IC Manufacturer to MCM Designer

Die Information Exchange (DIE) Format Reference Manual

Version 1.0

8 April, 1994

Please check the *<u>Notes for Reviewers</u>* *after the Table of Contents*

DIE Format Industry Group

(for more information, email a request to die-info@VHDL.org)

Prepared by:

Randolph E. Harr
Larry Saunders
Siuki Chan

Logic Modeling Group, Synopsys, Inc.

This document represents work in progress under funding from the ARPA/ESTO ASEM program under contract MDA972-93-C-0002 (nCHIP subcontract 240-01). Its distribution is made purely for the purposes of comment and industry feedback during the development phase of the contract.

Some of the drawings herein are based on original electronic work by MCC as part of the ASEM CAx Alliance.

Send all comments back to:

Randolph E. Harr
Logic Modeling Group, Synopsys Inc.
1520 McCandless Drive
Milpitas, CA 95035
408.945.9181 (fax)
randyh@lmc.com (email)
408.957.5241 (voice, for questions)

Table of Contents

Notes for Reviewers and Developers

reh, 8 April, 1994 (Version 1.0)

Italicized sections in this document represent notes about unclear areas or discussions still not finalized. They should not be considered part of the final specification.

This document is being released as a major release (1.0) to provide a stable platform from which to implement and test the concepts. It is expected a minor update release will occur fairly quickly to clean up any deficiencies discovered in use. Another major release is planned before turning the document over to a formal, standards body at the end of 1994.

Passives, discretes, and Analog signal pin electrical characteristics are not included in this draft.

Some semiconductor technologies such as GaAs, ECL or BiCMOS, as well as some bonding styles (Beam Lead and Solder Ball) may not be fully covered yet.

MCM Substrate test development support and timing information are not yet included.

This document and related information are available from the VHDL International Internet Services machine. Public access and download of files can occur in many ways. Each is described next:

***Email access**:*
There is an email FTP archive server on the machine. Send an email message to archive@VHDL.org. The subject is ignored. If a line in the body of the message begins with "help", then a descriptive help file of commands is sent. Basically, you communicate to the server through commands in the mail message body. It then responds to your commands via email. You should always add the command "path <your_email_address>" to any message to assure the return address is understood.

The following example assumes you have initiated a mail message to archive@VHDL.org. The body of the email message should be:.

path randyh@lmc.com	*Email address to send results to*
send pub die/die1-0/die1-0.ps	*To ask for a postscript file of this document*
send pub die/die1-0/die1-0.ps.Z	*To ask for the UNIX compressed version*
send pub die/die1-0/die1-0.exe	*To ask for the MS-DOS ZIP'ed version*

***Dial-Up access**:*
Dial-up the VHDL.org system at (408) 945-4170. Any baud (up to 14,400), parity, start & stop bits, and v. settings will do. Login using the "guest" account. Once in, simple UNIX commands such as "cd pub/die", "ls" and "cat" are available. You can download any desired files using "kermit", "xmodem" or "sz" (zmodem).*

Internet access:
Use "ftp VHDL.org" (or "ftp 198.31.14.3") and login as user "anonymous". Remember to set "binary" mode for any binary files you may select. Also, gopher is available and highly recommended if you have it available.

1. Introduction

The DIE Format is a human and computer sensible interchange format for information about unpackaged Integrated Circuits (IC's). The format can convey information about die devices (components) which are used by MCM designers and foundries. The information is intended for direct use in EDA and other computer-based software tools. Human readability has been maintained to aid in the verification and use of the information independent of the available tools. It is expected that the information in this format will be created and distributed by either the die manufacturer or a die broker.

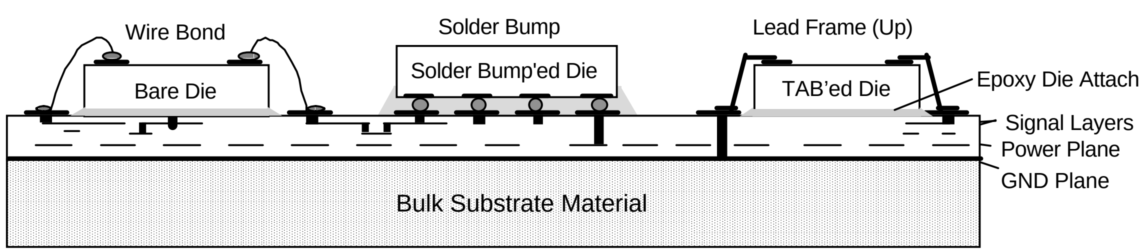
The DIE Format conveys the physical and functional characteristics of an unpackaged die. That is, those characteristics needed for place & route, thermal analysis, electrical signal analysis, power distribution design, physical bonding, behavior, test, and timing analysis. Existing formats are referenced and used where beneficial to avoid creating another standard for the same information.

Use of the DIE format is not intended to be a replacement for a data sheet nor to represent all the information needed to understand a die. The focus is on conveying that information which can be directly processed and is either time consuming to collect or error prone to enter into the tools. In limited cases, information important to the end user but not computer sensible has been included in the format to facilitate understanding and use of the data.

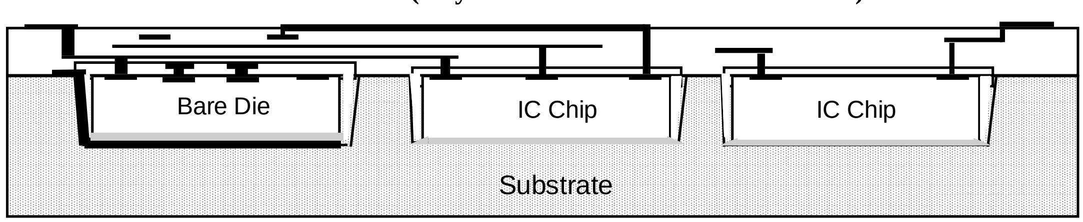
Information that is time consuming to manually collect, difficult to enter, or not generally available has a priority for inclusion. Information about bare die generally falls into this category.

An important aspect of the format are the levels of compliance. In some cases, it may be difficult for an information provider to gather or generally release all of the information needed for a given device. Therefore, the format categorizes information into one of three levels of compliance. Each higher level provides more detail about the device with level 0 being the basic, rudimentary data.

Die specifically covered by the format are pre-diced die (wafer form), bare die, and die that have been post-processed for pad attachment mechanisms such as solder bump**,** wire bond, lead frame (TAB and Ribbon)**,** and chips first. These various forms of die and their use in an MCM are shown in figure 1.



MCM-D Shown (only substrate different from MCM-C)



Chips First (High Density Interconnect) Shown

Figure 1: Typical MCM Die Attach and Pad Bonding Mechanisms

The DIE Format is intended to be comprehensive, complete and concise. This is a difficult task given the rapid change of technology. None the less, any use of the DIE Format should strictly adhere to the guidelines, or their intent, as described herein. A failure to do so violates the specification and is considered a non-conforming use of the format.

1.1. Background

This section is not part of the DIE Format Reference Manual specification and is supplied for information purposes only.

This document is the result of a study and series of workshops into the design practices and technologies of MCM and IC manufacturers, designers and EDA tools. It represents the data requirements that have been extracted from key companies and people in the industry. The requirements were used as the basis to form an interchange specification for die library information. See [REQUIRE] , [WORKSHOP1], [WORKSHOP2] and [WORKSHOP3] for more details on the decision process behind the generation of this format.

This format represents an interim solution to exchanging unpackaged device information. The primary focus in this version is on bare die digital IC's. The format is eventually meant to include all forms of unpackaged devices used in an MCM process **--** Analog ICs, discrete semiconductors, and passive devices. Once done, the Die Information Exchange format will become the Device Information Exchange format and be expanded in scope.

The resulting information model crudely defined by this document will be submitted for use to further refine more comprehensive standards. Specifically, it is expected to help drive the EIA EDIF [EIA548], CFI CIR Electronic Data Book (syntactically represented using [SGML8879]), and Standard Technique for the Exchange of Product Data (STEP) [ISO10303] emerging standards. A final standard in this area is not expected to be ratified before 1996, thus requiring this interim solution.

Whatever the final standard, the interim format is kept simple and human readable so as to make tool development simple and verification of content easy. Use of the standard without processing tools should be easy. Conversion to the final standard that develops (upward compatibility) will be a goal in development. IC manufacturers who start delivering to the format today are therefore guaranteed a long commitment into the future.

Consideration has been given to generating information from existing formats such as a GDS II file. This, along with the human readability of the format, should make it easy for IC manufacturers to create the necessary information about their die.

EDA vendors are committed to adopting the format as it exists today which further enhances the usefulness of this solution.

Before presenting the format in the following chapters, a basic model about the physical die and the document conventions for describing the DIE Format syntax are presented.

1.2. Basic Model

Any format or language can only describe a subset or abstraction of the real world. The basic model is intended to convey how the abstract model represented in the DIE Format maps to the real world the die exists in. Simplifying assumptions are given to aid in this understanding.

The format today is focused on unpackaged, Integrated Circuit devices, or die as it is termed here. Post-processed die which include a TAB lead frame, ribbon leads, or solder bumps are considered special, modified forms of a bare die. These special forms of unpackaged die use the basic bare die information for a majority of the detail and have optional, additional settings to describe the specific differences. A special setting exists at the beginning of any die description to indicate which type of die is being described.

From here on out, when the term <u>die</u> is used unannotated in this document, it implies <u>unpackaged die</u>. All form of die (such as bare die, flip die, TAB die, etc.) are included. The specific term for the type of die which contains no solder balls or electrical contact wires is termed a <u>bare die</u>. When considering unpackaged die along with discretes and passives as a whole, the term <u>device</u> is used.

Many times information is common for many die or many bond pads on the die. In these cases, the format allows for the separate definition of the common information and then the instancing of the definition with any necessary "local" customization given at that time. This is especially prevalent in describing the properties of the electrical connect pads.

1.2.1. Datum and Coordinate Systems

Geometrical figures (or objects) are defined in a two-dimensional, Euclidean coordinate view plane. The datum (base geometrical figure) is formed by an orthographic parallel projection of an object to the view plane. A view plane is parallel to the plane formed by either the die active surface or that formed by the MCM substrate when a device is mounted on it. All other properties of the object are described relative to this orientation. This is abstractly shown for a typical die in figure 2 below.

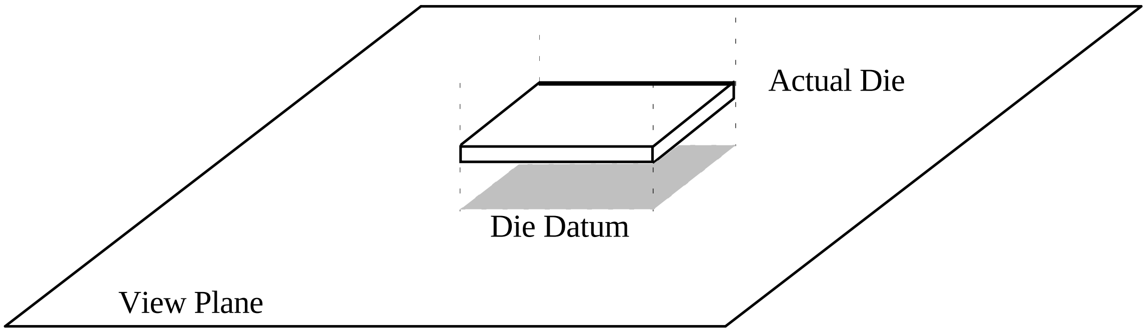


Figure 2: Die View Plane and Datum

An objects origin in the coordinate system is defined as the center of the smallest rectangle which bounds the datum on the view plane and is parallel to the view plane.

The orientation of the object (rotation about the Z axis) which defines the X and Y coordinate directions is arbitrary but must be consistently applied.

Note:

> The pad bonding diagram (see setting **die_bonding_diagram**) is usually a useful visual aid to establishing the orientation of a die in the coordinate system. An example bonding diagram for a die (without pad names or ID's) is shown in figure 3.

An objects X direction size is termed the width. The Y direction size is termed the height.

The die is considered to be composed of a stack of material layers. The material layers can each be described with a thickness. The die itself can be described with a total thickness.

The thickness is a dimension and does not define a relative displacement from an origin. No coordinate system origin in the Z direction is defined. The Z direction does not really exist as we are only defining a two dimensional, coordinate world.

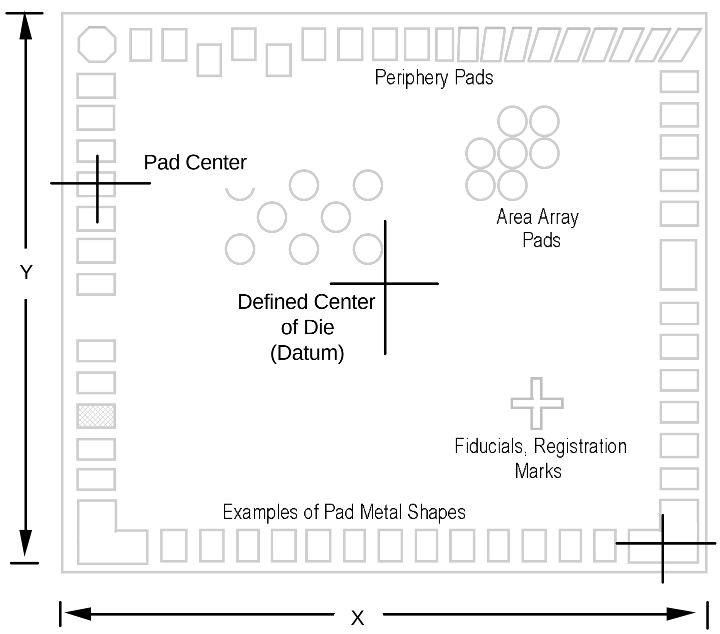


Figure 3: Die and Pad Datum (centers), bonding diagram

These concepts are highlighted in figure 4.

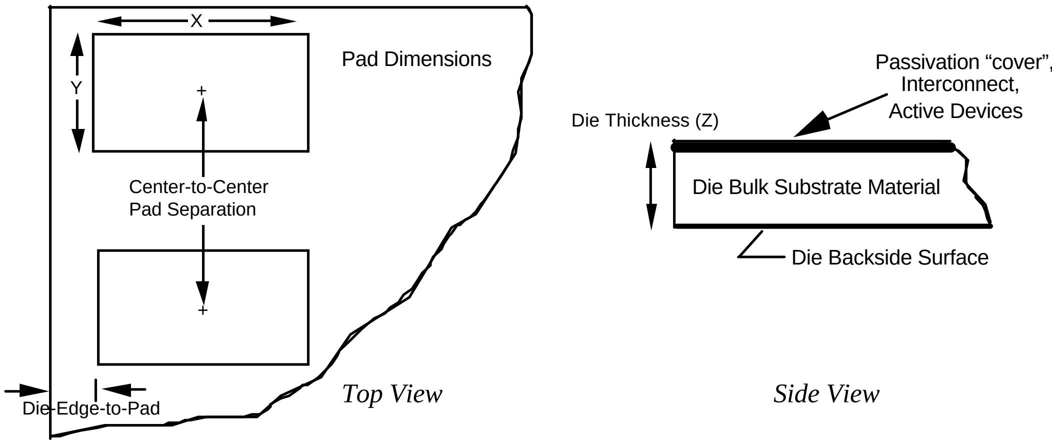


Figure 4: Top and Side Die Views

1.2.2. Die Material Layers

The third dimension of the die is described using a material layer stack form. The layers do not exist in any defined coordinate space with an origin. Each layer is considered existing in a "parallel" plane to the view plane. These parallel view planes are then considered stacked or ordered. A thickness dimension may be associated with any given layer.

The die layer orientation is strictly defined. The outermost surface of the die closest to the electrically active layers of material defines the "top" of the die. This "top" layer typically consists of an electrically passive material coating. This coating seals the active components of the die from the environment and is termed the "passivation layer".

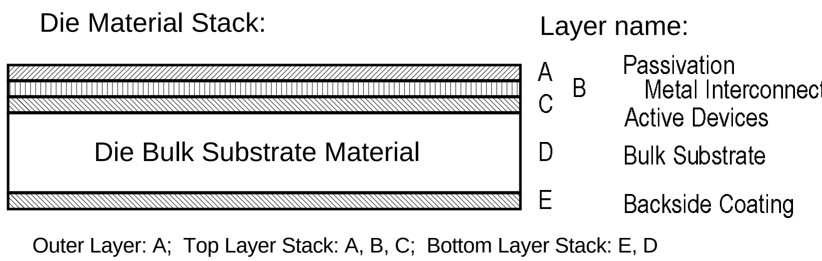


Figure 5: Simplified Die Material Layer Stack

There are usually thin film, metal interconnect layers with intervening electrical isolation type material between the outer passivation layer and the die active layers. These metal layers typically function as signal or power distribution layers.

The active layer which exists below these metal layers is slightly above and diffused into the bulk substrate material of the die.

The layers below the active layer are typically comprised of the bulk substrate material and the die backside surface. The bulk substrate material usually contributes to a the majority of a die's thickness.

The amount of detail given on the material layering in the die depends on the compliance level of the block of information.

When a layer stack is to be described, a consistent method of description is required. The required convention is that the layer furthest from the bulk material (and closest to either outside surface) is described first in a multiple layer description. Further layer specifications of a layer stack are given, in order, as one progresses into the material from this outer layer. Figure 5 diagrams this convention.

1.2.3. Bond Pads

A <u>pad</u> is defined as any exposed metal (not covered by the passivation material) of a possible connect point on the top surface of the device. For most die, this represents a passivation opening over a large, metal pad. For some die, the passivation opening may expose other materials besides the metallization. In this case, only the exposed metal (not the complete passivation opening) constitutes the pad.

For solder bump die with area array pads, the top layer of metal comprising the "pad" is usually on top of the protective, passivation layer. The pad in this case consists of all the exposed metal. A passivation opening still exists -- it is just smaller than the pad metal and provides the "contact cut" down to the protected metallization interconnect layer.

Solder bump pads that are provided for die stabilization or heat conducting purposes only would have their pad type defined as NO_CONNECT. A pad does not have to be an electrical connect point.

For solder bumped die, the solder ball characteristics are described in the **die_solder_bump** setting. This setting and the **die_pads** setting are required for die types of SOLDER_BUMP.

The various pad styles are diagrammed in figure 6.

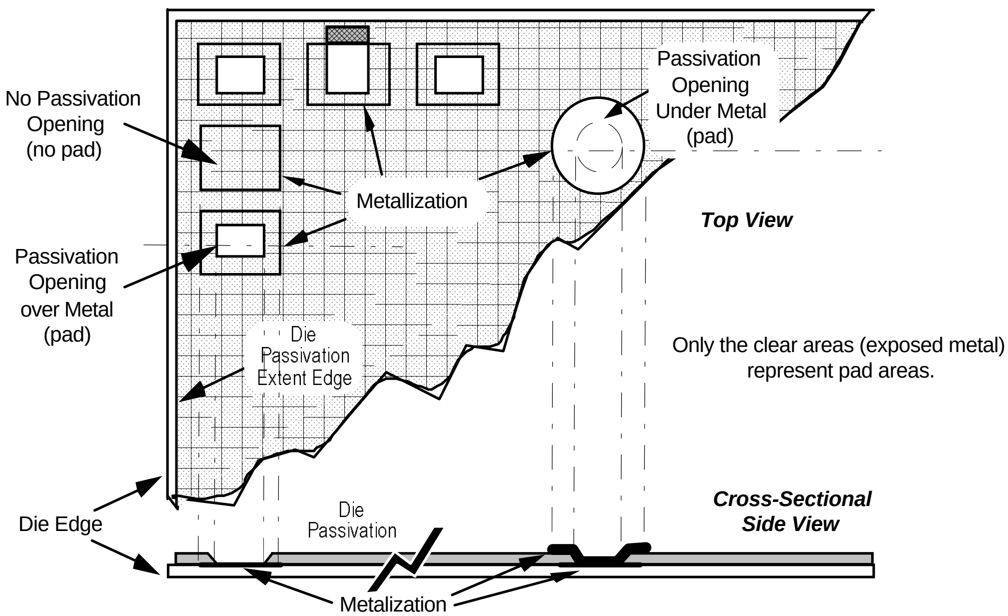


Figure 6: Various Pad Specifications and styles for bare die

For lead frame die, the pads described in the **die_pads** setting are describing the pad areas on the die itself. These are not the connect points to the lead frame -- sometimes termed the Outer Lead Bond (OLB)[1].

For ribbon lead frame die which are to be mounted face down (flip), the OLB may be interior to the lead and actually on the connecting tape portion. This is due to the cutting off or shortening of the leads during bonding.

The **die_lead_frame** setting is used to describe the lead frame characteristics, if they exist.

1.2.4. Tolerances and Accuracy

A physical dimension or a coordinate point has both a basic dimension value and a tolerance. The <u>basic dimension</u> is, by definition, a numerical value used to describe the theoretical size of an object. It is the basis (or datum) from which the tolerances and accuracy are defined.

A <u>tolerance</u> specifies the expected deviation from a given basic dimension for the physical structure being described -- either a maximum, minimum or both. The tolerance represents the deviation in the physical object from the datum.

---

[1] The lead frame is generally visualized as being divided into three different sections -- the Inner Lead Bond (ILB), the connecting tape, and the OLB. In general, only the OLB is of concern to a designer as long as the electrical characteristics of the lead frame have been captured in the die electrical description.

The tolerance for a physical dimension or a coordinate point is usually given by a separate setting **--** usually one with a higher level of compliance. If a tolerance value is not defined in a section or for a given level of compliance, then the tolerance is unknown.

Each numeric value has a precision and an accuracy. The precision is represented by the number of numeric digits used to represent the value. The accuracy is, by definition, plus or minus one half the value represented by the least, non-zero digit.

For example, if the last non-zero digit is in the 10^{-3} digit position, then the accuracy is ± .0005. Note that an accuracy of other than 1/2 a digit cannot be specified (although a tolerance for a dimension can be more specific).

When creating a block in the DIE Format, the appropriate precision to imply an accuracy that is close to the intended value should be used for all values.

The accuracy represents the deviation possible in the numeric value due to measurement, computation or other forms of introduced errors. The accuracy does not represent the tolerance of a value in representing a physical item.

1.2.5. Thermal Analysis Support

The goal of thermal analysis is to determine if the die, when used in the MCM, will operate within the thermal constraint recommended by the die manufacturer. The constraint is usually specified as the Operating Junction Temperature and used in steady state thermal analysis.

The assumption here is that the major thermal transfer from the die active region (source of dissipated energy) occurs by conduction through the die bulk substrate and die attach. A minor, but possibly significant, amount of thermal conduction can occur through the pad attach mechanisms.

The information about a die that is provided to aid in the analysis is the thermal conductivity of the die bulk substrate material, the geometry of the die and the die power dissipation. For a more accurate analysis, the pad and bond type information and die backside finish are required. The most critical of all the parameters is the power dissipation.

Die are very different in circuit technology, complexity, function, size, speed and application. Some die technologies have a very constant power dissipation which is independent of the switching speeds and circuit activity. This is very typical of bipolar technologies which have a constant current flow through the devices. Other technologies, such as those based on MOS switches, only consume significant current during switching or transition times. The current consumption is very dependent on the circuit activity. For these technologies, a single, accurate power dissipation formula is difficult to develop.

At minimum, the DIE Format information will contain a power dissipation value for a determined, representative set of operating conditions. To allow for a more detailed analysis at different conditions, the dependency on the conditions can be provided through the form of terms in an equation. This allows for a more accurate power dissipation to be determined for the expected operating conditions of a specific design.

Different manufacturers provide formula to calculate power dissipation in various degrees of detail. To provide a power dissipation value for the most die with a sufficient precision, a general equation is introduced here. It is meant to capture the most general and important features of the various power dissipation calculations. The power dissipation equation depends upon the die quiescent current, the load on the output signal drivers, the operating frequency, supply voltage(s) and the internal "bulk" capacitance.

