATUS	+8	U16	(reserved, runtime use)
FLAGS	+10	U16	Variable settings: 0x0001 Uncompressed TEXT field (NCF0).

© 2024 Robox SpA

Name	Offset	Type	Description
			0x0002 Uncompressed BITTEXT field (NCF1).
NAME	+12	U32	Offset variable name (relative to strings 24).
TEXT	+16	U32	Offset variable description (relative to compressed 25 strings, or strings 24 if NCF0).
BITTEXT	+20	U32	Offset variable bit descriptions, i.e., sequence of strings related to each individual bit separated by the ascii character 0xFF (related to compressed 25 strings, or strings 24 if NCF0).
ARRAY0	+24	U32	No. of indexes 1st level (0=No array) (#1) or index 0 for Alias 26.
ARRAY1	+28	U32	No. of indexes 2nd level (0=No array) (#1) or index 1 for Alias 26.
ARRAY2	+32	U32	No. of indexes 3rd level (0=No array) (#1) or index 2 for Alias 26.
ALIASID	+36	U16	Alias type 28 ID
ALIASBIT	+38	U16	No. of bits per alias (depends on the type of alias ID)
SIZE	+40	U32	Size of variable. If Alias 26, the size is 0.
EXTRA	+44	U32	Offset extra information (0=none) (relative to file).

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum index however should only be considered 16bit (i.e. maximum index 0xFFFF).

Table of blocks

CAUTION: This information refers to an earlier version of the file format (version 1.2).

The block table (BLOCKS) consists of a number of contiguous block definitions (BLOCKS), each of which contains all the information about a specific block and its contents on file.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 7.BLOCKSOFF and HEADER 7.BLOCKSCNT fields, respectively.

© 2024 Robox SpA

Block definition

A block (BLOCK) consists of the following data:

Name	Offset	Type	Description
BLOCKID	+0	U32	Unique ID of the block. Note: the values 0 and 0xFFFFFF are special meaning IDs.
STATUS	+4	U16	(reserved, runtime use)
FLAGS	+6	U16	Block settings: 0x0001 = Main block (ISMAIN). 0x0002 = Uncompressed TEXT field (NCF0). 0x0004 = Uncompressed COMMENT field (NCF1).
NAME	+8	U32	Offset block name (relative to strings 24). Note: If ISMAIN flag specified, name comes must be "main".
TEXT	+12	U32	Offset block description (relative to compressed 25 strings, or strings 24 if NCF0).
COMMENT	+16	U32	Offset comments for the block (relative to compressed 25 strings, or strings 24 if NCF1).
RUNGSOFF	+20	U32	Offset of the circuit table of the block 13 (relative to rows).
RUNGSCNT	+24	U32	No. of elements of the block circuit table 13.
	+28	B[36]	(reserved)

Table of block's circuits

CAUTION: This information refers to an earlier version of the file format (version 1.2).

The block circuit table (RUNGS) consists of a number of contiguous circuit definitions (RUNG), each of which contains all the information about a specific circuit and its contents.

The location (relative to the file) and the number of elements in the table are specified by the BLOCK 13.RUNGSOFF and BLOCK 13.RUNGSCNT fields, respectively.

© 2024 Robox SpA

Block circuit definition

A block circuit (RUNG) consists of the following information:

Name	Offset	Type	Description
RUNGID	+0	U32	Unique ID of the circuit. Note: the values 0 and 0xFFFFFF are special meaning IDs.
STATUS	+4	U16	(reserved, runtime use)
FLAGS	+6	U16	Circuit settings: 0x0001 Uncompressed TEXT field (NCF0).
REFID	+8	U32	Reference circuit ID (0=none).
ITEMSOFF	+12	U32	Offset of the circuit element table 14 (relative to the file).
ITEMSSIZE	+16	U32	Size of the circuit element table 14.
TEXT	+20	U32	Offset circuit description (related to compressed st 25 rings, or strings 24 if NCF1).
EXTRA	+24	U32	Offset extra information (0=none) (relative to file).

Table of circuit's items

CAUTION: This information refers to an earlier version of the file format (version 1.2).

The circuit element table (ITEMS) consists of a number of contiguous circuit definitions (ITEMs), each of which contains all the information about a specific element.

The location (relative to the file) and size of the table are specified in the following fields, respectively, depending on where used:

- by the RUNG 14.ITEMSOFF and RUNG 14.ITEMSSIZE fields, if used in block circuit (RUNG).
- by the fields LIVECMD 118.ITEMSOFF and LIVECMD 118.ITEMSSIZE, if used in live change commands (LIVECMD).

Circuit element definition

Circuit elements (ITEMs) have a common initial part, then a specific data area for each type of element. The common part is as follows:

Name	Offset	Type	Description
CODE	+0	U16	Item type code.

© 2024 Robox SpA

Name	Offset	Type	Description
SIZE	+2	U16	Full size of element (all fields).
DATE	+4	...	Specific data (optional).

The expected elements are as follows:

Symbolic	ID	Type	Data	Description
BB	0x0001			Start of branch.
NL	0x0002			Next branch level . Notes: must always be between a BB and an EB.
EB	0x0003			End of branch . Notes: must always correspond to a BB.
OC	0x0004		U32=Destination (#2): BOOL 26 types only.	Open contact.
CC	0x0005		U32=Destination (#2): BOOL 26 types only.	Closed contact.
RE	0x0006		U32=Destination (#2): EDGE 26 types only.	Rising edge.
FE	0x0007		U32=Destination (#2): EDGE 26 types only.	Falling edge.
AF	0x0008			Always false.
OUT	0x0009	F	U32=Destination (#2): BOOL 26 types only.	Output.
OL	0x000A	F	U32=Destination (#2): BOOL 26 types only.	Output latch (set).
OUL	0x000B	F	U32=Destination (#2): BOOL 26 types only.	Output unlatch (reset).
NOUT	0x000C	F	U32=Destination (#2): BOOL 26 types only	Output denied

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
CLR	0x000D	F	U32=Destination (#2): TIMER 26 and COUNTER 26 types only.	Zeroing (clear).
TP	0x000E	F	U32=Destination (#2): TIMER 26 types only U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer pulse.
TON	0x000F	F	U32=Preset value 2 (#2): TIMER 26 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer on delay.
TOFF	0x0010	F	U32=Preset value 2 (#2): TIMER 26 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer off delay.
TSQ	0x0011	F	U32=Preset value 2 (#2): TIMER 26 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable square wave timer.
TINT	0x0012	F	U32=Preset value 2 (#2): TIMER 26 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable integral timer.
CU	0x0013	F	U32=Preset value (#2): COUNTER 26 types only U32=Preset value (#4).	Count up.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
CD	0x0014	F	U32=Preset value (#2): COUNTER 26 types only U32=Preset value (#4).	Count down.
CUR	0x0015	F	U32=Preset value (#2): COUNTER 26 types only. U32=Preset value (#4).	Rotary count up.
CDR	0x0016	F	U32=Preset value (#2): COUNTER 26 types only. U32=Preset value (#4).	Count down rotary.
LD	0x0017	F	U32=Preset value (#2): COUNTER 26 types only.	Preset on counter.
MATH	0x0018	F	U32=Expression (#5) U32=Destination (#2) U8[]=Mathematical expression 21 data.	Math block.
EQ	0x0019		U32=Source A (#2) U32=Source B (#2).	Equality comparison block (A == B).
NE	0x001A		U32=Source A (#2) U32=Source B (#2)	Different comparison block (A != B).
GT	0x001B		U32=Source A (#2) U32=Source B (#2)	Major comparison block (A > B).
GE	0x001C		U32=Source A (#2) U32=Source B (#2)	Major/equal comparison block (A >= B).
LT	0x001D		U32=Source A (#2) U32=Source B (#2)	Minor comparison block (A < B).
LE	0x001E		U32=Source A (#2) U32=Source B (#2)	Minor/equal comparison block (A <= B).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
COPY	0x001F	F	U32=Source (#2) U32=Destination (#2) U32=N. Elements (#2)	Copies data from source to destination, for the specified no. of elements.
CALL	0x0020	F	U32 Block ID to call	Call of a block (sub-routine).
RET	0x0021	F		Return from a block to the inside.
ADD	0x0022	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Sum (A + B).
SUB	0x0023	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Subtraction (A - B).
MUL	0x0024	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Multiplication (A * B).
DIV	0x0025	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Division (A / B).
AND	0x0026	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	AND binary (A & B).
OR	0x0027	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Binary OR (A | B).
NOT	0x0028	F	U32=Source A (#2) U32=Destination (#2)	NOT binary (~A).
XOR	0x0029	F	U32=Source A (#2) U32=Source B (#2)	XOR binary (A ^ B).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			U32=Destination (#2)	
SQRT	0x002A	F	U32=Source A (#2) U32=Destination (#2)	Square root of A.
MOD	0x002B	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Module (A mod B).
POW	0x002C	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Power elevation (A pow B).
SIN	0x002D	F	U32=Source A (#2) U32=Destination (#2)	Sine of A.
COS	0x002E	F	U32=Source A (#2) U32=Destination (#2)	Cosine of A.
TAN	0x002F	F	U32=Source A (#2) U32=Destination (#2)	Tangent of A.
ASIN	0x0030	F	U32=Source A (#2) U32=Destination (#2)	Archosene of A.
ACOS	0x0031	F	U32=Source A (#2) U32=Destination (#2)	Arcocosene of A.
ATAN	0x0032	F	U32=Source A (#2) U32=Destination (#2)	Arcotangent of A.
ATAN2	0x0033	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Arcotangent of A, B.
LOG	0x0034	F	U32=Source A (#2) U32=Destination (#2)	Base-10 logarithm of A.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
LN	0x0035	F	U32=Source A (#2) U32=Destination (#2)	Base E logarithm of A.
ABS	0x0036	F	U32=Source A (#2) U32=Destination (#2)	Absolute value of A
SIGN	0x0037	F	U32=Source A (#2) U32=Destination (#2)	A sign.
ROUND	0x0038	F	U32=Source A (#2) U32=Destination (#2)	Rounding to the nearest integer to A.
MIN	0x0039	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Minimum between A and B.
MAX	0x003A	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Maximum between A and B.
RANGE	0x003B	F	U32=Source A (#2) U32=Source B (#2) U32=Source C (#2) U32=Destination (#2): BOOL 26 types only.	Checks whether A contained in the range B-C.
LIMIT	0x003C	F	U32=Source A (#2) U32=Source B (#2) U32=Source C (#2) U32=Destination (#2)	Limit A to the B-C range.
ROR	0x003D	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Right bit rotation (A >> B).
ROL	0x003E	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Left bit rotation (A << B).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
SHR	0x003F	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Shift bit to the right (A >> B).
SHL	0x0040	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#2)	Shift bit to the left (A << B).
MOV	0x0041	F	U32=Source A (#2) U32=Destination (#2)	Assign A.
TODEG	0x0042	F	U32=Source A (#2) U32=Destination (#2)	Conversion of A from RAD to DEG.
TORAD	0x0043	F	U32=Source A (#2) U32=Destination (#2)	Conversion of A from DEG to RAD.
NEG	0x0044			Negation of logical state.
NOP	0x0045	F		No operation.
EVAL	0x0046		U32=Expression (#5) U8[]=Mathematical expression 21 data	Expression evaluation (boolean)

((#1) Local bit variable (I32 or U32 type) with direct addressing (any constant indexes and number of bits).

(#2) Source/destination means either a variable name or the ascii representation of a numeric value (source only), or an offset to the string table 24.

(#4) Value (or preset value) means a numeric value in string format: empty string (or offset 0) means no value assigned.

(#5) Expression means string with offset to string table 24.

Mathematical expression data

Mathematical expression data (MATH element) is a series of contiguous operations (ordered in inverse polar) with the following common header:

Name	Offset	Type	Description
CODE	+0	U8	Operation code.
DATE	+1	...	Specific data (optional).

The planned operations are as follows:

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
PUSH	0x01	DBL=value	--	DBL	Push value into stack.
PUSHV	0x02	U32=Source (#1)	--	DBL	Push variable value into stack.
ADD	0x03		DBL0,DBL1	DBL	Sum.
SUB	0x04		DBL0,DBL1	DBL	Subtraction (#2).
MUL	0x05		DBL0,DBL1	DBL	Multiplication (#2).
DIV	0x06		DBL0,DBL1	DBL	Division (#2).
MOD	0x07		DBL	DBL	Module (#3).
ABS	0x08		DBL	DBL	Absolute value (#3).
SIN	0x09		DBL	DBL	Sine.
COS	0X0A		DBL	DBL	Cosine.
TAN	0x0B		DBL	DBL	Tangent.
ASIN	0x0C		DBL	DBL	Arcosene.
ACOS	0x0D		DBL	DBL	Arcocosene.
ATAN	0x0E		DBL	DBL	Arcotangent.
POW	0x0F		DBL	DBL	Exponent.
LOG10	0x10		DBL	DBL	Logarithm base 10.
LOG	0x11		DBL	DBL	Natural logarithm.
R_AND	0x12		U32	U32	Binary AND.
R_OR	0x13		U32	U32	OR binary.
R_NOT	0x14		U32	U32	NOT binary.
R_XOR	0x15		U32	U32	XOR binary.
POPV	0x16	U32=Destination (#1)	DBL	--	Pop value from stack to variable.
R_ROR	0x17		U32	U32	Right-hand bit rotation.
R_ROL	0x18		U32	U32	Left-hand bit rotation.

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
R_SHR	0x19		U32	U32	Shift bit to the right.
R_SHL	0x1A		U32	U32	Shift bit to the left.
TOBIN	0x1B		U32	U32	BCD to binary conversion.
TOBCD	0x1C		U32	U32	Binary to BCD conversion.
TODEG	0x1D		DBL	DBL	Conversion from radians to degrees.
TORAD	0x1E		DBL	DBL	Conversion from degrees to radians.
ATAN2	0x1F		DBL0,DBL1	DBL	Arcotangent with two parameters.
PUSHVB	0x20	U32=Source (#1)	DBL	--	Push variable bit value into stack.
POPVB	0x21	U32=Destination (#1)	DBL	--	Pop value from stack to variable bit.
POPL	0x22		BOOL	--	Pop logical value (0=false, otherwise true).
	0x23				(reserved)
EQ	0x24		DBL0,DBL1	BOOL	Comparison: equal.
NE	0x25		DBL0,DBL1	BOOL	Comparison: different.
LT	0x26		DBL0,DBL1	BOOL	Comparison: minor.
LE	0x27		DBL0,DBL1	BOOL	Comparison: less than or equal to.
GT	0x28		DBL0,DBL1	BOOL	Comparison: greater than.
GE	0x29		DBL0,DBL1	BOOL	Comparison: greater than or equal to.

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
AND	0x2A		BOOL,BOOL	BOOL	AND logical.
OR	0x2B		BOOL,BOOL	BOOL	OR logical.
NOT	0x2C		BOOL	BOOL	NOT logical.
XOR	0x2D		BOOL,BOOL	BOOL	logical XOR.
CHS	0x2E		DBL	DBL	Sign change.
SHR	0x2F		U32	U32	Logical shift to the right.
SHL	0x30		U32	U32	Logical shift to the left

(#1) Source/destination means name of a variable, i.e., an offset to the string table 24.

Table of variable's initializators

CAUTION: This information refers to an earlier version of the file format (version 1.2).

The variable initializer table (VARSINIT) consists of a number of contiguous definitions (VARINIT), each of which contains all the information about a specific variable initialization, also called the initial value.

The location (relative to the file) and the number of elements in the program variable table are specified by the HEADER 7.VARSINITOFF and HEADER 7.VARSINITSIZE fields, respectively.

Variable initializer definition

A variable initializer (VARINIT) consists of the following information:

Name	Offset	Type	Description
NAME	+0	U32	Offset variable name (relative to strings 24).
VALUE	+4	U32	Variable value offset (relative to strings 24).

Table of strings

CAUTION: This information refers to an earlier version of the file format (version 1.2).

The string table (STIRNGS) consists of a number of consecutive ascii strings, each terminated the ascii character \0 (zero) and of variable size. The table must contain at least one character and must be \0, so that strings with offset 0 actually point to a null string.

The location (relative to the file) and size of the table are specified in the following fields, respectively, depending on where used:

- by the HEADER 7.STRINGSOFF and HEADER 7.STRINGSSIZE fields, if used in ladder/diagram files.
- by the fields LIVEHEADER 117.STRINGSOFF and LIVEHEADER 117.STRINGSSIZE, if used in live edits.

© 2024 Robox SpA

Table of compressed strings

CAUTION: This information refers to an earlier version of the file format (version 1.2).

The compressed string table (CSTRINGS) consists of a number of consecutive ascii strings, each terminated the ascii character \0 (zero) and of variable size.

The location (relative to the file) and size of the table are specified by the HEADER 7 .STRINGSOFF and HEADER 7 .STRINGSSIZE fields, respectively: the compression state, which is optional, is instead specified by the HEADER 7 .CSTRINGSCTYPE field.

Annotations:

- in case of HEADER 7 .CSTRINGSCTYPE compression = 1, use the standard library functions (QT >= 3.3.4) qUncompress() to decompress the table properly.
- offsets referring to this table are always understood to refer to the uncompressed table.

Table of live changes

CAUTION: This information refers to an earlier version of the file format (version 1.2).

The embedded live changes table (EMBLIVES) consists of a number of contiguous (EMBLIVE) definitions, each of which contains all the information about a specific set of live changes to the ladder file itself: each set is actually an independent live changes 117 file but embedded in the ladder file itself.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 7 .EMBLIVESOFF and HEADER 7 .EMBLIVESCNT fields, respectively.

When loading, the various live change files are applied in the exact order in which they appear. The actual application of the changes is governed by the appropriate flag in EMBLIVE 25 .FLAGS (value 0x0001): only the last live change file can possibly have this flag set to false, indicating live changes not yet applied.

Definition of embedded live changes

An embedded live changes (EMBLIVE) consists of the following information:

Name	Offset	Type	Description
FILEOFF	+0	U32	Offset of the live edit 117 file (offset relative to the ladder file).
FILESIZE	+4	U32	Size of the live changes 117 file.
FLAGS	+8	U16	File settings: 0x0001 = Actual changes to be applied.
	+10	B[22]	(reserved)

Predefined data types

CAUTION: This information refers to an earlier version of the file format (version 1.2).

Data types are identified by a unique ID code, whether they are predefined or user-defined: IDs are divided as follows:

© 2024 Robox SpA

Range	Description
0x00000000-0x0000FFFF	Default data types
0x00010000-0xFFFFFFFE	User-defined data types
0xFFFFFF	Reserved

CAUTION: Default data types are never saved in the .LAD file, but are always considered to exist.

The default data types are as follows:

ID	Symbolic	Description	RTE (0x00)
0x00000000	(reserved)	Invalid data type	
0x00000001	I8	Signed 8bit integer	1
0x00000002	U8	Unsigned integer 8bit	1
0x00000003	I16	Integer 16bit with sign	2
0x00000004	U16	Unsigned 16bit integer	2
0x00000005	I32	Integer 32bit with sign	4
0x00000006	U32	Unsigned 32bit integer	4
0x00000007	I64	Integer 64bit with sign	
0x00000008	U64	Unsigned integer 64bit	
0x00000009	STRING	String (0 terminated)	...
0x00000A	FLOAT	Floating point 32bit	4
0x00000B	REAL	Floating point 64bit (double)	8
0x00000C	BOOL	Boolean	4 (#1)
0x00000D	TIMER	Default timer structure 27	32
0x00000E	COUNTER	Default counter structure 27	48
0x00000F	ALIAS	Alias (or literally) 26	0
0x00000010	EDGE	Special storage for rising edge (RE) and falling edge (FE) elements.	4
0x00000011	POWERSET	"Power Set" logic variable	4 (#2)
0x00000012	AXESGROUP	"Axes group" logic variable	4 (#2)
0x00000013	PATH	"Path" logic variable	4 (#2)

© 2024 Robox SpA

ID	Symbolic	Description	RTE (0x00)
0x00000014	POINT_L	"Library point" logical variable	4 (#2)

(#1) BOOL type in RTE is represented, for performance reasons, as I32, while by RDE3 it is seen as U8.

(#2) Logical variables in RTE are always represented as a pointer to memory (4bytes), even if they were embedded in user-defined structures. For further information on the composition (elements) of such variables, see the RTE technical documentation).

The representation and size of data types depend directly on the runtime destination ID(HEADER 7.TARGETID). The sizes of the data types are listed in the RTE columns and are expressed in bytes.

Default timer structure

The default TIMER structure consists of the following fields:

Type	Name	Description	Read	Write
BOOL	Q	Timer Outcome	X	
REAL	ET	Current value [ms]	X	X (#1)
REAL	PT	Preset value [ms]	X	X
REAL	PT2	Preset value 2 [ms]	X	X
BOOL	EN	Enable timer	X	

(#1) The ET field can be written only when the timer has not yet been assigned.

COUNTER default structure

The default COUNTER structure consists of the following fields:

Type	Name	Description	Read	Write
BOOL	QU	Outcome of count against preset	X	
BOOL	QD	Count outcome with respect to 0	X	
I32	CV	Current count value	X	X (#1)
I32	PV	Preset value	X	X
BOOL	CU	Status of the "count up" signal	X	
BOOL	CD	State of the "count down" signal	X	

(#1) The CV field can be written only when the counter has not yet been assigned.

