

BS EN 62258-2:2011



BSI Standards Publication

Semiconductor die products

Part 2: Exchange data formats

bsi.

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

National foreword

This British Standard is the UK implementation of EN 62258-2:2011. It is identical to IEC 62258-2:2011. It supersedes BS EN 62258-2:2005 which is withdrawn.

The UK participation in its preparation was entrusted to Technical Committee EPL/47, Semiconductors.

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

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

© BSI 2011

ISBN 978 0 580 61892 5

ICS 31.080.99

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

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 July 2011.

Amendments issued since publication

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

EUROPEAN STANDARD
NORME EUROPÉENNE
EUROPÄISCHE NORM

EN 62258-2

July 2011

ICS 31.080.99

Supersedes EN 62258-2:2005

English version

Semiconductor die products - Part 2: Exchange data formats (IEC 62258-2:2011)

Produits de puces de semiconducteurs -
Partie 2: Formats d'échange de données
(CEI 62258-2:2011)

Halbleiter-Chip-Erzeugnisse -
Teil 2: Datenaustausch-Formate
(IEC 62258-2:2011)

This European Standard was approved by CENELEC on 2011-06-29. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

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

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

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

CENELEC

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

Management Centre: Avenue Marnix 17, B - 1000 Brussels

© 2011 CENELEC - All rights of exploitation in any form and by any means reserved worldwide for CENELEC members.

Ref. No. EN 62258-2:2011 E

 **Documents**
Document Sharing Hub

Foreword

The text of document 47/2085/FDIS, future edition 2 of IEC 62258-2, prepared by IEC TC 47, Semiconductor devices, was submitted to the IEC-CENELEC parallel vote and was approved by CENELEC as EN 62258-2 on 2011-06-29.

This European Standard supersedes EN 62258-2:2005.

With respect to EN 62258-2:2005, the following parameters have been updated for EN 62258-2:2011:

Subclause	Parameter name
8.2.9	DEVICE_PICTURE_FILE
8.2.10	DEVICE_DATA_FILE
8.4.6	TERMINAL_GROUP
8.4.7	PERMUTABLE
8.5.1	TERMINAL_MATERIAL (was DIE_TERMINAL_MATERIAL)
8.5.2	TERMINAL_MATERIAL_STRUCTURE
8.6.2	MAX_TEMP_TIME
8.7.6	SIMULATOR_simulator_TERM_GROUP
8.8.3	ASSEMBLY
8.9.2	WAFER_THICKNESS
8.9.3	WAFER_THICKNESS_TOLERANCE
8.9.9	WAFER_INK
8.10.4	BUMP_SHAPE
8.10.5	BUMP_SIZE
8.10.6	BUMP_SPECIFICATION_DRAWING
8.10.7	BUMP_ATTACHMENT_METHOD
8.11.4	MPD_MSL_LEVEL
8.11.5	MPD_PACKAGE_DRAWING
8.12.1	QUALITY
8.12.2	TEST
8.13.1	TEXT
8.14.1	PARSE

This standard shall be read in conjunction with EN 62258-1.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CEN and CENELEC shall not be held responsible for identifying any or all such patent rights.

The following dates were fixed:

- latest date by which the EN has to be implemented
at national level by publication of an identical
national standard or by endorsement (dop) 2012-03-29
- latest date by which the national standards conflicting
with the EN have to be withdrawn (dow) 2014-06-29

Annex ZA has been added by CENELEC.

Endorsement notice

The text of the International Standard IEC 62258-2:2011 was approved by CENELEC as a European Standard without any modification.

Annex ZA (normative)

Normative references to international publications with their corresponding European publications

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

NOTE When an international publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
IEC 61360-4	2005	Standard data element types with associated classification scheme for electric components - Part 4: IEC reference collection of standard data element types and component classes	EN 61360-4 + corr. December 2005	2005
IEC 62258-1	-	Semiconductor die products - Part 1: Procurement and use	EN 62258-1	-
ISO 6093	1985	Information processing - Representation of numerical values in character strings for information interchange	-	-
ISO 8601	2004	Data elements and interchange formats - Information interchange - Representation of dates and times	-	-
ISO 10303-21	2002	Industrial automation systems and integration - Product data representation and exchange - Part 21: Implementation methods: Clear text encoding of the exchange structure	-	-
IPC/JEDEC J-STD-033B	2007	Handling, Packing, Shipping and Use of Moisture/Reflow Sensitive Surface Mount Devices	-	-