The power dissipation equation is:

$$\text{power\_dissipation} = I_{DDQ} * V_{DD} + (K * C_{load} + C_{PD}) * V_{DD}^2 * F$$

where the variables are:

Variable	Setting Providing Value	Description of Value
I_{DDQ}	die_quiescent_current	The die quiescent current.
V_{DD}		The power supply voltage at the power supply pad(s) (design specific).
K	die_load_factor	The average output load dependency for all signal pads.
C_{load}		The lump sum, output loading capacitance driven by the pads (design specific).
C_{PD}	die_power_capacitance	The frequency dependency of the internal power dissipation of the die.
F		The operating frequency of the device (design specific).

Table 1: Variables to Power Dissipation Equation

The formula variables V_{DD}, C_{load} and F are design (or die use) dependent and thus values are not provided in the format. The other values (I_{DDQ}, K and C_{PD}) are die specific (although possibly use specific also) and are provided via settings in the DIE Format.

Note:

The die load factor (K) is very dependent on the functional use of the die. In fact, its value is very pad (signal) specific. It is strongly recommended that a user determine the "K*C_{load}" value for the die. This could be done by developing a "K * C_{load}" value for each signal pad, and then summing the results for all such pads. The resultant, more accurate, lump-sum value can then be used in the power equation.

Note:

The current and voltage variables are denoted in what would normally be MOS circuit technology nomenclature. That is, the use of the "DD" subscript. The variables are expected to be applicable and equivalent to variables in other technologies. For example, Bipolar technology would normally use the "CC" subscript in place of the "DD". The user should apply common sense in determining the variable values.

The specific heat capacity is provided in a setting to support more accurate transient thermal analysis.

In basic thermal analysis supported by the DIE Format, power dissipation is assumed to be uniform across the die. If the distribution of power dissipation is not uniform, then cells of higher power dissipation should be identified. A more detailed thermal analysis can be done to obtain a more accurate junction temperature profile by using the power cell setting provided for this purpose.

The maximum power dissipation is provided in a setting to aid in worst case analysis.

1.2.6. Supply and Signal Integrity (Electrical) Analysis

Signal integrity analysis in a high performance MCM design is usually very important. The operating speed of these systems are typically high enough to require taking into account transmission line effects of the die to die interconnection. To aid the process of doing this analysis, information about the electrical characteristics of the die pads is available.

Some of the more detailed, electrical properties of pads are not directly included and defined in the DIE Format. Instead the IBIS format [IBISv1.1] is used to provide these parameters. An IBIS file can be textually included in the DIE Format and referenced from die and pad descriptions.

Besides signal integrity analysis, supply distribution and switching noise analysis are required to verify the electrical design of an MCM system. Support for these additional forms of analysis is provided through various parameters in the DIE Format.

It is expected that IBIS format version 2.0 or later will incorporate more accurate specifications of electrical characteristics currently captured by some of the DIE Format settings.

1.2.7. Test Structures

The DIE Format supports description of JTAG Boundary Scan structures as defined by the IEEE 1149.1 [JTAG1149] standard. These structures are described through the reference of a Boundary Scan Description Language (BSDL) external model [BSDL1149]. See the setting **die_BSDL** for more details.

There is no further, direct support for MCM test development in this version of the specification.

1.2.8. Logic Simulation

The DIE Format supports the inclusion of VHDL source code simulation models. These models are meant to conform to existing standards for digital logic simulation models such as the VHDL Commercial Component Model Standard [EIA567] and the U.S. DoD Military Handbook 59 [MILHBK59]. The intent of the included VHDL models is to provide an EDA tool independent specification of function and timing for the die.

Note:

> The model(s) supplied are not expected to be useful for MCM module fault simulation nor analysis of propagation of faults. Analysis like this usually requires either a hardware model or an accurate, gate-equivalent model of the die. The VHDL model referenced here is only expected to accurately reflect the good behavior of a digital circuit.

Note:

> There is no mechanism (or referenced standard using VHDL) to include analog behavior simulation models, at this time. Future versions may allow references to other forms of logic simulation models (for example, Verilog, object code, etc.). VHDL models usable with multiple verification tools (such as Logic simulation, Fault simulation and Dynamic Timing Analysis) may be supported in the future through updates in the modeling standards.

1.3. Document Conventions

Throughout the rest of the document, the syntax of the DIE Format is introduced using a variant of Backus-Naur Form (BNF) [BNF1964]. The conventions adopted are covered here for those readers unfamiliar with this format.

A definition in the syntax description is shown starting on a new line with a name of the item being defined, followed by a '::=' character sequence, and finally the body of the definition. A definition may span multiple lines.

```
<item_definition> ::= <body>
```

A body consists of a list of items, possibly separated or grouped by the special characters described below.

Items in the body are either terminals or non-terminals. A non-terminal item is shown as a name between angle brackets ('<' and '>'). Any item being defined is, by definition, a non-terminal.

A non-terminal item is semantically defined in the document. A terminal item is either a keyword, an enumeration name value or a character shown between single quotes. Terminal items are shown in **boldface** and **UPPER-CASE** characters in the syntax description[2]. In the text, keywords are bolded and enumerated name values are in upper case. Example syntax items are shown below:

```
<non_terminal>
TERMINAL_SUCH_AS_KEYWORD
'1' 'A' '<'
```

When zero or one occurrences (normally termed "optional") of an item or group of items can exist, the item(s) are grouped within square brackets ('[' and ']'). When zero, one or more occurrences of an item or group of items can occur, they are grouped within curly braces ('{' and '}'). One or more occurrences is defined by putting the item(s) first and then again inside curly braces.

```
Zero or One Occurrence:            [ <item(s)> ]
Zero, One or More Occurrences:     { <item(s)> }
```

[2] In general, the DIE format is case insensitive. The only exception are the numeric UNITS and MULTIPIERS described in the numeric value section later on.

```
One or More Occurrences:          <item(s)> { <item(s)> }
```

When a fixed number of items, a fixed number range of items, or a fixed minimum number of items is required; this is represented by putting the items in curly braces followed immediately by the number or range designation (n, n-m, or n+; respectively).

```
n Occurrences exactly:              { <item(s)> }n
Zero, One or up to n Occurrences:   { <item(s)> }-n
n to m Occurrences:                 { <item(s)> }n-m
n or More Occurrences:              { <item(s)> }n+
```

Sometimes there is the possibility of a choice between several different items. Each item may be as simple as a value or as complex as a non-terminal. When there is an option or choice between a list of different items, the items are separated by a vertical pipe ('|') character. All items between two vertical pipes are part of the same choice. All items between the definition start (::=) and the first vertical pipe are part of the first choice. All items after the last vertical pipe to the end of the definition comprise the last choice.

```
<choice_example> ::= <item1a> <item1b> | <item2> | <item3a> <item3b>
```

Non-terminals that are tokens (lexicons or literals) of the language are described further in the lexical elements section comprising chapter 3.

Notes (Note:), *italicized phrases*, and examples (**Example**:) are considered descriptive comments (informative) and not part of the actual specification (which is normative).

As terms are introduced and defined, they are underlined.

The specifics of the DIE Format are further unveiled in the document chapters that follow. Chapter two outlines a top-down structure of the DIE Format. A bottom-up description of the lexical elements is then given in chapter three. The sections are described in chapter four and the settings in chapter five. The settings are presented in reference form: alphabetic order with a minimum of one item per page.

The first three chapters should be read for an overview, followed by a detailed reading of chapters four and five. Following the last chapter is a reference list and appendices. The appendices are collected information provided for convenience.

2. DIE Format Structure

The top level unit of information in the DIE Format is termed a block. A block is the minimum unit of information that can be exchanged. Blocks are comprised of sections which, in turn, are comprised of settings. Settings are comprised of attributes, and attributes are a collection of one or more values. Values are lexical tokens in the format and carry the bulk of die specific information.

2.1. Block

A block is the basic unit of information exchange defined in the DIE Format. A block is a collection of die descriptions. A die description is composed of an actual die section along with any referenced pad definitions and external model descriptions.

There is no meaning attached to die described in the same block. There is a benefit to putting similar die in the same block though. Common sections (such as pad geometry definitions) need only be defined once and then can be used by many die descriptions.

The order of blocks in an information exchange shall have no effect on the interpretation of the contents -- the format is order independent. It is up to a tool processing a block of information to determine (from names and revision information inside the block) which die described in the block are of concern. If duplicate die descriptions exist (as defined by the keys for a die section), the most current version of information should normally be used.

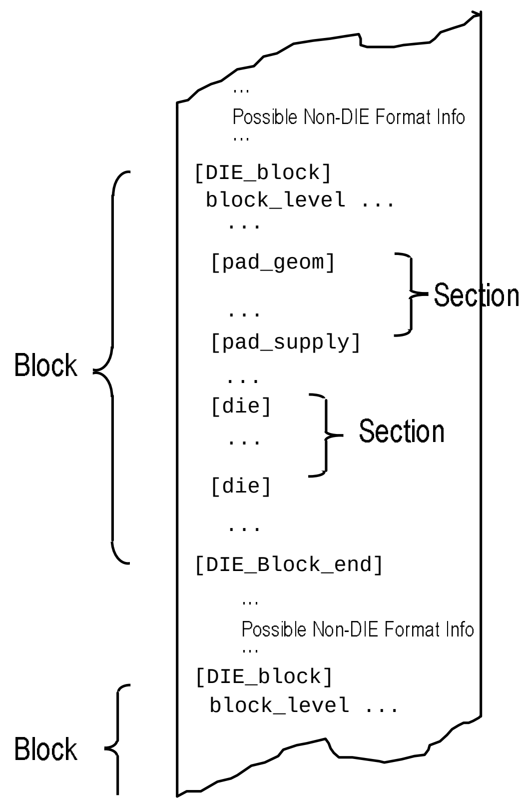


Figure 7: EDI Stream Message or OS File containing DIE Format Block(s)

Note:

It is expected that a block will be created dynamically from a potentially large database or electronic library of die information. A block can be used to transfer a subset of a die library to another party. As such, a block can be considered similar to a die carrier (such as a waffle pack), or a databook. A block is a collection of die information gathered and delivered for some specific purpose. For this reason, the block is defined purely to support the information exchange of individual die data.

Note:

The DIE Format does not limit itself to the file structures of a computer operating system (OS) or format of magnetic media. It is intended that one or more blocks can be embedded in an Electronic Data Interchange (EDI) message or grouped together in a common OS file as desired. A parser for the format should be built to be capable of skipping over extraneous information before, between, and after blocks embedded in a stream of characters for processing. This stream of characters can be provided in whatever mechanism the tool is built to handle -- shared memory, UNIX sockets, email message, magnetic tape, or OS files; for example. Figure 7 gives an abstract view of this structure.

Block Syntax

A block, in general, is an ordered collection of sections:

```
<block> ::= <section> { <section> }
```

Specifically, a block consists of a header section, a body of sections, and a trailer section.

```
<block> ::= <header_section> { <block_body> }1+ <trailer_section>
<header_section> ::= <DIE_Block>
```

```
<block_body> ::= <pad_geom> | <pad_digital> | <pad_supply> |
                  <die> | <model> | <model_end>
```

```
<trailer_section> ::= <DIE_Block_end>
```

A block is composed of three or more sections.

2.2. Sections

All information content in the DIE Format is partitioned into sections. A section is a major collection of related information. A section is used to define the begin and end of a DIE Block also.

Section Types

There are three primary section types which can exist in a block. They are DIE Block, Die, and DIE Block end. At least one of each section will be in every block. There can exist one or more Die sections in a block, each describing a different die (or version of die information).

There are three definition section types which allow information common to one or more pads to be defined once and referenced. These sections are independent of any die section. They are referenced by die sections through the use of a name defined in the definition section. The definition sections currently allowed are pad geometry, pad digital (signal, electrical), and pad supply (electrical).

To aid in providing a library of pad information (multiple definition sections) without concern to which elements of the library are needed, a block may include definition sections which are never referenced.

Definition sections must occur in a block before any die section referencing them (that is, earlier in the block's stream of characters).

Note:

> These pad definition sections comprise an unstructured pad library capability in the DIE Format. A library of pad types common across a technology family can be included in a block by copying all the pad definition sections into the beginning of the block. Only those pad sections referenced from a given die section will have relevance. A pad section need not be referenced to exist in the block.

Finally, there is an external model section type which allows inclusion of descriptive, processable information which is defined in an existing standard format. These sections can occur anywhere in the block and their linkage and interpretation is defined by the settings in the die or pad section(s) which reference them.

The external model section, *model*, is used to include I/O Buffer Information Specification (IBIS) files [IBISv1.1], Logic Simulation models [VHDL1076], Boundary Scan Description (BSDL) models [BSDL1149], and wafer lot specific measurements of a die [TIMEASURE]. IBIS models are referenced by a pad electrical definition (**pad_digital** section) while VHDL, BSDL, and measure files are referenced from a **die** section.

Note:

> Because these external formats are not understood by the DIE Format or parser, care should be taken to make sure the model end section token (**[model_end]**) does not occur in the included model file.

Note:

> The external model sections are provided to allow inclusion of an external, standard format of information in the block. This is in lieu of providing a file reference mechanism. File reference mechanisms allow for the possibility of misplacing the information referenced by the block during the information transfer. Therefore, only direct inclusion of the actual information is supported.

Section Syntax

The general syntax for the section structure is shown below:

```
<section> ::= '[' <section_keyword> ']' <section_body>

<section_keyword> ::= DIE_Block | die | DIE_Block_end
pad_geom | pad_digital | pad_supply | model | model_end
```

Sections start with a section keyword and continue until the start of the next section. Section keywords are enclosed in square brackets for easy identification and skipping of undesired sections. Keywords used elsewhere within the DIE Format are not bracketed.

A **DIE_Block** section is always the first section in a block. A **DIE_Block_end** section must always be the last section in the block. These two special sections cannot occur anywhere else in a block, by definition, and must occur only once per block.

Each section type is described further in chapter four.

Section Referencing

Some sections can reference information created in other sections. This allows common information to be defined once and used multiple times within a section or in different, related section. The section referencing allowed is diagrammed in figure 8.

A **Die** section can reference any **model** (BSDL, IBIS, measure, or VHDL file) or any pad definition (**pad_digital**, **pad_geom**, or **pad_supply**) section. A **pad_digital** section can reference an IBIS **model** section also.

At least one **pad_geom** section must always occur and be referenced from a **die** section.

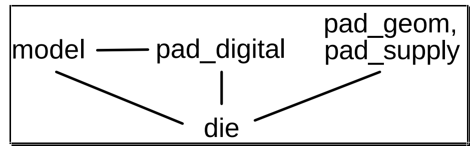


Figure 8: Section Referencing

Note:

There are usually few, unique pad geometry's per die or family of die. The pad geometry can be defined once and then referenced many times in the die description.

2.3. Settings

A section is composed of one or more settings. A setting is a collection of information that is self consistent and useful to understanding the object described by the section. The actual medium (or carrier) of information transfer in the DIE Format is through the settings.

Settings are identified by the setting keyword which they begin with. Settings are generally order independent -- serving to identify (somewhat) independent facts about the object described in the section.

Only one setting of a given identification (keyword) is allowed per section. That is, settings cannot be repeated within a section.

A setting is unique to the section type it is defined for. That is, settings are mutually exclusive to (partitioned among) section types. It is an error for a setting to be located in a section type it is not defined for.

Because of the large number of settings, they are described, in detail, in a reference style in chapter five later on in the document. Only the general structure of the setting format is clarified further here.

Setting Syntax

A setting consists of a keyword identifying the setting type, then followed by one or more attributes, and finally terminated with a semicolon. A basic attribute is made up of one or more values.

The general form of a setting is shown below:

```
<setting> ::= <setting_keyword> { <attribute> } ';'
```

Attributes define the information associated with the setting keyword and about a particular section item they are described in.

2.4. Attributes

An attribute is a collection of one or more values and represents a basic unit of information.

An attribute, in general, can consist of either a single value, a fixed number of values, or a variable number of values.

When a variable number of values is possible, there is confusion as to when one attribute ends and another begins. Some method of grouping the values in an attribute must be available so as to separate one attributes' values from another. Therefore, when there is the possibility of confusion, an attribute may be enclosed in parenthesis or be separated from a following attribute by a comma (','). This form of an attribute is termed a complex attribute.

Attribute Syntax

The general form of an attribute is as follows:

```
<attribute> ::= <attribute_basic> | <attribute_complex>

<attribute_complex> ::=
'(' <attribute_basic> ')' {'(' <attribute_basic> ')'} |
<attribute_basic> {',' <attribute_basic> }

<attribute_basic> ::= <value_list> |
<attribute_basic> { <attribute_basic> }

<value_list> ::= <value> { <value> }
```

Basic attributes can be hierarchical. That is, composed of basic attributes themselves.

Note:

This description of attributes is confusing at first but is needed to describe the free-form syntax. What the <attribute> mechanism accomplishes is being able to have no required, intervening syntax (other than tokenizing white space) between fixed length lists of values or lists of values that are of some unknown length -- (n+m*x) where x is unknown.

Not all settings require a unique, identifiable attribute definition. In these cases, the attribute is not uniquely identified and the value is just described directly.

The attributes are described for the specific setting type they occur in. There are some attributes, however, which occur in two or more settings or are important enough to be described separately. These special attributes are described further here.

2.4.1. Coordinate Point

Coordinate points are given in the coordinate system defined for the section -- usually either the die or pad coordinate system. All coordinates exist in a Cartesian, two dimensional plane system. The origin for a given coordinate system is defined by the basic model description. It is usually the center of the object being described.

Syntax

```
<point> ::= <x_coord> <y_coord>

<x_coord> ::= <numeric_value>

<y_coord> ::= <numeric_value>
```

The x_coord and y_coord attributes give the cardinal position in the X-Y coordinate system. Unlike dimensions, the coordinate point values can be comprised of positive, zero or negative numeric values.

The default unit and multiplier for a coordinate point attribute is microns (um), unless otherwise specified. Only units of meters, inches or mils are allowed.

Example(s):

```
1234.0um -637.5um
```

The example details a point 1234.0 microns in the X direction and -637.5 microns in the Y direction away from the origin.

2.4.2. Dimension

A dimension represents a distance associated with some object. Dimensions are always either a positive or zero numeric value, never negative. A dimension is not a vector (that is, does not have an explicit direction component although a specific value may be given in reference to a specific coordinate direction)

Syntax

```
<dimension> ::= <numeric_value>
```

The default unit and multiplier for a dimension attribute is microns (um), unless otherwise specified. Only units of meters, inches or mils are allowed.

Example(s):

```
400um
```

The example represents a dimension attribute value of 400 microns.

2.4.3. Material List

A material list attribute consists of a variable length description of the layers composing the material being defined by the setting.

Syntax

```
<material_list> ::= <single_layer> | <multiple_layers>

<single_layer> ::= <material_name> [ <thickness> ]

<multiple_layers> ::= <number_of_layers> { <layer> }n

<number_of_layers> ::= <numeric_value>

<layer> ::= <material_name> <thickness>

<material_name> ::= <name_value> | <string_value>

<thickness> ::= <dimension>
```

In the single layer form, the material list simplifies to the listing of the outermost material only. A thickness is optionally provided.

In the multiple layer list form, the layer stack and thickness of each layer is required. The numeric value indicating the number of layers determines which form of the material list is occurring if both are allowed. There is a restriction in that a material name_value cannot start with a numeric character to aid in parsing this construct.

A material name consists of a name_value or a string_value.

The material name is not necessarily meant to be computer sensible although it should be fully understandable to a material science expert. The material name must be a primary element or composite description. For composite materials, a string_value should be used to list the primary elements the material consists of along with the percentage contribution of each. If an understood composite is used (for example, Polyimide), then only the name need be used. If the material compound is proprietary and not to be disclosed, then this should be so stated inside the string_value or notes setting.

A material thickness is a standard numeric value with units of either meters, inches, or mils. Microns (um) is the default multiplier and unit.

Example(s):

```
"Al 99.7% Cu 0.3%"
Polyimide 2.5
2 "Al 99.7% Cu 0.3%" 5 Au 3um
```

The examples represent an Aluminum / Copper alloy, a Polyimide layer that is 2.5 microns thick, and a multiple layer material consisting of an outer layer of an Aluminum / Copper alloy that is 5 microns thick and then a Gold layer 3 microns thick.

2.4.4. Polygonal Area

In many cases, a general, polygonal shaped area is required to be described. For example, pad shapes and power cell area settings. In these cases, the general polygonal area attribute can be used as defined here.

Syntax

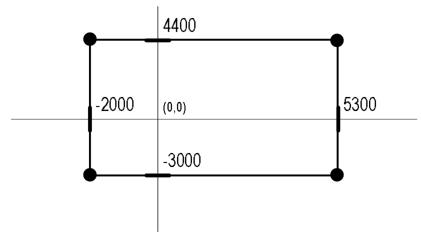
```
<polygonal_area> ::= <num_points> { <point> }4+

<num_points> ::= <numeric_value>
```

Num_points must be the number of coordinate points following. The point list describes the vertices of the polygonal area. The list of points are ordered such that two adjacent points in the list define a side.

The polygon described by this attribute defines a closed polygon. Therefore, the last point is always equal to the first. Also, if 'n' points are described, this represents 'n-1' vertices and 'n-1' sides to the polygonal area.

The polygon should be proper. That is, it can have concave or convex regions but no side should intersect another.

The default unit and multiplier is microns (um). Only units of meters, inches, and mils are allowed.



Example(s):

```
5 5300  4400
  5300 -3000
 -2000 -3000
 -2000  4400
  5300  4400
```

2.4.5. Tolerance

A tolerance attribute specifies a plus-or-minus (±) value range possible on the dimension(s) or point(s) in a related attribute. See the section 1.2.4 on Tolerances and Accuracy for a more thorough discussion of the implications of tolerances to the physical world.

Syntax

```
<tolerance> ::= <numeric_value>
```

The <u>tolerance</u> attribute is a special numeric_value that must always be positive (never zero or negative). The default unit and multiplier is micron (um). Only units of meters, inches or mils are allowed.

Example(s):

```
1.0mil
```

2.4.6. Version

The <u>version</u> attribute specifies the creation and therefore effective date and time of the item being attributed.

Note:

In some cases, the creation date may also imply an obsolescence date. If a block is created from an internal library or database, any extracted information is instantly outdated as the database may change at any time.

Syntax

```
<version_attribute> ::=  <revision_value> [ <date_value> [ <time_value> ]]
```

A version attribute consists of up to three values: a revision_value followed by an optional date_value and time_value. See the lexical analysis section for more details on each of these values.

Example(s):

```
1A.3 26/2/1994
```

2.5. Levels of Compliance

MCM fabrication and design processes vary widely. To accommodate the differing levels of detail required by these variances, the DIE Format is divided into increasingly complex levels of detail. These levels are termed compliance levels as they are also used to define the amount and type of information expected in a given block.

The DIE Format is designed to be an evolving specification. The whole purpose is to allow the common definition for interchange of die information from supplier to user in a machine processable format. Even if processable information is only released under confidence and legal restrictions, it should be processable in the same way as commonly available information. So tool and die suppliers are not necessarily burdened with all the levels of possible detail which may or may not be included, the DIE Format is partitioned into three levels of compliance.

Compliance level 0 consists of the common, indisputable information required by any die user. It represents the minimum set of information which must be conveyed in a block about a die. Although level 0 information may be expanded in future revisions of the specification, it is intended to be the most static level of information defined.

Compliance level 1 contains information which is either a) derivable from level 0 information, b) not needed except by MCM users pushing the edge of technology, or c) not as well accepted by the general industry as level 0 information is. Level 1 may contain sensitive information which is not generally distributed to the community at large. In a few cases, level 1 contains refinements of level 0 settings (for example, adding a tolerance to a value supplied in a level 0 setting). It is expected that level 1 type die information will be under tighter distribution rules or used as a competitive edge by some die suppliers to better support the user base.

Compliance level 2 contains information either highly proprietary in nature or so unique that it will probably only be delivered under special request directly from the producer to the consumer. Examples may be measurement data for a given processing lot of die.

Each compliance level is a superset of the one before it. Therefore, they build on one another. To be level 1 compliant, a block must contain all level 1 and 0 information.

Levels of information compliance are enforced by defining settings unique to a section and compliance level. Therefore, a level 1 setting for a given section would be optional in a level 0 block but required in a level 1 or 2 block. By definition, a level 0 setting is always required in a block anytime the section exists. A section inherits the required compliance level from the block it is contained within.