© 2024 Robox SpA

Alias (or literally)

The list of data types that can be used as aliases are as follows:

ID	Name	Type	Description
0x0001	INP(n)	BOOL	Input channel
0x0002	INP_W(n)	U16	Input word 16bit
0x0003	INP_W(n).b	BOOL	Input word 16bit, bit access
0x0004	INP_DW(n)	U32	Input word 32bit
0x0005	INP_DW(n).b	BOOL	Input word 32bit, bit access
0x0006	OUT	BOOL	Output channel
0x0007	OUT_W(n)	U16	Output word 16bit
0x0008	OUT_W(n).b	BOOL	Output word 16bit, bit access
0x0009	OUT_DW(n)	U32	Output word 32bit
0x000A	OUT_DW(n).b	BOOL	Output word 32bit, bit access
0x000B	R(n)	I32	Global integer register (volatile)
0x000C	R(n).b	BOOL	Global integer (volatile) register, bit access
0x000D	NVR(n)	I32	Global integer register (nonvolatile)
0x000E	NVR(n).b	BOOL	Global integer (nonvolatile) register, bit access
0x000F	RR(n)	REAL	Global real (volatile) register
0x0010	NVRR(n)	REAL	Real global register (non-volatile)
0x0011	AM(n)	U32	Alarm mask
0x0012	AM(n).b	U32	Alarm mask, bit access

Annotations:

- if the variable is of array type, the array index (n) can be either a constant or an unindexed variable.
- if the variable involves bit access, .b can be either a constant or a variable and must be in the compatible range the size of the reference variable.

© 2024 Robox SpA

Predefined variables

CAUTION: This information refers to an earlier version of the file format (version 1.2).

Variables are identified by a unique ID code, whether predefined or user-defined: IDs are broken down as follows:

Range	Description
0x00000000-0x0000FFFF	Default variable
0x0001000-0xFFFFFFFE	User-defined variable
0xFFFFFF	Reserved

CAUTION: Default variables are never saved in the .LAD file, but are always considered to exist.

The default variables are as follows:

ID	Name	Type	Description
0x00000001	INP(n)	BOOL	Input channel
0x00000002	INP_W(n)	U16	Input word 16bit
0x00000003	INP_DW(n)	U32	Input word 32bit
0x00000004	OUT	BOOL	Output channel
0x00000005	OUT_W(n)	U16	Output word 16bit
0x00000006	OUT_DW(n)	U32	Output word 32bit
0x00000007	R(n)	I32	Global integer register (volatile)
0x00000008	NVR(n)	I32	Global integer register (non-volatile)
0x00000009	RR(n)	REAL	Real global register (volatile)
0x00000A	NVRR(n)	REAL	Real global register (non-volatile)
0x00000B	AM(n)	U32	Alarm mask

Annotations:

- if the variable is of array type, the array index (n) can be either a constant or an unindexed variable.
- RDE3 creates additional predefined variables with IDs assigned completely arbitrarily, but greater than 0x00000100: RTE recognizes such variables by name and not by that ID.

© 2024 Robox SpA

File format (version 1.3)

File format (version 1.3)

A ladder/diagram programming language program is a single file (extension .LAD) containing both editing (source) and execution (runtime) information.

A ladder program consists primarily of a header 30, then several sections of which the main ones are:

- table of structures 33, which are the user-defined data types.
- table of local variables 34, i.e., the general variables of the program.
- table of OB classes 49, i.e., the object block classes referenced by the program.
- table of blocks 37, i.e., the sub-routines in other programming languages;
- table of live changes 53, i.e., any embedded live changes files;
- table of strings 52;
- table of compressed strings 53.

In the block list, there is always a main block called MAIN and possibly additional blocks (comparable to sub-routines in other programming languages): each block contains a list of circuits 38 that constitute the actual ladder program: each circuit in turn is a logically structured list of elements 38.

NOTE: All numeric data stored in this file are to be considered in little indian (x86 standard) format.

File header

At the beginning of the file (offset 0) is a general header (HEADER) that contains access information for all remaining components and sub-sections of the file.

Name	Offset	Type	Description
SIGN	+0	U32	File signature ('LADF').
VERSION	+4	U32	Version of the file (1.3.1).
GENERATOR	+8	U32	Name and version of the file generator (offset relative to string table).
TYPE	+12	U16	Execution type: 0 = Synchronous execution (SY). 1 = High priority execution (HP). 2 = Normal priority execution (NP). 3 = Low priority execution (LP).
FREQUENCY	+14	U16	Execution frequency in [hz], from 1hz to 3000hz, default 200hz .

© 2024 Robox SpA

Name	Offset	Type	Description
FLAGS	+16	U32	General settings: 0x000001 = The file contains formal verification errors. 0x000002 = The file contains references to external .OBB files (OB classes).
BUILDYEAR	+20	U16	Year last generation (4 digits).
BUILDMONTH	+22	U8	Month last generation (1-12).
BUILDDAY	+23	U8	Day last generation (1-31).
BUILDHOUR	+24	U8	Last hour generation (0-23).
BUILDMINUTE	+25	U8	Last minute compilation (0-59).
BUILDSECOND	+26	U8	Second last compilation (0-59).
TARGETID	+27	U8	Runtime destination ID: 0x00 = Firmware RTE (PowerPC G2) 0x01 = RTE firmware (PowerPC P2020)
VID	+28	U32	File version ID.
PLVID	+32	U32	Version ID of file pre live edits (if present, otherwise worth 0).
IARECNT	+36	U32	No. of ARE type elements present in the file.
IAFECNT	+40	U32	No. of AFE type elements present in the file.
	+44	U8[84]	(reserved)
BLOCKSOFF	+128	U32	Offset of the block table 37 (relative to the file).
BLOCKSCNT	+132	U32	No. of elements of the block table 37.

© 2024 Robox SpA

Name	Offset	Type	Description
STRINGSOFF	+136	U32	Offset of the string table 52 (relative to the file).
STRINGSSIZE	+140	U32	Size of the string table 52.
CSTRINGSOFF	+144	U32	Offset of the compressed string table 53 (relative to the file).
CSTRINGSSIZE	+148	U32	Size of the compressed string table 53.
CSTRINGSCTYPE	+152	U8	Type of compression used for ta compressed string table 53:. 0 = No compression. 1 = Standard compression Qt 3.3.x.
	+153	U8[3]	(reserved)
EMBLIVESOFF	+156	U32	Offset of the embedded live changes table 53 (relative to the file) .
EMBLIVESCNT	+160	U32	No. of elements of the embedded live changes table 53.
STRUCTSOFF	+164	U32	Offset of the structures table 33 (relative to the file).
STRUCTSCNT	+168	U32	No. of elements of the structures table 33.
VARSOFF	+172	U32	Offset of variable table 34 (relative to file).
VARSCNT	+176	U32	No. of elements of the variable table 34.
OBCLASSESOFF	+180	U32	Offset of the table classes OB 49 (relative to the file).
OBCLASSESCNT	+184	U32	No. of elements of the OB classes table 49 (relative to the file)
LASTBLOCKID	+188	U32	Last block ID used.
LASTRUNGID	+192	U32	Last circuit ID used.

© 2024 Robox SpA

Name	Offset	Type	Description
LASTSTRUCTID	+196	U32	Last structure ID used.
LASTVARID	+200	U32	Last variable ID used.
LASTOBCLASSID	+204	U32	Last OB class ID used.
	+208	U8[304]	(reserved)

Table of structures

The structures table (STRUCTS) consists of a number of contiguous definitions (STRUCTs), each of which contains all the information about a specific user-defined structure: there are no default data types 53 in this table.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 30.STRUCTSOFF and HEADER 30.STRUCTSCNT fields, respectively.

Structure definition

A structure (STRUCT) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	Structure type ID Note: only values 0x00010000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved, runtime use)
FLAGS	+6	U16	Structure settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 Uncompressed MODIFIER field (NCF1).
NAME	+8	U32	Offset structure name (relative to strings 52).
TEXT	+12	U32	Offset structure description (relative to compressed 53 strings, or strings 52 if NCF0).
VARSOFF	+16	U32	Variable table 34 offset (relative to file) (#1)
VARSCNT	+20	U32	Variable table 34 size
SIZE	+24	U32	Structure size [bytes]
SRCID	+28	U32	Source identifier 57 of the structure

© 2024 Robox SpA

Name	Offset	Type	Description
OBCID	+32	U32	Reference OB class ID (0=none)(#2,#3)
ATTRIB	+36	U32	Structure attributes: 0x000001 Structure referred to OB class (AREFOBC)(#2) 0x000002 Main structure OB class (AISOBC)(#2,#3)
MODIFIER	+40	U32	Offset modifiers of structure (related to compressed 53 strings, or strings 52 if NCF1).
REFID	+44	U32	Structure type reference ID (#4)

(#1) The structure definition uses a normal variable definition table to define its elements. In this case, variable IDs are meaningless and should always be 0.

(#2) At equal non-0 OBCID value with AREFOBC attribute, there can be only one structure with AISOBC flag, while there can be 0 or more structures without AISOBC attribute.

(#3) A structure is called an "orphan" structure if it is in the condition of AREFOBC attribute set and OBCID field = 0: this condition happens when updating/removing an OB class, there are still variables (or other) referencing it.

(#4) In case of AREFOBC attribute the REFID field contains the ID of the structure as defined of the source OBB file (0 in case of AISOBC attribute).

Table of variables

The variable table (VARS) consists of a number of contiguous definitions (VARs), each of which contains all the information about a specific user-defined variable: there are no default variables 57 in this table.

The location (relative to the file) and number of elements in the program variable table are specified in the following fields, respectively, depending on where used

- by the HEADER 30.VARSOFF and HEADER 30.VARSCNT fields if used in the file header;
- by the fields STRUCT 33.VARSOFF and STRUCT 33.VARSCNT if used in structure definition;
- by the fields LIVECMD 124.VARSOFF and LIVECMD 124.VARSCNT if used in command definition.

Variable definition

A variable (VAR) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	Variable ID (#2) Note: only values 0x00010000-0xFFFFFE; value 0xFFFFFF is a

© 2024 Robox SpA

Name	Offset	Type	Description
			special meaning ID; values 0x000001-0x0000FFFF are reserved for default variables.
TYPEID	+4	U32	Variable type ID Note: Can be a default type 53 or user-defined 33.
	+8	U8[2]	(reserved, runtime use)
FLAGS	+10	U16	Variable settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 BITTEXT field uncompressed (NCF1) 0x0004 Uncompressed MODIFIER field (NCF2)
NAME	+12	U32	Offset variable name (relative to strings 52).
TEXT	+16	U32	Offset variable description (relative to compressed 53 strings, or strings 52 if NCF0).
BITTEXT	+20	U32	Offset variable bit descriptions, i.e., sequence of strings related to each individual bit separated by the ascii character 0xFF (related to compressed 53 strings, or strings 52 if NCF0). (#2)
DIM0	+24	U32	1st array size (0=unspecified) (#1) or index 0 for Alias 54 (TYPEID = 0x00000F).
DIM1	+28	U32	2nd array size (0=unspecified) (#1) or index 1 for Alias 54 (TYPEID = 0x00000F).
DIM2	+32	U32	3rd array size (0=unspecified) (#1) or index 2 for Alias 54 (TYPEID = 0x00000F).
ALIASID	+36	U16	Alias type 55 ID (#2)

© 2024 Robox SpA

Name	Offset	Type	Description
ALIASBIT	+38	U16	No. of bits per alias (depends on the type of alias ID)(#2)
SIZE	+40	U32	Size of variable. (#3)
SRCID	+44	U32	Source identifier 57 of the variable.(#2,#4)
VALUE	+48	U32	Offset initial value of variable (relative to strings 52, 0=no value).
ATTRIB	+52	U32	Variable attributes: 0x000001 Variable can be written (WRITE) 0x000002 The variable can be read (READ) 0x000004 The variable can be forced (FORCE) 0x000008 The variable can be released (RELEASE) 0x000010 The variable is constant (CONST) 0x000020 The variable has no bit access (NOBITS)
MODIFIER	+56	U32	Variable modifier offsets (related to compressed st 53 rings, or strings 52 if NCF2).
DIM0L	+60	U32	Offset label for 1st array size (relative to strings 52). (#5)
DIM1L	+64	U32	Offset label for 2nd array size (relative to strings 52). (#5)
DIM2L	+68	U32	Offset label for 3rd array dimension (relative to strings 52). (#5)

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum size should still be considered only 16bit (i.e., maximum index 0xFFFF).

(#2) Field not used (value 0) if used in structure definition.

(#3) Field not used (value 0) if used by defining an Alias (TYPEID = 0x00000F)

(#4) Field not used (value 0) if used in live changes command definition.

© 2024 Robox SpA

(#5) Label means the name of a variable (constant value) or the ascii representation of a numeric value, i.e., an offset to the string table 52.

Table of blocks

The block table (BLOCKS) consists of a number of contiguous block definitions (BLOCKS), each of which contains all information about a specific block and its contents on file.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 30.BLOCKSOFF and HEADER 30.BLOCKSCNT fields, respectively.

Block definition

A block(**BLO**CK) consists of the following data:

Name	Offset	Type	Description
ID	+0	U32	Unique ID of the block. Note: the values 0 and 0xFFFFFF are special meaning IDs.
STATUS	+4	U16	(reserved, runtime use)
FLAGS	+6	U16	Block settings: 0x0001 = Main block (ISMAIN). 0x0002 = Uncompressed TEXT field (NCF0). 0x0004 = Uncompressed COMMENT field (NCF1).
NAME	+8	U32	Offset block name (relative to strings 52). Note: If ISMAIN flag specified, name comes must be "main".
TEXT	+12	U32	Offset block description (relative to compressed 53 strings, or strings 52 if NCF0).
COMMENT	+16	U32	Offset comments for the block (relative to compressed 53 strings, or strings 52 if NCF1).
RUNGSOFF	+20	U32	Offset of the circuit table of the block 38 (relative to rows).
RUNGSCNT	+24	U32	No. of elements of the block circuit table 38.

© 2024 Robox SpA

Name	Offset	Type	Description
	+28	B[36]	(reserved)

Table of block's circuits

The block circuit table (RUNGS) consists of a number of contiguous circuit definitions (RUNGs), each of which contains all the information about a specific circuit and its contents.

The location (relative to the file) and the number of elements in the table are specified by the BLOCK 37.RUNGSOFF and BLOCK 37.RUNGSCNT fields, respectively.

Block circuit definition

A block circuit (RUNG) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	Unique ID of the circuit. Note: the values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved, runtime use)
FLAGS	+6	U16	Circuit settings: 0x0001 Uncompressed TEXT field (NCF0).
REFID	+8	U32	Reference circuit ID (0=none).
ITEMSOFF	+12	U32	Offset of the circuit element table 38 (relative to the file).
ITEMSSIZE	+16	U32	Size of the circuit element table 38.
TEXT	+20	U32	Offset circuit description (related to compressed st 53 rings, or strings 52 if NCF1).
	+24	U8[4]	(reserved)

Table of circuit's items

La circuit element table (ITEMS) consists of a number of contiguous circuit element definitions (ITEMs), each of which contains all the information about a specific element.

The location (relative to the file) and size of the table are specified in the following fields, respectively, depending on where used:

- by the RUNG 38.ITEMSOFF and RUNG 38.ITEMSSIZE fields, if used in block circuit (RUNG).
- by the fields LIVECMD 124.ITEMSOFF and LIVECMD 124.ITEMSSIZE, if used in live change commands (LIVECMD).

© 2024 Robox SpA

Circuit element definition

Circuit elements (ITEM) have a common initial part, then a specific data area for each type of element. The common part is as follows:

Name	Offset	Type	Description
CODE	+0	U16	Element type code.
SIZE	+2	U16	Full size of element (all fields).
DATE	+4	...	Specific data (optional).

The expected elements are as follows:

Symbolic	ID	Type	Data	Description
BB	0x0001			Start of branch.
NL	0x0002			Next branch level . Notes: must always be between a BB and an EB.
EB	0x0003			End of branch . Notes: must always correspond to a BB.
OC	0x0004		U32=Destination (#3): BOOL 54 types only.	Open contact.
CC	0x0005		U32=Destination (#3): BOOL 54 types only.	Closed contact.
RE	0x0006		U32=Destination (#3): EDGE 54 types only.	Rising edge.
FE	0x0007		U32=Destination (#3): EDGE 54 types only.	Falling edge.
AF	0x0008			Always false.
OUT	0x0009	F	U32=Destination (#3): BOOL 54 types only.	Output.
OL	0x000A	F	U32=Destination (#3): BOOL 54 types only.	Output latch (set).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
OUL	0x000B	F	U32=Destination (#3): BOOL 54 types only.	Output unlatch (reset).
NOUT	0x000C	F	U32=Destination (#3): BOOL 54 types only	Output denied
CLR	0x000D	F	U32=Destination (#3): TIMER 54 and COUNTER 54 types only.	Zeroing (clear).
TP	0x000E	F	U32=Destination (#3): TIMER 54 types only U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer pulse.
TON	0x000F	F	U32=Preset value (#3): TIMER 54 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer on delay.
TOFF	0x0010	F	U32=Preset value (#3): TIMER 54 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer off delay.
TSQ	0x0011	F	U32=Preset value (#3): TIMER 54 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable square wave timer.
TINT	0x0012	F	U32=Preset value (#3): TIMER 54 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable integral timer.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
CU	0x0013	F	U32=Preset value (#3): COUNTER 54 types only U32=Preset value (#4).	Count up.
CD	0x0014	F	U32=Preset value (#3): COUNTER 54 types only U32=Preset value (#4).	Count down.
CUR	0x0015	F	U32=Preset value (#3): COUNTER 54 types only. U32=Preset value (#4).	Rotary count up.
CDR	0x0016	F	U32=Preset value (#3): COUNTER 54 types only. U32=Preset value (#4).	Count down rotary.
LD	0x0017	F	U32=Preset value (#3): COUNTER 54 types only.	Preset on counter.
MATH	0x0018	F	U32=Expression (#5) U32=Destination (#3) U8[]=Mathematical expression 46 data.	Math block.
EQ	0x0019		U32=Source A (#2) U32=Source B (#2).	Equality comparison block (A == B).
NE	0x001A		U32=Source A (#2) U32=Source B (#2)	Different comparison block (A != B).
GT	0x001B		U32=Source A (#2) U32=Source B (#2)	Major comparison block (A > B).
GE	0x001C		U32=Source A (#2) U32=Source B (#2)	Major/equal comparison block (A >= B).
LT	0x001D		U32=Source A (#2) U32=Source B (#2)	Minor comparison block (A < B).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
LE	0x001E		U32=Source A (#2) U32=Source B (#2)	Minor/equal comparison block (A <= B).
COPY	0x001F	F	U32=Source (#2) U32=Destination (#3) U32=N. Elements (#2)	Copies data from source to destination, for the specified no. of elements.
CALL	0x0020	F	U32 Block ID to call	Call of a block (sub-routine).
RET	0x0021	F		Return from a block to the inside.
ADD	0x0022	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Sum (A + B).
SUB	0x0023	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Subtraction (A - B).
MUL	0x0024	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Multiplication (A * B).
DIV	0x0025	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Division (A / B).
AND	0x0026	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	AND binary (A & B).
OR	0x0027	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Binary OR (A | B).
NOT	0x0028	F	U32=Source A (#2) U32=Destination (#3)	NOT binary (~A).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
XOR	0x0029	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	XOR binary (A ^ B).
SQRT	0x002A	F	U32=Source A (#2) U32=Destination (#3)	Square root of A.
MOD	0x002B	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Module (A mod B).
POW	0x002C	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Power elevation (A pow B).
SIN	0x002D	F	U32=Source A (#2) U32=Destination (#3)	Sine of A.
COS	0x002E	F	U32=Source A (#2) U32=Destination (#3)	Cosine of A.
TAN	0x002F	F	U32=Source A (#2) U32=Destination (#3)	Tangent of A.
ASIN	0x0030	F	U32=Source A (#2) U32=Destination (#3)	Archosene of A.
ACOS	0x0031	F	U32=Source A (#2) U32=Destination (#3)	Arcocosene of A.
ATAN	0x0032	F	U32=Source A (#2) U32=Destination (#3)	Arcotangent of A.
ATAN2	0x0033	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Arcotangent of A, B.
LOG	0x0034	F	U32=Source A (#2)	Base 10 logarithm of A.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			U32=Destination (#3)	
LN	0x0035	F	U32=Source A (#2) U32=Destination (#3)	Base E logarithm of A.
ABS	0x0036	F	U32=Source A (#2) U32=Destination (#3)	Absolute value of A
SIGN	0x0037	F	U32=Source A (#2) U32=Destination (#3)	A sign.
ROUND	0x0038	F	U32=Source A (#2) U32=Destination (#3)	Rounding to the nearest integer to A.
MIN	0x0039	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Minimum between A and B.
MAX	0x003A	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Maximum between A and B.
RANGE	0x003B	F	U32=Source A (#2) U32=Source B (#2) U32=Source C (#2) U32=Destination (#3): BOOL 54 types only.	Checks whether A contained in the range B-C.
LIMIT	0x003C	F	U32=Source A (#2) U32=Source B (#2) U32=Source C (#2) U32=Destination (#3)	Limit A to the B-C range.
ROR	0x003D	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Right bit rotation (A >> B).
ROL	0x003E	F	U32=Source A (#2) U32=Source B (#2)	Left bit rotation (A << B).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			U32=Destination (#3)	
SHR	0x003F	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Shift bit to the right (A >> B).
SHL	0x0040	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Shift bit to the left (A << B).
MOV	0x0041	F	U32=Source A (#2) U32=Destination (#3)	Assign A.
TODEG	0x0042	F	U32=Source A (#2) U32=Destination (#3)	Conversion of A from RAD to DEG.
TORAD	0x0043	F	U32=Source A (#2) U32=Destination (#3)	Conversion of A from DEG to RAD.
NEG	0x0044			Negation of logical state.
NOP	0x0045	F		No operation.
EVAL	0x0046		U32=Expression (#5) U8[]=Mathematical expression 46 data.	Evaluation expression (boolean).
OBCALL	0x0047	F	U32=Instance name (#6) U32=Complete method name (#9) U32=List of parameter names (#7) U32=List of parameter values (#7) U32=Result (#3) U8[]=Mathematical expression 46 data (#8)	Call to method of an OB instance.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
FT	0x0048			Initial execution
ARE	0x0049			Rising edge (anonymous)
AFE	0x004A			Falling edge (anonymous)

(#1) Local bit variable (type I32 or U32) with direct addressing (any constant indexes and number of bits).

(#2) Source/destination means variable name or ascii representation of a numeric value (source only), or an offset to the string table 52.