CONTENTS

INTRODUCTION.....	7
1 Scope and object.....	8
2 Normative references.....	8
3 Terms and definitions	9
4 Requirements	9
5 Device Data eXchange format (DDX) file goals and usage.....	9
6 DDX file format and file format rules	9
6.1 Data validity.....	10
6.2 Character set.....	10
6.3 SYNTAX RULES.....	10
7 DDX file content.....	11
7.1 DDX file content rules.....	11
7.1.1 Block structure.....	11
7.1.2 Parameter types	11
7.1.3 Data types	11
7.1.4 Forward references.....	12
7.1.5 Units.....	12
7.1.6 Co-ordinate data	12
7.1.7 Reserved words	12
7.2 DDX DEVICE block syntax.....	13
7.3 DDX data syntax.....	14
8 Definitions of DEVICE block parameters	14
8.1 BLOCK DATA	15
8.1.1 DEVICE_NAME Parameter	15
8.1.2 DEVICE_FORM Parameter	16
8.1.3 BLOCK_VERSION Parameter	16
8.1.4 BLOCK_CREATION_DATE Parameter	16
8.1.5 VERSION Parameter	16
8.2 DEVICE DATA.....	16
8.2.1 DIE_NAME Parameter	16
8.2.2 DIE_PACKAGED_PART_NAME Parameter	16
8.2.3 DIE_MASK_REVISION Parameter	17
8.2.4 MANUFACTURER Parameter	17
8.2.5 DATA_SOURCE Parameter	17
8.2.6 DATA_VERSION Parameter	17
8.2.7 FUNCTION Parameter	17
8.2.8 IC_TECHNOLOGY Parameter.....	18
8.2.9 DEVICE_PICTURE_FILE Parameter	18
8.2.10 DEVICE_DATA_FILE Parameter	18
8.3 GEOMETRIC DATA.....	19
8.3.1 GEOMETRIC_UNITS Parameter	19
8.3.2 GEOMETRIC_VIEW Parameter	19
8.3.3 GEOMETRIC_ORIGIN Parameter	19
8.3.4 SIZE Parameter	20
8.3.5 SIZE_TOLERANCE Parameter.....	20

8.3.6	THICKNESS Parameter	21
8.3.7	THICKNESS_TOLERANCE Parameter	21
8.3.8	FIDUCIAL_TYPE Parameter	21
8.3.9	FIDUCIAL Parameter	23
8.4	TERMINAL DATA	24
8.4.1	TERMINAL_COUNT Parameter	24
8.4.2	TERMINAL_TYPE_COUNT Parameter	24
8.4.3	CONNECTION_COUNT Parameter	24
8.4.4	TERMINAL_TYPE Parameter	25
8.4.5	TERMINAL Parameter	26
8.4.6	TERMINAL_GROUP Parameter	29
8.4.7	PERMUTABLE Parameter	31
8.5	MATERIAL DATA	32
8.5.1	TERMINAL_MATERIAL Parameter	32
8.5.2	TERMINAL_MATERIAL_STRUCTURE Parameter	32
8.5.3	DIE_SEMICONDUCTOR_MATERIAL Parameter	32
8.5.4	DIE_SUBSTRATE_MATERIAL Parameter	33
8.5.5	DIE_SUBSTRATE_CONNECTION Parameter	33
8.5.6	DIE_PASSIVATION_MATERIAL Parameter	33
8.5.7	DIE_BACK_DETAIL Parameter	34
8.6	ELECTRICAL AND THERMAL RATING DATA	34
8.6.1	MAX_TEMP Parameter	34
8.6.2	MAX_TEMP_TIME Parameter	34
8.6.3	POWER_RANGE Parameter	34
8.6.4	TEMPERATURE_RANGE Parameter	34
8.7	SIMULATION DATA	35
8.7.1	Simulator MODEL FILE Parameter	35
8.7.2	Simulator MODEL FILE DATE Parameter	35
8.7.3	Simulator NAME Parameter	35
8.7.4	Simulator VERSION Parameter	35
8.7.5	Simulator COMPLIANCE Parameter	36
8.7.6	Simulator TERM_GROUP Parameter	36
8.8	HANDLING, PACKING, STORAGE and ASSEMBLY DATA	36
8.8.1	DELIVERY_FORM Parameter	36
8.8.2	PACKING_CODE Parameter	36
8.8.3	ASSEMBLY Parameters	36
8.9	WAFER SPECIFIC DATA	37
8.9.1	WAFER_SIZE Parameter	37
8.9.2	WAFER_THICKNESS Parameter	37
8.9.3	WAFER_THICKNESS_TOLERANCE Parameter	37
8.9.4	WAFER_DIE_STEP_SIZE Parameter	38
8.9.5	WAFER_GROSS_DIE_COUNT Parameter	38
8.9.6	WAFER_INDEX Parameter	38
8.9.7	WAFER_RETICULE_STEP_SIZE Parameter	38
8.9.8	WAFER_RETICULE_GROSS_DIE_COUNT Parameter	39
8.9.9	WAFER_INK Parameters	39
8.10	BUMP TERMINATION SPECIFIC DATA	39
8.10.1	BUMP_MATERIAL Parameter	39

8.10.2	BUMP_HEIGHT Parameter	40
8.10.3	BUMP_HEIGHT_TOLERANCE Parameter	40
8.10.4	BUMP_SHAPE Parameter	40
8.10.5	BUMP_SIZE Parameter	40
8.10.6	BUMP_SPECIFICATION_DRAWING Parameter	41
8.10.7	BUMP_ATTACHMENT_METHOD Parameter	41
8.11	MINIMALLY PACKAGED DEVICE (MPD) SPECIFIC DATA	41
8.11.1	MPD_PACKAGE_MATERIAL Parameter	41
8.11.2	MPD_PACKAGE_STYLE Parameter	41
8.11.3	MPD_CONNECTION_TYPE Parameter	42
8.11.4	MPD_MSL_LEVEL Parameter	42
8.11.5	MPD_PACKAGE_DRAWING Parameter	42
8.12	QUALITY, RELIABILITY and TEST DATA	42
8.12.1	QUALITY Parameters	42
8.12.2	TEST Parameters	43
8.13	OTHER DATA	43
8.13.1	TEXT Parameters	43
8.14	CONTROL DATA	43
8.14.1	PARSE Parameters	43
Annex A (informative)	An example of a DDX DEVICE block	47
Annex B (informative)	Groups and Permutation	49
Annex C (informative)	A Typical CAD view from the DDX file block example given in Annex A	52
Annex D (informative)	Properties for Simulation	53
Annex E (informative)	TERMINAL and TERMINAL_TYPE graphical usage for CAD/CAM systems	55
Annex F (informative)	Cross-reference with IEC 61360-4	58
Annex G (informative)	Notes on VERSION and NAME parameters	61
Annex H (informative)	Notes on WAFER parameters	62
Annex I (informative)	Additional notes	64
Annex J (informative)	DDX Version history	65
Annex K (informative)	Parse Control	68
Figure 1	Relationship between geometric centre and geometric origin	20
Figure C.1	CAD representation of DDX example from Annex A	52
Figure E.1	Highlighting the MX and MY orientation properties	56
Figure E.2	Highlighting the angular rotational orientation properties	57
Figure H.1	Illustrating the WAFER parameters	63
Table 1	Terminal shape types	25
Table 2	Terminal shape co-ordinates	26
Table 3	Terminal IO types	28
Table 4	Substrate Connection Parameters	33
Table F.1	Parameter List	58
Table J.1	Parameter Change History List	65