All settings for a given compliance level are required in a block defined at or above that level. Settings for higher compliance levels can be used as desired. That is, a block labeled as compliant level 0 must have all level 0 settings but may optionally have some level 1 or 2 settings also included. A manufacturer may wish to supply some compliance level 1 information in its level 0 distribution to help clarify and improve the usability of the die information while not supplying all level 1 information. Only when all the settings for a given level (and all previous levels) are included can the block be labeled as compliant to that level.

Note:

> Tools processing the blocks should take this into account and understand how to skip over information deemed not necessary or incomplete. If a tool is designed to only accept level 0 or 1 information, than settings of higher levels encountered can be ignored without an error indication to the user.

Note:

> A "delta" release of a block (that is, one containing only the information that may have changed from a previous release) is not representable in the DIE Format. Each block and the corresponding sections must be complete with each release no matter how small the changes. It is up to the user or tool receiving DIE Format information to determine how to process it -- whether to search for the delta information or just replace the whole die description.

Note:

> The user creating a die section is encouraged to put descriptive comments in a block to create a history of changes associated with each new version created. This would be of benefit to those manually perusing the die section of information. The same suggestion applies to other sections also.

Note:

> Due to the fact that level 2 settings can be included in level 1 or level 0 blocks, and due to the fact that level 2 is comprised mostly of special, unique data; it is not expected to ever receive a DIE Format block with level 2 compliance.

3. Lexical Elements

The DIE Format is kept as free of syntax as possible for simplicity, readability, and flexibility. The downside is a parser, in general, will not be able to recover as easily once there is an error. Given that the format is expected to be machine generated, in most cases, and only hand checked or read; the benefits of this clarity outweigh the drawbacks.

3.1. Character Set

The DIE Format is comprised of a stream of bytes interpreted as characters from the International Organization for Standardization (ISO) eight bit coded character set (ISO 8859-1:1987 (E) [ISO8859]). All bytes of information will be interpreted as belonging to this set.

A few non-terminals are used in the syntax but not defined in terms of BNF here. <ISO_digit> implies any of the ten graphic character digits (0 through 9) from the ISO standard. <ISO_alphabetic> implies any graphic character representing an alphabet character from the same standard. <ISO_alphanumeric> implies either an <ISO_digit> or <ISO_alphabetic>. <ISO_graphic> implies any ISO graphic character which would normally occur under non-binary, 7-bit ASCII text files on a system.

3.2. Comments

Comments provide a free form mechanism to embellish the DIE Format description for the human reader. Comments are not allowed to contain directions or information intended for automatic processing by the computer.

Comments may be inserted anywhere in a block between tokens. Comments should not modify the interpretation of the block content in any way.

Comments begin with the vertical bar (or also termed pipe) character ('|') and continues until a new line, carriage return, or line feed character. Comments are removed from processing during lexical analysis.

Syntax

```
<comment> ::= '|' { <ISO_character> } <record_terminator>

<record_terminator> ::= <new_line> | <carriage_return> | <line_feed>
```

Examples:

```
|-------------------------------------------------------------------
|    These three lines are comments in the DIE Format.     ...
|-------------------------------------------------------------------
```

3.3. Tokens

A token is the minimum recognizable unit which comes out of the lexical analyzer. The character stream is broken up into tokens by intervening white space or by special character tokens.

White space is ignored except to designate token termination in a character stream, to terminate a comment, or when inside a string_value or text_string (defined later).

Tokens are case insensitive. Therefore, keywords shown in mixed or upper case are done purely for aesthetic reasons and are not to be interpreted as being required in mixed, upper or lower alphabetic case.

White space consists of the non-graphic ISO characters new line, carriage return, line feed, form feed, space, and horizontal tab. No other non-graphic ISO characters are allowed within a block. Multiple white space characters in a row are treated as a single character for tokenizing purposes.

```
<white_space> ::= <new_line> | <carriage_return> | <line_feed> |
                  <form_feed> | <space> | <htab>
```

Special character tokens can act as delimiters without requiring surrounding white space to designate their termination or the termination of the token before them. These special tokens are the semicolon (';'), comma (','), the left and right parenthesis ('(' or ')'), the left and right square bracket ('[' or ']'), and the vertical bar ('|').

```
<special_character> ::= ';' | ',' | '(' | ')' | '[' | ']' | '|'
```

3.4. Keywords

Keywords are special tokens. Keywords are used to start and identify types of sections and settings. Keywords are reserved and may not be used as name values or enumerated type values for attributes. A keyword can occur inside a comment or string value but not as a value in an attribute.

A keyword is comprised of an initiating alphabetic character, followed by any number of alphanumeric or underline ('_') characters.

Only pre-defined keywords can be used where keywords are required. They are uniquely defined and belong to either a section or setting type specifier.

To aid in upward compatibility and prevent existing forms of exchange and parsers from being outdated, a parser may ignore a section or setting which contains an unknown keyword if and only if the DIE Format version specified in the **DIE_Block** section is greater than the version the parser was designed to handle. In all other cases, unknown keywords encountered should be treated as errors.

It is an error for a block to contain a keyword that is not defined in any DIE Format version.

Syntax

```
     <keyword> ::= <ISO_alphabetic> { <keyword_body> }
<keyword_body> ::= <ISO_alphabetic> | <ISO_digit> | '_'
```

See the collected keyword list in appendix B for a list of the reserved keywords.

3.5. Values

Values are the primitive, single quanta of information transfer allowed. Values are tokens.

An attribute consists, at minimum, of one or more values. Each value is a token and, as such, cannot have any intervening white space or special token delimiter characters. (The only exception to this is the string value listed below.) Values can be of type date, ID, name, numeric, revision, rotmir, string, text, and time.

3.5.1. Date Value

A date value identifies a day out of the continuum of time. Traditionally, a date value is represented in a large variety of formats. To make sure the information is internationally understood, a specific numeric format using the Gregorian calendar is the only form allowed.

Syntax

```
<date_value> ::== <day> '/' <month> '/' <year>
```

The day is an integer in the range from 1 to 31 and must correspond to a legal day in the month and year specified.

The month is an integer in the range from 1 to 12 representing the months January through December, respectively.

The year is the full (four digit) year from the Roman calendar.

Note that a single digit day or month integer may have a leading zero for clarity.

Examples:

```
23/06/1993
1/1/1993
```

3.5.2. ID value

An ID value is a special name introduced into the format purely to allow cross referencing of sections. An ID defined in one section is possibly referenced in other sections to show an association or relationship of the information. Currently, ID's are used solely to identify pads.

Syntax

```
<ID_value> ::= <ISO_digit> { <ISO_digit> }
```

An ID consists of a series of one or more digits.

Examples:

```
34
1
```

3.5.3. Name Value

Name_values are used to create identification tags for use in associating items of information within the format or items in the format with real world objects.

Unlike name values in computer programming languages (which need to distinguish between numeric and name values in expressions during parsing), a name value in this format can begin with and contain almost any ISO graphic character. This flexibility is needed to represent the free form names contained in the sources the die information is extracted from.

As with all tokens, the alphabetic case is ignored for determining uniqueness in a name. Also, a name cannot be the same as a keyword.

There are a few characters not allowed in a (base) name to facilitate error recovery in parsing or the ability to distinguish tokens. The first 25 characters in the ISO character set (commonly termed control characters or non-printable ASCII) are not allowed. This includes all form of white space characters. The special characters are not allowed so names can be terminated by other than white space characters. Characters numbered 127 (the DEL) and 128 through 159 are not allowed.

Names which represent multi-bit values of a digital signal can utilize a postfix bit notation defined here. This notation, derived from the original ISP notation, consists of a sequence of bit ranges enclosed in angle brackets ('<' and '>'). All numeric values given in the notation must be either in ascending or descending order. Numeric values must be non-negative integers. The first value in the notation is always considered the most-significant-bit (MSB). The interpretation of the bit field represented by the multi-bit designation (unsigned integer, signed integer, mantissa, etc.) is not defined. The bit range notation purely assigns unique names to each bit in sequence. Gaps in the numeric sequence between range specifications do not indicate a gap or grouping in the bits, merely a break in the orderly sequence of numeric names assigned the bits.

Syntax

The name value format and bit field notation syntax is given below.

```
<name_value> ::= <base_name> [ <bit_notation> ]

<bit_notation> ::= '<' <bit_range> { ',' <bit_range> } '>'

<bit_range> ::= <numeric_value> [ ':' <numeric_value> ]

<base_name> ::= <ISO_graphic>
```

Examples:

```
RectangularPad
S18245TJ
1IOPADn
add<31:16,14:0>
```

3.5.4. Numeric Value

A numeric value can be an integer or real number followed immediately, with no intervening token delimiters, by a multiplier and/or unit combination. A numeric value does not need a multiplier or unit if there is a default defined for the attribute, setting, section or block.

All numeric values are expected to be in SI units [ISO1000] (with appropriately identified prefixes) where possible or in customary inch-pound units [IEEE260].

Polar coordinate values are expected to be in customary degrees (not SI radian units) unless otherwise specified.

Unit and multiplier symbols are case sensitive -- these are the only alphabetic case sensitive constructs in the DIE Format.

Prefix Symbol	Prefix Name	Factor
f	femto	x 10^{-15}
p	pico	x 10^{-12}
n	nano	x 10^{-9}
u	micro	x 10^{-6}
m	milli	x 10^{-3}
c	centi	x 10^{-2}
d	deci	x 10^{-1}
		x 10^{0}
da	deca	x 10^{1}
h	hecto	x 10^{2}
k	kilo	x 10^{3}
M	mega	x 10^{6}
G	giga	x 10^{9}
T	tera	x 10^{12}
P	peta	x 10^{15}

Table 2: Allowed numeric multipliers (subset of SI)

Unit Symbol	Implied Units	Units of
A	Amperes	SI Current
C	°Celsius	Temperature
F	Farad	SI Capacitance
g	gram	SI Mass
H	Henry	SI Inductance
in	inch	Length
J	Joule	SI Energy
J/kg-C	Joule / kilo-gram • °Celsius	SI Heat Capacity
K	Kelvin	SI Temperature
m	meter	SI Length
mil	mil (10^{-3} in)	Dimension
Ohm	Ohm	SI Resistance
s	second	SI Time
V	Volt	SI Voltage
W	Watt	SI Power
W/cm-C	Watt / centi-meter • °Celsius	Heat Conductivity

Table 3: Allowed numeric units (subset of [IEEE260] allowed units)

Tables two and three define the units and multiplicative factors (multipliers) acceptable in the DIE Format. The symbols are the allowable characters. The tables are derived from the American National Standard Letter Symbols for Units and Measurement [IEEE260]. This standard is the U.S.A. interpretation of the ISO 1000 SI Units standard, but includes the inch-pound units as well as symbol designations when limited character sets exist, as is the case here.

Syntax

The syntax for the numeric value is shown below:

```
<numeric_value> ::==  [ '+' | '-' ] <ISO_digit> { <ISO_digit> }
                        [ '.' <ISO_digit> { <ISO_digit> } ]
                         [ [ <multiplier> ] <unit> ]
```

All multipliers and units must be composed of a symbol listed in table two and three; respectively.

Exceptions to the Standards

Unlike what is stated in the referenced standards, there should be no space inserted between the digits and multiplier and / or units. Also, compound units are not allowed expect as designated in the table below.

In the symbol for micro, per section 6.1 of [IEEE260], the letter 'u' is used to substitute for the Greek symbol μ due to the limited character set.

In the symbol for degrees Celsius, per section 6.2 of [IEEE260], the degree symbol is dropped due to the limited character set.

In the symbol for Ohms, per section 6.1 of [IEEE260], the word 'Ohm' is used to substitute for the Greek symbol Ω due to the limited character set.

In the symbol for mass, the SI units actually reference kilograms but then allow factors associated with just the base letter 'g' representing grams. We define the base unit of grams here for formal BNF description purposes.

SI Unit's require parenthesis around some terms. The parenthesis are dropped in the DIE Format.

The multiplicative dot (or alternative space) between units has been changed to a dash ('-'). Degree Celsius is utilized where Kelvin is expected.

Centimeters is utilized instead of meters for the heat conductivity.

Note:

> Microns are represented by 'um' and defined as the default unit and multiplier in some cases. The base units of inch and mil are both defined. Either can be a default unit.

Note:

> Scientific notation values are not supported in the DIE Format. Real numbers with multipliers should be sufficient for all numeric specifications.

Examples:

```
1.1nm
6s
5
```

3.5.5. Revision Value

A revision value represents the identification of an element in a sequence history of some object. It is purposely free form to allow for differing revision designation formats in use by the different manufactures. But it is formally defined in forms of its ranking with other revision values to allow unambiguous processing.

Syntax

A revision value is made up of one or more revision marks which are separated by revision separators. A revision mark is a variable length of characters. The alphabetic characters which make up a mark are case insensitive.

```
<revision_value> ::== <revision_mark> { <revsep> <revision_mark> }

<revision_mark> ::== <ISO_alphanumeric> { <ISO_alphanumeric> }

<revsep> ::= '.' | ':' | '-'
```

For ranking purposes, a revision mark is considered to be more significant than the one following. Each character is considered to be more significant than the one following. Alphabetic characters are considered to be upper case.

To determine revision ranking, the ISO character set numeric values of the characters are used to compare revision marks. The higher the numeric value (or ranking), the newer the revision. Therefore, revision B is newer than revision A. Revision 1 is newer than revision 0. Also, revision A is newer (ranked higher) than revision 0.

If two revision values of differing lengths (that is, number of revision marks) are to be compared, then the shorter revision value should be padded with revision marks composed of the '0' character until the lengths match. If revision marks of differing character lengths are being compared, then the shorter mark should be extended with trailing zero ('0') characters.

The type of revision separator (revsep) is ignored when comparing revisions and used purely to indicate the separation of revision marks.

Examples:

```
1.0a
A1
C-4
```

3.5.6. Rotation and Mirror (rotmir) Value

The rotation and mirror value describes the transformation of a geometrical object during instantiation. A rotation is always performed before any mirroring.

The rotation specifies an amount to rotate the object counter-clockwise about the origin. The specification is in customary units of degrees.

The mirror specifies a reflection of an object with respect to a coordinate axis. A horizontal (**H**) mirror is about the X axis; a vertical (**V**) mirror about the Y axis. In coordinate terms, a horizontal mirror causes the numeric sign of all Y coordinates to invert (positive to negative, and vice-versa). Similarly, a vertical mirror causes all X coordinate numerical signs to invert.

With just a rotation value and an optional, single mirror specification, all necessary orthogonal orientations of an object can be created.

Syntax

<rotmir> ::= <rotation> [<mirror>]

<rotation> ::= **0 | 90 | 180 | 270**

<mirror> ::= **H | V**

Note:
Some specifications of this value have the same resultant effect. For example, a **180H** value has the same effect as a **0V**. Similarly, a **180V** value has the same result as **0H**. Also, the value **0** has no effect at all whereas **0V** or **0H** has a definite effect.

Examples:

```
180H
90V
0
```

3.5.7. String Value

Arbitrary, non EDA tool sensible strings of characters are allowed as values and treated as tokens. Such strings (or string values) are delimited by an opening and closing double quote character (""). Any ISO character with a graphical representation or white space designation can occur within these quotes.

The double quote itself, if occurring inside the string value, must be repeated to avoid being treated as the closing double quote character.

A string_value is allowed anywhere a text string is indicated. The string_value substitutes for the occurrence of the text_string value.

Syntax

<string_value> ::== '"' { <ISO_character> } '"'

Examples:

```
"This is a string_value"
"This die information is not ""guaranteed"" and is provided for modeling purposes only.  No liability is assumed by the company generating it."
```

3.5.8. Text Strings

If a string_value does not contain the setting termination character (the semicolon (';')) and is the last value allowed in a setting, then the double quote delimiters may be dropped. This reduces the syntax burden for the human reader in almost all uses of the string_value token and probably improves the chances for error recovery in a computer parser[3]. This special form of a string_value is termed a text_string.

Note:

The special text string value makes it very difficult for the parser to understand what is a token during analysis but is deemed necessary to keep the format usable and readable.

Syntax

```
<text_string> ::== <string_value> | { <ISO_character> }
```

Examples:

```
This is a text_string
```

3.5.9. Time Value

Syntax

A time-value allows the identification of a specific point in time relative to some date.

Syntax

A time-value is specified in numeric format as follows:

```
<time_value> ::== <hour> ':' <minute> [':' <second> ] [ AM | PM ]
```

<hour> is represented as a two-digit integer from 00 to 23. <minute> and <second> are each represented as a two digit integer from 00 to 59 (single digits have a leading 0 added to expand them to two digits). If the seconds are not specified, then a value of 00 is assumed.

The <hour> can be either in 12-hour or 24-hour numeric format. If 12-hour, then the trailing AM/PM indication is required and only the two-digit integers from 01 to 12 can be used. The integer 12 used in conjunction with the **AM** indicator is equivalent to the integer 00 in 24 hour format. 12:00**AM** represents midnight; 12:00**PM** represents Noon.

In the 24 hour format, time starts at Midnight. Therefore, 00:00:00 represents Midnight. Time counts sequentially from Midnight as is standard practice.

Examples:

```
01:13
13:34:13
03:14AM
```

---

[3] Note:

By removing the double quote delimiters, and not allowing an embedded semicolon, and making the special text_string be the last allowed value in a setting; then any semicolon encountered can be treated as the terminator for the setting. This is more likely to lead to earlier detection of an error than if a missing trailing double-quote in a normal string_value token occured.

4. Sections

The DIE Format consists of blocks which are sequences of sections. Each section is composed of settings. Each setting type is defined uniquely to a section and occurs only once in a section. It is settings which are defined as belonging to a given level of compliance and, as such, are the basis for determining what must exist in a compliant block and section.

Setting Presentation Format

Within each section and level described below, the settings are presented in a table format. The general form of the table is described here to aid in understanding the information. An example to embellish the description is given in table 4.

The keyword which identifies the setting is presented in the "Setting keyword" column. Immediately next to this keyword is a short description of the purpose or content of the setting. This aids in understanding the information covered from a global perspective.

In general, the settings are presented in alphabetical order in a given table. For the die section, due to the large number of settings, the table is split into three--one for each compliance level -- to ease the understanding of what is required.

Header describing column:	Setting keyword	Description	Level	Key
Primary keys for	setting_keyword_a	Textual description of setting	0	pri
section content	setting_keyword_b	"	0	pri
General setting for a level	setting_keyword_c	"	0	
Dependent setting	setting_keyword_n		1	dep
	setting_keyword_?	"	2	

Table 4: Sample Setting Description Summary Table

<u>Key</u> settings (in the flavor of relational database methodology) are used to identify a section of information uniquely. Key settings for a section are identified in the "Key" column in the table. Keys are listed first in the table and separated from the other settings via a double line. They contain an indication of "pri" in the "Key" column of the table.

The value of a key setting(s) must be unique for a given section within a DIE Format block. For this reason, key settings must always exist and are defined as level 0 compliant settings. A setting may also be labeled as a "dependent" setting in the key column of the table. A dependent setting implies that the setting is optional and not always required. Its use is dependent on some other condition such as the existence of another setting, the value of another setting, or a specification of the die not expressed in a setting. See the individual setting descriptions for an explanation of the dependent condition of use.

Each setting in a section is defined for a given minimum compliance level. If a block indicates a compliance level of 0, then all level 0 settings for each section encountered must exist. A level 1 compliant block will require all level 0 and level 1 settings in any section. A level 2 compliant block requires all settings defined for a section to exist in an occurrence of the section. The compliance level of a given setting is defined in the column labeled "Level" in the table.

Section Name Scoping Rules

Sections are referenced through names defined in a setting internal to the section. These names have visibility (or scope) from the place where they are defined till the end of a DIE Block. Therefore, names are unique to a block -- once a name is used, it cannot be reused.

Names are classified by the section they are defined in. Therefore, two identical names can be used in two different section types in the same block. Names are unique to a section type within the block.

The name for the die section is special in that it is not referenced any where else. Given it is only one piece of the key information for a section, the **die** section name does not have to be unique to the block. But all primary keys for a given section type must be unique among all section instance's of that type in a block. For most sections, the name defines the primary key.

Each section and its corresponding settings are introduced next.

4.1. DIE Block Sections

A DIE Block section begins a block description and is identified with the keyword **DIE_Block**. It contains information to identify the block's DIE Format version, level of compliance, and general information about the block creation and delivery. This sections occurs once and only once in a block and identifies the start of a DIE Format block.

Any settings defined in the DIE_Block section are provided purely to support the creation, transfer and delivery of die information contained within the other sections of the block. For this reason, a **DIE_Block** section has no keys.

All settings for the DIE Block section are defined as compliance level 0 settings and thus always required. They are introduced in table 5 below.

The **DIE_Block_End** section is special in that it contains no settings. It is the last section in the block, by definition, and serves to terminate the DIE Format block description.

Setting keyword	Description	Level	Key
block_DIE_format_version	DIE Format Version	0	
block_disclaimer	Legal disclaimers about information in the block	0	
block_level	Block's DIE Format level of compliance	0	
block_notes	General notes about a block's information	0	
block_source	Description of block information source (person, company, etc.)	0	
block_version	Version of the release of this block of die information	0	

Table 5: DIE Block Level 0 Settings

To facilitate processing a block, the settings **block_DIE_format_version** and **block_level** are always required first and in this order in the **DIE_Block** section of any given block.

The other settings in the **DIE_Block** section may occur in any order.

The **block_version** setting is provided purely as a means of creating a date and time stamp of the creation date of the block. The revision value does not have meaning in its use here.

The **block_notes** and **block_disclaimer** are added to provide distinguishable, comment sections that may be saved with the die data. Their bodies, as can be seen in the detailed descriptions of the settings, can be empty. The settings are required (not optional) to force a user to explicitly ignore them if not used.

Syntax

```
<die_block> ::= '[' DIE_Block ']' <block_die_format_version> <block_level>
                        { <other_header_setting> }

<other_header_setting> ::= <block_disclaimer> | <block_notes> |
                              <block_source> | <block_version>
```

```
<DIE_Block_end> ::= '[' DIE_Block_End ']'
```

Each <other_header_setting> must occur once and only once in a DIE Block section (due to the level 0 status), but they can occur in any order desired.

4.2. Die

A Die section starts with the keyword **die**. The section is used to group information identified with a particular die. The **die** section can be further classified using the **die_type** setting. Some of the other die section settings are dependent on the value of the **die_type** setting.

A **die** section will reference any other needed sections, such as a **pad_geom**etry section. There must be at least one **die** section in every DIE Format block.

A given **die** section must occur later in a block than any referenced pad definition sections.

The **die_type**, **die_name**, **die_manufacturer** and **die_mask_version** settings are the primary keys for the die information. These keys are meant to uniquely identify the die being described. These settings along with the other compliance level 0 settings are given in table 6 below.

Setting Keyword	Description	Level	Key
die_name	Manufacturers published name for die	0	pri
die_manufacturer	Manufacturer's name	0	pri
die_mask_version	Manufacturers published mask step version of die	0	pri
die_type	Kind of die	0	pri
die_junction_temperature	Recommended die junction temperature	0	
die_lead_frame	Beam lead frame description	0	dep
die_notes	General notes about a die	0	
die_packaged_part_name	Name(s) of conventionally packaged components which use this die.	0	
die_pads	Instances of pads for die	0	
die_power_max	Maximum die power dissipation	0	
die_power_nom	Nominal die power dissipation	0	
die_section_version	Version of die information provided in this section	0	
die_size	Die width and height	0	
die_solder_bump	Solder bump description	0	dep
die_source	Source of information about the die.	0	
die_substrate_connection	Electrical connection of the die substrate.	0	
die_substrate_material	Bulk substrate material of the die	0	

die_technology	Manufacturer published process name / technology of the die	0	
die_thickness	Thickness of the die	0	

Table 6: Die Section Level 0 Settings

The **die** section is the main, information section for describing a bare die. The level 0 settings represent the minimum information required to describe a die. Note that some of the settings are quite complex (like **die_pads**) while others are quite simple (like the **die_thickness**).

The default thermal properties of the substrate material (and thus the die as a whole) can be determined from the **die_substrate_material** setting. Specific thermal properties of the die substrate are provided in Level 1 settings. If these settings exist, their indicated values override any values derived from the **die-substrate-material** setting. The **die** section has two dependent level 0 settings. These are the **die-lead-frame** and **die-solder-bump**. They exist only if the **die-type** setting in the section is either LEAD_FRAME or SOLDER_BUMP; respectively.