(#3) By destination means name of a variable with offset to the string table 52.

(#4) Value (or preset value) means a numeric value in string format: empty string (or offset 0) means no value assigned.

(#5) Expression means string with offset to string table 52.

(#6) Instance name means name of a variable with offset to the string table 52*, whose TYPEID refers to a user structure with FISOBC flag.*

(#7) List means a string (with offset to the string table 52*), consisting of a number of sub-strings separated with ascii character 0xFF: the two lists (names/values) MUST always have the same no. of elements or be both empty (=0).*

(#8) The call to a method of an OB instantiation is actually translated as a mathematical expression, in which any parameters (values in natural order) are prepared on the stack, the call to the method is made, and then any result is found from the stack.

(#9) Full name of the method (unique), understood as a string with offset to the string table 52.

Mathematical expression data

Mathematical expression data (used by MATH, COND, and OBCALL elements), is a series of contiguous operations (ordered in inverse polity) with the following common header:

Name	Offset	Type	Description
CODE	+0	U8	Operation code.
DATE	+1	...	Specific data (optional).

The expected operations are as follows:

Symbolic	ID	Data	In (stack)	Out (stack)	Description
PUSH	0x01	DBL=value	--	DBL	Push value into stack.
PUSHV	0x02	U32=Source (#1)	--	DBL	Push variable value into stack.
ADD	0x03		DBL0,DBL1	DBL	Sum.
SUB	0x04		DBL0,DBL1	DBL	Subtraction (#2).

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
MUL	0x05		DBL0,DBL1	DBL	Multiplication (#2).
DIV	0x06		DBL0,DBL1	DBL	Division (#2).
MOD	0x07		DBL	DBL	Module (#3).
ABS	0x08		DBL	DBL	Absolute value (#3).
SIN	0x09		DBL	DBL	Sine.
COS	0X0A		DBL	DBL	Cosine.
TAN	0x0B		DBL	DBL	Tangent.
ASIN	0x0C		DBL	DBL	Arcosene.
ACOS	0x0D		DBL	DBL	Arcocosene.
ATAN	0x0E		DBL	DBL	Arcotangent.
POW	0x0F		DBL	DBL	Exponent.
LOG10	0x10		DBL	DBL	Logarithm base 10.
LOG	0x11		DBL	DBL	Natural logarithm.
R_AND	0x12		U32	U32	Binary AND.
R_OR	0x13		U32	U32	OR binary.
R_NOT	0x14		U32	U32	NOT binary.
R_XOR	0x15		U32	U32	XOR binary.
POPV	0x16	U32=Destination (#1)	DBL	--	Pop value from stack to variable.
R_ROR	0x17		U32	U32	Right-hand bit rotation.
R_ROL	0x18		U32	U32	Left-hand bit rotation.
R_SHR	0x19		U32	U32	Shift bit to the right.
R_SHL	0x1A		U32	U32	Shift bit to the left.
TOBIN	0x1B		U32	U32	BCD to binary conversion.
TOBCD	0x1C		U32	U32	Binary to BCD conversion.

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
TODEG	0x1D		DBL	DBL	Conversion from radians to degrees.
TORAD	0x1E		DBL	DBL	Conversion from degrees to radians.
ATAN2	0x1F		DBL0,DBL1	DBL	Arcotangent with two parameters.
PUSHVB	0x20	U32=Source (#1)	DBL	--	Push variable bit value into stack.
POPVB	0x21	U32=Destination (#1)	DBL	--	Pop value from stack to variable bit.
POPL	0x22		BOOL	--	Pop logical value (0=false, otherwise true).
	0x23				(reserved)
EQ	0x24		DBL0,DBL1	BOOL	Comparison: equal.
NE	0x25		DBL0,DBL1	BOOL	Comparison: different.
LT	0x26		DBL0,DBL1	BOOL	Comparison: minor.
LE	0x27		DBL0,DBL1	BOOL	Comparison: less than or equal to.
GT	0x28		DBL0,DBL1	BOOL	Comparison: greater than.
GE	0x29		DBL0,DBL1	BOOL	Comparison: greater than or equal to.
AND	0x2A		BOOL,BOOL	BOOL	AND logical.
OR	0x2B		BOOL,BOOL	BOOL	OR logical.
NOT	0x2C		BOOL	BOOL	NOT logical.
XOR	0x2D		BOOL,BOOL	BOOL	logical XOR.
CHS	0x2E		DBL	DBL	Sign change.

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
SHR	0x2F		U32	U32	Logical shift to the right.
SHL	0x30		U32	U32	Logical shift to the left
OBCALL	0x31	U32=Method (#2)	?	?	Direct method call to OB instantiation
POP	0x32		*	--	Pop value (skip)
PUSHP	0x33	U32=Source (#1)	U32	U32	Push pointer to variable

(#1) Source/destination means name of a variable, i.e., an offset to the string table 52.

(#2) By method means string, i.e., an offset to the string table 52, with format "<instance_name>.#<hexid_method>" or "<instance_name>[<index>].#<hexid_method>" (hexid = 32bit hexadecimal ascii number, without prefix 0x. Example: #000F00A09).

Table of OB classes

The OB classes table (OBCLASSES) consists of a number of contiguous definitions (OBCLASS), each of which contains all the information about a specific user-defined object block class.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 30.OBCLASSOFF and HEADER 30.OBCLASSCNT fields, respectively.

OB class definition

An OB class (OBCLASS) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	OB class ID Note: the values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved)
FLAGS	+6	U16	OB class settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 Uncompressed MODIFIER field (NCF1).
NAME	+8	U32	Offset OB class name (relative to strings 52).
TEXT	+12	U32	Offset description of OB class (relative to

© 2024 Robox SpA

Name	Offset	Type	Description
			compressed 53 strings, or strings 52 if NCF0).
VERSION	+16	U32	OB class version (nvMake format).
INTFID	+20	U32	Unique obect block interface identifier.
PATHID	+24	U32	Path identifier of the .OBB file: 0=absolute path 1=(reserved) 2=Path relative to workspace 3=Path related to factory RDE3
FILENAME	+28	U32	Offset file name .OBB (relative to strings 52) (#1).
SRCID	+32	U32	Source identifier 57 of the OB class.
METHODSOFF	+36	U32	Offset of the OB methods table 50 (relative to the file).
METHODSCNT	+40	U32	OB methods table 50 size
ATTRIB	+44	U32	OB class attributes.
MODIFIER	+48	U32	Offset modifiers of class OB (relative to compressed st 53 rings, or strings 52 if NCF1).

(#1) In case of PATHID other than 0, the .OBB filename has no paths within it, but is simply the filename with relative to the PATHID path: in case of PATHID equal to 0 instead, the .OBB filename might contain a relative/absolute path.

Table of OB methods

The OB methods table (OBMETHODS) consists of a number of contiguous (OBMETHOD) definitions, each of which contains all the information about a specific OB method related to the OB class to which it belongs.

The location (relative to the file) and the number of elements in the table are specified by the OBCLASS 49.METHODSOFF and OBCLASS 49.METHODSCNT fields, respectively.

OB method definition

An OB method (OBMETHOD) consists of the following information:

© 2024 Robox SpA

Name	Offset	Type	Description
ID	+0	U32	OB method ID. Note: the values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved)
FLAGS	+6	U16	OB method settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 Uncompressed MODIFIER field (NCF1).
NAME	+8	U32	Offset OB method name (related to strings 52).
KNAME	+12	U32	Offset key name of OB method (relative to strings 52).
TEXT	+16	U32	Offset description of OB method (relative to compressed 53 strings, or strings 52 if NCF0).
CALLID	+20	U32	Method ID in call (#1)
PARAMSOFF	+24	U32	Offset of parameter table for OB method 51 (relative to file).
PARAMSCNT	+28	U32	Size of parameter table by OB method 51.
RETTYPEID	+32	U32	Return variable type ID. Note: Can be a default 53 or user-defined 33 type 53.
ATTRIB	+36	U32	Attributes method.
MODIFIER	+40	U32	OB method modifier offsets (relative to compressed st 53 rings, or strings 52 if NCF1).

(#1) The value of the CALLID field is derived from the value of the METHOD.ID field, in the source .OBB file.

Parameter table for OB method

The parameter table by OB method (OBMETHODPS) consists of a number of contiguous definitions (OBMETHODP), each of which contains all the information about a specific parameter related to the OB method to which it belongs.

The location (relative to the file) and the number of elements in the table are specified by the OBMETHOD 50.PARAMSOFF and OBMETHOD 50.PARAMSCNT fields, respectively.

© 2024 Robox SpA

Parameter definition for OB method

A parameter for OB method (OBMETHODP) consists of the following information:

Name	Offset	Type	Description
TYPEID	+0	U32	Variable type ID Note: Can be a predefined type 53 or user-defined 33.
ATTRIB	+4	U16	Parameter attributes: 0x0001 Parameter passed by reference (ABYREF) 0x0002 Parameter passed by pointer (ABYPTR)
FLAGS	+6	U16	Parameter settings: 0x0001 Uncompressed TEXT field (NCF0).
NAME	+8	U32	Offset parameter name (relative to strings 52).
TEXT	+12	U32	Offset parameter description (relative to compressed 53 strings, or strings 52 if NCF0).
DIM0	+16	U32	1st array size (0=unspecified)(#1).
DIM1	+20	U32	2nd array size (0=unspecified)(#1).
DIM2	+24	U32	3rd array size (0=unspecified)(#1).

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum size should still be considered only 16bit (i.e., maximum index 0xFFFF).

Table of strings

The string table (STRINGS) consists of a number of consecutive ascii strings, each terminated the ascii character \0 (zero) and of variable size. The table must contain at least one character and must be \0, so that strings with offset 0 actually point to a null string.

The location (relative to the file) and size of the table are specified in the following fields, respectively, depending on where used:

- by the HEADER 30.STRINGSOFF and HEADER 30.STRINGSSIZE fields, if used in ladder/diagram files.
- by the fields LIVEHEADER 123.STRINGSOFF and LIVEHEADER 123.STRINGSSIZE, if used in live edits.

© 2024 Robox SpA

Table of compressed strings

The compressed string table (CSTRINGS) consists of a number of consecutive ascii strings, each terminated the ascii character \0 (zero) and of variable size.

The location (relative to the file) and size of the table are specified by the HEADER 30 .STRINGSOFF and HEADER 30 .STRINGSSIZE fields, respectively: the compression state, which is optional, is instead specified by the HEADER 30 .CSTRINGSCTYPE field.

Annotations:

- in case of HEADER 30 .CSTRINGSCTYPE compression = 1, use the standard library functions (QT >= 3.3.4) qUncompress() to decompress the table properly.
- offsets referring to this table are always understood to refer to the uncompressed table.

Table of live changes

The embedded live changes table (EMBLIVES) consists of a number of contiguous (EMBLIVE) definitions, each of which contains all the information about a specific set of live changes to the ladder file itself: each set is actually an independent live changes 122 file but embedded in the ladder file itself.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 30 .EMBLIVESOFF and HEADER 30 .EMBLIVESCNT fields, respectively.

When loading, the various live changes files are applied in the exact order in which they appear. The actual application of changes is governed by the appropriate flag in EMBLIVE 53 .FLAGS (value 0x0001): only the last live changes file may possibly have this flag set to false, indicating live changes not yet applied.

Definition of embedded live changes

An embedded live changes (EMBLIVE) consists of the following information:

Name	Offset	Type	Description
FILEOFF	+0	U32	Offset of the live changes 122 file (offset relative to the ladder file).
FILESIZE	+4	U32	Size of the live changes 122 file.
FLAGS	+8	U16	File settings: 0x0001 = Actual changes to be applied.
	+10	B[22]	(reserved)

Predefined data types

Data types are identified by a unique ID code, whether they are predefined or user-defined: IDs are divided as follows:

Range	Description
0x00000000-0x0000FFFF	Default data types (#1)

© 2024 Robox SpA

Range	Description
0x00010000-0xFFFFFFFE	User-defined data types
0xFFFFFF	Reserved

(#1) default data types are never saved in the .LAD file, but are always considered to exist. The default data types are as follows:

ID	Symbolic	Description	RTE (0x00)
0x00000000	(reserved)	Invalid data type	
0x00000001	I8	Signed 8bit integer	1
0x00000002	U8	Unsigned integer 8bit	1
0x00000003	I16	Integer 16bit with sign	2
0x00000004	U16	Unsigned 16bit integer	2
0x00000005	I32	Integer 32bit with sign	4
0x00000006	U32	Unsigned 32bit integer	4
0x00000007	I64	Integer 64bit with sign	
0x00000008	U64	Unsigned integer 64bit	
0x00000009	STRING	String (0 terminated)	...
0x00000A	FLOAT	Floating point 32bit	4
0x00000B	REAL	Floading point 64bit (double)	8
0x00000C	BOOL	Boolean	4 (#1)
0x00000D	TIMER	Default timer structure 55	32
0x00000E	COUNTER	Default counter structure 55	48
0x00000F	ALIAS	Alias (or literally) 54	0
0x00000010	EDGE	Special storage for rising edge (RE) and falling edge (FE) elements.	4
0x00000011	POWERSET	"Power Set" logic variable	4 (#2)
0x00000012	AXESGROUP	"Axes group" logic variable	4 (#2)
0x00000013	PATH	"Path" logic variable	4 (#2)
0x00000014	POINT_L	"Library point" logical variable	4 (#2)

© 2024 Robox SpA

(#1) BOOL type in RTE is represented, for performance reasons, as I32, while by RDE3 it is seen as U8.

(#2) Logical variables in RTE are always represented as a pointer to memory (4bytes), even if they were embedded in user-defined structures. For further information on the composition (elements) of such variables, see the RTE technical documentation).

The representation and size of data types depend directly on the runtime destination ID(HEADER 30.TARGETID). The sizes of the data types are listed in the RTE columns and are expressed in bytes.

Default timer structure

The default TIMER structure consists of the following fields:

Type	Name	Description	Read	Write
BOOL	Q	Timer Outcome	X	
REAL	ET	Current value [ms]	X	X (#1)
REAL	PT	Preset value [ms]	X	X
REAL	PT2	Preset value 2 [ms]	X	X
BOOL	EN	Enable timer	X	

(#1) The ET field can be written only when the timer has not yet been assigned.

COUNTER default structure

The default COUNTER structure consists of the following fields:

Type	Name	Description	Read	Write
BOOL	QU	Outcome of count against preset	X	
BOOL	QD	Count outcome with respect to 0	X	
I32	CV	Current count value	X	X (#1)
I32	PV	Preset value	X	X
BOOL	CU	Status of the "count up" signal	X	
BOOL	CD	State of the "count down" signal	X	

(#1) The CV field can be written only when the counter has not yet been assigned.

Alias (or literally)

The list of data types that can be used as aliases are as follows:

© 2024 Robox SpA

ID	Name	Type	Description
0x0001	INP(n)	BOOL	Input channel
0x0002	INP_W(n)	U16	Input word 16bit
0x0003	INP_W(n).b	BOOL	Input word 16bit, bit access
0x0004	INP_DW(n)	U32	Input word 32bit
0x0005	INP_DW(n).b	BOOL	Input word 32bit, bit access
0x0006	OUT	BOOL	Output channel
0x0007	OUT_W(n)	U16	Output word 16bit
0x0008	OUT_W(n).b	BOOL	Output word 16bit, bit access
0x0009	OUT_DW(n)	U32	Output word 32bit
0x000A	OUT_DW(n).b	BOOL	Output word 32bit, bit access
0x000B	R(n)	I32	Global integer register (volatile)
0x000C	R(n).b	BOOL	Global integer (volatile) register, bit access
0x000D	NVR(n)	I32	Global integer register (nonvolatile)
0x000E	NVR(n).b	BOOL	Global integer (nonvolatile) register, bit access
0x000F	RR(n)	REAL	Global real (volatile) register
0x0010	NVRR(n)	REAL	Real global register (non-volatile)
0x0011	AM(n)	U32	Alarm mask
0x0012	AM(n).b	U32	Alarm mask, bit access

Annotations:

- if the variable is of array type, the array index (n) can be either a constant or an unindexed variable.
- if the variable involves bit access, .b can be either a constant or a variable and must be in the compatible range the size of the reference variable.

© 2024 Robox SpA

Predefined variables

Variables are identified by a unique ID code, whether they are predefined or user-defined: IDs are divided as follows:

Range	Description
0x00000000-0x0000FFFF	Default variable
0x0001000-0xFFFFFFFE	User-defined variable
0xFFFFFF	Reserved

CAUTION: Default variables are never saved in the .LAD file, but are always considered to exist.

The default variables are as follows:

ID	Name	Type	Description
0x00000001	INP(n)	BOOL	Input channel
0x00000002	INP_W(n)	U16	Input word 16bit
0x00000003	INP_DW(n)	U32	Input word 32bit
0x00000004	OUT	BOOL	Output channel
0x00000005	OUT_W(n)	U16	Output word 16bit
0x00000006	OUT_DW(n)	U32	Output word 32bit
0x00000007	R(n)	I32	Global integer register (volatile)
0x00000008	NVR(n)	I32	Global integer register (non-volatile)
0x00000009	RR(n)	REAL	Real global register (volatile)
0x00000A	NVRR(n)	REAL	Real global register (non-volatile)
0x00000B	AM(n)	U32	Alarm mask

Annotation:

- if the variable is of array type, the array index (n) can be either a constant or an unindexed variable.
- RDE3 creates additional predefined variables with completely arbitrary assignment of IDs (in the range 0x00000100-0x00010000): RTE recognizes such variables by name and not by that ID.

Source identifiers

Source identifiers are used to uniquely identify the source of a certain entity (or element) within the ladder file, such as structures, variables, or OB classes. The value of the U32 identifier is composed as follows:

- U16 high, group identifier;

© 2024 Robox SpA

- U16 low, element identifier in the group.

The expected source identifiers are as follows:

Group	Element	Description
0x0000	0x0000	User defined entity manually
0x0001	0x0000	Entity imported from external file: generic file
0x0002	0x0000	Entity imported (forced) from RTE project: generic source
0x0002	0x0001	Imported entity (forced) from RTE project: configurator source

File format (version 1.4)

File format (version 1.4)

A ladder/diagram programming language program is a single file (extension .LAD) containing both editing (source) and execution (runtime) information.

A ladder program consists primarily of a header 58, then several sections of which the main ones are:

- table of structures 61, which are the user-defined data types.
- table of local variables 63, i.e., the general variables of the program.
- table of OB classes 78, i.e., the object block classes referenced by the program.
- table of blocks 65, i.e., the sub-routines in other programming languages;
- table of live changes 82, i.e., any embedded live change files;
- table of strings 81;
- table of compressed strings 81.

In the block list, there is always a main block called MAIN and possibly additional blocks (comparable to sub-routines in other programming languages): each block contains a list of circuits 66 that constitute the actual ladder program: each circuit in turn is a logically structured list of elements 67.

NOTE: All numeric data stored in this file are to be considered in little indian (x86 standard) format.

File header

At the beginning of the file (offset 0) is a general header (HEADER) that contains access information for all remaining components and sub-sections of the file.

Name	Offset	Type	Description
SIGN	+0	U32	File signature ('LADF').
VERSION	+4	U32	Version of the file (1.4.0).

© 2024 Robox SpA

Name	Offset	Type	Description
GENERATOR	+8	U32	Name and version of the file generator (offset relative to string table).
TYPE	+12	U16	Execution type: 0 = Synchronous execution (SY). 1 = High priority execution (HP). 2 = Normal priority execution (NP). 3 = Low priority execution (LP).
FREQUENCY	+14	U16	Execution frequency in [hz], from 1hz to 3000hz, default 200hz .
FLAGS	+16	U32	General settings: 0x000001 = The file contains formal verification errors. 0x000002 = The file contains references to external .OBB files (OB classes). 0x000004 = The file uses UTF-8 encoding.
BUILDYEAR	+20	U16	Year last generation (4 digits).
BUILDMONTH	+22	U8	Month last generation (1-12).
BUILDDAY	+23	U8	Day last generation (1-31).
BUILDHOUR	+24	U8	Last hour generation (0-23).
BUILDMINUTE	+25	U8	Last minute compilation (0-59).
BUILDSECOND	+26	U8	Second last compilation (0-59).
TARGETID	+27	U8	Runtime destination ID: 0x00 = Firmware RTE (PowerPC G2) 0x01 = RTE firmware (PowerPC P2020)

© 2024 Robox SpA

Name	Offset	Type	Description
VID	+28	U32	File version ID.
PLVID	+32	U32	Version ID of file pre live edits (if present, otherwise worth 0).
IARECNT	+36	U32	No. of ARE type elements present in the file.
IAFECNT	+40	U32	No. of AFE type elements present in the file.
	+44	U8[84]	(reserved)
BLOCKSOFF	+128	U32	Offset of the block table 65 (relative to the file).
BLOCKSCNT	+132	U32	No. of elements of the block table 65.
STRINGSOFF	+136	U32	Offset of the string table 81 (relative to the file).
STRINGSSIZE	+140	U32	Size of the string table 81.
CSTRINGSOFF	+144	U32	Offset of the compressed string table 81 (relative to the file).
CSTRINGSSIZE	+148	U32	Size of the compressed string table 81.
CSTRINGSCTYPE	+152	U8	Type of compression used for ta compressed string table 81:. 0 = No compression. 1 = Standard compression Qt 3.3.x.
	+153	U8[3]	(reserved)
EMBLIVESOFF	+156	U32	Offset of the embedded live changes table 82 (relative to the file) .
EMBLIVESCNT	+160	U32	No. of elements of the embedded live changes table 82.
STRUCTSOFF	+164	U32	Offset of the structures table 61 (relative to the file).