INTRODUCTION

This International Standard is based on the work carried out in the ESPRIT 4th Framework project GOODDIE which resulted in publication of the ES 59008 series of European specifications. Organisations that helped prepare this document include the ESPRIT ENCAST and ENCASIT projects, the Die Products Consortium, JEITA, JEDEC and ZVEI.

The structure of this International Standard as currently conceived is as follows:

Under main title: IEC 62258: Semiconductor die products

- Part 1: Procurement and use
- Part 2: Exchange data formats
- Part 3: Recommendations for good practice in handling, packing and storage (Technical report)
- Part 4: Questionnaire for die users and suppliers (Technical report)
- Part 5: Requirements for information concerning electrical simulation
- Part 6: Requirements for information concerning thermal simulation
- Part 7: XML schema for data exchange (Technical report)
- Part 8: EXPRESS model schema for data exchange (Technical report)

Further parts may be added as required.

SEMICONDUCTOR DIE PRODUCTS –

Part 2: Exchange data formats

1 Scope and object

This Part of IEC 62258 specifies the data formats that may be used for the exchange of data which is covered by other parts of the IEC 62258 series, as well as definitions of all parameters used according to the principles and methods of IEC 61360. It introduces a Device Data Exchange (DDX) format, with the prime goal of facilitating the transfer of adequate geometric data between die manufacturer and CAD/CAE user and formal information models that allow data exchange in other formats such as STEP physical file format, in accordance with ISO 10303-21, and XML. The data format has been kept intentionally flexible to permit usage beyond this initial scope.

It has been developed to facilitate the production, supply and use of semiconductor die products, including but not limited to:

- wafers,
- singulated bare die,
- die and wafers with attached connection structures,
- minimally or partially encapsulated die and wafers.

This standard reflects the DDX data format at version **1.3.0**

2 Normative references

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

IEC 62258-1, *Semiconductor die products – Part 1: Procurement and use*

IEC 61360-4:2005, *Standard data element types with associated classification scheme for electric components – Part 4: IEC reference collection of standard data element types, component classes* 303-21

ISO 8601:2004, *Data elements and interchange formats – Information interchange – Representation of dates and times*

ISO 6093:1985, *Information processing – Representation of numerical values in character strings for information interchange*

IPC/JEDEC J-STD-033B:2007, *Handling, Packing, Shipping and Use of Moisture/Reflow Sensitive Surface Mount Devices*

ISO 10303-21:2002, *Industrial automation systems and integration – Product data representation and exchange – Part 21: Implementation methods: Clear text encoding of the exchange structure*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62258-1 apply.

4 Requirements

Specific reference for parameter variables is made to the IEC 61360 data element type (DET) codes, which are defined in Part 4 of IEC 61360.

5 Device Data eXchange format (DDX) file goals and usage

5.1 To facilitate the transferral of data by electronic media from the device vendor to the end-user for use within a CAD or CAE system, a data file format, **Device Data eXchange, (DDX)**, shall be used. This data file format has been deliberately kept flexible, to permit further enhancements and additions for future use.

5.2 It is strongly recommended that **Device Data eXchange** files have the three letter **DDX** file extension, and a **Device Data eXchange** file shall hereon be referred to as a **DDX** file.

5.3 Data that are to be transferred from a device vendor to a user shall be contained in a single computer-readable DDX file, and the minimum contents of this file shall suffice a geometric CAD/CAE software design system. The file shall be textually readable, to permit simple manual verification.

5.4 The DDX file and its data contents shall be independent of both computer machine and operating system.

5.5 The DDX file contents shall include mechanical and interconnectivity information, but may additionally include electrical and functional data.

5.6 The DDX file may contain data for one or more devices, and shall be capable of being used as a library file by a CAD/CAE software design system. The file may contain one or more sets of data for the same device type, each having different delivery forms, such as bumped die, bare die, and Chip-Scale packaging.

5.7 The DDX file shall be capable of being simply or automatically generated, such as by an ASCII text editor or a spreadsheet.

5.8 The DDX file shall be capable of referencing additional external files, such as simulation and thermal model files.

5.9 All data shall be defined in such a way that conversion to or from other exchange formats is possible, such as GDSII and CIF for geometric data of die. As close compatibility to the existing DIE (Die Information Exchange) data as possible is desired, to facilitate simple translation of partial DIE data files.