A **die_type** of SOLDER_BUMP is used to indicate that the die has solder bumps attached to the pads.

Note:

This type specification cannot be used to indicate the adding of additional thin-film layers to create area array pads. These die would have to be described as new, unique die due to the change in the pad locations and geometry's.

Note:

The SOLDER_BUMP **die_type** setting can be used to document TAB die that have only been partially prepared. That is, die meant for lead frame mounting using solder bonds but where the lead frame has not yet been attached.

Die which have been post processed by attaching a lead frame, but not encapsulated or packaged, are described in this section using a **die_type** classification of LEAD_FRAME. All forms of lead frame attachments to the die are meant to be covered. This would include Tape Automated Bonding (TAB) lead frames, ribbon beam leads and any other formed lead mechanism.

Setting keyword	Description	Level	Key
die_backside_finish	Die backside finish and optional coating material list	1	
die_bonding_diagram	Reference to available bonding diagram or description	1	
die_BSDL	BSDL model reference	1	dep
die_conditions_storage	Suggested not to exceed conditions for storage.	1	
die_fiducials	Die identification marks	1	
die_IBIS	IBIS model reference	1	
die_packaged_part_bonding	Pad bonding method(s) used for packaged die	1	
die_pad_pitch	Minimum center-to-center spacing of pads on die	1	
die_pads_supply_grouping	Supply pad groups	1	
die_pads_tolerance	Tolerance of pad placement	1	
die_pads_VHDL_map	VHDL port to pad ID mapping	1	
die_passivation_material	Material list of passivation layer	1	

die_size_tolerance	Die width and height tolerance(s)	1	
die_specific_heat_capacity	Specific heat capacity of the substrate material	1	
die_thermal_conductivity	Thermal conductivity of the substrate material	1	
die_thickness_tolerance	Tolerance of the die thickness	1	
die_VHDL	VHDL model reference	1	
die_wafer_scribe_line	Width of wafer scribe lines	1	dep
die_wafer_size	Size of wafer before die separation.	1	dep
die_wafer_step_and_repeat	Die step and repeat dimensions of the wafer	1	dep

Table 7: Die Section Level 1 Settings

It is desired by MCM designers that die suppliers supply all level 1 information for any die described.

The **die_wafer_*** settings are required if a die is delivered undiced, in wafer form. The user of the die information should re-examine values which may be dependent on the dicing process if these settings exist.

5 The **die_BSDL** setting is only required if the die contains JTAG boundary scan pins and functionality. This functionality must be compliant with IEEE 1149.1 [JTAG1149].

Setting keyword	Description	Level	Key
die_bonding_sequence	Suggested sequence to bond pads	2	
die_conditions_bonding	Suggested not to exceed conditions during single pad bonding (pad vicinity conditions)	2	
die_conditions_process	Suggested not to exceed conditions for die attach processing.	2	
die_conditions_sealing	Suggested not to exceed conditions for processing of die (vicinity conditions during sealing)	2	
die_conditions_special	Any special conditions for handling or use of the die	2	
die_description	Textual description of die features	2	
die_flatness	Indication of die surface warpage	2	
die_load_factor	Output load dependency of the die power dissipation.	2	
die_lot	Lot number(s) of die this information pertains to.	2	

die_manufacturer_cage	CAGE code of manufacturer	2	
die_measure	Lot specific die measurements	2	
die_military_spec	Military part specification, if it exists	2	
die_packaged_part_attach	Packaged part method(s) of attaching bare die to carrier.	2	
die_pad_dielectric	Composition and thickness of the dielectric material under the pads	2	
die_pad_metal	Pad metal layer composition and thickness	2	
die_pads_jumpers	Suggested pad interconnects.	2	
die_pads_noncontact_area	Areas on pad metal surface which are not suitable for contact.	2	
die_passivation_extent_size	Extent of passivation if different from die size.	2	
die_power_capacitance	Frequency dependency of the internal die power dissipation	2	
die_power_cell	Nominal power dissipation of specific areas	2	
die_quiescent_current	Quiescent current consumption	2	
die_saw_step_error	Maximum saw or scribe step error during dicing	2	
die_saw_width	Width of saw (cut), if die separated by saw	2	
die_separation	Method used to separate die from wafer.	2	

Table 8: Die Section Level 2 Settings

The level 2 **die** section settings are considered to divulge proprietary data about the die, information that is not normally obtainable or released by the die manufacturer, or information very specific to a special MCM technology or process.

Syntax

```
<die> ::= '[' die ']' { <die_level_0_setting> |
                        <die_level_1_setting> |
                        <die_level_2_setting> }

<die_level_0_setting> ::=
            <die_junction_temperature>
            <die_lead_frame>|
            <die_manufacturer> |
            <die_mask_version> |
            <die_name> |
            <die_notes> |
            <die_packaged_part_name> |
            <die_pads> |
            <die_power_max> |
            <die_power_nom> |
```

```
            <die_section_version> |
            <die_size> |
            <die_solder_bump> |
            <die_source> |
            <die_substrate_connection> |
            <die_substrate_material> |
            <die_technology> |
            <die_thickness> |
            <die_type>

<die_level_1_setting> ::=
            <die_backside_finish> |
            <die_bonding_diagram> |
            <die_BSDL> |
            <die_conditions_storage> |
            <die_fiducials> |
            <die_IBIS> |
            <die_packaged_part_bonding> |
            <die_pad_pitch> |
            <die_pads_supply_grouping> |
            <die_pads_tolerance> |
            <die_pads_VHDL_map> |
            <die_passivation_material> |
            <die_size_tolerance> |
            <die_specific_heat_capacity> |
            <die_thermal_conductivity> |
            <die_thickness_tolerance> |
            <die_VHDL> |
            <die_wafer_scribe_line> |
            <die_wafer_size> |
            <die_wafer_step_and_repeat>

<die_level_2_setting> ::=
            <die_bonding_sequence> |
            <die_conditions_bonding> |
            <die_conditions_process> |
            <die_conditions_sealing> |
            <die_conditions_special> |
            <die_description> |
            <die_flatness> |
            <die_load_factor> |
            <die_lot> |
            <die_manufacturer_cage> |
            <die_measure> |
            <die_military_spec> |
            <die_packaged_part_attach> |
            <die_pad_dielectric> |
            <die_pad_metal> |
            <die_pads_jumpers> |
            <die_pads_noncontact_area> |
            <die_passivation_extent_size> |
            <die_power_capacitance> |
            <die_power_cell> |
            <die_quiescent_current> |
            <die_saw_step_error> |
            <die_saw_width> |
            <die_separation>
```

4.3. Pad Digital

The pad digital section is used to define a set of electrical settings for one or more digital signal pads. The section starts with the **pad_digital** keyword and is terminated by any section keyword that follows.

A pad digital signal section must occur in a block stream before it is referenced in a **die** section. Each **pad_digital** section must be unique to a given block. This is done by making the **pad_digital_name** setting (associated with it internally) unique among all similar sections in the block.

A pad digital definition can be referenced zero or more times by one or more **die** sections in a block. A pad digital section should only be referenced by a pad of type SIGNAL_DIGITAL.

Note:

> An unreferenced pad digital section is extraneous information, but it is not specifically excluded. This makes it easier to support pad libraries included within a block without concern as to which elements of the library are being referenced. Along the same line of thought, pad digital sections occurring in a block after all other die sections are unreferenceable (hence meaningless) but still allowed.

Setting keyword	Description	Level	Key
pad_digital_name	Pad digital signal section name	0	pri
pad_digital_circuit	Describes the pad circuit	0	
pad_digital_IBIS_model	IBIS model reference	1	
pad_digital_pull_down	Pad output driving capability at logic low level	0	dep
pad_digital_pull_up	Pad output driving capability at logic high level	0	dep
pad_digital_threshold	Pad input switching characteristics	0	dep

Table 9: Pad Digital Section Settings

A **pad_digital** section is uniquely identified by the **pad_digital_name** primary key setting.

Unlike most other sections, some of the settings in this section are dependent. The use of a setting depends on the digital pad type (defined in the **pad_digital_circuit** setting). A **pad_digital_threshold** setting is required when the pad circuit type includes the INPUT name. The **pad_digital_pull_down** and **pad_digital_pull_up** settings are only required when the pad circuit type includes any active driver type (that is, OUTPUT, TRISTATE, or one of the OPEN_ type keywords).

The settings of the level 0 pad digital section provide basic information about the pad logic circuitry.

The **pad_digital** section information is refined through the level 1 **pad_digital_IBIS_model** setting. This setting provides a reference to an IBIS model occurring elsewhere in a **Model** section. The IBIS model provides a more detailed, simulation-model source description of the pads' electrical characteristics.

There are no level 2 settings for the **pad_digital** section.

Syntax

```
<pad_digital> ::= '[' pad_digital ']'
                   { <pad_digital_level_0_setting> | <pad_digital_IBIS_model> }
<pad_digital_level_0_setting> ::=
             <pad_digital_circuit> |
             <pad_digital_name> |
             <pad_digital_pull_down> |
             <pad_digital_pull_up> |
             <pad_digital_threshold>
```

The **pad_digital_name** and **pad_digital_circuit** settings are always required. The other three level 0 settings are dependent on the pad circuit type defined in the **pad_digital_circuit** setting.

4.4. Pad Geometry

The pad geometry section is used to define a set of common geometric properties used by one or more pads in a die. There must be at least one pad geometry section in every block.

A pad geometry section starts with the **pad_geom** keyword.

A pad geometry definition must occur before it is referenced in a **die** section. Each **pad_geom** section must be unique to a given block by making the **pad_geom_name** setting unique. The pad geometry's name scope is over a whole block.

A pad geometry can be referenced zero or more times by one or more **die** sections in a block.

Note:

> An unreferenced pad geometry section is extraneous information; but it is not specifically excluded. This makes it easier to support pad libraries included within a block without concern as to which elements of the library are being referenced. Along the same line of thought, pad geometry sections occurring in a block after all other die sections are unreferenceable (hence meaningless) but still allowed.

Setting keyword	Description	Level	Key
pad_geom_name	Pad identification name	0	pri
pad_geom_bond_sites	Number of intended bond sites and their locations	2	
pad_geom_metal_extent	Pad metal shape	1	
pad_geom_passivation_extent	Passivation opening shape	1	
pad_geom_shape	Pad geometry shape	0	
pad_geom_tolerance	Pad geometry tolerance for sizes	1	

Table 10: Pad Geometry Section Settings

The pad geometry section level 0 settings are simple at this time and consist only of an identifying name and the associated shape description for a pad. See each setting description for more details. There will be a new **pad_geom** section for each unique pad geometry encountered.

Note:

> The **pad_geom** section is not version tagged. When the information of a **pad_geom** changes is pertinent only to a referencing **die** section. Therefore, it is sufficient to define that whenever the **pad_geom** information changes, any corresponding **die** sections which reference the **pad_geom** must have their version updated to reflect the change.

The pad geometry given by the level 0 **pad_geom_shape** setting is a simplification of the actual pad shape. It only represents the shape of the exposed metal of the pad. The level 1 settings for the **pad_geom** section further refine this model by providing the actual shape of the full metal pad and the passivation opening.

Note:

> These settings apply to all pads **--** even those prepared for solder bump bonding where the final pad metal is possibly over the passivation layer.

The level 2 setting **pad_geom_bond_sites** allows the specification of more than a single bond site for a given pad; something usually considered difficult to provide given the dependency on the bond technology.

Syntax

```
<pad_geom> ::= '[' pad_geom ']' {
            <pad_geom_level_0_setting> |
            <pad_geom_level_1_setting> |
            <pad_geom_level_2_setting> }
```

```
<pad_geom_level_0_setting> ::= <pad_geom_name> | <pad_geom_shape>

<pad_geom_level_1_setting> ::=
            <pad_geom_metal_extent> |
            <pad_geom_passivation_extent> |
            <pad_geom_tolerance>

<pad_geom_level_2_setting> ::=
            <pad_geom_bond_sites>
```

4.5. Pad Supply

The pad supply section is used to define a set of electrical settings for one or more supply pads. The section starts with the **pad_supply** keyword.

A pad supply section must occur in a block before it is referenced in a **die** section.

Each pad supply section must be unique to a given block. This is done by making the **pad_supply_name** setting unique among all pad supply sections in the block.

A pad supply definition can be referenced zero or more times by one or more **die** sections in a block. A pad supply section should only be referenced by a die pad_type of SUPPLY_GROUND or SUPPLY_POWER.

Setting keyword	Description	Level	Key
pad_supply_name	Power supply section name	0	pri
pad_supply_current_max	Maximum current limit	0	
pad_supply_voltage	Supply voltage specification	0	

Table 11: Pad Supply Section Settings

Note:

An unreferenced pad supply section is extraneous information, but it is not specifically excluded. This makes it easier to support pad libraries included within a block without concern as to which elements of the library are being referenced. Along the same line of thought, pad supply sections occurring in a block after all other die or other sections are unreferenceable (hence meaningless) but still allowed.

Syntax

```
<pad_supply> ::= '[' pad_supply ']' { <pad_supply_level_0_setting> }

<pad_supply_level_0_setting> ::=
            <pad_supply_name> |
            <pad_supply_current_max> |
            <pad_supply_voltage>
```

All the settings of the **pad_supply** section are compliance level 0 and required at all times.

4.6. Model Sections

The **model** section is used to include IBIS, VHDL, BSDL, and TI Measure external format model files.

The content and name of a given model section is defined by the model section settings. The setting **model_type** specifies the kind of model data contained in the section. The setting **model_name** provides a reference name that can be used by other settings in the DIE Format to identify the model section of concern. This section can occur zero or more times within a block.

IBIS models are I/O Buffer Information Specification (IBIS) format files. The format is described in [IBISv1.1]. Both the [Component] and the [Model] sections of an IBIS file are referenced in the DIE Format. The **die_IBIS** setting of the **die** section references the IBIS [Component] section. The IBIS [component] section contains the link between the die pads and the IBIS pins. The **pad_digital_IBIS_mode**l setting references the IBIS [Model] section and the DIE Format **[Model]** section. See the individual setting descriptions for more details.

VHDL models describe the behavior or function of the die and contain one or more VHDL design entities as defined in the IEEE Std 1076-1987 Language Reference Manual [VHDL1076]. VHDL models are expected to conform to the EIA guidelines for component models [EIA567]. It is assumed that any included VHDL design units will be in the proper analysis order and complete. The setting **die_VHDL** contains any VHDL top level entity-architecture or configuration names as associated with the die model and references the model section(s) containing the VHDL data. Any top level, VHDL model contained in the VHDL data should also contain a BSDL-style "package pin definition" to map the bare die pad IDs to VHDL entity ports. The setting **die_pads_VHDL_map** references this BSDL contract to link the die pads to the VHDL model ports.

BSDL models contain the boundary scan description for an unpackaged die as defined in Supplement (B) of the IEEE 1149.1-1990 standard [BSDL1149]. The BSDL model must use the **die_pads** ID's (defined in the referencing **die** section) in place of the packaged part pin numbers in the BSDL "package pin definition". The BSDL package pin description then becomes the association mechanism of the die bond pads to the logical ports of the BSDL description. The setting **die_BSDL** contains the name of the BSDL package pin definition which associates the **die_pads** ID's to the BSDL entity ports.

TI Measure files contain precise information about the characteristics of a given group of die from a common wafer lot. By storing the information in an external section, more precise information can be added and supported without requiring a revision of the original die information. Manufacturing lots of die are measured because some MCM processes require very precise knowledge of die sizes and pad locations. Note that zero or more die measure model sections may exist and be referenced by a given die section.

An unreferenced model section is extraneous information, but it is not specifically excluded.

Setting keyword	Description	Level	Key
model_name	Reference name of this model section	0	pri
model_type	Type of external format model	0	

Table 12: Model Sections Settings

A model section is uniquely identified by the **model_name** key setting.

The settings in a model section provide basic information which a die or pad section would use to reference it.

Model_end is a special section. It contains no settings in its body and is used purely to succinctly terminate the previous **Model** section. Any section may follow it immediately.

Syntax

```
<model> ::= '[' Model ']' <model_name> <model_type> { <ISO_character> }

<model_end> ::= '[' Model_end ']'
```

Note that the ISO_character is special here in that any number of characters can occur until a **Model_end** keyword appears. This includes allowing setting termination characters (semicolons **-** ';') and DIE Format keywords. The ISO_character sequence should always comprise a valid model in the defined format.

Any included, external format model cannot contain the character sequence "**[model_end]**" (or any variations with embedded white space or different alphabetic case).

Note:

Recognition of the **model_end** section may be tricky for a parser. For example, IBIS models contain similar syntax to the DIE Format, including a [model] construct. Care should be taken to correctly identify the **[model_end]** construct in the DIE Format.

5. Settings

A majority of the information defined and transferred occurs through the settings. The settings are the primitive cells that the compliance levels apply to also. Due to the large number of settings, they are introduced in a reference page style: in alphabetical order with each setting starting at the top of a new page.

5.1. block_DIE_format_version

Section:	**DIE_Block**	Level:	⓿
DIE Format Version			

Syntax

```
<block_DIE_format_version> ::=
            BLOCK_DIE_FORMAT_VERSION <version_attribute> ';'
```

The version attribute is comprised of the revision_value token followed by an optional date_value and time_value token. See the attribute document section for more details.

Model

This setting identifies the version of the DIE Format specification used to create this block. At minimum, it must include the revision_value as defined in the specification. The date_value can be optionally included. The time_value has no meaning here and should be ignored if encountered.

Only the revision information is considered accurate. It is not an error if the date specified does not match the date given for the indicated revision of the DIE Format specification .

Example(s)

```
      block_die_format_version 1.0 1/1/1994 8:10am ;
```

See Also

5.2. block_disclaimer

Section:	**DIE_Block**	Level:	⓿
Legal disclaimers about information contained in the block.			

Syntax

```
<block_disclaimer> ::= BLOCK_DISCLAIMER [ <text_string> ] ';'
```

The text can be a text_string or empty -- the value is optional. See the lexical analysis document section for more details.

Model

The block disclaimer provides an identified, clearly labeled area to list any disclaimers regarding the information contained within the block. In general, such disclaimer information should be kept with the die information as it is transferred between people, tools or systems. The disclaimer is not meant to be computer-sensible; only captured and possibly presented to a user.

Although this setting is required, the actual text_string value is optional. This allows an empty value for the setting and forces a conscious decision to leave the disclaimer information blank.

Example(s)

```
block_disclaimer The information contained within is provided for
      modeling purposes only and is not guaranteed in any way.  The
      information is subject to change at any time. ;

block_disclaimer ;  | explicitly ignored disclaimer setting
```

See Also

5.3. block_level

Section:	**DIE_Block**	Level: ⓿
Block's DIE Format level of compliance		

Syntax

```
<block_level> ::= BLOCK_LEVEL <compliance_level> ';'
<compliance_level> ::= '0' | '1' | '2'
```

There are only three block compliance levels defined: 0, 1 or 2. The level must always be one of these values.

Model

Identifies the compliance level of the block. See the description on compliance levels elsewhere in this document for more details.

Example(s)

```
block_level 1;
```

See Also

5.4. block_notes

Section: **DIE_Block**	Level: ⓿
General notes about a block's information.	

Syntax

```
<block_notes> ::= BLOCK_NOTES [ <text_string> ] ';'
```

The text can be a text_string or empty **--** the value is optional. See the lexical analysis document section for more details.

Model

The block notes setting stores information of general interest to the users of the block. Unlike comments which would probably be stripped out when the block is processed, the **block_notes** information should be retained with the other data.

The notes section may contain special descriptions related to the die grouped together in the block, technology information related to the die, or possibly additional explanations as to the source of the die information.

Although this setting is required, the actual text_string value is optional. This allows an empty value for the setting and forces a conscious decision to leave the notes information blank.

Note:
This setting is a useful location to give revision history, for example, if the block has such history.

Example(s)

```
block_notes Revision history:
        2.1   16/9/92    LCW   Corrected die 1 corner pads center
        2.0   31/8/92    REH   Added description of saw for die 3
        1.0   14/3/92    SC    Initial version for Joe's Machine, Inc. ;
```

See Also

die_notes

5.5. block_source

Section:	**DIE_Block**	Level:	⓿
Description of block information source (person, company, etc.)			

Syntax

```
<block_source> ::= BLOCK_SOURCE <text_string> ';'
```

The text must be a text_string. See the lexical analysis document section for more details.

Model

The block source setting allows block information to be tagged with the identity of the producer. The loose "text" definition probably precludes further interpretation or processing of the source information by a computer though. In general, it is expected the authors name (or initials), division, and company would be included here.

The block source is intended to be a human readable comment field that may be presented to the user during processing of the block information or saved with the processed data for future display. No additional interpretation is given.

The block source may also contain a databook or information source reference from which the die section information within was extracted.

This setting complements the notes setting which is meant to contain information of general interest.

Example(s)

```
block_source REH, Modeling Group, Logic Modeling Corporation
      All die from The TTL Databook, Texas Instruments, 1986. ;
```

See Also

5.6. block_version

Section:	**DIE_Block**	Level:	⓿
Version of the release of this block of die information			

Syntax

```
<block_version> ::= BLOCK_VERSION <version_attribute> ';'
```

The version attribute is comprised of the revision_value token followed by an optional date_value and time_value token. See the attribute document section for more details.

Model

This setting identifies the block's creation date and time only -- the revision_value information is considered extraneous. The block itself is not named thus precluding identification and therefore versioning. It is the die sections which contain unique names and therefore revision values for the die information.

If blocks are to be used as storage medium (for example, a collection of die descriptions from a whole databook) then the version attribute should be updated as follows:

The block version should be updated any time a) a **die** section within a block is modified,, b) a **die** section is added or deleted, c) a section referenced by a **die** section is updated (this will cause the die_section_version setting to be updated which would cause the block version to update), or d) when a **DIE_Block** section is modified. If any section not referenced in a **die** or block section is updated, the block version should not be updated as the section is considered extraneous and unneeded information and should not affect the real information content of the block.

Example(s)

```
block_version  A-2 23/06/93;
```

See Also

die_section_version

5.7. die_backside_finish

Section:	**die**	Level:	❶
Die backside finish and optional coating material list			

Syntax

```
<die_backside_finish> ::= DIE_BACKSIDE_FINISH  <backside_finish> ';'
<backside_finish> ::=
            POLISHED | LAPPED | METALLIZED <material_list>
```

The backside finish may identified by one of the enumerated name values POLISHED, or LAPPED, or METALLIZED See the description of a material_list attribute in the attribute document section for more details.

Model

The die backside finish is crucial to understanding possible die attachment mechanisms, thermal properties of the attachment, and electrical connectivity needs for biasing purposes.

The backside finish may be the natural bulk material which has been either polished or lapped. It may also have been metallized.

For a metallized surface, it is required to specify the coating of the backside of the die (wafer). The coating material layer(s) composition and possibly thickness should be described. If only a single layer is specified, then this must be the outer layer of material of a possible material stack.

The material name is either a periodic table element or compound element name. If a compound, and the portions of elements making up the compound are not implicitly understood, then the compound name should be further elaborated in the **die_notes** setting.

Example(s)

```
die_backside_finish  polished ;
die_backside_finish  metallized 2 "Al 99.7% Cu 0.3%" 5 Au 3 ;
```

See Also

die_substrate_connection

die_substrate_material

5.8. die_bonding_diagram

Section: **Die**	Level: ❶
Reference to available bonding diagram or description	

Syntax

```
<die_bonding_diagram> ::= DIE_BONDING_DIAGRAM <text_string> ';'
```

The bonding diagram location is indicated by a non computer-sensible text_string. See the lexical analysis document section for more details.

Model

The bonding diagram location is indicated with this setting. A bonding diagram generally provides a graphic representation of any metal pads, fiducials, or other identifying features on the die which are visible with 20x or less magnification. The diagram must contain an outline of the die and be oriented so the X axis is horizontal and the Y axis vertical when viewed upright. The X and Y axis should be labeled on the diagram. The text string is meant to provide information as to where the bonding diagram can be found -- either in computer or paper document form.