© 2024 Robox SpA

Name	Offset	Type	Description
STRUCTSCNT	+168	U32	No. of elements of the structures table 61.
VARSOFF	+172	U32	Offset of variable table 63 (relative to file).
VARSCNT	+176	U32	No. of elements of the variable table 63.
OBCLASSESOFF	+180	U32	Offset of the table classes OB 78 (relative to the file).
OBCLASSESCNT	+184	U32	No. of elements of the OB classes table 78 (relative to the file)
LASTBLOCKID	+188	U32	Last block ID used.
LASTRUNGID	+192	U32	Last circuit ID used.
LASTSTRUCTID	+196	U32	Last structure ID used.
LASTVARID	+200	U32	Last variable ID used.
LASTOBCLASSID	+204	U32	Last OB class ID used.
	+208	U8[304]	(reserved)

Table of structures

The structures table (STRUCTS) consists of a number of contiguous definitions (STRUCTs), each of which contains all the information about a specific user-defined structure: there are no default data types 82 in this table.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 58.STRUCTSOFF and HEADER 58.STRUCTSCNT fields, respectively.

Structure definition

A structure (STRUCT) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	Structure type ID Note: only values 0x00010000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved, runtime use)
FLAGS	+6	U16	Structure settings:

© 2024 Robox SpA

Name	Offset	Type	Description
			0x0001 Uncompressed TEXT field (NCF0). 0x0002 Uncompressed MODIFIER field (NCF1).
NAME	+8	U32	Offset structure name (relative to strings 81).
TEXT	+12	U32	Offset structure description (relative to compressed 81 strings, or strings 81 if NCF0).
VARSOFF	+16	U32	Variable table 63 offset (relative to file) (#1)
VARSCNT	+20	U32	Variable table 63 size
SIZE	+24	U32	Structure size [bytes]
SRCID	+28	U32	Source identifier 86 of the structure
OBCID	+32	U32	Reference OB class ID (0=none)(#2,#3)
ATTRIB	+36	U32	Structure attributes: 0x000001 Structure referred to OB class (AREFOBC)(#2) 0x000002 Main structure OB class (AISOBC)(#2,#3) 0x000004 Structure (ref.a) externally defined (AEXTERNAL)
MODIFIER	+40	U32	Offset modifiers of structure (relative to compressed 81 strings, or strings 81 if NCF1).
REFID	+44	U32	Structure type reference ID (#4).
EXTNAME	+48	U32	Offset external structure name (relative to strings 81).
	+52	U8[16]	(reserved)

(#1) The structure definition uses a normal variable definition table to define its elements. In this case, variable IDs are meaningless and should always be 0.

(#2) At equal non-0 OBCID value with AREFOBC attribute, there can be only one structure with AISOBC flag, while there can be 0 or more structures without AISOBC attribute.

© 2024 Robox SpA

(#3) A structure is called an "orphan" structure if it is in the condition of AREFOBC attribute set and OBCID field = 0: this condition happens when updating/removing an OB class, there are still variables (or other) referencing it.

(#4) In case of AREFOBC attribute the REFID field contains the ID of the structure as defined of the source OBB file (0 in case of AISOBC attribute).

Table of variables

The variable table (VARS) consists of a number of contiguous definitions (VARs), each of which contains all the information about a specific user-defined variable: there are no default variables 85 in this table.

The location (relative to the file) and number of elements in the program variable table are specified in the following fields, respectively, depending on where used

- by the HEADER 58.VARSOFF and HEADER 58.VARSCNT fields if used in the file header;
- by the fields STRUCT 61.VARSOFF and STRUCT 61.VARSCNT if used in structure definition;
- by the fields LIVECMD 131.VARSOFF and LIVECMD 131.VARSCNT if used in command definition.

Variable definition

A variable (VAR) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	Variable ID (#2) Note: only values 0x00010000-0xFFFFFE; value 0xFFFFFF is a special meaning ID; values 0x000001-0x0000FFFF are reserved for default variables.
TYPEID	+4	U32	Variable type ID Note: Can be a default 82 or user-defined 61 type 82.
	+8	U8[2]	(reserved, runtime use)
FLAGS	+10	U16	Variable settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 BITTEXT field uncompressed (NCF1) 0x0004 Uncompressed MODIFIER field (NCF2)
NAME	+12	U32	Offset variable name (relative to strings 81).
TEXT	+16	U32	Offset variable description (relative to

© 2024 Robox SpA

Name	Offset	Type	Description
			compressed 81 strings, or strings 81 if NCF0).
BITTEXT	+20	U32	Offset variable bit descriptions, i.e., sequence of strings related to each individual bit separated by the ascii character 0xFF (related to compressed 81 strings, or strings 81 if NCF0). (#2)
DIM0	+24	U32	1st array size (0=unspecified) (#1) or index 0 for Alias 83 (TYPEID = 0x00000F).
DIM1	+28	U32	2nd array size (0=unspecified) (#1) or index 1 for Alias 83 (TYPEID = 0x00000F).
DIM2	+32	U32	3rd array size (0=unspecified) (#1) or index 2 for Alias 83 (TYPEID = 0x00000F).
ALIASID	+36	U16	Alias type 84 ID (#2)
ALIASBIT	+38	U16	No. of bits per alias (depends on the type of alias ID)(#2)
SIZE	+40	U32	Size of variable. (#3)
SRCID	+44	U32	Source identifier 86 of the variable.(#2,#4)
VALUE	+48	U32	Offset initial value of variable (relative to strings 81, 0=no value).
ATTRIB	+52	U32	Variable attributes: 0x000001 Variable can be written (WRITE) 0x000002 The variable can be read (READ) 0x000004 The variable can be forced (FORCE) 0x000008 The variable can be released (RELEASE) 0x000010 The variable is constant (CONST)

© 2024 Robox SpA

Name	Offset	Type	Description
			0x000020 The variable has no bit access (NOBITS)
MODIFIER	+56	U32	Variable modifier offsets (related to compressed st 81 rings, or strings 81 if NCF2).
DIM0L	+60	U32	Offset label for 1st array size (relative to strings 81). (#5)
DIM1L	+64	U32	Offset label for 2nd array size (relative to strings 81). (#5)
DIM2L	+68	U32	Offset label for 3rd array dimension (relative to strings 81). (#5)
	+72	U8[16]	(reserved)

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum size should still be considered only 16bit (i.e., maximum index 0xFFFF).

(#2) Field not used (value 0) if used in structure definition.

(#3) Field not used (value 0) if used by defining an Alias (TYPEID = 0x00000F)

(#4) Field not used (value 0) if used in live changes command definition.

(#5) Label means the name of a variable (constant value) or the ascii representation of a numeric value, i.e., an offset to the string table 81.

Table of blocks

The block table (BLOCKS) consists of a number of contiguous block definitions (BLOCKS), each of which contains all the information about a specific block and its contents on file.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 58.BLOCKSOFF and HEADER 58.BLOCKSCNT fields, respectively.

Block definition

A block (BLOCK) consists of the following data:

Name	Offset	Type	Description
ID	+0	U32	Unique ID of the block. Note: the values 0 and 0xFFFFFF are special meaning IDs.
STATUS	+4	U16	(reserved, runtime use)
FLAGS	+6	U16	Block settings: 0x0001 = Main block (ISMAIN).

© 2024 Robox SpA

Name	Offset	Type	Description
			0x0002 = Uncompressed TEXT field (NCF0). 0x0004 = Uncompressed COMMENT field (NCF1).
NAME	+8	U32	Offset block name (relative to strings 81). Note: If ISMAIN flag specified, name comes must be "main".
TEXT	+12	U32	Offset block description (relative to compressed 81 strings, or strings 81 if NCF0).
COMMENT	+16	U32	Offset comments for the block (relative to compressed 81 strings, or strings 81 if NCF1).
RUNGSOFF	+20	U32	Offset of the circuit table of the block 66 (relative to rows).
RUNGSCNT	+24	U32	No. of elements of the block circuit table 66.
	+28	B[36]	(reserved)

Table of block's circuits

The block circuit table (RUNGS) consists of a number of contiguous circuit definitions (RUNG), each of which contains all the information about a specific circuit and its contents.

The location (relative to the file) and the number of elements in the table are specified by the BLOCK 65.RUNGSOFF and BLOCK 65.RUNGSCNT fields, respectively.

Block circuit definition

A block circuit (RUNG) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	Unique ID of the circuit. Note: the values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved, runtime use)
FLAGS	+6	U16	Circuit settings:

© 2024 Robox SpA

Name	Offset	Type	Description
			0x0001 Uncompressed TEXT field (NCF0).
REFID	+8	U32	Reference circuit ID (0=none).
ITEMSOFF	+12	U32	Offset of the circuit element table 67 (relative to the file).
ITEMSSIZE	+16	U32	Size of the circuit element table 67.
TEXT	+20	U32	Offset circuit description (related to compressed st 81 rings, or strings 81 if NCF1).
	+24	U8[4]	(reserved)

Table of circuit's items

The circuit element table (ITEMS) consists of a number of contiguous circuit definitions (ITEMs), each of which contains all the information about a specific element.

The location (relative to the file) and size of the table are specified in the following fields, respectively, depending on where used:

- by the RUNG 66.ITEMSOFF and RUNG 66.ITEMSSIZE fields, if used in block circuit (RUNG).
- by the fields LIVECMD 131.ITEMSOFF and LIVECMD 131.ITEMSSIZE, if used in live change commands (LIVECMD).

Circuit element definition

Circuit elements (ITEMs) have a common initial part, then a specific data area for each type of element. The common part is as follows:

Name	Offset	Type	Description
CODE	+0	U16	Element type code.
SIZE	+2	U16	Full size of element (all fields).
DATE	+4	...	Specific data (optional).

The expected elements are as follows:

Symbolic	ID	Type	Data	Description
BB	0x0001			Start of branch.
NL	0x0002			Next branch level .

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
				Notes: must always be between a BB and an EB.
EB	0x0003			End of branch . Notes: must always correspond to a BB.
OC	0x0004		U32=Destination (#3): BOOL 83 types only.	Open contact.
CC	0x0005		U32=Destination (#3): BOOL 83 types only.	Closed contact.
RE	0x0006		U32=Destination (#3): EDGE 83 types only.	Rising edge.
FE	0x0007		U32=Destination (#3): EDGE 83 types only.	Falling edge.
AF	0x0008			Always false.
OUT	0x0009	F	U32=Destination (#3): BOOL 83 types only.	Output.
OL	0x000A	F	U32=Destination (#3): BOOL 83 types only.	Output latch (set).
OUL	0x000B	F	U32=Destination (#3): BOOL 83 types only.	Output unlatch (reset).
NOUT	0x000C	F	U32=Destination (#3): BOOL 83 types only	Output denied
CLR	0x000D	F	U32=Destination (#3): TIMER 83 and COUNTER 83 types only.	Zeroing (clear).
TP	0x000E	F	U32=Destination (#3): TIMER 83 types only U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer pulse.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
TON	0x000F	F	U32=Preset value (#3): TIMER 83 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer on delay.
TOFF	0x0010	F	U32=Preset value (#3): TIMER 83 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer off delay.
TSQ	0x0011	F	U32=Preset value (#3): TIMER 83 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable square wave timer.
TINT	0x0012	F	U32=Preset value (#3): TIMER 83 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable integral timer.
CU	0x0013	F	U32=Preset value (#3): COUNTER 83 types only U32=Preset value (#4).	Count up.
CD	0x0014	F	U32=Preset value (#3): COUNTER 83 types only U32=Preset value (#4).	Count down.
CUR	0x0015	F	U32=Preset value (#3): COUNTER 83 types only. U32=Preset value (#4).	Rotary count up.
CDR	0x0016	F	U32=Preset value (#3): COUNTER 83	Count down rotary.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			types only. U32=Preset value (#4).	
LD	0x0017	F	U32=Preset value (#3): COUNTER 83 types only.	Preset on counter.
MATH	0x0018	F	U32=Expression (#5) U32=Destination (#3) U8[]=Mathematical expression 75 data.	Math block.
EQ	0x0019		U32=Source A (#2) U32=Source B (#2).	Equality comparison block (A == B).
NE	0x001A		U32=Source A (#2) U32=Source B (#2)	Different comparison block (A != B).
GT	0x001B		U32=Source A (#2) U32=Source B (#2)	Major comparison block (A > B).
GE	0x001C		U32=Source A (#2) U32=Source B (#2)	Major/equal comparison block (A >= B).
LT	0x001D		U32=Source A (#2) U32=Source B (#2)	Minor comparison block (A < B).
LE	0x001E		U32=Source A (#2) U32=Source B (#2)	Minor/equal comparison block (A <= B).
COPY	0x001F	F	U32=Source (#2) U32=Destination (#3) U32=N. Elements (#2)	Copies data from source to destination, for the specified no. of elements.
CALL	0x0020	F	U32 Block ID to call	Call of a block (sub-routine).
RET	0x0021	F		Return from a block to the inside.
ADD	0x0022	F	U32=Source A (#2) U32=Source B (#2)	Sum (A + B).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			U32=Destination (#3)	
SUB	0x0023	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Subtraction (A - B).
MUL	0x0024	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Multiplication (A * B).
DIV	0x0025	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Division (A / B).
AND	0x0026	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	AND binary (A & B).
OR	0x0027	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Binary OR (A | B).
NOT	0x0028	F	U32=Source A (#2) U32=Destination (#3)	NOT binary (~A).
XOR	0x0029	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	XOR binary (A ^ B).
SQRT	0x002A	F	U32=Source A (#2) U32=Destination (#3)	Square root of A.
MOD	0x002B	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Module (A mod B).
POW	0x002C	F	U32=Source A (#2) U32=Source B (#2)	Power elevation (A pow B).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			U32=Destination (#3)	
SIN	0x002D	F	U32=Source A (#2) U32=Destination (#3)	Sine of A.
COS	0x002E	F	U32=Source A (#2) U32=Destination (#3)	Cosine of A.
TAN	0x002F	F	U32=Source A (#2) U32=Destination (#3)	Tangent of A.
ASIN	0x0030	F	U32=Source A (#2) U32=Destination (#3)	Archosene of A.
ACOS	0x0031	F	U32=Source A (#2) U32=Destination (#3)	Arcocosene of A.
ATAN	0x0032	F	U32=Source A (#2) U32=Destination (#3)	Arcotangent of A.
ATAN2	0x0033	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Arcotangent of A, B.
LOG	0x0034	F	U32=Source A (#2) U32=Destination (#3)	Base 10 logarithm of A.
LN	0x0035	F	U32=Source A (#2) U32=Destination (#3)	Base E logarithm of A.
ABS	0x0036	F	U32=Source A (#2) U32=Destination (#3)	Absolute value of A
SIGN	0x0037	F	U32=Source A (#2) U32=Destination (#3)	A sign.
ROUND	0x0038	F	U32=Source A (#2) U32=Destination (#3)	Rounding to the nearest integer to A.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
MIN	0x0039	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Minimum between A and B.
MAX	0x003A	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Maximum between A and B.
RANGE	0x003B	F	U32=Source A (#2) U32=Source B (#2) U32=Source C (#2) U32=Destination (#3): BOOL 83 types only.	Checks whether A contained in the range B-C.
LIMIT	0x003C	F	U32=Source A (#2) U32=Source B (#2) U32=Source C (#2) U32=Destination (#3)	Limit A to the B-C range.
ROR	0x003D	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Right bit rotation (A >> B).
ROL	0x003E	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Left bit rotation (A << B).
SHR	0x003F	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Shift bit to the right (A >> B).
SHL	0x0040	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Shift bit to the left (A << B).
MOV	0x0041	F	U32=Source A (#2) U32=Destination (#3)	Assign A.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
TODEG	0x0042	F	U32=Source A (#2) U32=Destination (#3)	Conversion of A from RAD to DEG.
TORAD	0x0043	F	U32=Source A (#2) U32=Destination (#3)	Conversion of A from DEG to RAD.
NEG	0x0044			Negation of logical state.
NOP	0x0045	F		No operation.
EVAL	0x0046		U32=Expression (#5) U8[]=Mathematical expression 75 data.	Evaluation expression (boolean).
OBCALL	0x0047	F	U32=Instance name (#6) U32=Complete method name (#9) U32=List of parameter names (#7) U32=List of parameter values (#7) U32=Result (#3) U8[]=Mathematical expression 75 data (#8)	Call to method of an OB instance.
FT	0x0048			Initial execution
ARE	0x0049			Rising edge (anonymous)
AFE	0x004A			Falling edge (anonymous)

(#1) Local bit variable (type I32 or U32) with direct addressing (any constant indexes and number of bits).

(#2) Source/destination means variable name or ascii representation of a numeric value (source only), or an offset to the string table 81.

(#3) By destination means name of a variable with offset to the string table 81.

(#4) Value (or preset value) means a numeric value in string format: empty string (or offset 0) means no value assigned.

(#5) Expression means string with offset to string table 81.

(#6) Instance name means name of a variable with offset to the string table 81, whose TYPEID refers to a user structure with FISOBC flag.

© 2024 Robox SpA

(#7) List means a string (with offset to the string table 81), consisting of a number of sub-strings separated with ascii character 0xFF: the two lists (names/values) MUST always have the same no. of elements or be both empty (=0).

(#8) The call to a method of an OB instantiation is actually translated as a mathematical expression, in which any parameters (values in natural order) are prepared on the stack, the call to the method is made, and then any result is found from the stack.

(#9) Full name of the method (unique), understood as a string with offset to the string table 81.

Mathematical expression data

Mathematical expression data (used by MATH, COND, and OBCALL elements), is a series of contiguous operations (ordered in inverse polity) with the following common header:

Name	Offset	Type	Description
CODE	+0	U8	Operation code.
DATE	+1	...	Specific data (optional).

The expected operations are as follows:

Symbolic	ID	Data	In (stack)	Out (stack)	Description
PUSH	0x01	DBL=value	--	DBL	Push value into stack.
PUSHV	0x02	U32=Source (#1)	--	DBL	Push variable value into stack.
ADD	0x03		DBL0,DBL1	DBL	Sum.
SUB	0x04		DBL0,DBL1	DBL	Subtraction (#2).
MUL	0x05		DBL0,DBL1	DBL	Multiplication (#2).
DIV	0x06		DBL0,DBL1	DBL	Division (#2).
MOD	0x07		DBL	DBL	Module (#3).
ABS	0x08		DBL	DBL	Absolute value (#3).
SIN	0x09		DBL	DBL	Sine.
COS	0X0A		DBL	DBL	Cosine.
TAN	0x0B		DBL	DBL	Tangent.
ASIN	0x0C		DBL	DBL	Arcosene.
ACOS	0x0D		DBL	DBL	Arcocosene.
ATAN	0x0E		DBL	DBL	Arcotangent.
POW	0x0F		DBL	DBL	Exponent.

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
LOG10	0x10		DBL	DBL	Logarithm base 10.
LOG	0x11		DBL	DBL	Natural logarithm.
R_AND	0x12		U32	U32	Binary AND.
R_OR	0x13		U32	U32	OR binary.
R_NOT	0x14		U32	U32	NOT binary.
R_XOR	0x15		U32	U32	XOR binary.
POPV	0x16	U32=Destination (#1)	DBL	--	Pop value from stack to variable.
R_ROR	0x17		U32	U32	Right-hand bit rotation.
R_ROL	0x18		U32	U32	Left-hand bit rotation.
R_SHR	0x19		U32	U32	Shift bit to the right.
R_SHL	0x1A		U32	U32	Shift bit to the left.
TOBIN	0x1B		U32	U32	BCD to binary conversion.
TOBCD	0x1C		U32	U32	Binary to BCD conversion.
TODEG	0x1D		DBL	DBL	Conversion from radians to degrees.
TORAD	0x1E		DBL	DBL	Conversion from degrees to radians.
ATAN2	0x1F		DBL0,DBL1	DBL	Arcotangent with two parameters.
PUSHVB	0x20	U32=Source (#1)	DBL	--	Push variable bit value into stack.
POPVB	0x21	U32=Destination (#1)	DBL	--	Pop value from stack to variable bit.
POPL	0x22		BOOL	--	Pop logical value (0=false,

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
					otherwise true).
	0x23				(reserved)
EQ	0x24		DBL0,DBL1	BOOL	Comparison: equal.
NE	0x25		DBL0,DBL1	BOOL	Comparison: different.
LT	0x26		DBL0,DBL1	BOOL	Comparison: minor.
LE	0x27		DBL0,DBL1	BOOL	Comparison: less than or equal to.
GT	0x28		DBL0,DBL1	BOOL	Comparison: greater than.
GE	0x29		DBL0,DBL1	BOOL	Comparison: greater than or equal to.
AND	0x2A		BOOL,BOOL	BOOL	AND logical.
OR	0x2B		BOOL,BOOL	BOOL	OR logical.
NOT	0x2C		BOOL	BOOL	NOT logical.
XOR	0x2D		BOOL,BOOL	BOOL	logical XOR.
CHS	0x2E		DBL	DBL	Sign change.
SHR	0x2F		U32	U32	Logical shift to the right.
SHL	0x30		U32	U32	Logical shift to the left
OBCALL	0x31	U32=Method (#2)	?	?	Direct method call to OB instantiation
POP	0x32		*	--	Pop value (skip)
PUSHP	0x33	U32=Source (#1)	U32	U32	Push pointer to variable

(#1) Source/destination means name of a variable, i.e., an offset to the string table 81.

(#2) By method means string, i.e., an offset to the string table 81*, with format "<instance_name>.#<hexid_method>" or "<instance_name>[<index>].#<hexid_method>" (hexid = 32bit hexadecimal ascii number, without prefix 0x. Example method no. 0xFA9 of OB instance 'myob' : "myob#00000FA9".*

© 2024 Robox SpA

Table of OB classes

The OB classes table (OBCLASSES) consists of a number of contiguous definitions (OBCLASS), each of which contains all the information about a specific user-defined object block class.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 58.OBCLASSOFF and HEADER 58.OBCLASSCNT fields, respectively.

OB class definition

An OB class (OBCLASS) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	OB class ID Note: the values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved)
FLAGS	+6	U16	OB class settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 Uncompressed MODIFIER field (NCF1).
NAME	+8	U32	Offset OB class name (relative to strings 81).
TEXT	+12	U32	Offset description of OB class (relative to compressed 81 strings, or strings 81 if NCF0).
VERSION	+16	U32	OB class version (nvMake format).
INTFID	+20	U32	Unique obect block interface identifier.
PATHID	+24	U32	Path identifier of the .OBB file: 0=absolute path 1=(reserved) 2=Path relative to workspace 3=Path related to factory RDE3
FILENAME	+28	U32	Offset file name .OBB (relative to strings 81) (#1).
SRCID	+32	U32	Source identifier 86 of the OB class.