5.10 Definitions of parameters shall be in conformity with IEC 61360 (refer to Clause 5 of IEC 62258-1).

6 DDX file format and file format rules

NOTE 1 Version 1.2.1 of DDX supersedes version 1.0.0 contained in ES 59008-6-1.

NOTE 2 Version 1.3.0 of DDX supersedes version 1.2.1 contained in IEC 62258-2:2005.

Refer to Clause 1 for the DDX version of this standard.

6.1 Data validity

6.1.1 All data not complying with the data syntax (refer to 7.3) shall be treated as a remark and, as such, ignored.

6.1.2 All mandatory data shall be present. Missing data shall be flagged as an error, rendering that data unusable.

6.1.3 Mathematical operations, calculations or formulae shall not be permitted within numeric data.

6.2 Character set

6.2.1 The DDX file shall be an ASCII compatible text file with suitable line termination. Line termination will depend upon the operating system. DOS/Windows[®] generally uses a carriage/line-feed <CR/LF> terminator (ASCII 0Dh/0Ah), whereas UNIX[®] invariably relies solely upon a line-feed <LF> (ASCII 0x0A) terminator, the carriage return <CR> (ASCII 0x0D) being present by implication.

6.2.2 ASCII characters 0x00 to 0x7F are permitted, ASCII characters 0x80 to 0xFF shall be ignored.

6.2.3 All text data shall be case independent.

6.2.4 Space characters (ASCII 20h) and tab characters (ASCII 09h) shall both be treated as space separators, multiple space and tab characters will syntactically be treated as a single space separator.

6.3 SYNTAX RULES

6.3.1 All data lines shall be terminated with a semicolon: “;”.

6.3.2 A comma “,” shall be used as a data separator.

6.3.3 Lines beginning with a hash “#” shall be treated as an intentional comment. All data on that line shall be ignored.

6.3.4 Underscores “_” shall be ignored in a variable or property name, and may be used as intermediate name separators. Underscores are valid within textual string and name data.

6.3.5 Braces are used to open and close structures or BLOCKs. An open brace “{” shall be used to begin a structure or block, and a close brace “}” shall be used to terminate a structure or block.

6.3.6 Brackets “()” shall be permitted, then ignored, in numeric data for clarity (e.g. in co-ordinate pairs).

6.3.7 To accommodate typical spreadsheet CSV (Comma Separated Variable) format outputs, textual data may be inside double quotes “”, and matching pairs of double quotes shall be ignored.

6.3.8 There is no specific line continuation character. A textual string opened with a double quote “” shall close with a matching double quote “”, irrespective of the number of line breaks within that text. As all DDX commands terminate with a semicolon, the non-textual data will be

deemed to have ended at that semicolon. Textual data will be deemed to have ended at the semicolon following the closing double quote. Textual data not enclosed within double quotes may not include line break or control characters, and shall terminate at the first occurrence of a semicolon. Textual data following this semicolon will be treated as erroneous data and discarded.

6.3.9 For practicality, readability and ease of line parsing, is it recommended that the line length (between line termination characters) does not exceed 255 characters. It is further strongly recommended that a maximum limit of 1 023 characters per line be imposed to prevent other parsing software from having an input buffer overrun error.

7 DDX file content

7.1 DDX file content rules

7.1.1 Block structure

Data shall only exist within a block structure, referred to as a DEVICE block, and one or more DEVICE blocks, each containing data, may exist within a single file. Each DEVICE block is unique, and shall only contain data relevant to a single device, having a specific device form. All data within each DEVICE block shall be treated as being local and unique only to that block. (Refer to 6.3.4)

7.1.2 Parameter types

There are two types of parameters use for data, structures and variables, and these parameters shall only exist with a DEVICE block:

- a structure determines a set or multiple sets of data having different data types.
- a variable is equated to a single or multiple data of a single data type.

7.1.3 Data types

Data types are as follows.

7.1.3.1 Textual string data

All ASCII characters from ASCII 20h to ASCII 7Fh are permitted within textual data, characters including and above ASCII 80h shall be ignored. Consideration may be given to special print and display control characters to permit the printing of underscore or overscore characters. It is advised that textual string data is placed within pairs of double quotes, refer to 6.3.7.

7.1.3.2 Textual name data

All names shall be unique, and shall only consist of the following characters from the ASCII character set: -

A-Z a-z 0-9 \$ - % & ! @ _ .

When textual name data are used to form a file name, it is advisable for the name to be limited to eight characters for the file name and to three characters for the file extension, with a point “.” used as the name/extension delimiter, in line with many common operating systems. It is advisable for textual name data to be placed within pairs of double quotes (refer to 6.3.7).

Note that all textual name data is case independent, and spaces are not permitted within a textual name.

7.1.3.3 Real numeric data

Real numeric data shall comply with ISO 6093:1985, and shall consist of the characters: -

0-9 + - . E e

The data values may be signed, and use engineering or scientific notation, but shall not include dimensional units, e.g.

90008, 9000.80, 9.0008E5, -5207, -5.207E3, 0.102, 102E-3

Note that a comma “,” is used as a data separator, and therefore shall not be used as a replacement for a decimal point “.”.

7.1.3.4 Integer numeric data

Integer numeric data values shall comply to ISO 6093:1985, and only the characters **0** to **9** are permitted. Integers shall be unsigned, and shall not include dimensional units.