Example(s)

```
die_bonding_diagram See file s18245t.tif for a TIFF format image;
die_bonding_diagram Order publication 00AA55ZZ from 1-800-DIE-INFO;
```

See Also

die_fiducials

die_pads

pad_geom_metal_extent

5.9. die_bonding_sequence

Section: **Die**	Level: ❷
Suggested sequence to bond pads.	

Syntax

```
<die_bonding_sequence> ::=
            DIE_BONDING_SEQUENCE <bonding_sequence> ';'

<bonding_sequence> ::= '(' <pads_list> ')' {'(' <pads_list> ')'}  |
                           <pads_list>     {',' <pads_list>    }

<pads_list> ::= <pad_ID> { <pad_ID> }

<pad_ID> ::= <ID_value>
```

The pad_ID is an ID_value. See the lexical analysis document section for more details.

Model

The die bonding sequence gives a preferred order for bonding to bond pads. A pad_ID here must match with a corresponding pad_ID in the **die_pads** setting. The bond sequence should include any supply and signal type pads. It would be an error to list the pad_ID for a no_connect or not_defined type pad.

The pad_ID's are partitioned into mutually exclusive groups via the parenthesis or comma's. A pad_ID can appear only once in the bond sequence attribute. All bond pads within a group are assumed to be bondable in any order. The manufacturer is suggesting by this setting that all bond pads in a group should be bonded before bonding to any pads in any group following.

Not all pads have bond sites that are bondable. Therefore, not all pads (pad_ID's) are required in this list. Any pads not in the list are considered to exist in a common pads_list group which is inserted at the very end of the bond sequence attribute.

Note:

There may be multiple bond sites within a bondable pad. There is no way to distinguish different bond sites within a bondable pad with this setting.

Example(s)

```
die_bonding_sequence (4 3 1) (2 5) ;
die_bonding_sequence 4 3 1,2 5;
```

See Also

pad_geom_bond_sites

5.10. die_BSDL

Section:	**Die**	Level:	❶
BSDL information about this die			dep

Syntax

```
<die_BSDL> ::= DIE_BSDL
                  <model_ref> <BSDL_entity_name> <BSDL_package_decl> ';'
<model_ref> ::= <name_value>
<BSDL_entity_name> ::= <name_value>
<BSDL_package_decl> ::= <name_value>
```

The name value is described in the lexical analysis section of this document.

Model

This setting associates a BSDL entity name and its corresponding die pads / entity port mapping information with a die. For more information about BSDL, see [BSDL1149]. Only one BSDL, model specification can be associated with each die section.

The model_ref references a DIE Format **model** section containing a BSDL model. The **model** section must exist in the block if referenced by this setting. The reference occurs by matching the model_ref name_value with a **model** section model_section_name name_value.

The BSDL entity name is identified (and must match) the BSDL "entity" name found in the model. This name must be a legal BSDL identifier.

The BSDL package declaration identifies the appropriate package construct inside the BSDL entity. This name must be a legal BSDL identifier. The BSDL package declaration associates the BSDL entity ports with the die pad ID's ("pins" in BSDL terminology) from the die pads setting.

Example(s)

```
die_BSDL scan18245t scan18245t mda_package;
```

See Also

5.11. die_conditions_bonding

Section:	**Die**	Level:	❷
Suggested not to exceed conditions during single pad bonding (pad vicinity conditions)			

Syntax

```
<die_conditions_bonding> ::= DIE_CONDITIONS_BONDING <text_string> ';'
```

The bonding conditions are specified in a non-computer sensible form as a text_string. See the lexical analysis document section for more details.

Model

These are the suggested not to exceed conditions for the die before or during subsequent processing and use.

The conditions specified in this setting are for the individual pad bonding process, like wire bonding, where temperature and pressure is applied to the pads sequentially, one at a time and each for a certain duration.

The conditions described in the text string should at least include the temperature, pressure, and voltage (differential with reference to supply pads) along with the maximum duration for each. Humidity and any additional environmental conditions may also be specified.

Example(s)

```
die_conditions_bonding 150 degrees C max, 5 grams / um**2 max,
      15 volts over a 30 ms time period ;
```

See Also

die_conditions_process

5.12. die_conditions_process

Section:	**Die**	Level:	❷
Suggested not to exceed conditions for die attach processing			

<u>Syntax</u>

```
<die_conditions_process> ::= DIE_CONDITIONS_PROCESS <text_string> ';'
```

The process conditions are specified in a non-computer sensible form as a text_string. See the lexical analysis document section for more details.

<u>Model</u>

These are the suggested <u>not to exceed</u> conditions for the die before or during subsequent processing and use.

The conditions specified in this setting are for the die attach (and possibly bonding for non-bare die forms) process, where temperature and pressure is applied evenly over the whole die for a certain duration.

The conditions described in the text string should at least include the temperature, pressure, and voltage (differential with reference to supply pads); along with the maximum duration for each. Humidity and any other environmental conditions may also be specified.

<u>Example(s)</u>

```
die_conditions_process 350 degrees C max, 5 grams / um**2 max,
      no voltage differential over a 45 minute time period ;
```

<u>See Also</u>

die_conditions_process

die_conditions_sealing

die_conditions_special

5.13. die_conditions_sealing

Section:	**Die**	Level:	❷
Suggested not to exceed vicinity conditions for processing of die (vicinity conditions during sealing)			

Syntax

```
<die_conditions_sealing> ::= DIE_CONDITIONS_SEALING <text_string> ';'
```

The sealing conditions are specified in a non-computer sensible form as a text_string. See the lexical analysis document section for more details.

Model

These are the suggested not to exceed conditions for the die before or during subsequent processing and use.

The conditions specified in this setting are for the sealing process, where temperature and pressure is applied in the vicinity of the die but not directly to the die.

The conditions described in the text string should at least include the temperature, pressure, and voltage (differential with reference to supply pads; along with the maximum duration for each. Humidity and any other environmental conditions may also be specified.

Example(s)

```
die_conditions_sealing 350 degrees C max, 5 grams / um**2 max,
      no voltage differential over a 45 minute time period ;
```

See Also

die_conditions_process

5.14. die_conditions_special

Section:	**Die**	Level:	❷
Any special conditions for handling or use of the die			

Syntax

```
<die_conditions_special> ::= DIE_CONDITIONS_SPECIAL <text_string> ';'
```

The special conditions are specified in a non-computer sensible form as a text_string. See the lexical analysis document section for more details.

Model

These are the suggested special conditions for the die before or during subsequent processing and use.

The conditions specified in this setting are for any special cases not covered in other settings and that the manufacturer of a die desires to highlight for the user.

Note:
It is suggested that any electrostatic sensitivity be clearly identified and stated here if desired and not covered anywhere else.

Example(s)

```
die_conditions_special Not to exceed 50 volts potential on any pin ;
```

See Also

die_conditions_bonding

die_conditions_processing

die _conditions_sealing

die_conditions_storage

5.15. die_conditions_storage

Section:	**Die**	Level:	❶
Suggested not to exceed conditions for storage			

Syntax

<die_conditions_storage> ::= **DIE_CONDITIONS_STORAGE** <text_string> ';'

The storage conditions are specified in a non-computer sensible form as a text_string. See the lexical analysis document section for more details.

Model

These are the suggested not to exceed storage conditions for the bare die during subsequent storage and handling. Usually these values represent absolute maximum conditions over longer periods of time then specified in the various process condition settings.

The conditions described in the text string should be at least include the temperature. Humidity and any other environmental conditions may also be specified.

Example(s)

`die_conditions_storage` `250 degrees C max, -70 deg C min ;`

See Also

die_condition_process

die_condition_special

5.16. die_description

Section: **Die**	Level: ❷
Textual description of die features	

Syntax

```
<die_description> ::= DIE_DESCRIPTION <text_string> ';'
```

The die description is a non-computer sensible text_string. See the lexical analysis document section for more details.

Model

The die description is provided to present targeted, feature information about a die. This setting could be utilized to further comfort the user that they have the correct die description information.

Note:

It would be useful to include in the description a classification of the die and / or list of keywords useful for choosing or understanding a given part.

Example(s)

```
die_description SCAN18245T MDA: Serially Access Network Non-inverting
 Transceiver with TRI-STATE® outputs.  TRI-STATE is a registered
 trademark of National Semiconductor Inc. ;
| note that the ® symbol is an ISO character !
```

See Also

die_notes

die_packaged_part_name

5.17. die_fiducials

Section: **Die**	Level: ❶
Die identification marks	

Syntax

```
<die_fiducials> ::= DIE_FIDUCIALS  <polygonal_area_list> ';'

<polygonal_area_list> ::=
            '(' <polygonal_area> ')' {'(' <polygonal_area> ')'}  |
                <polygonal_area>     {',' <polygonal_area>    } ';'
```

See the polygonal_area attribute description in the lexical analysis document section for more information.

Model

This setting identifies any metal feature on the die that is covered by passivation material but still visible. The setting is primarily used to disclose the existence of visible, possibly nonfunctional, metal shapes on the die. Commonly these features will be registration marks, company logo's or other identifying features. They serve to help determine orientation or identification of the die during handling.

There may be more than one of these features visible on the die. Visibility is determined by the ability to distinguish and recognize the feature at 20x magnification or less.

The polygonal_area attribute defines the shape and orientation of one of these metal features by specifying the coordinates of each of the vertices of the feature's outline. The coordinates are in die coordinate space. For complex objects (such as a manufacturer's logo or designer's initials) only a reasonable approximation bounding box of the mark need be described.

Example(s)

```
die_fiducials
      5 150 250 250 250 250 350 150 350 150 250, | a company logo
      7 400 800 450 800 450 1000 650 1000 650 1050 400 1050 400 800 ;
      | a passivation covered metal shape
```

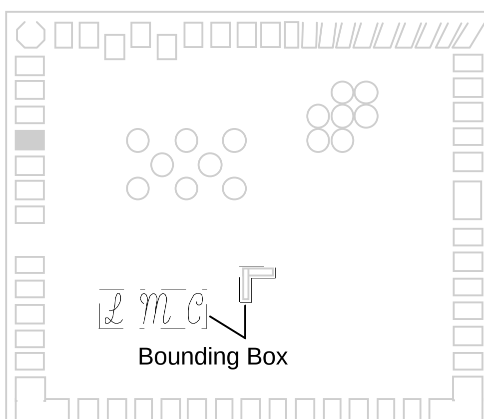


Figure 9: Non functional metal features

See Also

die_bonding_diagram

5.18. die_flatness

Section:	**Die**	Level:	❷
Indication of die surface warpage			

Syntax

```
<die_flatness> ::= DIE_FLATNESS <flatness> ';'
<flatness> ::= <dimension>
```

The flatness is expressed as a dimension attribute. See the dimension attribute document section for more details.

Model

A flat surface is an ideal situation such that all elements of the surface lie in a plane. The top surface of a die may not be flat enough for use in some MCM processes. Often the die may warp after going through the many cycles of heating and cooling typical of the thin film processing. This setting is provided for those MCM processes which consider the flatness of the die top surface critical to their processing technology.

Flatness can be expressed as the minimum separation of 2 parallel planes such that every element on the top surface of the outermost layer of the die must lie in the space between the 2 planes. This property specifies the flatness of the die outer layer.

Note:

The die thickness may or may not contain the warpage of the die in its value. The thickness tolerance (if it exists) may include the possible warpage, if the warpage is minor. Also note that no real indication is given of the "roughness" of the die top surface, especially if the warpage across a large die is less significant than the unevenness due to etching of buried thin-film layers.

Example(s)

```
die_flatness .1 ;
```

See Also

die_thickness

5.19. die_IBIS

Section:	**Die**	Level:	❶
IBIS information about this die			

Syntax

<die_IBIS> ::= **DIE_IBIS** <model_ref> <IBIS_component_name> ';'

<model_ref> ::= <name_value>

<IBIS_component_name> ::= <name_value>

The name value is described in the lexical analysis section of this document.

Model

This setting associates an IBIS file and its corresponding [Component] section with a die. Only one IBIS [Component] section can be associated with a given **die** section. For more information about coding an IBIS file see [IBISv1.1].

The model_ref attribute references a DIE Format model section name; one whose enclosing section contains IBIS data.

The IBIS_component_name attribute identifies the IBIS [Component] section name within the IBIS file that is associated with this die. This name must be a legal IBIS identifier.

Note:

> The associated IBIS [Pin] section within the IBIS [Component] section could, but does not yet, relate IBIS [model] sections with the pads on the die. IBIS has not yet addressed unpackaged die and, as such, has not defined the meaning of the pin number for bare die. It is recommended that the DIE Format pad_ID's be used in the IBIS [Pin] section to facilitate understanding. Currently, pad_ID's are associated to IBIS [Model] sections via the **pad_digital_IBIS_model** setting. An exception to using pad ID's as IBIS pins would be for LEAD_FRAME die when it has its OLB bond locations are identified with a pad ID or pin designation different than the pad itself.

Example(s)

```
die_IBIS scan18245t.ibs SCAN18245T-MDA;
```

See Also

pad_digital_IBIS_model

5.20. die_junction_temperature

Section: **Die**	Level: ⓿
Recommended die junction temperature.	

Syntax

```
<die_junction_temperature> ::=
      DIE_JUNCTION_TEMPERATURE <min_temp> <max_temp> [ <abs_max_temp> ] ';'

<min_temp> ::= <numeric_value>

<max_temp> ::= <numeric_value>

<abs_max_temp> ::= <numeric_value>
```

The minimum, maximum, and absolute maximum temperatures are numeric values. The default unit is C (degrees Celsius). The only units allowed are C or K. See the lexical analysis section of this document for more details.

Model

The setting represents the recommended minimum and maximum operating junction temperature of the die. The die junction temperature is the temperature expected in the electrically active portions of the die. The temperature range specified should not be exceeded for long periods of time during periods of operation. Operation is defined as supply and possibly clock current available to the appropriate pins.

The setting also includes an optional specification for the absolute maximum junction temperature -- a not to exceed condition. This value is specified over any length of time and independent of whether the die is under operation (powered or not). The intent is to provide a guideline for a temperature which, if exceeded, will cause permanent damage to the die.

Example(s)

```
die_junction_temperature 0 100;

die_junction_temperature -55 125 175;
```

See Also

die_conditions_storage

5.21. die_lead_frame

Section: **Die**	Level: ⓿
Beam lead frame description	dep

Syntax

```
<die_lead_frame> ::= DIE_LEAD_FRAME <text_string> ';'
```

The text_string value is not computer-sensible and described in the lexical analysis document section.

Model

The die lead frame setting should describe the physical characteristics of any beam leads already attached to the die. This may be a Tape Automated Bonding (TAB) flexible frame, flat ribbon leads, or even wire bond (circular wire) leads that have been attached to the die and delivered as a component in this form. At minimum, the dimensions of the leads and the material used should be described.

The lead frame description is only required of die designated as LEAD_FRAME in the **die_type** setting.

Note:

In the future, it is expected this setting will become computer sensible. The non-computer-sensible string is provided in this version for backward compatibility.

Example(s)

```
die_lead_frame formed leads,.5in long, aluminum;
```

See Also

die_type

5.22. die_load_factor

Section:	**Die**	Level:	❷
Output load dependency of the die power dissipation			

Syntax

```
<die_load_factor> ::= DIE_LOAD_FACTOR <factor> ';'
<factor> ::= <numeric_value>
```

The factor is a unit-less numeric value between 0 and 1.

Model

This setting is a numeric constant created to take two factors into account. The first is to provide an estimate of the probability that a signal output pad will switch within a clock period. The second is to account for the effect that the output voltage does not switch from rail-to-rail. This setting represents the value for K in the power dissipation equation of document section 1.2.5.

The value is meant to be used in the power dissipation equation only and is not intended to have meaning in any other context.

Note:

> This factor can represent the summed then averaged factors for each individual signal pad. It is suggested a user determine a more plausible estimate for each signal pad individually and multiply the value by the pad's actual load capacitance in a design. The results for each pad can then be summed together to create a more accurate $K * C_{LOAD}$ value for the power dissipation equation.

Example(s)

```
die_load_factor 0.2;
```

See Also

die_power_capacitance

die_quiescent_current

5.23. die_lot

Section:	**Die**	Level:	❷
Lot number's of die this information pertains to			

Syntax

```
<die_lot> ::= DIE_LOT <text_string> ';'
```

The die lot is a non-computer sensible text_string. See the lexical analysis document section for more details.

Model

The setting contains information describing the wafer processing lot(s) that die conforming to the die section of information pertain to. This would normally be a more refining designation than the mask version specification. It is only used if a manufacturer wishes to qualify information to a specific subset of lots that use the same processing masks.

Note:
This setting may be used by some manufacturers as an additional primary key for the **die** section.

Example(s)

```
die_lot Lot #'s 922403 - 923401 are covered by this section ;
```

See Also

die_mask_version

5.24. die_manufacturer

Section: **Die**	Level: ⓿
Manufacturer's name	pri

Syntax

```
<die_manufacturer> ::= DIE_MANUFACTURER <text_string> ';'
```

The manufacturer name is represented by the general text_string value. See the lexical analysis document section for more details.

Model

A non computer-sensible but interpreted string identifying the die manufacturer, division and possibly site.

The manufacturer is not a processable piece of information in that corporate, division, and product category naming issues are not defined. The information is provided here purely as comment form for human consumption. A parser can identify the information and pass it on for informational message purposes to help the user understand what is being processed.

Also, the manufacturer name is one of the keys used to uniquely identify die information. In that respect, the names should be consistent as possible among different die from the same manufacturer.

Note:

The information is considered current as of when the die information is created. The user should always check with the manufacturer on the current availability of any device.

Example(s)

```
die_manufacturer Hewlett-Packard Corporation, Roseville Division ;
```

See Also

die_manufacturer_cage

die_mask_version

die_name

die_type

5.25. die_manufacturer_cage

Section: **Die**	Level: ❷
CAGE code of manufacturer	

Syntax

```
<die_manufacturer_cage> ::= DIE_MANUFACTURER_CAGE <cage_code> ';'
<cage_code> ::= { <ISO_digit> }4-5
```

A cage code is a series of four or five digits.

Model

The Commercial and Government Entity (CAGE) code associated with the manufacturer of the die. This is provided to further assist in identifying the source of the die and possibly die information contained within the same section. This value is computer sensible.

Example(s)

```
die_manufacturer_cage 54673;
```

See Also

die_manufacturer

5.26. die_mask_version

Section: **Die**	Level: ⓿
Manufacturer's published mask step version of die	pri

Syntax

```
<die_mask_version> ::= DIE_MASK_VERSION <version_attribute> ';'
```

The version_attribute is described in detail in the document section on attributes.

Model

The manufacturer's published mask step version should be documented in this setting. The date and time are optional, informational values. The revision value should be specified in as close a format to the original die manufacturer version specification as possible.

Given this setting is a primary key of the **die** section, it should be used to identify unique die products. If there is a change in a die product that may or may not cause the die information to change, then the mask version should be updated. If there is solely a change to die information without a change to the product, the **die_section_version** setting should be updated.

Note:

> The value generally represents the published or marketing assigned mask step version. The internal step version used by engineering generally changes often and may not affect the information in the DIE Format. Therefore, the published name is the most useful guideline to use.

Example(s)

```
die_mask_version C-1 12/2/92 ;
```

See Also

die_manufacturer

die_name

die_type

die_section_version

5.27. die_measure

Section:	**Die**	Level:	❷
Lot specific die measurements			

Syntax

```
<die_measure> ::= DIE_MEASURE <model_ref> { <model_ref> } ';'
<model_ref> :: = <name_value>
```

The name value is described in the lexical analysis section of this document.

Model

This setting identifies any included TI Measure file external format models that may be attached the **die** section. For more information on the lot specific, die measure file see [TIMEASURE].

The model_ref attribute(s) specify one or more DIE Format **model** sections which contain TI Measure file data. The referenced models' section type should be MEASURE. The pad ID's identified in the measure file should correspond to the ID's in the **die_pads** setting.

Note:

The measure file provides lot specific measured data of pad locations and other information. This has mainly been developed for the chips-first community which need accurate data to form thin film processing masks or to guide direct laser write systems. Their technology needs to make contact cuts (through thin film polymers) to the pads for module pad bonding.

Example(s)

```
die_measure s18245_lot9812 s18245_lot9815;
```

See Also

die_lot

die_pads_noncontact_area

die_separation

5.28. die_military_spec

Section: **Die**	Level: ❷
Military part specification, if it exists	

Syntax

<die_military_spec> ::= **DIE_MILITARY_SPEC** <text_string> ';'

The military part specification is a text_string. See the lexical analysis document section for more details.

Model

The military specification(s) part name is provided to assist in gathering additional information about the die, such as the conventionally packaged part data sheets or Military Spec (slash ('/') or SMD) sheets.

One or more names may be listed, if they apply. This is not intended to be a computer-sensible setting although it should be human understandable.

Example(s)

```
die_military_spec 38510-2010 ;
```

See Also

die_packaged_part_name

5.29. die_name

Section: **Die**	Level: ⓿
Manufacturer's published name for die	pri

Syntax

```
<die_name> ::= DIE_NAME <text_string> ';'
```

The die name is a text_string. The text_string value is described in the lexical analysis document section.

Model

A unique name for the die. Although not computer sensible, the name must be unique enough to be used as a key for the **die** section information.

This value should normally be the manufacturer's order number or a subset of this number. Sub-setting of the number can occur by dropping designations such as the delivery format (for example, tape or reel) or quality version (such as MIL STD 883 processed, burned-in, etc.). The name should be what uniquely identifies and ties together a physical die and its description in a DIE Format block.

Example(s)

```
die_name scan18245T-MDA;
```

See Also

die_manufacturer

die_mask_version

die_type

5.30. die_notes

Section: **die**	Level: ⓿
General notes about a die	

Syntax

<die_notes> ::= **DIE_NOTES** [<text_string>] ';'

The text can be a text_string or empty **--** the value is optional. See the lexical analysis document section for more details.

Model

The die notes setting stores information of general interest to the users of the die. Unlike comments which would probably be stripped out when a die section is processed, the **die_notes** information should be retained with the other data.

The notes setting may contain special descriptions about the technology information related to this die, or possibly additional explanations as to the source of the die information.

Although this setting is required, the actual text_string value is optional. This allows an empty value for the setting and forces a conscious decision to leave the notes information blank.

Example(s)

```
die_notes Revision history:
      2.1   16/9/92     LCW   Corrected die 1 corner pads center
      2.0   31/8/92     REH   Added description of saw for die 3
      1.0   14/3/92     SC    Initial version for Joe's Machine, Inc. ;
```

See Also

block_notes

5.31. die_packaged_part_attach

Section: **Die**	Level: ❷
Packaged part method(s) of attaching bare die to carrier	

Syntax

<die_packaged_part_attach> ::= **DIE_PACKAGED_PART_ATTACH** <text_string> ';'

The text_string is not computer sensible. See the lexical analysis document section for more detail.

Model

Specifies the material and method used for die attachment in conventionally packaged parts by the manufacturer. This is only serves as a possible suggestion for a die attachment material or method. The decision on the method for others to use depends on the MCM process and technology requirements.

This setting is not intended to indicate the bonding method. Therefore solder bumped or flip chip die would not necessarily utilize this setting in the **die** section.

Note:

The description can be as detailed or vague as the manufacturer desires. The value should not be construed by the user as a recommendation; only a documentation of what was used under one set of circumstances.

Example(s)

die_packaged_part_attach epoxy;

See Also

die_backside_finish

die_conditions_process

die_packaged_part_bonding

die_packaged_part_name

5.32. die_packaged_part_bonding

Section:	**Die**	Level:	❶
Pad bonding method(s) used for packaged die			

Syntax

```
<die_packaged_part_bonding> ::= DIE_PACKAGED_PART_BONDING <text_string> ';'
```

The bonding method value is a text_string that is not computer sensible. See the lexical analysis document section for more detail.

Model

Specifies the material and method used for bonding to a bare die pad in conventionally packaged parts by the die manufacturer. This is only serves as a suggestion for a pad bonding material or method. The decision on the method for others to use depends on the MCM process and technology requirements.

This setting can be used to indicate how a bare die is attached to a lead frame by the die manufacturer.

Suggestive, descriptive terms common to the industry should be used. Ones like "wire_bond", "wire_bond_wedge", "wire_bond_ball", "wire_bond_ultrasonic", "wire_bond_thermosonic", "wire_bond_thermocompression", "TAB", "Solder TAB", "TAB_Thermode_solder", "TAB_Thermode_conductive_adhesive", "TAB_single_point_bond", "C4", and "Solder Ball".

