

Binary EDS File Format v2.00

Document version 2.00, AA, September 2014

© Embedded Systems Academy, Inc.

1 OVERVIEW

This document defines a binary file format that describes an EDS. It can be used by CANopen stacks for run-time configuration of the Object Dictionary.

2 TOP LEVEL FORMAT

The file is arranged into the following top-level sections:

Offset Into File (bytes)	Size (bytes)	Description
0	128	Header
128	32	Table Offsets
160	<i>n</i>	Tables
160 + <i>n</i>	2	CRC

Each top-level section is described in turn in rest of this document.

All multi-byte values are stored in little-endian format.

3 HEADER

Offset into file: 0 bytes

Length: 128 bytes

The header has the following structure:

Offset Into Header (bytes)	Size (bytes)	Description
0	2	File format major version
2	2	File format minor version
4	4	'P', 'O', 'C', 'M'
8	4	Functionality of node (FUNC)
12	2	Baudrate in kbps
14	1	Node ID
15	1	Reserved, always 0x00
16	2	Number of RPDOs
18	2	Number of TPDOs
20	4	Process image size in bytes
24	4	Reserved, always 0x00000000
28	4	Reserved, always 0x00000000

32	96	Identification string
----	----	-----------------------

This document describes file format “2.00”, i.e. the major version is two and the minor version is zero.

The FUNC value is 32-bits in size, which are used as follows:

Bits	Description
0	LSS supported : No = 0, Yes = 1
1	Autostart : No = 0, Yes = 1
2	Bootup manager (NMT master) : No = 0, Yes = 1
3	CANopen manager : No = 0, Yes = 1
4	Process image min/max values included : No = 0, Yes = 1
5	Read node ID and baudrate from hardware: No = 0, Yes = 1
6 - 31	Reserved, always 0

The identification string is stored in ISO-8859-1 “latin1” encoding. It does not have a null terminator. If it is less than 96 characters in length then it is padded with 0x00 at the end until it is 96 characters in length.

The intention is that the first eight bytes in the file identify the file type and version number of its internal structure. Using this information the rest of the file can be parsed.

4 TABLE OFFSETS

Offset into file: 128 bytes

Length: 32 bytes

The table offsets have the following structure:

Offset Into Table Offsets (bytes)	Size (bytes)	Description
0	4	Offset from start of file to SDO Reply table
4	4	Offset from start of file to OD Entry table
8	4	Offset from start of file to Generic OD Entry table
12	4	Offset from start of file to Process Image Defaults table
16	4	Offset from start of file to Process Image Maximums table
20	4	Offset from start of file to Process Image Minimums table
24	4	Offset from start of file to RPDO Configuration table
28	4	Offset from start of file to TPDO Configuration table

Each offset is a 32-bit value taken from the start of the file. The order of tables in the file is not guaranteed. E.g. the RPDO Configuration table could have a smaller offset than the SDO Reply table, indicating that it appears earlier in the file.

5 TABLES

Offset into file: Starting at 160 bytes

Length: n bytes

All the tables are of variable length.

The tables might not be stored back-to-back in the file. Depending on the data alignment setting used when the file is generated there could be up to eight bytes of padding between the end of one table and the start of the next. Padding bytes are always 0x00. For example if the data alignment was set to four bytes then all offsets to the start of tables will be on a four-byte boundary. This is because some data in the tables is intended to be directly copied to transmit buffers and misaligned data would result in degraded performance.

If bit 4 of the FUNC value in the header is zero then the length of the process image maximums table and process image minimums table are zero bytes and they do not have any alignment padding.

5.1 SDO REPLY TABLE

Offset into file: Read from first entry of table offsets

The table consists of a set of records followed by a sentinel record.

Offset Into Table (bytes)	Size (bytes)	Description
0	8	Record 1
8	8	Record 2
...
m	8	Record p
$m + 8$	8	Sentinel record

Each record defines an object dictionary entry. Entries that are constant values one to four bytes in size are included in this table.

Bytes in Record	Size (bytes)	Description
0	1	First byte of expedited SDO response
1	2	Object Dictionary index
3	1	Object Dictionary subindex
4	4	Data

Each record defines all eight bytes of an expedited SDO response.

The sentinel record marks the end of the table and all eight bytes of the record are set to 0xFF.

5.2 OD ENTRY TABLE

Offset into file: Read from second entry of table offsets

The table consists of a set of records followed by a sentinel record.

Offset Into Table (bytes)	Size (bytes)	Description
0	6	Record 1
6	6	Record 2
...
m	6	Record p
$m + 6$	6	Sentinel record

Each record defines an object dictionary entry. Entries that have data in the process image and are four bytes or less in size are included in this table.