For practical purposes, an integer shall be limited to 16-bit resolution, i.e. integer values between and including 0 to 65536 only are acceptable.

7.1.3.5 Date data

Date data values shall comply with ISO 8601:2004 format, and may include time information as well, e.g.

“YYYY-MM-DD”, “YYYYMMDD”, “YYYY-MM-DDTHH:MM:SS”.

7.1.4 Forward references

To permit single-pass parsing, no variable identifier or variable name shall be referenced prior to being defined.

7.1.5 Units

All units shall belong to the SI system, apart from the geometric unit of the micron (10^{-6}m), the inch and the mil (10^{-3} inch). Only one unit of dimension shall be permitted within a single **DEVICE** block. Note that the inch and the mil are non-preferred units, and are only present due to continued common usage.

7.1.6 Co-ordinate data

In all co-ordinate data, the **X** co-ordinate shall precede the **Y** co-ordinate and the **Y** co-ordinate shall precede the **Z** co-ordinate (i.e. **X,Y** or **X,Y,Z**).

The **X** co-ordinate shall be the horizontal axis (numerically left to right), the **Y** co-ordinate shall be the vertical axis (numerically bottom to top), and the **Z** co-ordinate shall be depth axis (numerically near to far).

7.1.7 Reserved words

All parameter names shall be considered as reserved and no variable identifier or variable name shall be permitted to have the same name. This restriction does not apply to free form textual data within quotes or double quotes (as 7.1.3.1 and 7.1.3.2).

7.2 DDX DEVICE block syntax

```
DEVICE device_name device_form {  
    relevant die data .....  
}
```

The **DDX** file may contain one or more **DEVICE** blocks, all data pertaining to a particular device shall be embedded within the relevant block. (Refer to clause 6.1.1 and clause 7.1.1).

A **DEVICE** block is opened by the **DEVICE** keyword and opening brace "{", (as shown), and the **DEVICE** block is closed by the matching closing brace "}".

Data not within a **DEVICE** block structure shall be treated as a remark, permitting the future addition of checksum information, file creation date and historical data etc., within the **DDX** file, without affecting the actual device data.

The **device_name** is the given name by which the device shall be referred, and the **device_form** is the mechanical form of the device to which the block data pertains.

Valid data for the **device_form** variable are:

- **bare_die**,
- **bumped_die**,
- **lead_frame_die**
- **minimally_packaged_device** (or **MPD**).

Further **device_form** types may be added at a later stage, refer to IEC 61360-4:2005, AAD004-001, "die type code", for further details.

Only one **DEVICE** block having **device_name** of type **device_form** shall be present within the **DDX** file, but duplication of either **device_name** or **device_form** is permissible.

An example of a typical **DDX** file arrangement of **DEVICE** blocks:-

```
DEVICE name1 bare_die {  
    relevant data for device "name1" as a bare die..  
}  
DEVICE name1 bumped_die {  
    relevant data for device "name1" as a bumped die..  
}  
DEVICE name2 mpd {  
    relevant data for device "name2" as a minimally packaged device..  
}  
DEVICE name2 bare_die {  
    relevant data for device "name2" as a bare die..  
}  
DEVICE name1 mpd {  
    relevant data for device "name1" as a minimally packaged device..  
}  
DEVICE name3 bare_die {  
    relevant data for device "name3" as a bare die..  
}
```

In the above example, there are three occurrences of a **DEVICE** block for device "name1", and two occurrences of a **DEVICE** block for device "name2", but each of these **DEVICE** blocks specify a different **device_form**. The order or sequencing of the **DEVICE** blocks has no relevance.

7.3 DDX data syntax

Property = value [, value];

<property>*[equate separator]***<value/variable {separator <value/variable> }>***[data terminator]**[line terminator]*

<property>	::=	Parameter name
<i>[space]</i>	::=	{space character (20h) or tab character (09h)}0+
<i>[equate separator]</i>	::=	<i>[space]</i> {equal =} <i>[space]</i>
<i>[separator]</i>	::=	<i>[space]</i> {comma ,} <i>[space]</i>
<i>[data terminator]</i>	::=	<i>[space]</i> {semicolon;}
<i>[line terminator]</i>	::=	{CR or CR/LF}

For example:

```
thickness      =    100.0 ;
Thickness=470;
geometric_units=micron;
geometricunits      =    micron;
GeometricUnits= "millimetres";
terminal_type = T1, Circle, 220;
Terminal_Type = T2, Rectangle, 200 , 250;
TerminalType = T2, O, (200, 250);
TERMINALTYPE      =    T2, O, 200 , 250;
```

Thus, **terminal_type**, **Terminal_Type**, **TerminalType** and **TERMINALTYPE** will all reference the same parameter name.

8 Definitions of DEVICE block parameters

8.0 General usage notes

8.0.1 Device form notes

Where a parameter is unique to the **device_form**, as defined in the **DEVICE** block, the parameter will be preceded with the following ...

- 8.0.1.1 **DIE_** data parameter is unique to bare die or bumped die form
- 8.0.1.2 **BUMP_** data parameter is unique to only bumped die
- 8.0.1.3 **MPD_** data parameter is unique to a minimally packaged device, such as a CSP
- 8.0.1.4 **WAFER_** data parameter is unique to a die device delivered at wafer level
- 8.0.1.5 **LEAD_** data parameter is unique to a die device with attached lead frame.