Material names of the bond wire are also suggested such as "gold", "aluminum", and "lead/tin solder".

Example(s)

```
die_packaged_part_bonding wire_bond_ball gold ;
```

See Also

die_conditions_bonding

5.33. die_packaged_part_name

Section:	**Die**	Level:	⓿
Name(s) of conventionally packaged parts which use this die			

Syntax

<die_packaged_part_name> ::= **DIE_PACKAGED_PART_NAME** <text_string> ';'

The text_string that is not computer sensible. See the lexical analysis document section for more details.

Model

The packaged part name is provided to assist in gathering additional information about the die, such as conventionally packaged part data sheets or component information database references.

As described in the setting for **die name**, the **die_packaged_part_name** should be as close as makes sense to the order number(s) provided by the manufacturer. See the **die_name** setting for more details.

One or more applicable names may be listed.

Example(s)

```
die_packaged_part_name SN74A04AT;
```

See Also

die_military_spec

die_name

5.34. die_pad_dielectric

Section: **Die**	Level: ❷
Composition and thickness of dielectric material under the pad	

Syntax

<die_pad_dielectric> ::= **DIE_PAD_DIELECTRIC** <material_list> ';'

The material_list attribute is described in the lexical analysis document section.

Model

The die pad dielectric setting defines the composition and thickness of the dielectric material under the metal pad. The number and ordering of dielectric layers and the specific makeup and thickness of each layer is specified with the material_list attribute. The dielectric material layer(s) occur between the pad metal and the bulk substrate. If the bulk substrate is specially "doped" below the pad, then this should be described as a material layer also, if deemed necessary.

Example(s)

```
die_pad_dielectric   Polyimide 2.5;
```

See Also

die_pad_metal

die_substrate_material

pad_geom_metal_extent

5.35. die_pad_metal

Section:	**Die**	Level:	❷
Composition and thickness of pad metal layer			

Syntax

```
<die_pad_metal> ::= DIE_PAD_METAL  <material_list> ';'
```

For a detailed explanation of the material_list attribute, see the description in the lexical analysis section.

Model

The die pad metal setting defines the composition and thickness of the pad metal material. The number and ordering of metal layers and the specific makeup and thickness of each layer is specified with the material_list attribute. The pad metal material layer(s) occur above the dielectric material.

Example(s)

```
die_pad_metal   2  "Al 99.7% Cu 0.3%" 5  Au 3um;
```

See Also

die_pad_dielectric

die_passivation_material

pad_geom_metal_extent

pad_geom_shape

5.36. die_pad_pitch

Section:	**Die**	Level:	❶
Minimum center-to-center spacing of pads on the die			

Syntax

```
<die_pad_pitch> ::= DIE_PAD_PITCH <dimension> ';'
```

The dimension attribute is described in the attribute section of the document.

Model

The pad pitch setting describes the minimum pad-to-pad separation. The value should represent the minimum center to center distance between any two bonding points on the die pads. Although this value is derivable, it is provided as a convenience to aid in MCM manufacturing technology assessment.

Note:

The setting does not presume there is a regular pad pitch. When regularly spaced pads exist, this value should represent the regular pad pitch. The value is provided as a rough, quick technology assessment guide. The value is used most often with wire-bond pad connection techniques to determine if the pad pitch is course enough for the equipment to be used.

Note:

The value can be approximated by finding the minimum distance between any two pad center location coordinate points given in the **die_pads** setting. The setting is more accurately determined by taking into consideration the bonding sites described in the **pad_geom_bond_sites** setting.

Example(s)

```
die_pad_pitch 4mil;
```

See Also

die_pads

pad_geom_bond_sites

5.37. die_pads

Section: **Die**	Level: ⓿
Instances of pads for die	

Syntax

```
<die_pads> ::= DIE_PADS <num_pads> <pad_descr_list> ';'
<num_pads> ::= <numeric_value>
<pad_descr_list> ::=  <pad_descr>     { ',' <pad_descr> } |
                  '(' <pad_descr> ')' { '(' <pad_descr> ')' }
<pad_descr> ::= <pad_ID> <pad_geom_inst> <pad_elec_inst> <pad_info>
<pad_ID> ::= <ID_value>
<pad_geom_inst> ::= <pad_geom_ref> <point> <rotmir>
<pad_geom_ref> ::= <name_value>
<pad_elec_inst> ::= <pad_type> [ <pad_elec_ref> ]
<pad_type> ::= SUPPLY_POWER | SUPPLY_GROUND | SIGNAL_DIGITAL |
               SIGNAL_ANALOG | TEST_POINT | NO_CONNECT | NOT_DEFINED
<pad_elec_ref> ::= <name_value>
<pad_info> ::= [ <pad_common_name> [ <pad_swap_code> ] ]
<pad_common_name> ::= <name_value>
<pad_swap_code> ::= <swap_function> <pin_type> <pin_group>
<swap_function> ::= <numeric_value>
<pin_type> ::= <numeric_value>
<pin_group> ::= <numeric_value>
```

The num_pads must be a positive (non-zero, non-negative) integer. There should be the stated number of pad descriptions following.

A pad_ID must be an ID_value used to uniquely identify the pad in the list of pad_descr's.

The pad_geom_inst construct defines an instance of a previously defined pad geometry. The pad_geom_ref is a name_value previously defined in a **pad_geom** section. The point is a pair of coordinate values (X then Y) in the die's coordinate space. See the lexical analysis section for more details on the values; especially the rotation and mirror (rotmir) lexical construct.

The pad_type is one of a fixed list of enumerated name values. The pad_elec_ref is a name_value previously defined in a **pad_digital** or **pad_supply** section. The pad_elec_ref should not be used for non-supply or signal type pads.

The pad_common_name is an optional name_value. The pad_swap_code is a triplet of positive, integer numeric values: swap_function, pin_type, and pin_group.

See the lexical analysis and attribute document sections for more details.

Model

The **die_pads** setting describes the pads (passivation openings to metal electrical contact areas, see the model section) of the die. Note that this does not detail the number of bond sites for a given pad nor does it guarantee a pad listed is bondable. The number of pads to be listed is given first (num_pads) followed by a pad description (pad_descr) for each pad.

Pad Descriptors

Each pad descriptor (pad_descr) describes one of the pads. The pad_ID must be unique to all other pad_ID's in the setting. See the suggested convention for assigning pad_ID's to pads given later in this description. This setting defines a pad_ID for each pad.

An ID assigned to a pad should not consist of the single digit zero ('0'). This special ID value is used to indicate that no pad is specified in some settings with pad lists. Also, no pad ID should lead with a digit zero ('0') unless it is this special value.

Pad Geometry Instance

The pad_geom_inst is a reference to a previously defined pad_geom section. The reference is by matching a pad_geom_ref value to a **pad_geom_name** name value.

The point following the name reference defines the center of the pad geometry referenced in die coordinates. Graphically, it is the translation specification of the referenced geometry description from the pad's geometry coordinate space to the die coordinate space.

The rotation and mirror (rotmir) construct allows a pad geometry to be rotated and then mirrored. This is useful for regular shaped, rectangular or polygonal pads that are instanced in different places on the die with different orientations. See the basic model document section and the description of the rotmir construct for more details.

Pad Electrical Model Instance

The pad_elec_inst construct defines the pad electrical type and possibly a reference to a pad electrical model definition. The pad type indication is used to determine which type of section the pad_elec_inst references. The pad_elec_ref name must be identical to a previously defined pad electrical section definition name.

All pads must be described with a type. The type gives an indication as to the intended logical use of the pad.

SIGNAL pads provide the electrical connect points to utilize the functionality designed into the die. Analog and digital signal pads are differentiated due to the different conventions for specification and modeling taken. The pad_elec_ref name for a digital signal pad must have been previously defined in a **pad_digita**l section. Currently, SIGNAL_ANALOG type pads do not have a referenceable pad electrical definition. The name is not defined (but must still be specified) if the pad type is SIGNAL_ANALOG.

SUPPLY pads are intended to be passive electrical connect points which provide the electrical supply sources from which the active circuitry can reference. Only two types of supply pads are defined -- POWER and GROUND. Note that there can be more than a single voltage POWER pad types. By definition, all SUPPLY pads with the same pad_elec_ref name are meant to be jumped or connected to the same source externally. The pad_elec_ref name can only reference a previously defined **pad_supply** section name.

TEST_POINT pad types are meant to describe process characterization or other IC manufacturer specific test points. These pads are either not expected to be bonded or are only to be used during special, non-functional periods. The points are not further defined in the DIE Format unless the IC manufacturer chooses to do so through the **die_notes** setting. The pad_elec_ref is not meant to be a meaningful name when the pad type is TEST_POINT.

Note:

IEEE 1149.1 (JTAG) test access port pins are of type SIGNAL_DIGITAL (not TEST_POINT).

NO_CONNECT pad types define a pad which should be electrically isolated. It may be a test point that the manufacturer chose not to identify or provided for die stabilization on a SOLDER_BUMP die.

The NOT_DEFINED pads are just that, not defined in any way.

Only NO_CONNECT and NOT_DEFINED pads can avoid including a pad_elec_ref name. These pad descriptors must also not include any pad_info if the pad_elec_ref name is not included.

For SIGNAL_ANALOG and TEST_POINT, the pad_elec_ref is currently not defined. It is suggested a dummy name value place holder such as NO_REF be used.

Note:

A pad is defined as a passivation opening coincident with a metal connect point. Passivation covered metal pads are not allowed to be defined here but should usually be defined in the **die_fiducials** setting. Other passivation openings which do not have metal pads coincident are not describable in the DIE Format.

Pad Common Names

The pad common name is an optional value and intended to represent the functional or symbol pad name normally associated with the data sheet or other descriptive information. The common name need not be unique. All supply pads may be designated as Vcc for example. There is no interpretation given to this name. It is an error if the common name is defined for use or association with other computer sensible, die information.

The following conventions are encouraged:

<base_name> [<positive, integer index>]	To represent a bit in a bus or other grouping of pads with a common <base_name>
Vcc, GND, Vee, etc.	For supply pins (see EIA JEDEC standards)
<base_name>*	To represent a low-true signal_digital type (otherwise assumed high-true)

Swap code

The optional swap code values -- swap_function, pin_type, and pin_group -- represent important layout and routing information usually associated with a package pin number or schematic symbol pin name. The codes allow a router to swap external signal connections to the pads in a defined, logical manner that will not affect the final, intended functionality of the die but which may allow an automatic router to more optimally route the connections to the die.

Note:

Usually this "swappability" is limited to small or medium scale integration parts (buffers, gates, etc.). But sometimes, large microprocessor or memory devices can have their address or data bus lines defined as swappable without affecting the final functionality.

The swap_function is used to partition groups of pads which are associated with essentially disjoint functional units within an IC. All the pads with the same swap_function value can be swapped for all the corresponding pads with a different (but same) swap_function value. Correspondence is determined via the pin_type and pin_group designations.

The different functional units may share common control or supply pads. Swap function code 0 is reserved for common supply and signal pins. The pin type within this special swap_function grouping is used to identify the differences (such as ground, power supply 1, power supply 2, etc.).

Pads defined with the common swap_function code cannot be used to select functionality uniquely or differently in one functional unit than from the other. That is, the individual functional units must act electrically as if they were on separate die and electrically connected to the common signals the same way with no intervening logic.

Note:

There is no correspondence intended here between swap code designations on one die versus those on another. The codes are not intended to define similar functional units between two different die.

The pin_group is used to identify common group of pads within a functional unit. This is the next level of partitioning of pads -- partitioning within a swap_function unit. Pads within a functional unit but in different pin_group's can be swapped with each other.

The special pin_group value of '0' is reserved for those pins which are not part of any swappable group. They are unique to the functional unit and as such should be left alone unless the functional unit itself is swapped with another functional unit.

The pin_type is used to classify pads with a pin_group which must be matched up when swapping two different pin groups. Zero ('0') is a pin_type value reserved for those pins which are not swappable under any circumstances.

Common electrical pads should be defined with a common pin_type. All pads with the same swap_function, pin_group and pin_type value triplet are swappable among themselves without causing any change to the functionality. For example, the common inputs to any basic logic gate (and, nand, or, nor or xor) can all be defined with the same swap_function code.

It is an error for different functional units to have unique pin_type or pin_group values -- they must be common among functional units that are swappable.

Functional units with no possibility for swapping of any pads can either not have this value specified or all pads should have a code specified of '0 0 0' which essentially identifies every pad as being unique and un-swappable. Any pad descriptor without a swap code specified has a default value of '0 0 0'.

Common supply pads can be defined as swappable pads, if desired. They should be given a swap_function and pin_group of zero ('0'). Then each supply group (that is, with an identical pad_elec_ref name) should be given a unique (to the die), non-zero pad_type. This would defined pads which are swappable among themselves and global to any swappable functional units. Similarly, common signal pads could be defined in the same way.

For example, if a two dual buffers are defined in the same package, then the pads for each dual-buffer would be defined with a common swap_function value. Within a dual-buffer (functional_unit) the first buffer would have its input and output pads defined with the same pin_group value. Correspondingly, the second buffers' pads would be defined with a second pin_group value. Finally, the input pads of either of the buffers of a dual-buffer functional unit would have the same pin_type value; the output pads another common pin_type value. The input pads' pin_type value in one dual buffer must be identical to the corresponding pads' pin_type value in the other dual-buffer if the functional unit's are to be swappable.

Pad ID assignment sequence

There is a suggested convention for assigning pad_ID's. Although not required, the following practice will ease the absorption of the information by end users. In the end, best judgment must be used by the information provider on how to convey the most information.

For periphery pad die, it is suggested that the number 1 be used for the pad that is left and upper most in the die. If no pad exists in the corner, the one that would be just below the corner (on the left side) should be used. Increasing numeric ID's should be assigned as one progresses down the left side of the die, across the bottom, up the right side and across the top; all in a counter-clockwise direction. If two or more rows of pads on the periphery exist, then the outer row should be assigned first, followed by the second, and so on -- thus creating a spiraling-in effect for assigning ID's.

For array pad die, it is suggested that the number 1 be used for the pad that is the left most in the top row of pads on the die. The pads should then be traversed a horizontal row at a time in a downward vertical direction -- scanning in a similar fashion to the scan on video monitors. Within each horizontal row, ID's of increasing numerical order should be assigned starting with the leftmost pad and proceeding to the right. The rightmost pad in the last row (lowest vertical) should have the highest ID.

For irregularly spaced or placed pads, determining a "row" or "column" may be difficult. Although subjective, it is suggested that if, when viewing the bonding diagram, two pads appear in the same row or column, they be considered to be in the same row or column. If there appears to be a single row or column of pads on the periphery, but some are offset, then the offset pads should be considered as part of the main row or column. If a pad is obviously offset far away from the main row or column, then consider it a separate row. "Obviously" might be judged as more than a two pad width (or height) center-to-center separation between pads.

Example(s)

```
die_pads 4
      1 4umX4um  240 1700 0 supply_power   VCC_model   Vcc        0 0 1,
      2 4umX4um  240  240 0 signal_digital cmos_input  Serial_in  1 0 3,
      3 4umX4um 1700  240 0 supply_ground  GND_model   GND        0 0 2,
      4 4umX4um 1700 1700 0 signal_digital cmos_output Serial_out 1 0 4;
```

5.38. die_pads_jumpers

Section: **Die**	Level: ❷
Suggested pad interconnects.	

Syntax

```
<die_pads_jumpers> ::=  DIE_PADS_JUMPERS <interconnect_lists> ';'
<interconnect_lists> ::=
             '(' <jumper_list> ')' {'(' <jumper_list> ')'}  |
                 <jumper_list>     {',' <jumper_list>    }
<jumper_list> ::= <pad_ID>  <pad_ID> { <pad_ID> }
<pad_ID> ::= <ID_value>
```

ID_value's are described in the lexical analysis section of this document.

Model

This setting lists all the die pads that should be jumper'ed together at the MCM assembly level. Each unique interconnect group is defined with a separate jumper_list attribute. As many jumper_list attributes as desired can be defined.

Die pads within each group are referenced via their respective pad_ID. A pad_ID here must match with a corresponding pad_ID in the **die_pads** setting. It would be an error to list the pad_ID for a no_connect or not_defined type pad. A pad_ID can only appear once in the **die_pads_jumpers** setting. Not all pads need to be jumper'ed together and not all pads are required in these lists. Any pads not listed in this setting are considered not to need direct MCM assembly level interconnections for proper operation.

Each interconnect group is syntactically separated from the others via either enclosing parenthesis or separating comma's. A jumper_list group must contain at least two pad_ID's. All pads within a group are assumed to be jumper'ed together in the order defined in the **die_bonding_sequence** setting.

Note:

> If a pad needs to be connected to a supply pad, it is sufficient to list only a single supply pad of the appropriate type, voltage, etc. If two different pads each need jumpering to supply pads and are really independent specifications, then different supply pads can be used. Otherwise, both pads would need to be in the same group as the supply pad.

Example(s)

```
die_pads_jumpers (4 3 1) (2 5) ;
die_pads_jumpers 5 4 3 1, 2 5;
```

See Also

die_bonding_sequence

die_pads

5.39. die_pads_noncontact_area

Section:	**Die**	Level:	❷
Areas on pad metal surface which are not suitable for contact			

Syntax

```
<die_pads_noncontact_area> ::=
      DIE_PADS_NONCONTACT_AREA <num_areas> <area> { <area> } ';'

<num_areas> ::= <numeric_value>

<area> ::= <pad_ID> <diameter> <point>

<diameter> ::= <dimension>
```

The num_areas is a positive, integer numeric value. There should be num_areas area attributes following this value.

The area description is composed of a fixed number of values. The pad_ID is a reference to an ID_value. The diameter is defined simply with a dimension attribute. The point attribute is in the die's coordinate system.

See the attribute and lexical analysis sections of the document for more details on each attribute or value.

Model

This setting is used to describe the areas on the surface of the metal pads which are no longer original or "smooth" as would be immediately after the thin film processing used to develop the pad. Most likely the surface has been changed due to probing. For some bond pad attach processes, the probe corrupted area cannot be used as a bond site (for example, a thin film "cut" down to surface) and so must be described as a non-contact area.

The pad_ID references a pad defined in the **die_pads** setting in the same die section. There is no order required on the pad_ID's referenced in the area attributes.

The diameter defines the diameter of a circular region that is intended to cover or enclose the non-contact area. Note that the circular geometry referenced here has no relationship to the actual pad geometry referenced in the **die_pads** setting. It is not an error if the geometry defined here overlaps with the defined **pad_geom_shape** boundary.

The x and y coordinate point is the location of the center of the circle defining the non-contact area. The diameter and point are described in the die's coordinate system.

Example(s)

```
      die_pads_noncontact_area 1 3 .5um -1 -1 ;
```

See Also

5.40. die_pads_supply_grouping

Section:	**Die**	Level:	❶
Supply pad groups.			

Syntax

```
<die_pads_supply_grouping> ::=
            DIE_PADS_SUPPLY_GROUPING  <supply_group_list> ';'

<supply_group_list> ::=
            '(' <supply_descr> ')' { '(' <supply_descr> ')' }  |
                <supply_descr>     { ',' <supply_descr>     }

<supply_descr> ::= <pad_ID> <power_ID> <ground_ID>

<pad_ID> ::= <ID_value>

<power_ID> ::= <ID_value>

<ground_ID> ::= <ID_value>
```

ID_value's are described in the lexical analysis section of this document.

Model

This setting lists the power and ground supply pads which are most electrically significant to the referenced signal pad. The indication provides a rough approximation to build a circuit model for studying the pad signal and supply integrity of the die in a design. All signal pads must be included in this list although no particular ordering is required.

If no electrically significant connection exists between a supply rail and the pad, a "0" should be used. For example an open-drain output has no connection from the power supply rail to the output transistor and should use '0' for the power_ID value.

Die pads are referenced via their respective pad_ID and a pad_ID here must match with a corresponding pad_ID in the **die_pads** setting. The pad type description in the die_pads settings must match its use here. Each supply group is syntactically separated from the others via either parenthesis' or comma's.

Example(s)

```
die_pads_supply_grouping
            1     5     6,
            2     5     6,
            3     0     6,
            4     0     6;
```

See Also

die_IBIS

die_pads

5.41. die_pads_tolerance

Section: **Die**	Level: ❶
Tolerance of pad placement	

Syntax

<die_pads_tolerance> ::= **DIE_PADS_TOLERANCE** <tolerance> [<tolerance>] ';'

The tolerance is presented in detail in the attribute section of this document.

Model

The die pads tolerance represents the tolerance for the placement (the center points) of the die pad geometry.

When only a single tolerance is specified, the single tolerance is for the X and Y dimensions. When two tolerances are specified, each specifies a separate tolerance for the X and Y dimension. The X tolerance always comes first if both are present.

Note that the tolerance for the pad geometry itself is specified in the pad_geom section.

Example(s)

```
die_pads_tolerance  .1 ;
```

See Also

die_pads

pad_geom_tolerance

5.42. die_pads_VHDL_map

Section:	**Die**	Level:	❶
VHDL entity port to pad ID mapping			

Syntax

```
<die_pads_VHDL_map> ::=
            DIE_PADS_VHDL_MAP  <VHDL_map_pair> { <VHDL_map_pair> } ';'
<VHDL_map_pair> ::= <VHDL_entity_name> <BSDL_package_decl>
<VHDL_entity_name> ::= <name_value>
<BDSL_package_decl> ::= <name_value>
```

The name value is described in the lexical analysis section of this document.

Model

This setting defines the mapping between VHDL entity ports and die pads. More than one mapping may exist because there may be several VHDL entities that represent the die functionality.

The VHDL entity name value identifies a VHDL entity that represents the die behavior. This entity must be contained in one of the VHDL model sections specified by the **die_VHDL** setting. The BSDL_package_decl attribute identifies a BSDL-style package pin definition name (a VHDL constant) within the entity declaration named as part of the map. See [BSDL1149] for more information on how to code BSDL package pin definitions. The BSDL package pin definition associates the entity ports in the VHDL model with the die pads on the die via the pad_ID numbers.

Since multiple VHDL entities may exist that represent the die, additional map pairs are allowed so as to describe the (possibly) different pin mappings for each entity.

All name values must be legal VHDL identifiers.

Example(s)

```
die_pads_VHDL_map s18245t_e1  mda_package  s18245t_e2  mdb_package ;
```

See Also

die_VHDL

die_BSDL

5.43. die_passivation_extent_size

Section:	**Die**	Level:	❷
Extent of passivation if different from die size			

Syntax

```
<die_passivation_extent_size> ::=
            DIE_PASSIVATION_EXTENT_SIZE  <width> <height> <point> ';'
<width> ::= <dimension>
<height> ::= <dimension>
```

For a detailed explanation of dimension and point, see the respective description in the attribute document section.

Model

This setting defines the location and size of the die passivation cap over the surface of the actual die substrate. Width and height specify the rectangular size of the passivation region. The point specifies the horizontal (X) and vertical (Y) offset from the center of the die to the center of the passivation cap rectangle. It is presumed the die and passivation cap have parallel edges.

This value will be different than the die size anytime the expected or actual die substrate extends beyond the passivation cap.

Example(s)

```
die_passivation_extent_size  4100 3900   50 -75;
```

See Also

die_size

die_step_and_repeat

die_passivation_material

5.44. die_passivation_material

Section: **Die**	Level: ❶
Material list of passivation layer	

Syntax

```
<die_passivation_material> ::= DIE_PASSIVATION_MATERIAL <material_list> ';'
```

See the description of the material_list attribute in the attribute section for more details. Either the multi or single layer form of the attribute is allowed.

Model

The passivation material setting describes the outermost coating(s) of the die. The description, at minimum, consists of the outermost layer material name with an optional thickness specification.

The multi-layer passivation material setting describes the top layers of the die above the active surface material. The description consists of a material name and thickness specification for each layer.

Sometimes, multiple material layers are used to provide a better insulating and hermetic sealing than provided for with a single layer. The attributes multi-layer form allows a more detailed description of the material layers to occur.

Note:
The multiple layer form is expected for SOLDER_BUMP die type descriptions.