© 2024 Robox SpA

Name	Offset	Type	Description
METHODSOFF	+36	U32	Offset of the OB methods table 79 (relative to the file).
METHODSCNT	+40	U32	OB methods table 79 size
ATTRIB	+44	U32	OB class attributes.
MODIFIER	+48	U32	Offset modifiers of class OB (related to compressed 81 strings, or strings 81 if NCF1).
	+52	U8[16]	(reserved)

(#1) In case of PATHID other than 0, the .OBB filename has no paths within it, but is simply the filename with relative to the PATHID path: in case of PATHID equal to 0 instead, the .OBB filename might contain a relative/absolute path.

Table of OB methods

The OB methods table (OBMETHODS) consists of a number of contiguous (OBMETHOD) definitions, each of which contains all the information about a specific OB method related to the OB class to which it belongs.

The location (relative to the file) and the number of elements in the table are specified by the OBCLASS 78.METHODSOFF and OBCLASS 78.METHODSCNT fields, respectively.

OB method definition

An OB method (OBMETHOD) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	OB method ID. Note: the values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved)
FLAGS	+6	U16	OB method settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 Uncompressed MODIFIER field (NCF1).
NAME	+8	U32	Offset OB method name (related to strings 81).
KNAME	+12	U32	Offset key name of OB method (relative to strings 81).

© 2024 Robox SpA

Name	Offset	Type	Description
TEXT	+16	U32	Offset description of OB method (relative to compressed 81 strings, or strings 81 if NCF0).
CALLID	+20	U32	Method ID in call (#1)
PARAMSOFF	+24	U32	Offset of parameter table for OB method 80 (relative to file).
PARAMSCNT	+28	U32	Size of parameter table by OB method 80.
RETTYPEID	+32	U32	Return variable type ID. Note: Can be a default 82 or user-defined 61 type 82.
ATTRIB	+36	U32	Attributes method.
MODIFIER	+40	U32	OB method modifier offsets (related to compressed st 81 rings, or strings 81 if NCF1).
	+44	U8[16]	(reserved)

(#1) The value of the CALLID field is derived from the value of the METHOD.ID field, in the source .OBB file.

Parameter table for OB method

The parameter table per OB method (OBMETHODPS) consists of a number of contiguous definitions (OBMETHODP), each of which contains all the information about a specific parameter related to the OB method to which it belongs.

The location (relative to the file) and the number of elements in the table are specified by the OBMETHOD 79.PARAMSOFF and OBMETHOD 79.PARAMSCNT fields, respectively.

Parameter definition for OB method

A parameter for OB method (OBMETHODP) consists of the following information:

Name	Offset	Type	Description
TYPEID	+0	U32	Variable type ID Note: Can be a predefined type 82 or user-defined 61.
ATTRIB	+4	U16	Parameter attributes: 0x0001 Parameter passed by reference (ABYREF)

© 2024 Robox SpA

Name	Offset	Type	Description
			0x0002 Parameter passed by pointer (ABYPTR)
FLAGS	+6	U16	Parameter settings: 0x0001 Uncompressed TEXT field (NCF0).
NAME	+8	U32	Offset parameter name (relative to strings 81).
TEXT	+12	U32	Offset parameter description (relative to compressed 81 strings, or strings 81 if NCF0).
DIM0	+16	U32	1st array size (0=unspecified)(#1).
DIM1	+20	U32	2nd array size (0=unspecified)(#1).
DIM2	+24	U32	3rd array size (0=unspecified)(#1).
	+28	B[16]	(reserved)

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum size should still be considered only 16bit (i.e., maximum index 0xFFFF).

Table of strings

The string table (STRINGS) consists of a number of consecutive ascii strings, each terminated the ascii character \0 (zero) and of variable size. The table must contain at least one character and must be \0, so that strings with offset 0 actually point to a null string.

The location (relative to the file) and size of the table are specified in the following fields, respectively, depending on where used:

- by the HEADER 58.STRINGSOFF and HEADER 58.STRINGSSIZE fields, if used in ladder/diagram files.
- by the fields LIVEHEADER 130.STRINGSOFF and LIVEHEADER 130.STRINGSSIZE, if used in live edits.

Table of compressed strings

The compressed string table (CSTRINGS) consists of a number of consecutive ascii strings, each terminated the ascii character \0 (zero) and of variable size.

The location (relative to the file) and size of the table are specified by the HEADER 58 .STRINGSOFF and HEADER 58.STRINGSSIZE fields, respectively: the compression state, which is optional, is instead specified by the HEADER 58.CSTRINGSCTYPE field.

Annotations:

- in case of HEADER 58.CSTRINGSCTYPE compression = 1, use the standard library functions (QT >= 3.3.4) qUncompress() to decompress the table properly.

© 2024 Robox SpA

- offsets referring to this table are always understood to refer to the uncompressed table.

Table of live changes

The embedded live changes table (EMBLIVES) consists of a number of contiguous (EMBLIVE) definitions, each of which contains all the information about a specific set of live changes to the ladder file itself: each set is actually an independent live changes 122 file but embedded in the ladder file itself.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 58 .EMBLIVESOFF and HEADER 58 .EMBLIVESCNT fields, respectively.

When loading, the various live changes files are applied in the exact order in which they appear. The actual application of the changes is governed by the appropriate flag in EMBLIVE 82 .FLAGS (value 0x0001): only the last live changes file may possibly have this flag set to false, indicating live changes not yet applied.

Definition of embedded live changes

An embedded live changes (EMBLIVE) consists of the following information:

Name	Offset	Type	Description
FILEOFF	+0	U32	Offset of the live edit 129 file (offset relative to the ladder file).
FILESIZE	+4	U32	Size of the live modification 129 file.
FLAGS	+8	U16	File settings: 0x0001 = Actual changes to be applied.
	+10	U8[22]	(reserved)

Predefined data types

Data types are identified by a unique ID code, whether they are predefined or user-defined: IDs are divided as follows:

Range	Description
0x00000000-0x0000FFFF	Default data types (#1)
0x00010000-0xFFFFFFFE	User-defined data types
0xFFFFFF	Reserved

(#1) default data types are never saved in the .LAD file, but are always considered to exist.

The default data types are as follows:

ID	Symbolic	Description	RTE (0x00)
0x00000000	(reserved)	Invalid data type	
0x00000001	I8	Signed 8bit integer	1

© 2024 Robox SpA

ID	Symbolic	Description	RTE (0x00)
0x00000002	U8	Unsigned integer 8bit	1
0x00000003	I16	Integer 16bit with sign	2
0x00000004	U16	Unsigned 16bit integer	2
0x00000005	I32	Integer 32bit with sign	4
0x00000006	U32	Unsigned 32bit integer	4
0x00000007	I64	Integer 64bit with sign	
0x00000008	U64	Unsigned integer 64bit	
0x00000009	STRING	String (0 terminated)	...
0x00000A	FLOAT	Floating point 32bit	4
0x00000B	REAL	Floading point 64bit (double)	8
0x00000C	BOOL	Boolean	4 (#1)
0x00000D	TIMER	Default timer structure 83	32
0x00000E	COUNTER	Default counter structure 84	48
0x00000F	ALIAS	Alias (or literally) 83	0
0x00000010	EDGE	Special storage for rising edge (RE) and falling edge (FE) elements.	4
0x00000011	POWERSET	"Power Set" logic variable	4 (#2)
0x00000012	AXESGROUP	"Axes group" logic variable	4 (#2)
0x00000013	PATH	"Path" logic variable	4 (#2)
0x00000014	POINT_L	"Library point" logical variable	4 (#2)

(#1) BOOL type in RTE is represented, for performance reasons, as I32, while by RDE3 it is seen as U8.

(#2) Logical variables in RTE are always represented as a pointer to memory (4bytes), even if they were embedded in user-defined structures. For further information on the composition (elements) of such variables, see the RTE technical documentation).

The representation and size of data types depend directly on the runtime destination ID(HEADER 58.TARGETID). The sizes of the data types are listed in the RTE columns and are expressed in bytes.

Default timer structure

The default TIMER structure consists of the following fields:

© 2024 Robox SpA

Type	Name	Description	Read	Write
BOOL	Q	Timer Outcome	X	
REAL	ET	Current value [ms]	X	X (#1)
REAL	PT	Preset value [ms]	X	X
REAL	PT2	Preset value 2 [ms]	X	X
BOOL	EN	Enable timer	X	

(#1) The ET field can be written only when the timer has not yet been assigned.

COUNTER default structure

The default COUNTER structure consists of the following fields:

Type	Name	Description	Read	Write
BOOL	QU	Outcome of count against preset	X	
BOOL	QD	Count outcome with respect to 0	X	
I32	CV	Current count value	X	X (#1)
I32	PV	Preset value	X	X
BOOL	CU	Status of the "count up" signal	X	
BOOL	CD	State of the "count down" signal	X	

(#1) The CV field can be written only when the counter has not yet been assigned.

Alias (or literally)

The list of data types that can be used as aliases are as follows:

ID	Name	Type	Description
0x0001	INP(n)	BOOL	Input channel
0x0002	INP_W(n)	U16	Input word 16bit
0x0003	INP_W(n).b	BOOL	Input word 16bit, bit access
0x0004	INP_DW(n)	U32	Input word 32bit
0x0005	INP_DW(n).b	BOOL	Input word 32bit, bit access
0x0006	OUT	BOOL	Output channel

© 2024 Robox SpA

ID	Name	Type	Description
0x0007	OUT_W(n)	U16	Output word 16bit
0x0008	OUT_W(n).b	BOOL	Output word 16bit, bit access
0x0009	OUT_DW(n)	U32	Output word 32bit
0x000A	OUT_DW(n).b	BOOL	Output word 32bit, bit access
0x000B	R(n)	I32	Global integer register (volatile)
0x000C	R(n).b	BOOL	Global integer (volatile) register, bit access
0x000D	NVR(n)	I32	Global integer register (nonvolatile)
0x000E	NVR(n).b	BOOL	Global integer (nonvolatile) register, bit access
0x000F	RR(n)	REAL	Global real (volatile) register
0x0010	NVRR(n)	REAL	Real global register (non-volatile)
0x0011	AM(n)	U32	Alarm mask
0x0012	AM(n).b	U32	Alarm mask, bit access

Annotations:

- if the variable is of array type, the array index (n) can be either a constant or an unindexed variable.
- if the variable involves bit access, .b can be either a constant or a variable and must be in the compatible range the size of the reference variable.

Predefined variables

Variables are identified by a unique ID code, whether they are predefined or user-defined: IDs are divided as follows:

Range	Description
0x00000000-0x0000FFFF	Default variable
0x0001000-0xFFFFFFFE	User-defined variable
0xFFFFFF	Reserved

CAUTION: Default variables are never saved in the .LAD file, but are always considered to exist.

© 2024 Robox SpA

The default variables are as follows:

ID	Name	Type	Description
0x00000001	INP(n)	BOOL	Input channel
0x00000002	INP_W(n)	U16	Input word 16bit
0x00000003	INP_DW(n)	U32	Input word 32bit
0x00000004	OUT	BOOL	Output channel
0x00000005	OUT_W(n)	U16	Output word 16bit
0x00000006	OUT_DW(n)	U32	Output word 32bit
0x00000007	R(n)	I32	Global integer register (volatile)
0x00000008	NVR(n)	I32	Global integer register (non-volatile)
0x00000009	RR(n)	REAL	Real global register (volatile)
0x00000A	NVRR(n)	REAL	Real global register (non-volatile)
0x00000B	AM(n)	U32	Alarm mask

Annotations:

- if the variable is of array type, the array index (n) can be either a constant or an unindexed variable.
- RDE3 creates additional predefined variables with completely arbitrary assignment of IDs (in the range 0x00000100-0x00010000): RTE recognizes such variables by name and not by that ID.

Source indentifiers

Source identifiers are used to uniquely identify the source of a certain entity (or element) within the ladder file, such as structures, variables, or OB classes. The value of the U32 identifier is composed as follows:

- U16 high, group identifier;
- U16 low, element identifier in the group.

The expected source identifiers are as follows:

Group	Element	Description
0x0000	0x0000	User defined entity manually
0x0001	0x0000	Entity imported from external file: generic file
0x0002	0x0000	Entity imported (forced) from RTE project: generic source

© 2024 Robox SpA

Group	Element	Description
0x0002	0x0001	Imported entity (forced) from RTE project: configurator source

File format (version 1.5)

File format (version 1.5)

A ladder/diagram programming language program is a single file (extension .LAD) containing both editing (source) and execution (runtime) information.

A ladder program consists primarily of a header 87, then several sections of which the main ones are:

- table of structures 90, which are the user-defined data types.
- table of local variables 92, i.e., the general variables of the program.
- table of OB classes 107, i.e., the object block classes referenced by the program.
- table of blocks 94, i.e., the sub-routines in other programming languages;
- table of live changes 112, i.e., any embedded live changes file;
- table of included files 110, i.e., any external files whose contents are included in the program;
- table of strings 111;
- table of compressed strings table 111.

In the block list, there is always a main block called MAIN and possibly additional blocks (comparable to sub-routines in other programming languages): each block contains a list of circuits 95 that constitute the actual ladder program: each circuit in turn is a logically structured list of elements 96.

NOTE: All numeric data stored in this file are to be considered in little indian (x86 standard) format.

File header

At the beginning of the file (offset 0) is a general header (HEADER) that contains access information for all remaining components and sub-sections of the file.

Name	Offset	Type	Description
SIGN	+0	U32	File signature ('LADF').
VERSION	+4	U32	Version of the file (1.5.0).
GENERATOR	+8	U32	Name and version of the file generator (offset relative to string table).
TYPE	+12	U16	Execution type: 0 = Synchronous execution (SY).

© 2024 Robox SpA

Name	Offset	Type	Description
			1 = High priority execution (HP). 2 = Normal priority execution (NP). 3 = Low priority execution (LP).
FREQUENCY	+14	U16	Execution frequency in [hz], from 1hz to 3000hz, default 200hz .
FLAGS	+16	U32	General settings: 0x000001 = The file contains formal verification errors. 0x000002 = The file contains references to external .OBB files (OB classes). 0x000004 = The file uses UTF-8 encoding.
BUILDYEAR	+20	U16	Year last generation (4 digits).
BUILDMONTH	+22	U8	Month last generation (1-12).
BUILDDAY	+23	U8	Day last generation (1-31).
BUILDHOUR	+24	U8	Last hour generation (0-23).
BUILDMINUTE	+25	U8	Last minute compilation (0-59).
BUILDSECOND	+26	U8	Second last compilation (0-59).
TARGETID	+27	U8	Runtime destination ID: 0x00 = Firmware RTE (PowerPC G2) 0x01 = RTE firmware (PowerPC P2020) 0x02 = RTE firmware (ARM Cortex A9)
VID	+28	U32	File version ID.
PLVID	+32	U32	Version ID of the file pre live edits (if present, otherwise worth 0).

© 2024 Robox SpA

Name	Offset	Type	Description
IARECNT	+36	U32	No. of ARE type elements present in the file.
IAFECNT	+40	U32	No. of AFE type elements present in the file.
CCFLAGS	+44	U32	Compiler settings: 0x000001 = Ignore inclusion file warning messages: import variables 0x000002 = Ignore inclusion file warning messages: import OB classes.
	+48	U8[80]	(reserved)
BLOCKSOFF	+128	U32	Offset of the block table 94 (relative to the file).
BLOCKSCNT	+132	U32	No. of elements of the block table 94.
STRINGSOFF	+136	U32	Offset of the string table 111 (relative to the file).
STRINGSSIZE	+140	U32	Size of the string table 111.
CSTRINGSOFF	+144	U32	Offset of the compressed string table 111 (relative to the file).
CSTRINGSSIZE	+148	U32	Size of the compressed string table 111.
CSTRINGSCTYPE	+152	U8	Type of compression used for ta compressed string table 111:. 0 = No compression. 1 = Standard compression Qt 3.3.x.
	+153	U8[3]	(reserved)
EMBLIVESOFF	+156	U32	Offset of the embedded live change table 112 (relative to the file) .
EMBLIVESCNT	+160	U32	No. of elements of the embedded live changes

© 2024 Robox SpA

Name	Offset	Type	Description
			table 112.
STRUCTSOFF	+164	U32	Offset of the structures table 90 (relative to the file).
STRUCTSCNT	+168	U32	No. of elements of the structures table 90.
VARSOFF	+172	U32	Offset of variable table 92 (relative to file).
VARSCNT	+176	U32	No. of elements of the variable table 92.
OBCLASSESOFF	+180	U32	Offset of the table classes OB 107 (relative to the file).
OBCLASSESCNT	+184	U32	No. of elements of the OB classes table 107 (relative to the file)
LASTBLOCKID	+188	U32	Last block ID used.
LASTRUNGID	+192	U32	Last circuit ID used.
LASTSTRUCTID	+196	U32	Last structure ID used.
LASTVARID	+200	U32	Last variable ID used.
LASTOBCLASSID	+204	U32	Last OB class ID used
INCFILESOFF	+208	U32	Offset of the inclusion file table 110 (relative to the file)
INCFILESCNT	+212	U32	No. of elements of the inclusion file table 110.
LASTINCFILEID	+216	U32	Last used inclusion file ID
	+220	U8[292]	(reserved)

Table of structures

The structures table (STRUCTS) consists of a number of contiguous definitions (STRUCTs), each of which contains all the information about a specific user-defined structure: there are no default data types 112 in this table.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 87.STRUCTSOFF and HEADER 87.STRUCTSCNT fields, respectively.

Structure definition

A structure (STRUCT) consists of the following information:

© 2024 Robox SpA

Name	Offset	Type	Description
ID	+0	U32	Structure type ID Note: only values 0x00010000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved, runtime use)
FLAGS	+6	U16	Structure settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 Uncompressed MODIFIER field (NCF1).
NAME	+8	U32	Offset structure name (relative to strings 111).
TEXT	+12	U32	Offset structure description (relative to compressed 111 strings, or strings 111 if NCF0).
VARSOFF	+16	U32	Variable table 92 offset (relative to file) (#1)
VARSCNT	+20	U32	Variable table 92 size
SIZE	+24	U32	Structure size [bytes]
SRCID	+28	U32	Source identifier 116 of the structure
OBCID	+32	U32	Reference OB class ID (0=none)(#2,#3)
ATTRIB	+36	U32	Structure attributes: 0x000001 Structure referred to OB class (AREFOBC)(#2) 0x000002 Main structure OB class (AISOBC)(#2,#3) 0x000004 Structure (ref.a) externally defined (AEXTERNAL)
MODIFIER	+40	U32	Offset modifiers of structure (relative to compressed 111 strings, or strings 111 if NCF1).
REFID	+44	U32	Structure type reference ID (#4).

© 2024 Robox SpA

Name	Offset	Type	Description
EXTNAME	+48	U32	Offset external structure name (relative to strings 111).
	+52	U8[16]	(reserved)

(#1) The structure definition uses a normal variable definition table to define its elements. In this case, variable IDs are meaningless and should always be 0.

(#2) At equal non-0 OBCID value with AREFOBC attribute, there can be only one structure with AISOBC flag, while there can be 0 or more structures without AISOBC attribute.

(#3) A structure is called an "orphan" structure if it is in the condition of AREFOBC attribute set and OBCID field = 0: this condition happens when updating/removing an OB class, there are still variables (or other) referencing it.

(#4) In case of AREFOBC attribute the REFID field contains the ID of the structure as defined of the source OBB file (0 in case of AISOBC attribute).

Table of variables

The variable table (VARS) consists of a number of contiguous definitions (VARs), each of which contains all the information about a specific user-defined variable: there are no default variables 115 in this table.

The location (relative to the file) and number of elements in the program variable table are specified in the following fields, respectively, depending on where used

- by the HEADER 87.VARSOFF and HEADER 87.VARSCNT fields if used in the file header;
- by the fields STRUCT 90.VARSOFF and STRUCT 90.VARSCNT if used in structure definition;
- by the fields LIVECMD 131.VARSOFF and LIVECMD 131.VARSCNT if used in command definition.

Variable definition

A variable (VAR) consists of the following information:

Name	*Offset*	*Type*	*Description*
ID	+0	U32	Variable ID (#2) Note: only values 0x00010000-0xFFFFFE; value 0xFFFFFF is a special meaning ID; values 0x000001-0x0000FFFF are reserved for default variables.
TYPEID	+4	U32	Variable type ID Note: Can be a default 112 or user-defined 90 type 112.
	+8	U8[2]	(reserved, runtime use)

© 2024 Robox SpA

Name	Offset	Type	Description
FLAGS	+10	U16	Variable settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 BITTEXT field uncompressed (NCF1) 0x0004 Uncompressed MODIFIER field (NCF2)
NAME	+12	U32	Offset variable name (relative to strings 111).
TEXT	+16	U32	Offset variable description (relative to compressed 111 strings, or strings 111 if NCF0).
BITTEXT	+20	U32	Offset variable bit descriptions, i.e., sequence of strings related to each individual bit separated by the ascii character 0xFF (related to compressed 111 strings, or strings 111 if NCF0). (#2)
DIM0	+24	U32	1st array size (0=unspecified) (#1) or index 0 for Alias 113 (TYPEID = 0x00000F).
DIM1	+28	U32	2nd array size (0=unspecified) (#1) or index 1 for Alias 113 (TYPEID = 0x00000F).
DIM2	+32	U32	3rd array size (0=unspecified) (#1) or index 2 for Alias 113 (TYPEID = 0x00000F).
ALIASID	+36	U16	Alias type 114 ID (#2)
ALIASBIT	+38	U16	No. of bits per alias (depends on the type of alias ID)(#2)
SIZE	+40	U32	Size of variable. (#3)
SRCID	+44	U32	Source identifier 116 of the variable.(#2,#4)
VALUE	+48	U32	Offset initial value of variable (relative to strings 111, 0=no value).