8.0.2 Data Parameter Items

Within the following list of data parameters (see 8.1 onwards), the following items are shown:

- 8.0.2.1 the parameter name, as used syntactically within the **DDX** file,
- 8.0.2.2 the parameter type, indicating either a variable or structure data type,
- 8.0.2.3 the parameter function, determining its usage and meaning,
- 8.0.2.4 the parameter value, indicating the type of data expected,
- 8.0.2.5 any parameter limitation, indicating any limitation within the **DEVICE** block,
- 8.0.2.6 parameter dependencies, highlighting parameters that need to be declared prior to invocation,
- 8.0.2.7 one or more practical examples, and

8.0.2.8 any relevant notes.

A brief table of parameters is given in Annex F, and a working example of a full **DDX DEVICE** block is given in Annex A, with its expected graphical output in Annex C.

All parameters shall conform to the relevant IEC 61360 Data Element Type (DET) codes, as defined in IEC 61360-4:2005. Refer to Annex F for a cross-reference table.

8.0.3 Terms and conventions

A point of electrical connection is called a terminal. This may be a bond-pad for a bare die, and may equally refer to the landing or connection footprint area required by an interconnection medium. It is a common convention for die to have the initial terminal, numbered 1, in the upper left hand corner of the die, and for terminal or pin numbering to continue counter-clockwise in sequence.

The X co-ordinate dimensions are for the length in the horizontal plane with increasing positive values to the right. The Y co-ordinate dimensions are for the width in the horizontal plane with increasing positive values away from the user's view. The Z co-ordinate dimensions are for height in the vertical direction with increasing positive values upwards (towards the viewer).

As a point of reference, all die components, including bumped die, are generally viewed from above with the active side upwards.

8.0.4 Summary of general rules

- 8.0.4.1 All valid data shall be contained within a **DEVICE** block (refer to 7.1.1).
- 8.0.4.2 Any local or unique parameter, such as a name, shall be defined prior to its usage.
- 8.0.4.3 All parameters shall conform to the relevant IEC 61360 DET codes, as defined in Part 4 of IEC 61360 (refer to 5.8 and Annex F).
- 8.0.4.4 The units of measurement, **GEOMETRIC_UNITS**, shall be defined before any geometric variable is defined.
- 8.0.4.5 The geometric origin, **GEOMETRIC_ORIGIN**, and the geometric view, **GEOMETRIC_VIEW**, shall be defined before any geometric co-ordinates are defined.
- 8.0.4.6 The **TERMINAL_COUNT** parameter shall be defined before any **TERMINAL** parameters are referred to and the number of **TERMINAL** parameters shall not exceed the **TERMINAL_COUNT** value.
- 8.0.4.7 The **TERMINAL_TYPE_COUNT** parameter shall be defined before any **TERMINAL_TYPE** parameters are referred to and the number of **TERMINAL_TYPE** parameters shall not exceed the **TERMINAL_TYPE_COUNT** value.

8.1 BLOCK DATA

8.1.1 **DEVICE_NAME** Parameter

This is defined within the **DEVICE** block heading as the **device_name** parameter

Parameter Name	device_name
Parameter Type	Variable, refer to 7.2 on DEVICE blocks
Parameter Function	Defines the device manufacturer's type number or the reference name.
Parameter Values	Textual name data, as 7.1.3.2
Example	SN74LS04
Reference	See 7.1.7, 7.2 and Annex G.

8.1.2 DEVICE_FORM Parameter

This is defined within the **DEVICE** block heading by the **device_form** parameter.

Parameter Name	device_form
Parameter Type	Variable, refer to 7.2 on DEVICE blocks
Parameter Function	Defines the physical form of the device.
Parameter Values	Textual string data, as 7.1.3.1
Example	<code>bare_die, bumped_die, MPD</code>
Reference	Subclause 7.2 and Annex G.

8.1.3 BLOCK_VERSION Parameter

Parameter Name	BLOCK_VERSION
Parameter Type	Variable
Parameter Function	Specifies the version number and/or issue number of the DEVICE block.
Parameter Values	Textual string data, as 7.1.3.1
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>BLOCK_VERSION = "1.0A";</code>
Reference	Annex G.

8.1.4 BLOCK_CREATION_DATE Parameter

Parameter Name	BLOCK_CREATION_DATE
Parameter Type	Variable
Parameter Function	Specifies the date that the DEVICE block was created and last edited.
Parameter Values	Date, as 7.1.3.5
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>BLOCK_CREATION_DATE = "1997-12-25";</code>
Reference	Annex G.

8.1.5 VERSION Parameter

Parameter Name	VERSION
Parameter Type	Variable
Parameter Function	Specifies the revision/version number of the DDX standard, (currently at version 1.3.0), to which this DEVICE block conforms.
Parameter Values	Textual string data, as 7.1.3.1
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>VERSION = "1.3.0";</code>
Notes	Refer to Clause 1 for the version of this standard document. Note that this document may not be the latest version, so refer to your Standards Authority if in doubt.
Reference	Annex G.

8.2 DEVICE DATA

8.2.1 DIE_NAME Parameter

Parameter Name	DIE_NAME
Parameter Type	Variable
Parameter Function	Specifies the name of the die or mask set from which the die was produced. This may be different to the DEVICE_NAME parameter.
Parameter Values	Textual string data, as 7.1.3.1
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>DIE_NAME = "XXC345";</code>
Reference	Annex G.