Example(s)

```
die_passivation_material silicon_nitride 2um;
die_passivation_material 2 silicon_nitride 2um silicon_dioxide 1;
```

See Also

die_backside_finish

die_substrate_material

die_passivation_extent_size

5.45. die_power_capacitance

Section:	Die	Level:	❷
Frequency dependency of the internal die power dissipation.			

Syntax

```
<die_power_capacitance> ::= DIE_POWER_CAPACITANCE <power_capacitance> ';'
<power_capacitance> ::= <numeric_value>
```

Power capacitance is a positive numeric value. The default and only unit allowed is farad.

Model

This setting represents the frequency dependency of the internal power dissipation of a die. It is expressed as the C_{PD} value in the power dissipation equation of section 1.2.5.

This value is intended to be used in the power dissipation equation only and is not intended to have meaning in any other context.

Note:

> C_{PD} is commonly provided with MOS technology parts as a power dissipation specification. For other technologies, it can be derived by solving the power dissipation equation for other values during static or near static operation with known loads.

Example(s)

```
die_power_capacitance 10pF;
```

See also

die_load_factor

die_quiescent_current

5.46. die_power_cell

Section: **Die**	Level: ❷
Nominal power dissipation of specific areas.	

Syntax

```
<die_power_cell> ::= DIE_POWER_CELL <power_area_list> ';'
<power_area_list> ::=
             '(' <power_area> ')' {'(' <power_area> ')'}  |
                 <power_area>     {',' <power_area>    }
<power_area> ::= <power> <polygonal_area>
<power> ::= <numeric_value>
```

Power is a positive numeric value. The default and only unit allowed is watts.

The polygonal area is an attribute and is specified in die coordinates. See the attribute section of this document for more details.

Model

This setting allows for a more detailed specification of the nominal power dissipation as distributed across the die. The nominal power dissipation for specific area's of the die can be listed. This refinement allows for a more detailed analysis by pinpointing potential hot areas on the die active surface.

Any area specified in this setting should be within the limits of the **die_size**.

The operating conditions specified in the **die_power_nom** setting apply to this setting also.

Note:
The power dissipation of any unspecified area's of the die should normally be the balance of **die_power_nom** settings' value minus the sum of the power of all the power_area specifications.

Example(s)

```
die_power_cell 0.2W
            5 5300  4400
              5300 -3000
             -2000 -3000
             -2000  4400
              5300  4400;
```

See Also

die_power_nom

die_power_max

die_size

5.47. die_power_max

Section: **Die**	Level: ⓿
Maximum thermal power dissipation	

Syntax

```
<die_power_max> ::= DIE_POWER_MAX <power> <condition> ';'
<power> ::= <numeric_value>
<condition> ::= <text_string>
```

Power is a positive numeric value. The default and only allowed unit is Watts.

The condition is a non-computer sensible text_string. See the lexical analysis section for more details.

Model

The setting conveys the absolute maximum power the die can be expected to dissipate when operated within the limits of the stated operating conditions.

The condition attribute is used to state any operating condition(s) within which the maximum power dissipation occurs. For example, the configuration of any internal state, the supply voltage, the junction temperature operating point or range, test patterns used, or any external loading may be described. The condition is used to provide additional information which the information provider considers important to the understanding of the power value. If no conditions are specified, then the die operating condition limits are presumed and other (unspecified) conditions are left at unknown values.

Note:

The maximum power dissipation is not the sum of the possible maximum power dissipation of individual areas of the die. Each die area maximum power dissipation may occur at different conditions.

Example(s)

```
die_power_max 2W f=1/tRC, CS=VIL, outputs open;
```

See Also

die_power_nom

die_quiescent_current

5.48. die_power_nom

Section: **Die**	Level: ⓿
Nominal thermal power dissipation	

Syntax

```
<die_power_nom> ::= DIE_POWER_NOM <power> <condition> ';'
<power> ::= <numeric_value>
<condition> ::= <text_string>
```

Power is a positive numeric value. The default and only unit allowed is Watts.

The condition is a non-computer sensible text_string. See the lexical analysis section for more details.

Model

The setting provides the nominal power dissipation expected of the die for the stated operating conditions.

The condition value is used to state any operating condition(s) within which the nominal power dissipation occurs. For example, the configuration of any internal state, the supply voltage, the junction temperature operating point or range, test patterns used, and of any external loading may be described. The condition is used to provide additional information which the information provider considers important to the understanding of the power value. If no conditions are specified, then the die operating condition limits are presumed and other (unspecified) conditions are left at unknown values.

Example(s)

```
die_power_nom 2W Vcc=5.0V Tj=25degC;
```

See Also

die_power_max

die_power_cell

die_quiescent_current

5.49. die_quiescent_current

Section: **Die**	Level: ❷
Quiescent current consumption	

Syntax

```
<die_quiescent_current> ::=
            DIE_QUIESCENT_CURRENT <current> [ <condition> ] ';'
```

```
<current> ::= <numeric_value>
```

```
<condition> ::= <text_string>
```

Current is a positive numeric value. The default unit is Ampere.

The condition is a non-computer sensible text_string. See the lexical analysis section for more details.

Model

The setting provides the expected supply current consumption of the die during static operation. Static operation is defined as when the supply is connected and all the signal pins are held steady at a known state.

The condition attribute should list the known state of any pads and any other conditions that would affect the determination of the current value.

Note:

For devices which always require a clock (or similar active) signal when supply current is present, it is recommended to extrapolate the expected power supply current for a static clock. This can be estimated by plotting a curve for the known current at various operating frequencies.

The value should be used for I_{DDQ} in the power dissipation equation of section 1.2.5.

Example(s)

```
die_quiescent_current 0.2A All pins at active low, minimum voltage;
```

See Also

die_load_factor

die_power_capacitance

die_power_max

die_power_nom

5.50. die_saw_step_error

Section:	**Die**	Level:	❷
Maximum saw or scribe step error during dicing			

Syntax

```
<die_saw_step_error> ::= DIE_SAW_STEP_ERROR  <dimension> ';'
```

For further details, see the dimension attribute description section in this document.

Model

This setting defines the maximum saw or scribe step error expected during dicing. The dimension specifies the size of the error relative to the center of the channel which separates die on a wafer.

Example(s)

```
die_saw_step_error  10um;
```

See Also

die_saw_width

5.51. die_saw_width

Section:	**Die**	Level:	❷
Width of saw (cut), if die separated by saw			

Syntax

<die_saw_width> ::= **DIE_SAW_WIDTH** <dimension> ';'

For further details, see the dimension attribute description section in this document.

Model

The **die_saw_width** setting defines the width of the saw cut during dicing, if the die is separated by a saw. The dimension specifies the width of the substrate material cut away during the saw process.

This value, when taken in conjunction with the unspecified channel width, may give an indication of the expected variation in size from that specified in the **die_size** setting. This is possible if the listed step and repeat distance indicates the channel size by being greater than the die passivation extent size. Otherwise, the **die_wafer_scribe_line** provides a specific setting to indicate the channel width.

Example(s)

```
die_saw_width 10mil;
```

See Also

die_measure

die_passivation_extent

die_saw_step_error

die_wafer_scribe_line

die_wafer_step_and_repeat

5.52. die_section_version

Section: **Die**	Level: ⓿
Version of die information provided in this section	

Syntax

```
<die_section_version> ::= DIE_SECTION_VERSION <version_attribute> ';'
```

The version attribute is defined further in the lexical analysis document section.

Model

The die section version associates a version stamp with the die section information itself (as opposed to the mask version which stamps the physical die revision). The die section version should be updated any time there is a change to any data contained in the die section or any referenced sections (such as a **pad_geom**). The die version is independent of the block it is contained in so that a die section can be extracted from one block and inserted in another without changing the version.

Note:

> This necessarily implies that any referenced sections (such as **pad_geom**'s) would also need to be either copied or verified to exist and be identical in any block which would contain a moved die section.

If a new step revision of a die is available, this requires the creation of a brand new die section.

There is no requirement that the die version start at a stated value for a new die section. The only requirement is that as information is updated, the revision value in the die version take on a greater, numerically ordered value. Also, the date and time should be set (if given) to a value later than might have been used previously (and presumably to the date and time the update is made).

Note:

> One could imagine adding the obsolete time as a version attribute or setting. It was the general consensus that die information, when released, by definition, is instantly outdated. Therefore, the version stamp on the die section can indicate the obsolete date and time as well.

Example(s)

```
die_section_version 1.0 12/12/92;
```

See Also

block_version

die_mask_version

5.53. die_separation

Section: **Die**	Level: ❷
Method used to separate die from wafer	

Syntax

```
<die_separation> ::= DIE_SEPARATION <sep_type> ';'
<sep_type> ::= SAWED | SCRIBE_AND_BREAK
```

Only the listed enumerated name values are allowed.

Model

The die separation setting specifies the method used to separate the die from the wafer.

Note:

Although interesting information for any die user, the setting value is critical for tight tolerance MCM processes which cannot deal with the burs left on die edges from scribe and break techniques. Chips first is one such technology.

Example(s)

```
die_separation sawed;
```

See Also

die_measure

die_saw_width

5.54. die_size

Section: **Die**	Level: ⓿
Die width and height	

Syntax

```
<die_size> ::= DIE_SIZE <width> <height> ';'
<width> ::= <dimension>
<height> ::= <dimension>
```

See the attribute document section for more details about the dimension attribute.

Model

The die size is specified with this setting. See the basic model section in this document for more details.

The width and height are for the horizontal (X) and vertical (Y) die coordinate directions; respectively. The orientation of the die is defined in the **die_bonding_diagram** setting and through the definition of pad locations and fiducials.

Example(s)

```
die_size 4300 4300;
```

See Also

die_bonding_diagram

die_fiducials

die_passivation_extent_size

die_size_tolerance

die_step_and_repeat

die_thickness

5.55. die_size_tolerance

Section:	**Die**	Level:	❶
Die width and height tolerance(s)			

Syntax

```
<die_size_tolerance> ::= DIE_SIZE_TOLERANCE <tolerance> [ <tolerance> ] ';'
```

The tolerance attribute is presented in detail in the attribute section of this document.

Model

The die size tolerance represents the tolerance for the width and height size specification of the die.

When only a single tolerance is specified, the single tolerance is for the width and height dimensions. When two tolerances are specified, each specifies a separate tolerance for the width and height dimension. The width tolerance always comes first if both are present.

Example(s)

```
die_size_tolerance .25 ;
```

See Also

die_size

5.56. die_solder_bump

Section: **Die**	Level: ⓿
Solder bump description	dep

Syntax

<die_solder_bump> ::= **die_solder_bump** <bump_type>
<bump_height> <bump_volume> <material_name> ';'

<bump_type> ::= **DEPOSITED | REFLOWED**

<bump_height> ::= <dimension>

<bump_volume> ::= <numeric_value>

The bump type is an enumerated name value.

The dimension attribute is described in the attribute document section.

The material_name is described in the material_list attribute document section.

Model

The solder bump setting describes the characteristics of solder bumps already attached to the die.

These bumps may be in some original shape (just deposited) or already reflowed to create the spherical shape. If deposited, the bump_volume, bump_height, as well as **pad_geom_shape** needs to be used to estimate the final height of the reflowed bump. The bump_type is used to understand whether this calculation need take place.

The bump volume is like a dimension except the unit is interpreted as a volume (raised to the third power) and not a linear dimension. The numeric value is only allowed to have units of inches, meters, or mils. Therefore, 3um would be interpreted as .000003 m^3. The volume represents the amount of material deposited as a bump.

The bump height is the nominal height from the pad centers' surface to the farthest point above the surface of the pad. This point must lie in a plane which is parallel to the die pad surface. This plane would, by definition, be the farthest one away from the pad plane which still touches the solder bump. The dimension represents the separation between the plane and pad surface.

The material name should describe the solder bump composition. See the material list attribute for a further description.

Example(s)

die_solder_bump reflowed 4 mil 10 mil "90% lead 10% tin" **;**

See Also

die_type

5.57. die_source

Section: **Die**	Level: ⓿
Source of information about the die.	

Syntax

```
<die_source> ::= DIE_SOURCE <text_string> ';'
```

The source is a text_string value. See the lexical analysis document section for more details.

Model

The **die_source** setting exists to allow a **die** section to be tagged as to the source of the information and the creator. The loose "text" definition precludes further interpretation or computer processing of the information. In general, it is expected the authors name (or initials), division, and company would be included; as well as a reference to any material used to generate the information, such as a databook.

Note that version information of the source might also be given here. The **die_section_version** setting only describes the version for the **die** section itself.

The die source is intended to be a human readable comment field that may be presented to the user during processing of the die information or saved with the processed data for future display. No additional interpretation is given.

Example(s)

```
die_source REH, Modeling Group, Logic Modeling Corporation
      The TTL Databook, Texas Instruments, 1986. ;
```

See Also

die_manufacturer

die_notes

5.58. die_specific_heat_capacity

Section:	**Die**	Level:	❶
Specific heat capacity of the substrate material			

Syntax

```
<die_specific_heat_capacity> ::=
                    DIE_SPECIFIC_HEAT_CAPACITY <specific_heat_capacity> ';'
<specific_heat_capacity> ::= <numeric_value>
```

The specific heat capacity is a positive, numeric value. The default and only allowed unit is J/kg-C.

Model

The setting provides the specific heat capacity of the die substrate material. The value is assumed to exist for a substrate temperature within the die operating junction temperature range. The value, when combined with the mass of the bare die, helps determine the die heat capacity. The heat capacity is used in transient thermal analysis.

Note:

> This setting is meant to provide the specific value of the die substrate specific heat capacity. Otherwise, the value may be inferred from the **die_substrate_material** level 0 setting.

Example(s)

```
die_specific_heat_capacity 1.2;
```

See Also

- **die_size**
- **die_substrate_material**
- **die_thermal_conductivity**
- **die_thickness**

5.59. die_substrate_connection

Section: **Die**	Level: ⓿
Electrical connection of the die substrate.	

Syntax

<die_substrate_connection> ::= **DIE_SUBSTRATE_CONNECTION** <connection_type> **';'**

<connection_type> ::= <isolated> | <optional> | <connected>

<isolated> ::= **ISOLATED**

<optional> ::= **OPTIONAL** <pad_electrical_name>

<connected> ::= **MUST_CONNECT** <pad_electrical_name>

<pad_electrical_name> ::= <name_value>

The pad electrical name is a name_value and described in the lexical analysis section of the document

Model

The **die_substrate_connection** setting describes the electrical connection required of the die substrate. Conventionally, this is done to bias the backside of the bulk substrate material for performance or operational reasons. This setting has 3 possible connection types:

The ISOLATED connection type specifies that the substrate should be electrically isolated.

The OPTIONAL connection type specifies that the substrate can either be electrically isolated or connected to the pad(s) listed. There should be no difference in the performance of the die whether the substrate is connected or not.

The MUST_CONNECT options specifies that the substrate must be connected to the pad(s) listed.

The pad_electrical_name indicates what the substrate should be connected too. It specifies the name of a **pad_supply** section which is associated with a group of the die supply pads or an on-chip bias generator. If the name is for a pad_supply section, then the substrate backside should be connected to the referenced supply. Otherwise, the pad referencing the same pad_supply section with the given name should be connected to the substrate backside.

The pad_electrical_name must have been previously defined in a **pad_supply_name** setting and should be associated with at least one of the die's pads in the **die_pads** setting.

A bias generator pad, if it exists, should be given a SUPPLY_POWER pad type and defined by a unique **pad_supply** section.

Example(s)

```
die_substrate_connection optional VCC;
```

See Also

die_backside_finish

die_substrate_material

pad_supply_name

5.60. die_substrate_material

Section: **Die**	Level: ⓿
Bulk substrate material of the die	

Syntax

```
<die_substrate_material> ::=
            DIE_SUBSTRATE_MATERIAL <material_list> ';'
```

The material list attribute is described in the attribute document section.

Model

The substrate material setting describes the bulk material of the thin-film process die base. In most cases, this will be specified as a single material, such as silicon. The setting is used for many technology analysis processes such as thermal conductivity, coefficient of (thermal) expansion, conductivity for biasing issues, and backside surface material.

If it is important to know the crystal properties of the material to determine the thermal, stress, or electrical properties of the material, these should be stated as part of the material "string_value" in the material layer attribute.

The thickness of the single material layer form of the material_list attribute is not required. If specified, it should be less in value than the **die_thickness** setting. The thicknesses of a multi-layer specification should sum to less than the value of the **die_thickness** setting.

Note:

> If the thickness of the die substrate material is given in the material layer attribute, this does not remove the requirement for the **die_thickness** setting.

Example(s)

```
die_substrate_material silicon;
```

See also

- **die_backside_finish**
- **die_specific_heat_capacity**
- **die_substrate_connection**
- **die_thermal_conductivity**

5.61. die_technology

Section:	**Die**	Level: ⓿
Manufacturer published process name / technology of the die		

Syntax

```
<die_technology> ::= DIE_TECHNOLOGY <text_string> ';'
```

The technology can be specified in a text_string. See the lexical analysis document section for more details.

Model

The **die_technology** setting exists to provide high level guidance to the design engineer as to what process technology the die is based on. The setting is expected, at minimum, to convey the generic process technology. Terms used for this setting could be CMOS, ECL, TTL, or GaAs, for example.

The setting may convey the information in a manufacturer specific process name as long as it conveys the general technology information also. Examples of this type may be **CHMOS-IV** or **SOS-Hard**.

The setting is not-computer sensible but may change in the future.

Example(s)

```
die_technology cmos ;
```

See Also

pad_digital_circuit

5.62. die_thermal_conductivity

Section:	**Die**	Level:	❶
Thermal conductivity of the substrate material			

Syntax

```
<die_thermal_conductivity> ::=
                  DIE_THERMAL_CONDUCTIVITY <thermal_conductivity> ';'
<thermal_conductivity> ::= <numeric_value>
```

The thermal conductivity is a positive, numeric value. The default and only allowable unit is W/cm-C.

Model

The setting provides the thermal conductivity of the die substrate material. The value is for a junction temperature within the specified, normal operating junction temperature range. This value, when combined with the geometric parameters of the bare die, helps determine the thermal resistance of the die. The die thermal resistance is used in many types of thermal analysis.

Note:

This setting is meant to provide the specific value of the die substrate thermal conductivity. Otherwise, the value may be inferred from the **die_substrate_material** level 0 setting.

Example(s)

```
die_thermal_conductivity 1.2;
```

See Also

die_size

die_specific_heat_capacity

die_substrate_material

die_thickness

5.63. die_thickness

Section: **Die**	Level: ⓿
Thickness of the die	

Syntax

```
<die_thickness> ::= DIE_THICKNESS <thickness> ';'
<thickness> ::= <dimension>
```

The thickness is a dimension attribute. See the attribute section of this document for further details.

Model

The thickness of the die in the Z dimension is described with this setting. It generally represents the thickness of the substrate material (bulk material) plus any additional thickness of the layers put on the substrate during wafer processing.

The thickness represents the minimum distance between two parallel planes, both parallel to the view plane, within which any part of the die will fit between or on the plane surface. At least one point of the die must lie on each plane. Note that this includes any warpage or surface roughness that might exist. The actual surface warpage is expressed in a separate setting.

The thickness is used for thermal modeling and die physical size considerations.

Example(s)

```
die_thickness 14mils ;
```

See Also

die_thickness_tolerance

5.64. die_thickness_tolerance

Section:	**Die**	Level:	❶
Tolerance of the die thickness			

Syntax

```
<die_thickness_tolerance> ::= DIE_THICKNESS_TOLERANCE <tolerance> ';'
```

The tolerance value is described in the lexical analysis section of this document.

Model

The tolerance on the die thickness value may be specified via this setting. The tolerance represents the possible plus or minus (±) variance of the die thickness (as specified in the **die_thickness** setting).

Note, as with all tolerance settings, there is no implicit default tolerance for any value if the setting is not used.

Example(s)

```
die_thickness_tolerance .25mils;
```

See Also

die_thickness

5.65. die_type

Section: **Die**	Level: ⓿
Kind of die	pri

<u>Syntax</u>

```
<die_type> ::= DIE_TYPE <kind> ';'
<kind> ::= BARE | SOLDER_BUMP | LEAD_FRAME
```

Kind is an enumerated name value.

<u>Model</u>

Die_type identifies the kind of die being described in the **die** section.

SOLDER_BUMP or LEAD_FRAME die should have been post processed to add the solder bumps or metal lead frame to the pads; respectively. Otherwise if the die is in its original form since being separated from the wafer, it is considered a BARE die.

The **die_type** setting is used to understand what other information is expected in the **die** section. It, along with the **die_name**, **die_manufacturer**, and **die_mask_version** form the key by which a **die** section is uniquely identified.

<u>Example(s)</u>

```
die_type  solder_bump ;
```

<u>See Also</u>

die_lead_frame

die_manufacturer

die_mask_version

die_solder_bump

5.66. die_VHDL

Section:	**Die**	Level:	❶
VHDL information about this die			

Syntax

```
<die_VHDL> ::= DIE_VHDL <model_ref> { <model_ref> } ','
                        <VHDL_entity_name> { <VHDL_entity_name> } ';'

<model_ref> ::= <name_value>

<VHDL_entity_name> ::= <name_value>
```

The name value is described in the lexical analysis section of this document.

Model

This setting identifies any included VHDL models and top level VHDL entity/configuration names associated with a die. For more information about coding VHDL see [VHDL1076].

The model_ref attribute(s) specify the DIE Format model section(s) containing VHDL data. There may be one or more model sections, each with one or more VHDL design units. All design units are assumed to be in compilation order starting with the first model section name and continuing to the last design unit in the last model section name.

The VHDL_entity_name attribute(s) identify the top level VHDL entity/configuration name(s) used for either simulation or synthesis purposes. This constitutes a complete list of all versions of the design contained in the VHDL data. Note that a VHDL configuration requires the existence of an associated entity but that a VHDL entity presupposes the absence of an associated configuration. List entity names only where there are no configurations defined for a specific version of the design. All VHDL_entity_name attributes must be legal VHDL identifiers.

The die pad to VHDL entity port mapping is defined in the **die_pads_VHDL_map** setting.

Example(s)

```
die_VHDL s18245t_cfg , s18245t_c  s18245t_e  s18245t_a;
```

See Also

die_pads_VHDL_map

5.67. die_wafer_scribe_line

Section:	**Die**	Level:	❶
Width of wafer scribe lines		dep	

Syntax

```
<die_wafer_scribe_line> ::=
            DIE_WAFER_SCRIBE_LINE  <width> [ <width> ] ';'
<width> ::= <dimension>
```

The width is a dimension attribute and described in the attribute section of this document.

Model

The setting describes the scribe line width(s) existing between the die on the wafer. It is only required if the die is delivered in wafer form (pre-diced).

When only a single width is specified, the value is used for both the horizontal and vertical dimensions. When two widths are specified, the horizontal and vertical dimensions are specified separately. The horizontal width always comes first if both are present.

Example(s)

```
die_wafer_scribe_line 10 10um ;
```

See Also

die_saw_width

die_wafer_step_and_repeat

5.68. die_wafer_size

Section:	**Die**	Level:	❶
Size of the wafer before die separation		dep	

Syntax

```
<die_wafer_size> ::= DIE_WAFER_SIZE <size> [ <text_string> ] ';'
<size> ::= <dimension>
```

The size is a dimension attribute and described in the attribute section of this document. The default unit for this size dimension attribute is inches (in). The default multiplier is none.

The text_string is described further in the lexical analysis document section.

Model

The die wafer size is the diameter for the undiced wafer and only required if the die is delivered undiced on a wafer.

Wafers are defined to be circular with a possible flat edge or clipped portion added for registration and processing purposes. The diameter measurement is the maximum distance between any two points on the wafer edge. This, by definition, precludes a point from being coincident with the flat edge.

Note:

The default unit and multiplier is contrary to the dimension attribute default which is microns (um).

For wafer orientation purposes, the flat edge is considered the "bottom" (least Y values, edge coincident with the X axis) of the wafer. If a wafer does not contain a flat edge, then some method of identifying the orientation (maybe through a wafer plot, recognized fiducial, or similar) should be described in the optional text string of the setting. The orientation is required if the scribe line width or step and repeat distance is specified differently for the horizontal direction than for the vertical.

Note

The die orientation on the wafer is not specified in any setting. It may be determined from the **die_size** and **die_wafer_step_and_repeat** distances.