© 2024 Robox SpA

Name	Offset	Type	Description
ATTRIB	+52	U32	Variable attributes: 0x000001 Variable can be written (WRITE) 0x000002 The variable can be read (READ) 0x000004 The variable can be forced (FORCE) 0x000008 The variable can be released (RELEASE) 0x000010 The variable is constant (CONST) 0x000020 The variable has no bit access (NOBITS)
MODIFIER	+56	U32	Variable modifier offsets (related to compressed st111rings, or strings111 if NCF2).
DIM0L	+60	U32	Offset label for 1st array size (relative to strings111). (#5)
DIM1L	+64	U32	Offset label for 2nd array size (relative to strings111). (#5)
DIM2L	+68	U32	Offset label for 3rd array dimension (relative to strings111). (#5)
	+72	U8[16]	(reserved)

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum size should still be considered only 16bit (i.e., maximum index 0xFFFF).

(#2) Field not used (value 0) if used in structure definition.

(#3) Field not used (value 0) if used by defining an Alias (TYPEID = 0x00000F)

(#4) Field not used (value 0) if used in live command definition.

(#5) Label means the name of a variable (constant value) or the ascii representation of a numeric value, i.e., an offset to the string table111.

Table of blocks

The block table (BLOCKS) consists of a number of contiguous block definitions (BLOCKS), each of which contains all the information about a specific block and its contents on file.

The location (relative to the file) and the number of elements in the table are specified by the HEADER87.BLOCKSOFF and HEADER87.BLOCKSCNT fields, respectively.

© 2024 Robox SpA

Block definition

A block (BLOCK) consists of the following data:

Name	Offset	Type	Description
ID	+0	U32	Unique ID of the block. Note: the values 0 and 0xFFFFFF are special meaning IDs.
STATUS	+4	U16	(reserved, runtime use)
FLAGS	+6	U16	Block settings: 0x0001 = Main block (ISMAIN). 0x0002 = Uncompressed TEXT field (NCF0). 0x0004 = Uncompressed COMMENT field (NCF1).
NAME	+8	U32	Offset block name (relative to strings 111). Note: If ISMAIN flag specified, name comes must be "main".
TEXT	+12	U32	Offset block description (relative to compressed 111 strings, or strings 111 if NCF0).
COMMENT	+16	U32	Offset comments for the block (relative to compressed 111 strings, or strings 111 if NCF1).
RUNGSOFF	+20	U32	Offset of the circuit table of the block 95 (relative to rows).
RUNGSCNT	+24	U32	No. of elements of the block circuit table 95.
	+28	B[36]	(reserved)

Table of block's circuits

The block circuit table (RUNGS) consists of a number of contiguous circuit definitions (RUNG), each of which contains all the information about a specific circuit and its contents.

The location (relative to the file) and the number of elements in the table are specified by the BLOCK 95.RUNGSOFF and BLOCK 95.RUNGSCNT fields, respectively.

© 2024 Robox SpA

Block circuit definition

A block circuit (RUNG) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	Unique ID of the circuit. Note: the values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved, runtime use)
FLAGS	+6	U16	Circuit settings: 0x0001 Uncompressed TEXT field (NCF0).
REFID	+8	U32	Reference circuit ID (0=none).
ITEMSOFF	+12	U32	Offset of the circuit element table 96 (relative to the file).
ITEMSSIZE	+16	U32	Size of the circuit element table 96.
TEXT	+20	U32	Offset circuit description (related to compressed st 111 rings, or strings 111 if NCF1).
	+24	U8[4]	(reserved)

Table of circuit's items

The circuit element table (ITEMS) consists of a number of contiguous circuit definitions (ITEMs), each of which contains all the information about a specific element.

The location (relative to the file) and size of the table are specified in the following fields, respectively, depending on where used:

- by the RUNG 96.ITEMSOFF and RUNG 96.ITEMSSIZE fields, if used in block circuit (RUNG).
- by the fields LIVECMD 131.ITEMSOFF and LIVECMD 131.ITEMSSIZE, if used in live change commands (LIVECMD).

Circuit element definition

Circuit elements (ITEMs) have a common initial part, then a specific data area for each type of element. The common part is as follows:

Name	Offset	Type	Description
CODE	+0	U16	Element type code.
SIZE	+2	U16	Full size of element (all fields).

© 2024 Robox SpA

Name	Offset	Type	Description
DATE	+4	...	Specific data (optional).

The expected elements are as follows:

Symbolic	ID	Type	Data	Description
BB	0x0001			Start of branch.
NL	0x0002			Next branch level . Notes: must always be between a BB and an EB.
EB	0x0003			End of branch . Notes: must always correspond to a BB.
OC	0x0004		U32=Destination (#3): BOOL 113 types only.	Open contact.
CC	0x0005		U32=Destination (#3): BOOL 113 types only.	Closed contact.
RE	0x0006		U32=Destination (#3): EDGE 113 types only.	Rising edge.
FE	0x0007		U32=Destination (#3): EDGE 113 types only.	Falling edge.
AF	0x0008			Always false.
OUT	0x0009	F	U32=Destination (#3): BOOL 113 types only.	Output.
OL	0x000A	F	U32=Destination (#3): BOOL 113 types only.	Output latch (set).
OUL	0x000B	F	U32=Destination (#3): BOOL 113 types only.	Output unlatch (reset).
NOUT	0x000C	F	U32=Destination (#3): BOOL 113 types only	Output denied
CLR	0x000D	F	U32=Destination (#3): TIMER 113 and	Zeroing (clear).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			COUNTER 113 types only.	
TP	0x000E	F	U32=Destination (#3): TIMER 113 types only U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer pulse.
TON	0x000F	F	U32=Preset value (#3): TIMER 113 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer on delay.
TOFF	0x0010	F	U32=Preset value (#3): TIMER 113 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable timer off delay.
TSQ	0x0011	F	U32=Preset value (#3): TIMER 113 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable square wave timer.
TINT	0x0012	F	U32=Preset value (#3): TIMER 113 types only. U32=Preset value 1 (#4) U32=Preset value 2 (#4).	Enable integral timer.
CU	0x0013	F	U32=Preset value (#3): COUNTER 113 types only U32=Preset value (#4).	Count up.
CD	0x0014	F	U32=Preset value (#3): COUNTER 113 types only	Count down.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			U32=Preset value (#4).	
CUR	0x0015	F	U32=Preset value (#3): COUNTER 113 types only. U32=Preset value (#4).	Rotary count up.
CDR	0x0016	F	U32=Preset value (#3): COUNTER 113 types only. U32=Preset value (#4).	Count down rotary.
LD	0x0017	F	U32=Preset value (#3): COUNTER 113 types only.	Preset on counter.
MATH	0x0018	F	U32=Expression (#5) U32=Destination (#3) U8[]=Mathematical expression 104 data.	Math block.
EQ	0x0019		U32=Source A (#2) U32=Source B (#2).	Equality comparison block (A == B).
NE	0x001A		U32=Source A (#2) U32=Source B (#2)	Different comparison block (A != B).
GT	0x001B		U32=Source A (#2) U32=Source B (#2)	Major comparison block (A > B).
GE	0x001C		U32=Source A (#2) U32=Source B (#2)	Major/equal comparison block (A >= B).
LT	0x001D		U32=Source A (#2) U32=Source B (#2)	Minor comparison block (A < B).
LE	0x001E		U32=Source A (#2) U32=Source B (#2)	Minor/equal comparison block (A <= B).
COPY	0x001F	F	U32=Source (#2) U32=Destination (#3)	Copies data from source to destination, for the specified no. of elements.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			U32=N. Elements (#2)	
CALL	0x0020	F	U32 Block ID to call	Call of a block (sub-routine).
RET	0x0021	F		Return from a block to the inside.
ADD	0x0022	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Sum (A + B).
SUB	0x0023	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Subtraction (A - B).
MUL	0x0024	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Multiplication (A * B).
DIV	0x0025	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Division (A / B).
AND	0x0026	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	AND binary (A & B).
OR	0x0027	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Binary OR (A | B).
NOT	0x0028	F	U32=Source A (#2) U32=Destination (#3)	NOT binary (~A).
XOR	0x0029	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	XOR binary (A ^ B).
SQRT	0x002A	F	U32=Source A (#2)	Square root of A.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			U32=Destination (#3)	
MOD	0x002B	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Module (A mod B).
POW	0x002C	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Power elevation (A pow B).
SIN	0x002D	F	U32=Source A (#2) U32=Destination (#3)	Sine of A.
COS	0x002E	F	U32=Source A (#2) U32=Destination (#3)	Cosine of A.
TAN	0x002F	F	U32=Source A (#2) U32=Destination (#3)	Tangent of A.
ASIN	0x0030	F	U32=Source A (#2) U32=Destination (#3)	Archosene of A.
ACOS	0x0031	F	U32=Source A (#2) U32=Destination (#3)	Arcocosene of A.
ATAN	0x0032	F	U32=Source A (#2) U32=Destination (#3)	Arcotangent of A.
ATAN2	0x0033	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Arcotangent of A, B.
LOG	0x0034	F	U32=Source A (#2) U32=Destination (#3)	Base 10 logarithm of A.
LN	0x0035	F	U32=Source A (#2) U32=Destination (#3)	Base E logarithm of A.

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
ABS	0x0036	F	U32=Source A (#2) U32=Destination (#3)	Absolute value of A
SIGN	0x0037	F	U32=Source A (#2) U32=Destination (#3)	A sign.
ROUND	0x0038	F	U32=Source A (#2) U32=Destination (#3)	Rounding to the nearest integer to A.
MIN	0x0039	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Minimum between A and B.
MAX	0x003A	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Maximum between A and B.
RANGE	0x003B	F	U32=Source A (#2) U32=Source B (#2) U32=Source C (#2) U32=Destination (#3): BOOL 113 types only.	Checks whether A contained in the range B-C.
LIMIT	0x003C	F	U32=Source A (#2) U32=Source B (#2) U32=Source C (#2) U32=Destination (#3)	Limit A to the B-C range.
ROR	0x003D	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Right bit rotation (A >> B).
ROL	0x003E	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Left bit rotation (A << B).
SHR	0x003F	F	U32=Source A (#2) U32=Source B (#2)	Shift bit to the right (A >> B).

© 2024 Robox SpA

Symbolic	ID	Type	Data	Description
			U32=Destination (#3)	
SHL	0x0040	F	U32=Source A (#2) U32=Source B (#2) U32=Destination (#3)	Shift bit to the left (A << B).
MOV	0x0041	F	U32=Source A (#2) U32=Destination (#3)	Assign A.
TODEG	0x0042	F	U32=Source A (#2) U32=Destination (#3)	Conversion of A from RAD to DEG.
TORAD	0x0043	F	U32=Source A (#2) U32=Destination (#3)	Conversion of A from DEG to RAD.
NEG	0x0044			Negation of logical state.
NOP	0x0045	F		No operation.
EVAL	0x0046		U32=Expression (#5) U8[]=Mathematical expression 104 data.	Evaluation expression (boolean).
OBCALL	0x0047	F	U32=Instance name (#6) U32=Complete method name (#9) U32=List of parameter names (#7) U32=List of parameter values (#7) U32=Result (#3) U8[]=Mathematical expression 104 data (#8)	Call to method of an OB instance.
FT	0x0048			Initial execution
ARE	0x0049			Rising edge (anonymous)
AFE	0x004A			Falling edge (anonymous)

© 2024 Robox SpA

(#1) Local bit variable (type I32 or U32) with direct addressing (any constant indexes and number of bits).

(#2) Source/destination means variable name or ascii representation of a numeric value (source only), or an offset to the string table.

(#3) By destination means name of a variable with offset to the string table.

(#4) Value (or preset value) means a numeric value in string format: empty string (or offset 0) means no value assigned.

(#5) Expression means string with offset to string table.

(#6) Instance name means name of a variable with offset to the string table, whose TYPEID refers to a user structure with FISOBC flag.

(#7) List means a string (with offset to the string table), consisting of a number of sub-strings separated with ascii character 0xFF: the two lists (names/values) MUST always have the same no. of elements or be both empty (=0).

(#8) The call to a method of an OB instantiation is actually translated as a mathematical expression, in which any parameters (values in natural order) are prepared on the stack, the call to the method is made, and then any result is found from the stack.

(#9) Full name of the method (unique), understood as a string with offset to the string table.

Mathematical expression data

Mathematical expression data (used by MATH, COND, and OBCALL elements), is a series of contiguous operations (ordered in inverse polity) with the following common header:

Name	Offset	Type	Description
CODE	+0	U8	Operation code.
DATE	+1	...	Specific data (optional).

The expected operations are as follows:

Symbolic	ID	Data	In (stack)	Out (stack)	Description
PUSH	0x01	DBL=value	--	DBL	Push value into stack.
PUSHV	0x02	U32=Source (#1)	--	DBL	Push variable value into stack.
ADD	0x03		DBL0,DBL1	DBL	Sum.
SUB	0x04		DBL0,DBL1	DBL	Subtraction (#2).
MUL	0x05		DBL0,DBL1	DBL	Multiplication (#2).
DIV	0x06		DBL0,DBL1	DBL	Division (#2).
MOD	0x07		DBL	DBL	Module (#3).
ABS	0x08		DBL	DBL	Absolute value (#3).
SIN	0x09		DBL	DBL	Sine.

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
COS	0X0A		DBL	DBL	Cosine.
TAN	0x0B		DBL	DBL	Tangent.
ASIN	0x0C		DBL	DBL	Arcosene.
ACOS	0x0D		DBL	DBL	Arcocosene.
ATAN	0x0E		DBL	DBL	Arcotangent.
POW	0x0F		DBL	DBL	Exponent.
LOG10	0x10		DBL	DBL	Logarithm base 10.
LOG	0x11		DBL	DBL	Natural logarithm.
R_AND	0x12		U32	U32	Binary AND.
R_OR	0x13		U32	U32	OR binary.
R_NOT	0x14		U32	U32	NOT binary.
R_XOR	0x15		U32	U32	XOR binary.
POPV	0x16	U32=Destination (#1)	DBL	--	Pop value from stack to variable.
R_ROR	0x17		U32	U32	Right-hand bit rotation.
R_ROL	0x18		U32	U32	Left-hand bit rotation.
R_SHR	0x19		U32	U32	Shift bit to the right.
R_SHL	0x1A		U32	U32	Shift bit to the left.
TOBIN	0x1B		U32	U32	BCD to binary conversion.
TOBCD	0x1C		U32	U32	Binary to BCD conversion.
TODEG	0x1D		DBL	DBL	Conversion from radians to degrees.
TORAD	0x1E		DBL	DBL	Conversion from degrees to radians.
ATAN2	0x1F		DBL0,DBL1	DBL	Arcotangent with two parameters.

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
PUSHVB	0x20	U32=Source (#1)	DBL	--	Push variable bit value into stack.
POPVB	0x21	U32=Destination (#1)	DBL	--	Pop value from stack to variable bit.
POPL	0x22		BOOL	--	Pop logical value (0=false, otherwise true).
	0x23				(reserved)
EQ	0x24		DBL0,DBL1	BOOL	Comparison: equal.
NE	0x25		DBL0,DBL1	BOOL	Comparison: different.
LT	0x26		DBL0,DBL1	BOOL	Comparison: minor.
LE	0x27		DBL0,DBL1	BOOL	Comparison: less than or equal to.
GT	0x28		DBL0,DBL1	BOOL	Comparison: greater than.
GE	0x29		DBL0,DBL1	BOOL	Comparison: greater than or equal to.
AND	0x2A		BOOL,BOOL	BOOL	AND logical.
OR	0x2B		BOOL,BOOL	BOOL	OR logical.
NOT	0x2C		BOOL	BOOL	NOT logical.
XOR	0x2D		BOOL,BOOL	BOOL	logical XOR.
CHS	0x2E		DBL	DBL	Sign change.
SHR	0x2F		U32	U32	Logical shift to the right.
SHL	0x30		U32	U32	Logical shift to the left
OBCALL	0x31	U32=Method (#2)	?	?	Direct method call to OB instantiation
POP	0x32		*	--	Pop value (skip)

© 2024 Robox SpA

Symbolic	ID	Data	In (stack)	Out (stack)	Description
PUSHP	0x33	U32=Source (#1)	U32	U32	Push pointer to variable

(#1) Source/destination means name of a variable, i.e., an offset to the string table 111.

(#2) By method means string, i.e., an offset to the string table 111, with format "<instance_name>.#<hexid_method>" or "<instance_name>[<index>].#<hexid_method>" (hexid = 32bit hexadecimal ascii number, without prefix 0x. Example method no. 0xFA9 of OB instance 'myob' : "myob#00000FA9".

Table of OB classes

The OB classes table (OBCLASSES) consists of a number of contiguous definitions (OBCLASS), each of which contains all the information about a specific user-defined object block class.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 87.OBCLASSOFF and HEADER 87.OBCLASSCNT fields, respectively.

OB class definition

An OB class (OBCLASS) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	OB class ID Note: the values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved)
FLAGS	+6	U16	OB class settings: 0x0001 Uncompressed TEXT field (NCF0). 0x0002 Uncompressed MODIFIER field (NCF1).
NAME	+8	U32	Offset OB class name (relative to strings 111).
TEXT	+12	U32	Offset description of OB class (relative to compressed 111 strings, or strings 111 if NCF0).
VERSION	+16	U32	OB class version (nvMake format).
INTFID	+20	U32	Unique obect block interface identifier.
PATHID	+24	U32	Path identifier of the .OBB file: 0=absolute path

© 2024 Robox SpA

Name	Offset	Type	Description
			1=(reserved) 2=Path relative to workspace 3=Path related to factory RDE3
FILENAME	+28	U32	Offset file name .OBB (relative to strings 111) (#1).
SRCID	+32	U32	Source identifier 116 of the OB class.
METHODSOFF	+36	U32	Offset of the OB methods table 108 (relative to the file).
METHODSCNT	+40	U32	OB methods table 108 size
ATTRIB	+44	U32	OB class attributes.
MODIFIER	+48	U32	Offset modifiers of class OB (related to compressed 111 strings, or strings 111 if NCF1).
	+52	U8[16]	(reserved)

(#1) In case of PATHID other than 0, the .OBB filename has no paths within it, but is simply the filename with relative to the PATHID path: in case of PATHID equal to 0 instead, the .OBB filename might contain a relative/absolute path.

Table of OB methods

The OB methods table (OBMETHODS) consists of a number of contiguous (OBMETHOD) definitions, each of which contains all the information about a specific OB method related to the OB class to which it belongs.

The location (relative to the file) and the number of elements in the table are specified by the OBCLASS 107.METHODSOFF and OBCLASS 107.METHODSCNT fields, respectively.

OB method definition

An OB method (OBMETHOD) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	OB method ID. Note: the values 0 and 0xFFFFFF are special meaning IDs.
	+4	U8[2]	(reserved)
FLAGS	+6	U16	OB method settings:

© 2024 Robox SpA

Name	Offset	Type	Description
			0x0001 Uncompressed TEXT field (NCF0). 0x0002 Uncompressed MODIFIER field (NCF1).
NAME	+8	U32	Offset OB method name (related to strings 111).
KNAME	+12	U32	Offset key name of OB method (relative to strings 111).
TEXT	+16	U32	Offset description of OB method (relative to compressed 111 strings, or strings 111 if NCF0).
CALLID	+20	U32	Method ID in call (#1)
PARAMSOFF	+24	U32	Offset of parameter table for OB method 109 (relative to file).
PARAMSCNT	+28	U32	Size of parameter table by OB method 109.
RETTYPEID	+32	U32	Return variable type ID. Note: Can be a default 112 or user-defined 90 type 112.
ATTRIB	+36	U32	Attributes method.
MODIFIER	+40	U32	OB method modifier offsets (related to compressed st 111 rings, or strings 111 if NCF1).
	+44	U8[16]	(reserved)

(#1) The value of the CALLID field is derived from the value of the METHOD.ID field, in the source .OBB file.

Parameter table for OB method

The parameter table by OB method (OBMETHODPS) consists of a number of contiguous definitions (OBMETHODP), each of which contains all the information about a specific parameter related to the OB method to which it belongs.

The location (relative to the file) and the number of elements in the table are specified by the OBMETHOD 108.PARAMSOFF and OBMETHOD 108.PARAMSCNT fields, respectively.

Parameter definition for OB method

A parameter for OB method (OBMETHODP) consists of the following information:

© 2024 Robox SpA

Name	Offset	Type	Description
TYPEID	+0	U32	Variable type ID Note: Can be a predefined type 112 or user-defined 90.
ATTRIB	+4	U16	Parameter attributes: 0x0001 Parameter passed by reference (ABYREF) 0x0002 Parameter passed by pointer (ABYPTR)
FLAGS	+6	U16	Parameter settings: 0x0001 Uncompressed TEXT field (NCF0).
NAME	+8	U32	Offset parameter name (relative to strings 111).
TEXT	+12	U32	Offset parameter description (relative to compressed 111 strings, or strings 111 if NCF0).
DIM0	+16	U32	1st array size (0=unspecified)(#1).
DIM1	+20	U32	2nd array size (0=unspecified)(#1).
DIM2	+24	U32	3rd array size (0=unspecified)(#1).
	+28	B[16]	(reserved)

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum size should still be considered only 16bit (i.e., maximum index 0xFFFF).

Table of include files

The inclusion file table (INCFILES) consists of a number of contiguous definitions (INCFILE), each of which contains all the information about a specific user-defined inclusion file.

The location (relative to the file) and the number of elements in the inclusion file table are specified by the HEADER 87.INCFILESOFF and HEADER 87.INCFILESCNT fields, respectively.