8.2.2 DIE_PACKAGED_PART_NAME Parameter

Parameter Name	DIE_PACKAGED_PART_NAME
----------------	-------------------------------

Parameter Type	Variable
Parameter Function	Specifies the manufacturers part name for the equivalent packaged part, where available or applicable. This may be different to the <code>DEVICE_NAME</code> parameter.
Parameter Values	Textual string data, as 7.1.3.1
Example	<code>DIE_PACKAGED_PART_NAME = "SN5405JN";</code>
Notes	Used to reference the identical packaged die part, not merely similar function, when supplied by the same manufacturer in packaged form.

8.2.3 DIE_MASK_REVISION Parameter

Parameter Name	DIE_MASK_REVISION
Parameter Type	Variable
Parameter Function	Specifies the mask revision details associated with that particular version on the die.
Parameter Values	Textual string data, as 7.1.3.1
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>DIE_MASK_REVISION = "RTDAC1";</code> <code>DIE_MASK_REVISION = "9033-2-101-M5/2";</code>
Notes	Intended to ensure that the die data matches the geometric version of the actual die.
Reference	Annex G.

8.2.4 MANUFACTURER Parameter

Parameter Name	MANUFACTURER
Parameter Type	Variable
Parameter Function	Specifies the manufacturer or fabrication house of the device.
Parameter Values	Textual string data, as 7.1.3.1
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>MANUFACTURER = "Fuzziwuz Logic Inc.";</code>

8.2.5 DATA_SOURCE Parameter

Parameter Name	DATA_SOURCE
Parameter Type	Variable
Parameter Function	Specifies the source of the device data, essentially where this is different from the manufacturer or fabrication house.
Parameter Values	Textual string data, as 7.1.3.1
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>DATA_SOURCE = "AnyChip Technology Ltd.";</code> <code>DATA_SOURCE = "Good-Die database";</code>

8.2.6 DATA_VERSION Parameter

Parameter Name	DATA_VERSION
Parameter Type	Variable
Parameter Function	Specifies the revision of the data source use to determine the parameters within the DEVICE block. This parameter is linked to 8.2.5, the <code>DATA_SOURCE</code> parameter.
Parameter Values	Textual string data, as 7.1.3.1
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>DATA_VERSION = "Initial Issue 1.0";</code>
Reference	Refer to Annex G.

8.2.7 FUNCTION Parameter

Parameter Name	FUNCTION
Parameter Type	Variable
Parameter Function	A brief description of the devices' function.
Parameter Values	Textual string data, as 7.1.3.1

Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>FUNCTION = "16 Bit Microprocessor";</code>
Notes	IEC 61360-4:2005 specifies certain functional and application classes that may be used.

8.2.8 IC_TECHNOLOGY Parameter

Parameter Name	IC_TECHNOLOGY
Parameter Type	Variable
Parameter Function	Specifies the fabrication technology of the device.
Parameter Values	Textual string data, as 7.1.3.1
Limitations	Shall be declared only once within a single DEVICE block.
Example	<pre>IC_TECHNOLOGY = "CMOS"; IC_TECHNOLOGY = "bipolar"; IC_TECHNOLOGY = "bicmos"; IC_TECHNOLOGY = "GaAs";</pre>

8.2.9 DEVICE_PICTURE_FILE Parameter

Parameter Name	DEVICE_PICTURE_FILE
Parameter Type	Variable
Parameter Function	Specifies the name(s) of the graphics or picture file(s) representing the device.
Parameter Values	Textual name data, as 7.1.3.2
Example	<pre>DEVICE_PICTURE_FILE = "DIE001.JPG"; DEVICE_PICTURE_FILE = "AA0B0C.SF", "FRED.GIF";</pre>
Notes	The picture file may be an actual photograph, or a pictorial representation of the device. The file extent should indicate the file format used. Only file names, without relative or absolute path names, shall be used. Multiple parameters (picture file names) may be introduced within a single declaration, or multiple declarations maybe be employed to the same effect. There is no scaling or positional data requirement.
Reference	Annex I, Notes 3 and 4.

8.2.10 DEVICE_DATA_FILE Parameter

Parameter Name	DEVICE_DATA_FILE
Parameter Type	Variable
Parameter Function	Specifies the name(s) of a file(s) containing pertinent data, such as technical specifications, procurement documents or general data-sheets.
Parameter Values	Textual name data, as 7.1.3.2
Example	<pre>DEVICE_DATA_FILE = "SpecSheet.PDF"; DEVICE_DATA_FILE = "AA0B0C.DOC", "AA0B0C.TXT";</pre>
Notes	The data file may be of any recognised format. The file extent should indicate the file format used. Only file names, without relative or absolute path names, shall be used. Multiple parameters (data file names) may be introduced within a single declaration, or multiple declarations maybe be employed to the same effect.
Reference	Annex I, Notes 3 and 4.