Bytes in Record	Size (bytes)	Description
0	2	Object Dictionary index
2	1	Object Dictionary subindex
3	1	Data size plus access type (DSAT)
4	2	Offset into process image from start of process image table

Object Dictionary entries are placed into the table in the following order:

1. Entries four bytes or less that are mapped into PDOs
2. Entries four bytes or less that are not mapped into PDOs

The DSAT byte is defined as follows:

Bits	Description
0 - 3	Object Dictionary entry size in bytes
4	Read access: No = 0, Yes = 1
5	Write access: No = 0, Yes = 1
6	PDO read mapping: No = 0, Yes = 1
7	PDO write mapping: No = 0, Yes = 1

The sentinel record marks the end of the table and all six bytes of the record are set to 0xFF.

5.3 GENERIC OD ENTRY TABLE

Offset into file: Read from third entry of table offsets

The table consists of a set of records followed by a sentinel record.

Offset Into Table (bytes)	Size (bytes)	Description
0	8	Record 1
8	8	Record 2
...
m	8	Record p
$m + 8$	8	Sentinel record

Each record defines an object dictionary entry. Entries that have data in the process image and are more than four bytes in size are included in this table.

All records are the same size.

Bytes in Record	Size (bytes)	Description
0	2	Object Dictionary index
2	1	Object Dictionary subindex
3	1	Access type (ACC)
4	2	Data size
6	2	Offset into process image from start of process image table

The ACC byte is defined as follows:

Bits	Description
0 - 3	Reserved, always set to zero
4	Read access: No = 0, Yes = 1
5	Write access: No = 0, Yes = 1
6	PDO read mapping: No = 0, Yes = 1
7	PDO write mapping: No = 0, Yes = 1

The sentinel record marks the end of the table and all eight bytes of the record are set to 0xFF.

5.4 PROCESS IMAGE DEFAULTS TABLE

Offset into file: Read from fourth entry of table offsets

The table consists of a set of sequential blocks of data of varying lengths containing the default values for process image data.

Each block of data may have padding before or after it to ensure correct data alignment. Padding bytes are always set to 0x00.

To find the starting offset and length of a block of data in the table use the OD Entry and Generic OD Entry tables.

5.5 PROCESS IMAGE MAXIMUMS TABLE

Offset into file: Read from fifth entry of table offsets

The table consists of a set of sequential blocks of data of varying lengths containing the maximum values for process image data.

Each block of data may have padding before or after it to ensure correct data alignment. Padding bytes are always set to 0x00.

To find the starting offset and length of a block of data in the table use the OD Entry and Generic OD Entry tables.

5.6 PROCESS IMAGE MINIMUMS TABLE

Offset into file: Read from sixth entry of table offsets

The table consists of a set of sequential blocks of data of varying lengths containing the minimum values for process image data.

Each block of data may have padding before or after it to ensure correct data alignment. Padding bytes are always set to 0x00.

To find the starting offset and length of a block of data in the table use the OD Entry and Generic OD Entry tables.

5.7 RPDO CONFIGURATION TABLE

Offset into file: Read from seventh entry of table offsets

The table consists of a set of records followed by a sentinel record.

Offset Into Table (bytes)	Size (bytes)	Description
0	12	Record 1
12	12	Record 2
...
<i>m</i>	12	Record <i>p</i>
<i>m</i> + 12	12	Sentinel record

Each record defines a receive PDO:

Bytes in Record	Size (bytes)	Description
0	1	Number of RPDO (0-indexed)
1	1	Transmission type
2	1	Length in bytes
3	1	Reserved. Always set to 0x00
4	4	COB-ID
8	4	Offset into process image from start of process image table for first mapped entry

The sentinel record marks the end of the table and all 12 bytes of the record are set to 0xFF.

5.8 TPDO CONFIGURATION TABLE

Offset into file: Read from eighth entry of table offsets

The table consists of a set of records followed by a sentinel record.

Offset Into Table (bytes)	Size (bytes)	Description
0	16	Record 1
16	16	Record 2
...

<i>m</i>	16	Record <i>p</i>
<i>m + 16</i>	16	Sentinel record

Each record defines a transmit PDO:

Bytes in Record	Size (bytes)	Description
0	1	Number of TPDO (0-indexed)
1	1	Transmission type
2	1	Length in bytes
3	1	Reserved. Always set to 0x00
4	4	COB-ID
8	4	Offset into process image from start of process image table for first mapped entry
12	2	Event time
14	2	Inhibit time

The sentinel record marks the end of the table and all 16 bytes of the record are set to 0xFF.

6 CRC

Offset into file: 160 + n bytes

Length: 2 bytes

The CRC is located in the last two bytes of the file and is stored in little-endian format.

The value is calculated over all of the preceding bytes in the file.

The CRC uses the SDO checksum algorithm and polynomial, $x^{16} + x^{12} + x^5 + 1$, with a starting value of zero.