Inclusion file definition

An inclusion file (INCFILE) consists of the following information:

Name	Offset	Type	Description
ID	+0	U32	File ID

© 2024 Robox SpA

Name	Offset	Type	Description
			Note: The values 0 and 0xFFFFFF are special meaning IDs.
FLAGS	+4	U32	File settings: (none)
PATHID	+8	U16	Source path ID: 0 = (reserved absolute path; not implemented) 1 = Path relative to container .LAD file.
	+10	U8[2]	(reserved)
FILENAME	+12	U32	Offset filename (relative to strings 111).
	+16	U8[16]	(reserved)

Table of strings

The string table (STRINGS) consists of a number of consecutive ascii strings, each terminated the ascii character \0 (zero) and of variable size. The table must contain at least one character and must be \0, so that strings with offset 0 actually point to a null string.

The location (relative to the file) and size of the table are specified in the following fields, respectively, depending on where used:

- by the HEADER 87.STRINGSOFF and HEADER 87.STRINGSSIZE fields, if used in ladder/diagram files.
- by the fields LIVEHEADER 130.STRINGSOFF and LIVEHEADER 130.STRINGSSIZE, if used in live edits.

Table of compressed strings

The compressed string table (CSTRINGS) consists of a number of consecutive ascii strings, each terminated the ascii character \0 (zero) and of variable size.

The location (relative to the file) and size of the table are specified by the HEADER 87 .STRINGSOFF and HEADER 87.STRINGSSIZE fields, respectively: the compression state, which is optional, is instead specified by the HEADER 87.CSTRINGSCTYPE field.

Annotations:

- in case of HEADER 87.CSTRINGSCTYPE compression = 1, use the standard library functions (QT >= 3.3.4) qUncompress() to decompress the table properly.
- offsets referring to this table are always understood to refer to the uncompressed table.

© 2024 Robox SpA

Table of live changes

The embedded live changes table (EMBLIVES) consists of a number of contiguous (EMBLIVE) definitions, each of which contains all the information about a specific set of live changes to the ladder file itself: each set is actually an independent live changes 122 file but embedded in the ladder file itself.

The location (relative to the file) and the number of elements in the table are specified by the HEADER 87.EMBLIVESOFF and HEADER 87.EMBLIVESCNT fields, respectively.

When loading, the various live change files are applied in the exact order in which they appear. The actual application of the changes is governed by the appropriate flag in EMBLIVE 112.FLAGS (value 0x0001): only the last live change file may possibly have this flag set to false, indicating live changes not yet applied.

Definition of embedded live changes

An embedded live changes (EMBLIVE) consists of the following information:

Name	Offset	Type	Description
FILEOFF	+0	U32	Offset of the live edit 137 file (offset relative to the ladder file).
FILESIZE	+4	U32	Size of the live modification 137 file.
FLAGS	+8	U16	File settings: 0x0001 = Actual changes to be applied.
	+10	U8[22]	(reserved)

Predefined data types

Data types are identified by a unique ID code, whether they are predefined or user-defined: IDs are divided as follows:

Range	Description
0x00000000-0x0000FFFF	Default data types (#1)
0x00010000-0xFFFFFFFE	User-defined data types
0xFFFFFF	Reserved

(#1) default data types are never saved in the .LAD file, but are always considered to exist.

The default data types are as follows:

ID	Symbolic	Description	RTE (0x00)
0x00000000	(reserved)	Invalid data type	
0x00000001	I8	Signed 8bit integer	1
0x00000002	U8	Unsigned integer 8bit	1
0x00000003	I16	Integer 16bit with sign	2

© 2024 Robox SpA

ID	Symbolic	Description	RTE (0x00)
0x00000004	U16	Unsigned 16bit integer	2
0x00000005	I32	Integer 32bit with sign	4
0x00000006	U32	Unsigned 32bit integer	4
0x00000007	I64	Integer 64bit with sign	
0x00000008	U64	Unsigned integer 64bit	
0x00000009	STRING	String (0 terminated)	...
0x00000A	FLOAT	Floating point 32bit	4
0x00000B	REAL	Floading point 64bit (double)	8
0x00000C	BOOL	Boolean	4 (#1)
0x00000D	TIMER	Default timer structure 113	32
0x00000E	COUNTER	Default counter structure 114	48
0x00000F	ALIAS	Alias (or literally) 113	0
0x00000010	EDGE	Special storage for rising edge (RE) and falling edge (FE) elements.	4
0x00000011	POWERSET	"Power Set" logic variable	4 (#2)
0x00000012	AXESGROUP	"Axes group" logic variable	4 (#2)
0x00000013	PATH	"Path" logic variable	4 (#2)
0x00000014	POINT_L	"Library point" logical variable	4 (#2)

(#1) BOOL type in RTE is represented, for performance reasons, as I32, while by RDE3 it is seen as U8.

(#2) Logical variables in RTE are always represented as a pointer to memory (4bytes), even if they were embedded in user-defined structures. For more information on the composition (elements) of such variables, see the RTE technical documentation).

The representation and size of data types depend directly on the runtime destination ID(HEADER 87.TARGETID). The sizes of the data types are listed in the RTE columns and are expressed in bytes.

Default timer structure

The default TIMER structure consists of the following fields:

Type	Name	Description	Read	Write
BOOL	Q	Timer Outcome	X	
REAL	ET	Current value [ms]	X	X (#1)
REAL	PT	Preset value [ms]	X	X
REAL	PT2	Preset value 2 [ms]	X	X
BOOL	EN	Enable timer	X	

(#1) The ET field can be written only when the timer has not yet been assigned.

COUNTER default structure

The default COUNTER structure consists of the following fields:

Type	Name	Description	Read	Write
BOOL	QU	Outcome of count against preset	X	
BOOL	QD	Count outcome with respect to 0	X	
I32	CV	Current count value	X	X (#1)
I32	PV	Preset value	X	X
BOOL	CU	Status of the "count up" signal	X	
BOOL	CD	State of the "count down" signal	X	

(#1) The CV field can be written only when the counter has not yet been assigned.

Alias (or literally)

The list of data types that can be used as aliases are as follows:

ID	Name	Type	Description
0x0001	INP(n)	BOOL	Input channel
0x0002	INP_W(n)	U16	Input word 16bit
0x0003	INP_W(n).b	BOOL	Input word 16bit, bit access
0x0004	INP_DW(n)	U32	Input word 32bit
0x0005	INP_DW(n).b	BOOL	Input word 32bit, bit access
0x0006	OUT	BOOL	Output channel

© 2024 Robox SpA

ID	Name	Type	Description
0x0007	OUT_W(n)	U16	Output word 16bit
0x0008	OUT_W(n).b	BOOL	Output word 16bit, bit access
0x0009	OUT_DW(n)	U32	Output word 32bit
0x000A	OUT_DW(n).b	BOOL	Output word 32bit, bit access
0x000B	R(n)	I32	Global integer register (volatile)
0x000C	R(n).b	BOOL	Global integer (volatile) register, bit access
0x000D	NVR(n)	I32	Global integer register (nonvolatile)
0x000E	NVR(n).b	BOOL	Global integer (nonvolatile) register, bit access
0x000F	RR(n)	REAL	Global real (volatile) register
0x0010	NVRR(n)	REAL	Real global register (non-volatile)
0x0011	AM(n)	U32	Alarm mask
0x0012	AM(n).b	U32	Alarm mask, bit access

Annotations:

- if the variable is of array type, the array index (n) can be either a constant or an unindexed variable.
- if the variable involves bit access, .b can be either a constant or a variable and must be in the compatible range the size of the reference variable.

Predefined variables

Variables are identified by a unique ID code, whether they are predefined or user-defined: IDs are divided as follows:

Range	Description
0x00000000-0x0000FFFF	Default variable
0x0001000-0xFFFFFFFE	User-defined variable
0xFFFFFF	Reserved

CAUTION: Default variables are never saved in the .LAD file, but are always considered to exist.

© 2024 Robox SpA

The default variables are as follows:

ID	Name	Type	Description
0x00000001	INP(n)	BOOL	Input channel
0x00000002	INP_W(n)	U16	Input word 16bit
0x00000003	INP_DW(n)	U32	Input word 32bit
0x00000004	OUT	BOOL	Output channel
0x00000005	OUT_W(n)	U16	Output word 16bit
0x00000006	OUT_DW(n)	U32	Output word 32bit
0x00000007	R(n)	I32	Global integer register (volatile)
0x00000008	NVR(n)	I32	Global integer register (non-volatile)
0x00000009	RR(n)	REAL	Real global register (volatile)
0x00000A	NVRR(n)	REAL	Real global register (non-volatile)
0x00000B	AM(n)	U32	Alarm mask

Annotations:

- if the variable is of array type, the array index (n) can be either a constant or an unindexed variable.
- RDE3 creates additional predefined variables with completely arbitrary assignment of IDs (in the range 0x00000100-0x00010000): RTE recognizes such variables by name and not by that ID.

Source indentifiers

Source identifiers are used to uniquely identify the source of a certain entity (or element) within the ladder file, such as structures, variables, or OB classes. The value of the U32 identifier is composed as follows:

- U16 high, group identifier;
- U16 low, element identifier in the group.

The expected source identifiers are as follows:

Group	Element	Description
0x0000	0x0000	User defined entity manually
0x0001	0x0000	Entity imported from external file: generic file
0x0002	0x0000	Entity imported (forced) from RTE project: generic source

© 2024 Robox SpA

Group	Element	Description
0x0002	0x0001	Imported entity (forced) from RTE project: configurator source
0x0003	0x0000	Entity imported from include file: generic file

Live changes format (version 1.1)

Live changes format (version 1.1)

CAUTION: This information refers to an earlier version of the live changes format (version 1.1).

Live changes are basically a file containing a list of commands to modify, in a certain sequence, a ladder/diagram program: these modifications are normally either embedded in the main ladder file or are transmitted by RDE3 to the linked numerical control when loading/testing the live changes themselves.

Live changes consist mainly of a header 117 then several sections among which the main ones are:

- live changes command table 118;
- string table 24.

NOTE: All numeric data stored in this file are to be considered in little indian (x86 standard) format.

Live changes header

CAUTION: This information refers to an earlier version of the live changes format (version 1.1).

At the beginning of the file (offset 0) is a general header (LIVEHEADER) that contains access information for all remaining components and sub-sections of the file.

Name	Offset	Type	Description
VERSION	+0	U32	Version of the file (1.1.0).
VID	+4	U32	Version ID of the file
PLVID	+8	U32	Version ID of the file before the live changes. If the changes are actual to be applied, this field should be worth 0.
	+12	U32	(reserved)
FLAGS	+16	U32	File settings: 0x000001 = File contains verification errors

© 2024 Robox SpA

Name	Offset	Type	Description
LIVEFLAGS	+20	U32	Live edit settings: (none)
STRINGSOFF	+24	U32	Offset of the string table 24 (relative to the file).
STRINGSSIZE	+28	U32	Size of the string table 24.
LIVECMDSOFF	+32	U32	Offset of the live changes command table 118 (relative to the file).
LIVECMDSSIZE	+36	U32	Size of the live changes command table 118.
LASTBLOCKID	+40	U32	Last block ID used.
LASTRUNGID	+44	U32	Last circuit ID used.
LASTSTRUCTID	+48	U32	Last structure ID used.
LASTVARID	+52	U32	Last used variable ID.
	+56	U8[20]	(reserved)

Live changes command table

CAUTION: This information refers to an earlier version of the live changes format (version 1.1).

The live changes command table (LIVECMDS) consists of a number of contiguous definitions (LIVECMDs), each of which contains all the information about a specific live command.

The location (relative to the file) and size of the table are specified by the LIVEHEADER 117 .ITEMSOFF and LIVEHEADER 117 .ITEMSSIZE fields, respectively.

Live changes command definition

Live commands (LIVECMD) have a common initial part, then a specific data area for each type of element. The common part is as follows:

Name	Offset	Type	Description
CODE	+0	U16	Command code
SIZE	+2	U32	Full size of the command (all fields).
DATA	+6	...	Command-specific data (optional).

The commands currently provided are:

© 2024 Robox SpA

- Command 0x0001 - Circuit insertion 119
- Command 0x0002 - Edit circuit 120
- Command 0x0003 - Remove circuit 120
- Command 0x0004 - Structure insertion 120
- Command 0x0005 - Variable insertion 121

Command 0x0001 - Circuit Insertion

This command allows a circuit to be inserted into a given block, at a certain position. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0001
SIZE	+2	U32	32 + ITEMSSIZE
BLOCKID	+6	U32	container block ID
RUNGID	+10	U32	ID of the affected circuit (goes into Inserting state)
REFID	+14	U32	ID of reference circuit (only for insertion locations #1, otherwise 0)
IFLAGS	+18	U16	Insertion settings: 0x000F = Insertion location: 0x0 = Before the circuit with REFID ID (#1) 0x1 = After the circuit with REFID ID (#1) 0x2 = As the first circuit of the block 0x3 = As the last circuit of the block
FLAGS	+20	U16	Circuit settings: 0x0001 (ignored in live changes)
ITEMSOFF	+22	U32	Offset of the circuit element table 14 (relative to the file)
ITEMSSIZE	+26	U32	Size of the circuit element table 14
TEXT	+30	U32	Offset circuit description ((relative to strings 24))

WARNING: the circuit element table must still be appended to the live command itself.

© 2024 Robox SpA

Command 0x0002 - Edit Circuit

This command allows to insert a modified circuit, removing the old one, in a given block. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0002
SIZE	+2	U32	32 + ITEMSSIZE
BLOCKID	+6	U32	container block ID
RUNGID	+10	U32	ID of the affected circuit (goes into Inserting state)
REFID	+14	U32	ID of previous circuit to be removed (goes into Dropping state)
	+18	U16	(reserved)
FLAGS	+20	U16	Circuit settings: 0x0001 (ignored in live changes)
ITEMSOFF	+22	U32	Offset of the circuit element table 14 (relative to the file)
ITEMSSIZE	+26	U32	Size of the circuit element table 14
TEXT	+30	U32	Offset circuit description (related to strings 24)

Command 0x0003 - Circuit Removal.

This command allows to remove an existing circuit. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0003
SIZE	+2	U32	14
BLOCKID	+6	U32	Container block ID
RUNGID	+10	U32	ID of the circuit to be removed (goes into Deleting state)

Command 0x0004 - Structure Insertion

This command allows you to create a new structure. The data of the command, are as follows:

© 2024 Robox SpA

Name	Offset	Type	Description
CODE	+0	U16	0x0004
SIZE	+2	U32	32 + (VARSCNT * sizeof(VAR 11))
TYPEID	+6	U32	User structure type ID Note: only values 0x0001000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
FLAGS	+10	U16	Structure settings: 0x0001 (ignored in live edits)
NAME	+12	U32	Offset structure name (relative to strings 24)
TEXT	+16	U32	Offset description of structure (relative to strings 24)
VARSOFF	+20	U32	Variable table 11 offset (file related) (#1)
VARSCNT	+24	U32	Variable table 11 size
SIZE	+28	U32	Size of structure [bytes]

(#1) The structure definition uses a normal variable definition table to define its elements. In this case, variable IDs are meaningless and should always be 0.

Command 0x0005 - Variable insertion.

This command allows you to create a new variable. The command data, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0005
SIZE	+2	U32	48
VARID		U32	Unique ID of the user variable. Note: only values 0x0001000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
TYPEID		U32	ID variable data type Note: can be a default 25 or user-defined 10 type 25.
FLAGS		U16	Variable settings:

© 2024 Robox SpA

Name	Offset	Type	Description
			0x0001 (ignored in live edits) 0x0002 (ignored in live edits)
NAME		U32	Offset variable name (relative to strings 24)
TEXT		U32	Offset description of the variable (relative to strings 24)
BITTEXT		U32	Offset variable bit descriptions, i.e., sequence of strings related to each individual bit separated by the ascii character 0xFF (related to strings 24).
ARRAY0		U32	No. of indexes 1st level (0=No array) (#1) or index 0 for Aliases 26.
ARRAY1		U32	No. of indexes 2nd level (0=No array) (#1) or index 1 for Alias 26.
ARRAY2		U32	No. of indexes 3rd level (0=No array) (#1) or index 2 for Alias 26.
ALIASID		U16	Alias type 28 ID
ALIASBIT		U16	No. of bits per alias (depends on the type of alias ID)
SIZE		U32	Size of variable. If Alias 26, the size is 0.

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum index however should only be considered 16bit (i.e. maximum index 0xFFFF).

Live changes format (version 1.3)

Live changes format (version 1.3)

Live changes are basically a file containing a list of commands to modify, in a certain sequence, a ladder/diagram program: these modifications are normally either embedded in the main ladder file or are transmitted from RDE3 to the linked numerical control when loading/testing the live changes themselves.

Live changes consist mainly of a header 123 then several sections among which the main ones are:

- live changes command table 124;

© 2024 Robox SpA

- string table 52.

NOTE: All numeric data stored in this file are to be considered in little indian (x86 standard) format.

Live changes header

At the beginning of the file (offset 0) is a general header (LIVEHEADER) that contains access information for all remaining components and sub-sections of the file.

Name	Offset	Type	Description
VERSION	+0	U32	File version (1.3.0).
VID	+4	U32	Version ID of the file
PLVID	+8	U32	Version ID of the file before the live changes. If the changes are actual to be applied, this field should be worth 0.
	+12	U32	(reserved)
FLAGS	+16	U32	File settings: 0x000001 = File contains verification errors
LIVEFLAGS	+20	U32	Live edit settings: (none)
STRINGSOFF	+24	U32	Offset of the string table 52 (relative to the file).
STRINGSSIZE	+28	U32	Size of the string table 52.
LIVECMDSOFF	+32	U32	Offset of the live changes command table 124 (relative to the file).
LIVECMDSSIZE	+36	U32	Size of the live changes command table 124.
LASTBLOCKID	+40	U32	Last block ID used.
LASTRUNGID	+44	U32	Last circuit ID used.
LASTSTRUCTID	+48	U32	Last structure ID used.
LASTVARID	+52	U32	Last variable ID used.
LASTOBCLASSID	+56	U32	Last OB class ID used.

© 2024 Robox SpA

Name	Offset	Type	Description
IARECNT	+60	U32	No. of ARE type elements present in the file.
IAFECNT	+64	U32	No. of AFE type elements present in the file.
	+68	U8[8]	(reserved)

Live changes command table

The live changes command table (LIVECMDS) consists of a number of contiguous definitions (LIVECMD), each of which contains all the information about a specific live command.

The location (relative to the file) and size of the table are specified by the LIVEHEADER 123 .ITEMSOFF and LIVEHEADER 123 .ITEMSSIZE fields, respectively.

Live changes command definition

Live commands (LIVECMD) have a common initial part, then a specific data area for each type of element. The common part is as follows:

Name	Offset	Type	Description
CODE	+0	U16	Command code
SIZE	+2	U32	Full size of the command (all fields).
DATA	+6	...	Command-specific data (optional).

The commands currently provided are:

- Command 0x0001 - Circuit insertion 124
- Command 0x0002 - Edit circuit 125
- Command 0x0003 - Remove circuit 126
- Command 0x0006 - Structure insertion 126
- Command 0x0007 - Variable insertion 127

Command 0x0001 - Circuit Insertion

This command allows a circuit to be inserted into a given block, at a certain position. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0001
SIZE	+2	U32	32 + ITEMSSIZE
BLOCKID	+6	U32	container block ID

© 2024 Robox SpA

Name	Offset	Type	Description
RUNGID	+10	U32	ID of the affected circuit (goes into Inserting state)
REFID	+14	U32	ID of reference circuit (only for insertion locations #1, otherwise 0)
IFLAGS	+18	U16	Insertion settings: 0x000F = Insertion location: 0x0 = Before the circuit with REFID ID (#1) 0x1 = After the circuit with REFID ID (#1) 0x2 = As the first circuit of the block 0x3 = As the last circuit of the block
FLAGS	+20	U16	Circuit settings: 0x0001 (ignored in live changes)
ITEMSOFF	+22	U32	Offset of the circuit element table 38 (relative to the file)
ITEMSSIZE	+26	U32	Size of the circuit element table 38
TEXT	+30	U32	Offset circuit description (relative to strings 52)

WARNING: the circuit element table must still be appended to the live command itself.

Command 0x0002 - Edit Circuit

This command allows to insert a modified circuit, removing the old one, in a given block. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0002
SIZE	+2	U32	32 + ITEMSSIZE
BLOCKID	+6	U32	container block ID
RUNGID	+10	U32	ID of the affected circuit (goes into Inserting state)

© 2024 Robox SpA

Name	Offset	Type	Description
REFID	+14	U32	ID of previous circuit to be removed (goes into Dropping state)
	+18	U16	(reserved)
FLAGS	+20	U16	Circuit settings: 0x0001 (ignored in live changes)
ITEMSOFF	+22	U32	Offset of the circuit element table 38 (relative to the file)
ITEMSSIZE	+26	U32	Size of the circuit element table 38
TEXT	+30	U32	Offset circuit description (related to strings 52)

Command 0x0003 - Circuit Removal.

This command allows to remove an existing circuit. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0003
SIZE	+2	U32	14
BLOCKID	+6	U32	Container block ID
RUNGID	+10	U32	ID of the circuit to be removed (goes into Deleting state)

Command 0x0006 - Structure Insertion

This command allows you to create a new structure. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0006
SIZE	+2	U32	44 + (VARSCNT * sizeof(VAR 34))
TYPEID	+6	U32	User structure type ID Note: only values 0x0001000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
FLAGS	+10	U16	Structure settings:

© 2024 Robox SpA

Name	Offset	Type	Description
			0x0001 (ignored in live edits) 0x0002 (ignored in live edits)
NAME	+12	U32	Offset structure name (relative to strings 52).
TEXT	+16	U32	Offset description of structure (relative to strings 52).
VARSOFF	+20	U32	Variable table 34 offset (file related) (#1)
VARSCNT	+24	U32	Variable table 34 size
SIZE	+28	U32	Structure size [bytes]
OBCID	+32	U32	Reference OB class ID (0=none)(#2,#3)
ATTRIB	+36	U32	Structure attributes: 0x000001 Structure referred to OB class (AREFOBC)(#2) 0x000002 Main structure OB class (AISOBC)(#2,#3)
MODIFIER	+40	U32	Offset structure modifiers (related to strings 52).