8.3 GEOMETRIC DATA

8.3.1 GEOMETRIC_UNITS Parameter

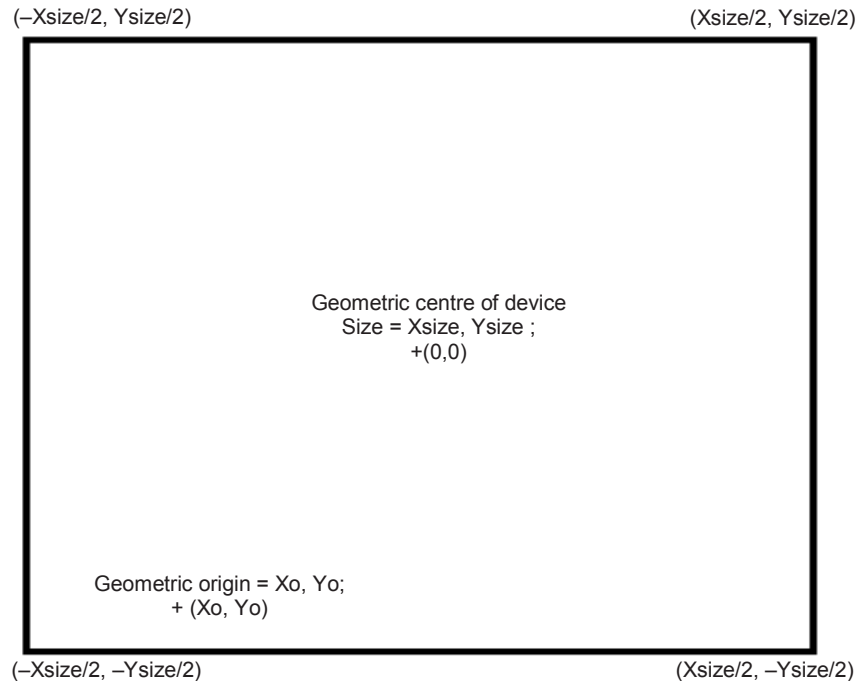
Parameter Name	GEOMETRIC_UNITS
Parameter Type	Variable
Parameter Function	Specifies the geometric units that shall apply to all geometric values within the DEVICE block.
Parameter Values	Textual string data, as 7.1.3.1. One of the following values: ▪ micrometre, or micron, ▪ metre, ▪ millimetre, ▪ inch ▪ mil (1.0E-3 inch)
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>GEOMETRIC_UNITS = microns;</code> <code>GEOMETRIC_UNITS = mil;</code>
Notes	This GEOMETRIC_UNITS parameter shall be declared before any geometric units are used. The micron is generally the default dimensional unit for die dimensions, and the inch and mil are non-preferred units.
Reference	Subclause 7.1.5 and Annex E.

8.3.2 GEOMETRIC_VIEW Parameter

Parameter Name	GEOMETRIC_VIEW
Parameter Type	Variable
Parameter Function	Specifies the geometric view that shall apply to all geometric shapes within the DEVICE block.
Parameter Values	Textual string data, as 7.1.3.1. One of the following values: • TOP meaning active side upwards, and • BOTTOM meaning active side downwards.
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>GEOMETRIC_VIEW = top;</code> <code>GEOMETRIC_VIEW = "bottom";</code>
Notes	The GEOMETRIC_VIEW parameter shall be declared before any geometric shapes are created. It would be common for a bare die and packaged part to be viewed in the "TOP" view, whereas bumped die may well be viewed from the "BOTTOM" (i.e. through the substrate).
Reference	Annex E.

8.3.3 GEOMETRIC_ORIGIN Parameter

Parameter Name	GEOMETRIC_ORIGIN
Parameter Type	Variable
Parameter Function	Determines the X- and Y-geometric origin from which all other co-ordinate pairs are referenced. The origin is given with respect to the geometric centre of the die in the units specified by the GEOMETRIC_UNITS parameter.
Parameter Values	Real X co-ordinate origin, Real Y co-ordinate origin, as 7.1.3.3
Dependencies	GEOMETRIC_UNITS, SIZE
Limitations	Shall be declared only once within a single DEVICE block.
Example	<code>GEOMETRIC_ORIGIN = -6000,-7500;</code>
Notes	Dimension values are in GEOMETRIC_UNITS. The origin co-ordinate pair values relate to the geometric die centre, refer to Figure 1. for further explanation.
Reference	Annex E.



IEC 895/11

Figure 1 – Relationship between geometric centre and geometric origin

The GEOMETRIC_ORIGIN is treated as an offset for all co-ordinate data, so that the GEOMETRIC_ORIGIN values are **added** to **all** individual co-ordinate data pairs to give the X- and Y- position relative to the geometric centre of the device.

The primary use of the GEOMETRIC_ORIGIN parameter is to permit the centring of the origin on a terminal or other geometric feature, rather than an arbitrary geometric position, as, in practice, SIZE may be subject to an asymmetric tolerance so that all related references to this geometric centre could therefore also be subject to a tolerance error.

8.3.4 SIZE Parameter

Parameter Name	SIZE
Parameter Type	Variable
Parameter Function	Determines the X- and Y- dimensions of the device, and optionally specifies its shape as an ellipse.
Parameter Values	Real X-dimension, Real Y-dimension, (as 7.1.3.3), { <u>E</u> llipse}
Dependencies	GEOMETRIC_UNITS, GEOMETRIC_VIEW
Limitations	Shall be declared only once within a single DEVICE block.
Example	<pre>SIZE = 250, 500.5; SIZE = 350, 350, E;</pre>
Notes	Dimension values are in GEOMETRIC_UNITS. In the latter example, the character “E” as the 3rd parameter defines an ellipse, in this instance a circular die of 350 GEOMETRIC_UNITS in diameter.
Reference	Annex E.

8.3.5 SIZE_TOLERANCE Parameter

Parameter Name	SIZE_TOLERANCE
Parameter Type	Variable
Parameter Function	Specify the geometric tolerance(s) of the SIZE parameter values.
Parameter Values	Real in GEOMETRIC_UNITS, as 7.1.3.3