(*#1) Structure definition uses a normal variable definition table to define its elements. In this case, variable IDs are meaningless and should always be 0.*

(#2) At equal non-0 OBCID value with AREFOBC attribute, there can be only one structure with AISOBC flag, while there can be 0 or more structures without AISOBC attribute.

(#3) A structure is called an "orphan" structure if it is in the condition of AREFOBC attribute set and OBCID field = 0: this condition happens when updating/removing an OB class, there are still variables (or other) referencing it.

Command 0x0007 - Variable insertion.

This command allows you to create a new variable with an initialization value. The command data, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0007
SIZE	+2	U32	60
VARID	+6	U32	Unique ID of the user variable.

© 2024 Robox SpA

Name	Offset	Type	Description
			Note: only values 0x0001000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
TYPEID	+10	U32	ID variable data type Note: can be a default 53 or user-defined 33 type 53.
FLAGS	+14	U16	Variable settings: 0x0001 (ignored in live edits) 0x0002 (ignored in live edits) 0x0004 (ignored in live edits)
NAME	+16	U32	Offset variable name (relative to strings 52)
TEXT	+20	U32	Offset description of the variable (relative to strings 52)
BITTEXT	+24	U32	Offset variable bit descriptions, i.e., sequence of strings related to each individual bit separated by the ascii character 0xFF (related to strings 52).
DIM0	+28	U32	1st array size (0=unspecified)(#1) or index 0 for Alias 54.
DIM1	+32	U32	2nd array size (0=unspecified)(#1) or index 1 for Alias 54.
DIM2	+36	U32	3rd array size (0=unspecified)(#1) or index 2 for Alias 54.
ALIASID	+40	U16	Alias type 55 ID
ALIASBIT	+42	U16	No. of bits per alias (depends on the type of alias ID)
SIZE	+44	U32	Size of variable. If Alias 54, the size is 0.

© 2024 Robox SpA

Name	Offset	Type	Description
VALUE	+48	U32	Offset initial value of variable (relative to strings 52, 0=no value).
ATTRIB	+52	U32	Variable attributes: 0x000001 Variable can be written (WRITE) 0x000002 The variable can be read (READ) 0x000004 The variable can be forced (FORCE) 0x000008 The variable can be released (RELEASE) 0x000010 The variable is constant (CONST) 0x000020 The variable has no bit access (NOBITS)
MODIFIER	+56	U32	Variable modifier offsets (relative to strings 52).
DIM0L	+60	U32	Offset label 1st array size (related to strings 52). (#2)
DIM1L	+64	U32	Offset label 2nd array size (relative to strings 52). (#2)
DIM2L	+68	U32	Offset label 3rd array size (relative to strings 52). (#2)

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum size should still only be considered to be 16bit (i.e. maximum index 0xFFFF).

(#2) Label means either a variable name (constant value) or the ascii representation of a numeric value, i.e., an offset to the string table 52.

Live changes format (version 1.4)

Live changes format (version 1.4)

Live changes modifications are basically a file containing a list of commands to modify, in a certain sequence, a ladder/diagram program: these modifications are normally either embedded in the main ladder file or are transmitted from RDE3 to the linked numerical control when loading/testing the live changes themselves.

Live changes consist mainly of a header 130 then several sections among which the main ones are:

- live changes command table 131;
- string table 81.

NOTE: All numeric data stored in this file are to be considered in little indian (x86 standard) format.

Live changes header

At the beginning of the file (offset 0) is a general header (LIVEHEADER) that contains access information for all remaining components and sub-sections of the file.

Name	Offset	Type	Description
VERSION	+0	U32	File version (1.4.0).
VID	+4	U32	Version ID of the file
PLVID	+8	U32	Version ID of the file before the live changes. If the changes are actual to be applied, this field should be worth 0.
	+12	U32	(reserved)
FLAGS	+16	U32	File settings: 0x000001 = File contains verification errors 0x000004 = The file uses UTF-8 encoding
LIVEFLAGS	+20	U32	Live edit settings: (none)
STRINGSOFF	+24	U32	Offset of the string table (relative to the file).
STRINGSSIZE	+28	U32	Size of the string table.
LIVECMDSOFF	+32	U32	Offset of the live changes command table 131 (relative to the file).
LIVECMDSSIZE	+36	U32	Size of the live changes command table 131.
LASTBLOCKID	+40	U32	Last block ID used.
LASTRUNGID	+44	U32	Last circuit ID used.
LASTSTRUCTID	+48	U32	Last structure ID used.
LASTVARID	+52	U32	Last variable ID used.
LASTOBCLASSID	+56	U32	Last OB class ID used.
IARECNT	+60	U32	No. of ARE type elements present in the file.

© 2024 Robox SpA

Name	Offset	Type	Description
IAFECNT	+64	U32	No. of AFE type elements present in the file.
	+68	U8[60]	(reserved)

Live changes command table

The live command table (LIVECMDS) consists of a number of contiguous definitions (LIVECMD), each of which contains all the information about a specific live command.

The location (relative to the file) and size of the table are specified by the LIVEHEADER 130 .ITEMSOFF and LIVEHEADER 130 .ITEMSSIZE fields, respectively.

live command definition

Live commands (LIVECMD) have a common initial part, then a specific data area for each type of element. The common part is as follows:

Name	Offset	Type	Description
CODE	+0	U16	Command code
SIZE	+2	U32	Full size of the command (all fields).
DATA	+6	...	Command-specific data (optional).

The commands currently provided are:

- Command 0x0001 - Circuit insertion 131
- Command 0x0002 - Edit circuit 132
- Command 0x0003 - Remove circuit 133
- Command 0x0008 - Structure insertion 133
- Command 0x0009 - Variable insertion 134

Command 0x0001 - Circuit Insertion

This command allows a circuit to be inserted into a given block, at a certain position. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0001
SIZE	+2	U32	32 + ITEMSSIZE
BLOCKID	+6	U32	container block ID
RUNGID	+10	U32	ID of the affected circuit (goes into Inserting state)

© 2024 Robox SpA

Name	Offset	Type	Description
REFID	+14	U32	ID of reference circuit (only for insertion locations #1, otherwise 0)
IFLAGS	+18	U16	Insertion settings: 0x000F = Insertion location: 0x0 = Before the circuit with REFID ID (#1) 0x1 = After the circuit with REFID ID (#1) 0x2 = As the first circuit of the block 0x3 = As the last circuit of the block
FLAGS	+20	U16	Circuit settings: 0x0001 (ignored in live changes)
ITEMSOFF	+22	U32	Offset of the circuit element table 67 (relative to the file)
ITEMSSIZE	+26	U32	Size of the circuit element table 67
TEXT	+30	U32	Offset circuit description (relative to strings 81)

WARNING: the circuit element table must still be appended to the live command itself.

Command 0x0002 - Edit Circuit

This command allows to insert a modified circuit, removing the old one, in a given block. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0002
SIZE	+2	U32	32 + ITEMSSIZE
BLOCKID	+6	U32	container block ID
RUNGID	+10	U32	ID of the affected circuit (goes into Inserting state)
REFID	+14	U32	ID of previous circuit to be removed (goes into Dropping state)
	+18	U16	(reserved)

© 2024 Robox SpA

Name	Offset	Type	Description
FLAGS	+20	U16	Circuit settings: 0x0001 (ignored in live changes)
ITEMSOFF	+22	U32	Offset of the circuit element table 67 (relative to the file)
ITEMSSIZE	+26	U32	Size of the circuit element table 67
TEXT	+30	U32	Offset circuit description (related to strings 81)

Command 0x0003 - Circuit Removal.

This command allows to remove an existing circuit. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0003
SIZE	+2	U32	14
BLOCKID	+6	U32	Container block ID
RUNGID	+10	U32	ID of the circuit to be removed (goes into Deleting state)

Command 0x0008 - Structure Insertion

This command allows you to create a new structure. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0008
SIZE	+2	U32	68 + (VARSCNT * sizeof(VAR 63))
TYPEID	+6	U32	User structure type ID Note: only values 0x0001000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
FLAGS	+10	U16	Structure settings: 0x0001 (ignored in live edits) 0x0002 (ignored in live edits)

© 2024 Robox SpA

Name	Offset	Type	Description
NAME	+12	U32	Offset structure name (relative to strings 81).
TEXT	+16	U32	Offset description of structure (relative to strings 81).
VARSOFF	+20	U32	Variable table 63 offset (file related) (#1).
VARSCNT	+24	U32	Variable table 63 size
SIZE	+28	U32	Size of the structure [bytes].
OBCID	+32	U32	Reference OB class ID (0=none).(#2,#3)
ATTRIB	+36	U32	Structure attributes: 0x000001 Structure referred to OB class (AREFOBC)(#2) 0x000002 Main structure OB class (AISOBC)(#2,#3) 0x000004 Structure with external name (AEXTNAME)
MODIFIER	+40	U32	Offset modifiers of structure (relative to strings 81).
REFID	+44	U32	Structure type reference ID (#4).
EXTNAME	+48	U32	Offset external structure name (relative to strings 81).
	+52	U8[16]	(reserved)

(#1) The structure definition uses a normal variable definition table to define its elements. In this case, variable IDs are meaningless and should always be 0.

(#2) At equal non-0 OBCID value with AREFOBC attribute, there can be only one structure with AISOBC flag, while there can be 0 or more structures without AISOBC attribute.

(#3) A structure is called an "orphan" structure if it is in the condition of AREFOBC attribute set and OBCID field = 0: this condition happens when updating/removing an OB class, there are still variables (or other) referencing it.

(#4) In case of AREFOBC attribute, the REFID field contains the ID of the structure as defined of the source OBB file (0 in case of AISOBC attribute).

Command 0x0009 - Variable insertion.

This command allows you to create a new variable with an initialization value. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0009
SIZE	+2	U32	76
VARID	+6	U32	Unique ID of the user variable. Note: only values 0x0001000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
TYPEID	+10	U32	ID variable data type Note: can be a default 82 or user-defined 61 type 82.
FLAGS	+14	U16	Variable settings: 0x0001 (ignored in live edits) 0x0002 (ignored in live edits) 0x0004 (ignored in live edits)
NAME	+16	U32	Offset variable name (relative to strings 81).
TEXT	+20	U32	Offset description of variable (relative to strings 81).
BITTEXT	+24	U32	Offset variable bit descriptions, i.e., sequence of strings related to each individual bit separated by the ascii character 0xFF (related to strings 52).
DIM0	+28	U32	1st array size (0=unspecified)(#1) or index 0 for Alias 83.
DIM1	+32	U32	2nd array size (0=unspecified)(#1) or index 1 for Alias 83.
DIM2	+36	U32	3rd array size (0=unspecified)(#1) or index 2 for Alias 83.
ALIASID	+40	U16	Alias type 84 ID

© 2024 Robox SpA

Name	Offset	Type	Description
ALIASBIT	+42	U16	No. of bits per alias (depends on the type of alias ID)
SIZE	+44	U32	Size of variable. If Alias 83, the size is 0.
VALUE	+48	U32	Offset initial value of variable (relative to strings 81, 0=no value).
ATTRIB	+52	U32	Variable attributes: 0x000001 Variable can be written (WRITE) 0x000002 The variable can be read (READ) 0x000004 The variable can be forced (FORCE) 0x000008 The variable can be released (RELEASE) 0x000010 The variable is constant (CONST) 0x000020 The variable has no bit access (NOBITS)
MODIFIER	+56	U32	Variable modifier offsets (relative to strings 81).
DIM0L	+60	U32	Offset label 1st array size (related to strings 81). (#2)
DIM1L	+64	U32	Offset label 2nd array size (relative to strings 81). (#2)
DIM2L	+68	U32	Offset label 3rd array size (relative to strings 81). (#2)
	+72	U8[16]	(reserved)

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum size should still be considered only 16bit (i.e., maximum index 0xFFFF).

(#2) Label means either a variable name (constant value) or the ascii representation of a numeric value, i.e., an offset to the string table .

© 2024 Robox SpA

Live changes format (version 1.5)

Live changes format (version 1.5)

Live changes are basically a file containing a list of commands to modify, in a certain sequence, a ladder/diagram program: these modifications are normally either embedded in the main ladder file or are transmitted by RDE to the linked numerical control when loading/testing the live changes themselves.

Live changes consist mainly of a header 137 then several sections among which the main ones are:

- live changes command table 138;
- string table 81.

NOTE: All numeric data stored in this file are to be considered in little indian (x86 standard) format.

Live changes header

At the beginning of the file (offset 0) is a general header (LIVEHEADER) that contains access information for all remaining components and sub-sections of the file.

Name	Offset	Type	Description
VERSION	+0	U32	File version (1.5.0).
VID	+4	U32	Version ID of the file
PLVID	+8	U32	Version ID of the file before the live changes. If the changes are actual to be applied, this field should be worth 0.
	+12	U32	(reserved)
FLAGS	+16	U32	File settings: 0x000001 = File contains verification errors 0x000004 = The file uses UTF-8 encoding
LIVEFLAGS	+20	U32	Live edit settings: (none)
STRINGSOFF	+24	U32	Offset of the string table (relative to the file).
STRINGSSIZE	+28	U32	Size of the string table.
LIVECMDSOFF	+32	U32	Offset of the live changes command table 138 (relative to the file).

© 2024 Robox SpA

Name	Offset	Type	Description
LIVECMDSSIZE	+36	U32	Size of the live changes command table 138.
LASTBLOCKID	+40	U32	Last block ID used.
LASTRUNGID	+44	U32	Last circuit ID used.
LASTSTRUCTID	+48	U32	Last structure ID used.
LASTVARID	+52	U32	Last variable ID used.
LASTOBCLASSID	+56	U32	Last OB class ID used.
IARECNT	+60	U32	No. of ARE type elements present in the file.
IAFECNT	+64	U32	No. of AFE type elements present in the file.
LASTINCFILEID	+68	U32	Last used include file ID.
	+72	U8[56]	(reserved)

Live changes command table

The live command table (LIVECMDS) consists of a number of contiguous definitions (LIVECMD), each of which contains all the information about a specific live command.

The location (relative to the file) and size of the table are specified by the LIVEHEADER 137 .ITEMSOFF and LIVEHEADER 137 .ITEMSSIZE fields, respectively.

Live changes command definition

Live commands (LIVECMD) have a common initial part, then a specific data area for each type of element. The common part is as follows:

Name	Offset	Type	Description
CODE	+0	U16	Command code
SIZE	+2	U32	Full size of the command (all fields).
DATA	+6	...	Command-specific data (optional).

The commands currently provided are:

- Command 0x0001 - Circuit insertion 139
- Command 0x0002 - Edit circuit 139
- Command 0x0003 - Remove circuit 140
- Command 0x0008 - Structure insertion 140

© 2024 Robox SpA

- Command 0x0009 - Variable insertion 142

Command 0x0001 - Circuit Insertion

This command allows a circuit to be inserted into a given block, at a certain position. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0001
SIZE	+2	U32	32 + ITEMSSIZE
BLOCKID	+6	U32	container block ID
RUNGID	+10	U32	ID of the affected circuit (goes into Inserting state)
REFID	+14	U32	ID of reference circuit (only for insertion locations #1, otherwise 0)
IFLAGS	+18	U16	Insertion settings: 0x000F = Insertion location: 0x0 = Before the circuit with REFID ID (#1) 0x1 = After the circuit with REFID ID (#1) 0x2 = As the first circuit of the block 0x3 = As the last circuit of the block
FLAGS	+20	U16	Circuit settings: 0x0001 (ignored in live changes)
ITEMSOFF	+22	U32	Offset of the circuit element table 67 (relative to the file)
ITEMSSIZE	+26	U32	Size of the circuit element table 67
TEXT	+30	U32	Offset circuit description (relative to strings 81)

WARNING: the circuit element table must still be appended to the live command itself.

Command 0x0002 - Edit Circuit

This command allows to insert a modified circuit, removing the old one, in a given block. The data of the command, are as follows:

© 2024 Robox SpA

Name	Offset	Type	Description
CODE	+0	U16	0x0002
SIZE	+2	U32	32 + ITEMSSIZE
BLOCKID	+6	U32	container block ID
RUNGID	+10	U32	ID of the affected circuit (goes into Inserting state)
REFID	+14	U32	ID of previous circuit to be removed (goes into Dropping state)
	+18	U16	(reserved)
FLAGS	+20	U16	Circuit settings: 0x0001 (ignored in live changes)
ITEMSOFF	+22	U32	Offset of the circuit element table 67 (relative to the file)
ITEMSSIZE	+26	U32	Size of the circuit element table 67
TEXT	+30	U32	Offset circuit description (related to strings 81)

Command 0x0003 - Circuit Removal.

This command allows to remove an existing circuit. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0003
SIZE	+2	U32	14
BLOCKID	+6	U32	Container block ID
RUNGID	+10	U32	ID of the circuit to be removed (goes into Deleting state)

Command 0x0008 - Structure Insertion

This command allows you to create a new structure. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0008

© 2024 Robox SpA

Name	Offset	Type	Description
SIZE	+2	U32	68 + (VARSCNT * sizeof(VAR 63))
TYPEID	+6	U32	User structure type ID Note: only values 0x0001000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
FLAGS	+10	U16	Structure settings: 0x0001 (ignored in live edits) 0x0002 (ignored in live edits)
NAME	+12	U32	Offset structure name (relative to strings 81).
TEXT	+16	U32	Offset description of structure (relative to strings 81).
VARSOFF	+20	U32	Variable table 63 offset (file related) (#1).
VARSCNT	+24	U32	Variable table 63 size
SIZE	+28	U32	Size of the structure [bytes].
OBCID	+32	U32	Reference OB class ID (0=none).(#2,#3)
ATTRIB	+36	U32	Structure attributes: 0x000001 Structure referred to OB class (AREFOBC)(#2) 0x000002 Main structure OB class (AISOBC)(#2,#3) 0x000004 Structure with external name (AEXTNAME)
MODIFIER	+40	U32	Offset modifiers of structure (relative to strings 81).
REFID	+44	U32	Structure type reference ID (#4).
EXTNAME	+48	U32	Offset external structure name (relative to strings 81).

© 2024 Robox SpA

Name	Offset	Type	Description
	+52	U8[16]	(reserved)

(#1) The structure definition uses a normal variable definition table to define its elements. In this case, variable IDs are meaningless and should always be 0.

(#2) At equal non-0 OBCID value with AREFOBC attribute, there can be only one structure with AISOBC flag, while there can be 0 or more structures without AISOBC attribute.

(#3) A structure is called an "orphan" structure if it is in the condition of AREFOBC attribute set and OBCID field = 0: this condition happens when updating/removing an OB class, there are still variables (or other) referencing it.

(#4) In case of AREFOBC attribute, the REFID field contains the ID of the structure as defined of the source OBB file (0 in case of AISOBC attribute).

Command 0x0009 - Variable insertion.

This command allows you to create a new variable with an initialization value. The data of the command, are as follows:

Name	Offset	Type	Description
CODE	+0	U16	0x0009
SIZE	+2	U32	76
VARID	+6	U32	Unique ID of the user variable. Note: only values 0x0001000-0xFFFFFE; values 0 and 0xFFFFFF are special meaning IDs.
TYPEID	+10	U32	ID variable data type Note: can be a default 82 or user-defined 61 type 82.
FLAGS	+14	U16	Variable settings: 0x0001 (ignored in live edits) 0x0002 (ignored in live edits) 0x0004 (ignored in live edits)
NAME	+16	U32	Offset variable name (relative to strings 81).
TEXT	+20	U32	Offset description of variable (relative to strings 81).
BITTEXT	+24	U32	Offset variable bit descriptions, i.e., sequence of strings

© 2024 Robox SpA

Name	Offset	Type	Description
			related to each individual bit separated by the ascii character 0xFF (related to strings 52).
DIM0	+28	U32	1st array size (0=unspecified)(#1) or index 0 for Alias 83.
DIM1	+32	U32	2nd array size (0=unspecified)(#1) or index 1 for Alias 83.
DIM2	+36	U32	3rd array size (0=unspecified)(#1) or index 2 for Alias 83.
ALIASID	+40	U16	Alias type 84 ID
ALIASBIT	+42	U16	No. of bits per alias (depends on the type of alias ID)
SIZE	+44	U32	Size of variable. If Alias 83, the size is 0.
VALUE	+48	U32	Offset initial value of variable (relative to strings 81, 0=no value).
ATTRIB	+52	U32	Variable attributes: 0x000001 Variable can be written (WRITE) 0x000002 The variable can be read (READ) 0x000004 The variable can be forced (FORCE) 0x000008 The variable can be released (RELEASE) 0x000010 The variable is constant (CONST) 0x000020 The variable has no bit access (NOBITS)
MODIFIER	+56	U32	Variable modifier offsets (relative to strings 81).
DIM0L	+60	U32	Offset label 1st array size (related to strings 81). (#2)

© 2024 Robox SpA

Name	Offset	Type	Description
DIM1L	+64	U32	Offset label 2nd array size (relative to strings 81). (#2)
DIM2L	+68	U32	Offset label 3rd array size (relative to strings 81). (#2)
	+72	U8[16]	(reserved)

(#1) Although ARRAYn fields are 32bit, in the current version of SW the maximum size should still be considered only 16bit (i.e., maximum index 0xFFFF).

(#2) Label means either a variable name (constant value) or the ascii representation of a numeric value, i.e., an offset to the string table .

© 2024 Robox SpA

Index

- B -

BLOCK 37
BLOCKS 37

- C -

CSTRINGS 53

- E -

EMBLIVE 53
EMBLIVES 53

- F -

File format (version 1.2) 7
File format (version 1.3) 30
File format (version 1.4) 58
File format (version 1.5) 87

- H -

HEADER 30

- I -

ITEM 39
ITEMS 38

- L -

Live changes format (version 1.1) 117
Live changes format (version 1.3) 122
Live changes format (version 1.5) 137
LIVECMD 118
LIVECMDS 118
LIVEHEADER 117

- M -

Minimal requirements 6

- R -

RUNG 38
RUNGS 38

- S -

STRINGS 52
STRUCT 33
STRUCTS 33

- T -

Tabella comandi LIVE 138

- V -

VAR 34
VARS 34

© 2024 Robox SpA