Example(s)

```
die_wafer_size 8in;
```

See Also

die_wafer_scribe_line

die_wafer_step_and_repeat

5.69. die_wafer_step_and_repeat

Section: **Die**	Level: ❶
Die step and repeat dimensions of the wafer	dep

Syntax

```
<die_wafer_step_and_repeat> ::=
                  DIE_WAFER_STEP_AND_REPEAT <step> [ <step> ] ';'
<step> ::= <dimension>
```

The step is a dimension attribute and described in the attribute section of this document.

Model

The step and repeat size is the distance taken when creating the mask reticules for wafer processing. It represents the distance between the same fixed point on two adjacent images of the die on the passivation mask. See the basic model description given earlier in the document for more details. The setting is only required when the die is delivered undiced in wafer form.

If a single step size is given, then this size is used for both the horizontal and vertical step dimension. If two sizes are given, then a separate specification for the horizontal and vertical dimension is given. The horizontal step size is always specified first.

Note:
The step and repeat size should normally be greater than the **die_passivation_extent_size** (for the respective dimension) and should be approximately equal to the **die_passivation_extent_size** plus the **die_saw_width** or plus the **die_wafer_scribe_line**.

Note:
The die orientation on a wafer is not explicitly given in the DIE Format. If the die is non-square, the orientation could possibly be inferred from the step and repeat dimension.

Example(s)

```
die_wafer_step_and_repeat 1700 1800 ;
```

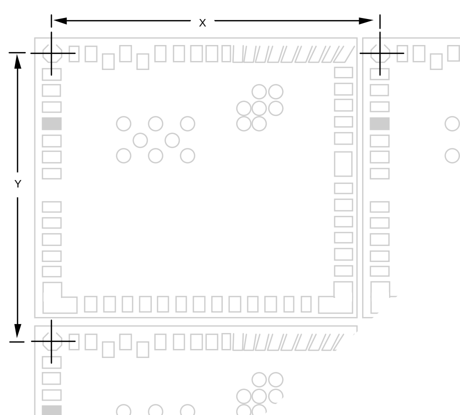


Figure 10: Die Step and Repeat Dimensions

See Also

die_passivation_extent_size

die_saw_width

die_size

die_wafer_scribe_line

die_wafer_size

5.70. model_name

Section: **Model**	Level: ⓿
Reference name of this model section	pri

Syntax

```
<model_name> ::= MODEL_NAME <model_section_name> ';'
<model_section_name> ::= <name_value>
```

Model

This setting defines a reference name that can be used by other Die Format settings to refer the information contained in a **model** section. The reference name must be unique to all other **model** sections in a DIE Block.

The **model_type** and **model_name** settings are the only two settings allowed in a model section and they must appear first, in that exact order, immediately after the **model** section keyword.

A model_section_name is referenced from a **die** section by either a **die_BSDL**, **die_IBIS**, **die_measure**, or **die_VHDL** setting.

Example(s)

```
model_name  s22333b.ibs ;
```

See Also

model_type

die_BSDL

die_IBIS

die_measure

die_VHDL

5.71. model_type

Section:	**Model**	Level:	⓿
Type of external format model			

Syntax

```
<model_type> ::= MODEL_TYPE <model_kind> ';'
<model_kind> ::= VHDL | BSDL | IBIS | MEASURE
```

Model

This setting identifies the kind of data included in the associated external format model section. **Model_type** must be set to a value of VHDL, BSDL, IBIS or MEASURE. For more information on these external formats see [VHDL1076], [BSDL1149], [IBISv1.1] and [TIMEASURE]; respectively.

The **model_type** and **model_name** settings are the only two settings allowed in a model section and they must appear first, in that exact order, immediately after the model section keyword.

A **model** section can be referenced from a **die** section. This is done with the **die** settings **die_BSDL**, **die_IBIS**, **die_measure**, or **die_VHDL**. The **model_type** setting indicates which type of external **model** is contained in the **model** section and therefore which **die** setting should reference the **model** section. It is an error if the **model** section content, **model_type** setting and any **die** section reference do not correspond.

An IBIS type **model** section, for consistency, can also be referenced by a **pad_digital_IBIS_model** setting.

Example(s)

```
model_type  IBIS ;
```

See Also

model_name

die_BSDL

die_IBIS

die_measure

die_VHDL

5.72. pad_digital_circuit

Section: **pad_digital**		Level: ⓿
Describes the pad circuit		

Syntax

```
<pad_digital_circuit> ::= PAD_DIGITAL_CIRCUIT <technology> <circuit_type> ';'
<technology> ::= ECL | TTL | CMOS | NMOS | PMOS
<circuit_type> ::= [ INPUT ] [ <active_driver> ] [ <passive_driver> ]
<active_driver> ::= OUTPUT | TRISTATE | OPEN_COLLECTOR | OPEN_EMITTER |
                        OPEN_DRAIN | OPEN_SOURCE
<passive_driver> ::= PULL_UP | PULL_DOWN
```

Only defined enumerated name values are allowed in this setting.

Model

There are 5 characteristic circuit technologies defined in this setting: ECL (Emitter Coupled Logic), TTL (Transistor-Transistor Logic), CMOS (Complementary Metal Oxide Silicon), NMOS (N-type Metal Oxide Silicon), and PMOS (P-type Metal Oxide Silicon). The setting here, unlike the **die_technology** setting, is used to indicate the general type of interface circuitry included with the die pads.

An INPUT type pad is a receiver used to transmit logic information from the external electrical connection to the internal circuitry of the die. Such a pad is identified by including the INPUT name value. The input logic threshold voltage must be defined in the **pad_digital_threshold** setting if the type INPUT is specified.

An OUTPUT type pad must contain an active driver used to transmit logic information from the internal circuitry of the die to the external electrical connection. An active driver is defined as one consisting of a transistor "switch" which can effectively change the current drive (or sinking) capability based on an internal state. The transistor switches from a high to low or low to high impedance state based on an internal state applied to the transistor base. Typically the other leads of the transistor are connected to supply rails and the output.

For saturation logic (TTL, CMOS, NMOS and PMOS), the active driver usually has 2 complementary transistor drivers -- one for each logic level. Only one of these drivers is active at a time. The driver can change from a high (inactive) to low (active) impedance state (or vice-versa). This driver type is indicated by the OUTPUT name value.

For non-saturation logic (ECL), the pad has a single common-emitter output driver which has a different output voltage for each logic level. ECL active drivers are indicated by the OPEN_EMITTER name value.

For saturation logic, sometimes the OUTPUT active driver can be configured so both drivers can be turned off simultaneously. This type of pad can effectively isolate the die electrically from the external connection. The TRISTATE name value indicates this type of driver circuit.

An OPEN_COLLECTOR driver type is a TTL or ECL output pad which has a single active driver for the logic low level only. Similarly, the OPEN_EMITTER driver type is a TTL or ECL output pad which has a single active driver for the logic high level only.

An OPEN_DRAIN driver type is a CMOS, NMOS, or PMOS output pad which has a single driver for the logic low level only. An OPEN_SOURCE driver type is a CMOS, NMOS or PMOS output pad which has a single driver for the logic high level only.

Sometimes a pad will consist of a receiver and transmitter. In this case, the INPUT name value will be specified along with an active driver name value.

In some cases, an electrical connection to one of the supply rails through a constant impedance is provided. This passive (or weak) driver has an impedance considerably higher than the active driver but considerably lower than the inactive driver. The passive driver is usually provided to keep the electrical connection associated with the pad at a specific voltage level when no driver is active. This type of connection is indicated by a PULL_UP or PULL_DOWN name value. These drivers represent a fixed impedance path to the power and ground supply rails; respectively.

Note:

An INPUT and OUTPUT pad would normally not exist, as one would always know what is being driven due to the active driver always being on. Therefore, dual type pads (receiver and transmitter; or bi-directional as they are conventionally known) typically exist with either TRISTATE or one of the OPEN_ output pad type.

Note:

It is very common for a digital signal receiver (INPUT type pad) to have a passive driver included with it. These pad types are usually not considered bi-directional due to the passive nature of the driver. Care should be taken to understand the "output driving" nature of such a pad although it is logically a receiver only.

Example(s)

```
pad_digital_circuit CMOS output;
```

See Also

pad_digital_IBIS_model

5.73. pad_digital_IBIS_model

Section:	**pad_digital**	Level:	❶
IBIS model reference			

Syntax

```
<pad_digital_IBIS_model> ::=
            PAD_DIGITAL_IBIS_MODEL <model_ref> <IBIS_model_name> ';'
<model_ref> ::= <name_value>
<IBIS_model_name> ::= <name_value>
```

The name value attribute is described further in the lexical analysis section of the document.

Model

The **pad_digital_IBIS_model** setting associates an IBIS [model] section of an IBIS file with a **pad_digital** section. The **die_pads** setting associates a pad with a **pad_digital** section. Therefore, this setting associates an IBIS [model] with a pad on the die.

The model_ref attribute identifies the DIE Format **model** section containing the IBIS data.

The IBIS_model_name attribute indicates the IBIS [model] section associated with this **pad_digital** section. The name must match the IBIS [model] name. The IBIS_model_name must conform to the IBIS name requirements.

The information content of the IBIS [model] section is defined by IBIS specification. For more information about IBIS models, see [IBISv1.1].

The IBIS model provides for a more detailed and directly usable specification of the electrical characteristics of a pad. The information given in the other **pad_digital** settings should be in congruence with the IBIS model specified.

The model_ref given here must match any corresponding IBIS model_ref given in a **die** section which references a **pad_digital** section containing this setting.

Note:

Unlike DIE Format section references, which must occur in the block before being referenced, the DIE Format **model** section being referenced here can occur anywhere in the block.

Example(s)

```
pad_digital_IBIS_model s22333.ibs CMOS_20MA_driver ;
```

See Also

die_IBIS

model_name

5.74. pad_digital_name

Section:	**pad_digital**	Level:	⓿
Pad digital signal section name.			pri

Syntax

```
<pad_digital_name> ::= PAD_DIGITAL_NAME <name_value> ';'
```

The name value attribute is described further in the lexical analysis section of the document.

Model

The pad digital name is an identifier used to uniquely identify the **pad_digital** section within a block. The name can be anything meaningful to the information provider.

The **pad_digital_name** is referenced in the **die_pads** setting in a **die** section. It is used to associate the electrical settings defined in the **pad_digital** section to an actual digital signal pad in the **die** section.

Example(s)

```
pad_digital_name output_buffer_20mA ;
```

See Also

die_pads

pad_digital_circuit

5.75. pad_digital_pull_down

Section:	**pad_digital**	Level: ⓿
Pad output driving capability at logic low level		dep

Syntax

```
<pad_digital_pull_down> ::=
            PAD_DIGITAL_PULL_DOWN <pull_down_list> ';'

<pull_down_list> ::=  <pull_down>     { ',' <pull_down> } |
                  '(' <pull_down> ')' { '(' <pull_down> ')' }

<pull_down> ::= <supply_voltage> <current> <voltage>

<supply_voltage> ::= <numeric_value>

<current> ::= <numeric_value>

<voltage> ::= <numeric_value>
```

Current is a numeric value. The default and only allowed unit is Ampere.

Voltage and supply_voltage are numeric values. The default and only allowed unit is Volt.

Model

The **pad_digital_pull_down** setting specifies the current and voltage characteristics for an output pad to maintain the low logic level at the specified supply voltage condition. Multiple supply_voltage conditions are supported.

The polarity of the current indicates its direction. A positive value indicates that current flows from the pad to the external electrical connection and a negative value indicates that current flows from the external electrical connection to the pad.

All voltage values are specified with reference to the ground supply.

For an OUTPUT active driver, this setting specifies the maximum current the driver can sink and still maintain the output voltage level below the voltage specified.

Example(s)

```
pad_digital_pull_down  5V -24mA 0.5V;

pad_digital_pull_down   3V   -20uA  0.1V,
                        4.5V -4.0mA 0.33V,
                        5.5V -5.2mA 0.33V;
```

See Also

pad_digital_circuit

pad_digital_IBIS_model

pad_digital_pull_up

5.76. pad_digital_pull_up

Section:	**pad_digital**	Level:	⓿
Pad output driving capability at logic high level			dep

Syntax

```
<pad_digital_pull_up> ::=
            PAD_DIGITAL_PULL_UP <pull_up_list> ';'


<pull_up_list> ::=  <pull_up>     { ',' <pull_up> }   |
                '(' <pull_up> ')' { '(' <pull_up> ')' }

<pull_up> ::= <supply_voltage> <current> <voltage>

<supply_voltage> ::= <numeric_value>

<current> ::= <numeric_value>

<voltage> ::= <numeric_value>
```

Current is a numeric value. The default and only allowed unit is Ampere.

Voltage and supply_voltage are a numeric values. The default and only allowed unit is Volt.

Model

The **pad_digital_pull_up** setting specifies the current and voltage characteristics for an output pad to maintain the high logic level at the specified supply voltage condition. Multiple supply_voltage conditions are supported.

The polarity of the current indicates its direction. A positive value indicates that current flows from the pad to the external electrical connection and a negative value indicates that current flows from the external electrical connection to the pad.

All voltage values are specified with reference to the ground supply.

For an output driver, this setting specifies the maximum current the driver can source and still maintain the output voltage level above the voltage specified.

Example(s)

```
pad_digital_pull_up 5V 3mA 2.4V;

pad_digital_pull_up
                        3V   20uA  1.9V,
                        4.5V 4.0mA 3.84V,
                        5.5V 5.2mA 5.34V;
```

See Also

pad_digital_circuit

pad_digital_IBIS_model

pad_digital_pull_down

5.77. pad_digital_threshold

Section:	**pad_digital**	Level:	⓿
Pad input switching characteristics		dep	

Syntax

```
<pad_digital_threshold> ::= PAD_DIGITAL_THRESHOLD <threshold_list> ';'
<threshold_list> ::=  <threshold>     { ',' <threshold> } |
                  '(' <threshold> ')' { '(' <threshold> ')' }
<threshold> ::= <supply_voltage> <logic_low> <logic_high>
<supply_voltage> ::= <numeric_value>
<logic_low> ::= <threshold_voltage> <max_current>
<logic_high> ::= <threshold_voltage> <max_current>
<threshold_voltage> ::= <numeric_value>
<max_current> ::= <numeric_value>
```

The max_current is a numeric value. The default and only allowed unit is Ampere.

The supply_voltage and threshold_voltage are numeric values. The default and only allowed unit is Volt.

See the lexical analysis section of this document for more information on the numeric value.

Model

For a logic_low specification, the threshold voltage is the highest steady state voltage which the input will still recognize as a logic low level.

For a logic_high specification, the threshold voltage is the lowest steady state voltage which the input will still recognize as a logic high level.

Note:

> This simple threshold specification does not take transition voltage characteristics (hysterisis of the input) into account. That is, those inputs which may have a different threshold voltage for a transition to the logic level as opposed to a transition out of the logic level.

The maximum current is what the driving source has to sink or source (depending on the polarity of the current value) from this input pad to maintain the logic value.

The polarity of the current indicates its direction of flow. A positive value indicates that current flows into the pad from the external electrical connection. A negative value indicates that current flows from the pad connection to the external electrical connection.

Note:

> The polarity for current specifications in settings describing output pads is opposite to the polarity used in this setting.

All voltages are specified with reference to the ground supply.

The supply voltage can only represent a single or nominal operating voltage.

Note:

> The max_current may be specified as 0A for an input pad which draws a negligible input current. This is typically the case for MOS inputs.

Example(s)

```
pad_digital_threshold  (5V 2.0V 20uA 0.8V -0.1mA) ;
```

```
pad_digital_threshold   3.0V 1.5V  0.0 0.5V  0.0,
                        4.5V 3.15V 0.0 1.35V 0.0,
                        5.5V 4.2V  0.0 1.8V  0.0;
```

See Also

5 **pad_digital_circuit**

pad_digital_IBIS_model

pad_digital_pull_down

pad_digital_pull_up

5.78. pad_geom_bond_sites

Section:	**Pad_geom**	Level:	❷
Number of intended bond sites and their locations			

Syntax

```
<pad_geom_bond_sites> ::=
            PAD_GEOM_BOND_SITES <num_points> { <point> } ';'
```

```
<num_points> ::= <numeric_value>
```

The num_points is a positive integer.

The list of points that follows is optional. If there are any points in the list, then there should be num_points coordinate points.

See the attribute and lexical analysis sections for more details on the specifics of the point and numeric_value; respectively.

Model

Most pads are designed with the specific intent of a given pad bonding method. Some pads are designed larger so multiple bonds can be made to a single pad to improve the electrical characteristics of the off-die interconnect. For example, it is often the case that supply pads are larger to accommodate multiple wire bonds with standard width wire to facilitate the carrying of larger currents.

The **pad_geom_bond_sites** setting indicates the number of bond sites on a pad and the center of each. In general, this will be based on the conventional packaged part bond information. This setting is not intended to represent a requirement for a given number of bonds or bonding method.

Note:
The coordinates are in **pad_geom** coordinate space **--** this is before placement and rotation of the defined shape.

Note:
A single bond point may be specified. This is useful for pads (such as parallelograms or L-shaped) where a bond point is not necessarily implicit or the center of the pad geometry.

Example(s)

```
pad_geom_bond_sites 2 0 -1 0 1 ;
```

See Also

pad_geom

5.79. pad_geom_metal_extent

Section:	**Pad_geom**	Level:	❶
Pad metal shape and extent			

Syntax

```
<pad_geom_metal_extent> ::= PAD_GEOM_METAL_EXTENT <center> <shape> ';'
<center> ::= <point>
```

The center is a point attribute. See the attribute document section for more details.

The shape attribute is described in the **pad_geom_shape** setting.

Model

This setting defines the shape and orientation of the actual metal pad. The shape attribute can be used to specify the footprint area of the actual metal pad. The metal extent is required because the **pad_geom_shape** setting only indicates the "exposed" pad metal; not necessarily the full metal area of the bond pad.

Some of the pad metal may be covered by passivation material. See the **pad_geom_passivation_extent** setting for a description of the passivation opening over or under the metal pad.

The center point defines any translation component for the metal pad with respect to the **pad_geom_shape** defined coordinate system. It may be that the pad defined center is not coincident with the shape of the pad metal itself (remember, a pad is the exposed metal only). This setting does not provide for the possibility of the pad metal being rotated with respect to the **pad_geom_shape**.

The shape attribute is more fully described in the **pad_geom_shape** setting description.

Note:

> The coordinates are in **pad_geom** coordinate space **--** this is before placement and rotation of the shape on the die surface.

Example(s)

```
pad_geom_metal_extent
            0.0 0.0 polygon 5 15 25 25 25 25 35 15 35 15 25;
```

See Also

pad_geom_passivation_extent

pad_geom_shape

5.80. pad_geom_name

Section:	**Pad_geom**	Level:	⓿
Pad identification name			pri

Syntax

```
<pad_geom_name> ::= PAD_GEOM_NAME <name_value> ';'
```

The name value attribute is described further in the lexical analysis section of the document.

Model

The pad geometry name is used to create a unique identifier for the pad geometry section it occurs in. It forms, in relational database terminology, the primary key for the table of information in the **pad_geom** section. The name can be anything meaningful to the information provider provided it conforms to the name_value requirements. The name must be unique to all other **pad_geom_name** values in the block to avoid confusion.

The **pad_geom_name** is referenced in a **die** sections' **die_pads** setting; and thus used to associate a pad geometrical definition with a pad instance on a die.

Note:

A pad geometry name is different than a pad instance ID. The ID is used to identify an instance of a pad using a numeric label. The pad name defined here is used to name the geometrical model for a pad definition.

Example(s)

```
pad_geom_name 4umX4um ;
```

See Also

die_pads

pad_geom_shape

5.81. pad_geom_passivation_extent

Section:	**Pad_geom**	Level:	❶
Passivation opening shape			

Syntax

```
<pad_geom_passivation_extent> ::=
            PAD_GEOM_PASSIVATION_EXTENT <center> <shape> ';'
```

```
<center> ::= <point>
```

The center is a point attribute. See the attribute document section for more details.

The shape attribute is described in the **pad_geom_shape** setting.

Model

This setting defines the shape and orientation of the passivation opening either over or under the actual metal pad. The shape attribute can be used to specify the footprint area of the actual passivation opening. The passivation opening description is required as the **pad_geom_shape** setting only indicates the "exposed" pad metal. The passivation cut may extend beyond the metal pad boundaries or occur under the metal pad and thus not be determinable from the pad geometry description.

This setting will generally be different than the actual metal pad shape. See the **pad_geom_metal_extent** setting for a description of the pad metal shape and extent.

The passivation extent is required to understand the total size of the hole created in the passivation cover for a pad. Area array pads for solder bump bond technology typically have the pad metal on top of and larger than the passivation opening. Pads intended for lead frame bonds may have a larger opening than the pad metal extent size. This parameter provides the direct indication of the opening (cut) size.

The center point defines any translation component for the passivation opening with respect to the **pad_geom_shape** defined coordinate system. It may be that the pad defined center is not coincident with the shape of the passivation opening itself (remember, a pad is the exposed metal only). This setting does not provide for the possibility of the passivation opening being rotated with respect to the **pad_geom_shape**.

The shape attribute is more fully described in the **pad_geom_shape** setting description.

Note:

> The coordinates are in **pad_geom** coordinate space **--** this is before placement and rotation of the shape on the die surface.

Example(s)

```
      pad_geom_passivation_extent
                  0.0 0.0 polygon 5 15 25 25 25 25 35 15 35 15 25;
```

See Also

- **die_passivation_material**
- **pad_geom_metal_extent**
- **pad_geom_shape**

5.82. pad_geom_shape

Section:	**Pad_geom**	Level:	0
Pad geometry shape			

Syntax

```
<pad_geom_shape> ::= PAD_GEOM_SHAPE <shape> ';'
<shape> ::= <circle> | <rectangle> | <polygon>
<circle>    ::= CIRCLE <diameter>
<rectangle> ::= RECTANGLE <width> <height>
<polygon>   ::= POLYGON <polygonal_area>
<diameter> ::= <dimension>
<width> ::= <dimension>
<height> ::= <dimension>
```

A circle is simply defined with a dimension attribute representing a diameter.

A rectangle is defined as a width and height. Width and height are dimension attributes.

A polygon is represented by a polygonal area attribute.

See the attribute section of this document for more details.

Model

The **pad_geom_shape** setting is used to describe the bondable area of the pad used for making electrical connections to the die. These pads may be setup for wire bond, solder ball, lead frame or any other bonding technique. The bondable area is defined as any exposed metal of a die pad. It does not include exposed areas which do not have metal pad materiel there.

For solder bump pads, ones where the metal may be laid on top of the passivation cap, the **pad_geom_shape** and **pad_geom_metal_extent** areas will be the same.

The bondable area usually represents the passivation opening over the metal pad. If the passivation "cut" is not vertical or the edges are irregular, then the pad should be described such that it represents the largest object (circle, rectangle, etc.) which can be projected onto the pad metal from the view plane without creating any overlap with the passivation material.

The **pad_geom_shape** is constructed with dimension attributes. The basic value of each dimension attribute represents the theoretical designed value associated with the pad.

The pads are described, as detailed in the basic model, in a coordinate system where (0, 0) is the center of the pad. The pad center is defined as the center of the smallest rectangle which will enclose the pad. Therefore, having just a diameter for a circular pad or size for a rectangular pad is enough to construct the pad shape within the pads coordinate system.

Most pads are either rectangular (mostly periphery pads intended for wire or lead frame bonding) or circular (arranged in arrays over the die surface for solder bump mounting). A polygon is used for all other possible bond pad structures such as parallelograms, 'L' shaped corner pads, hexagons, and others. See figure 11 for examples of pad shapes usually encountered.

The width and height of the rectangle shape are for the horizontal (X) and vertical (Y) directions; respectively. When the pad is instanced, it may get rotated and thus cause this initial reference to change.

Note:

The pad center, as defined, may not be located on the interior of the pad area (think of 'L' shaped corner pads or parallelogram pads). Therefore, the pad center cannot be assumed to be a bond point for polygonal pads.

If a pad is elliptical, than either a polygon or circle can be used to approximate the ellipse. The circle should be the largest which will fit inside the ellipse. It is up to the user to determine which form will more closely represent the intended geometry.

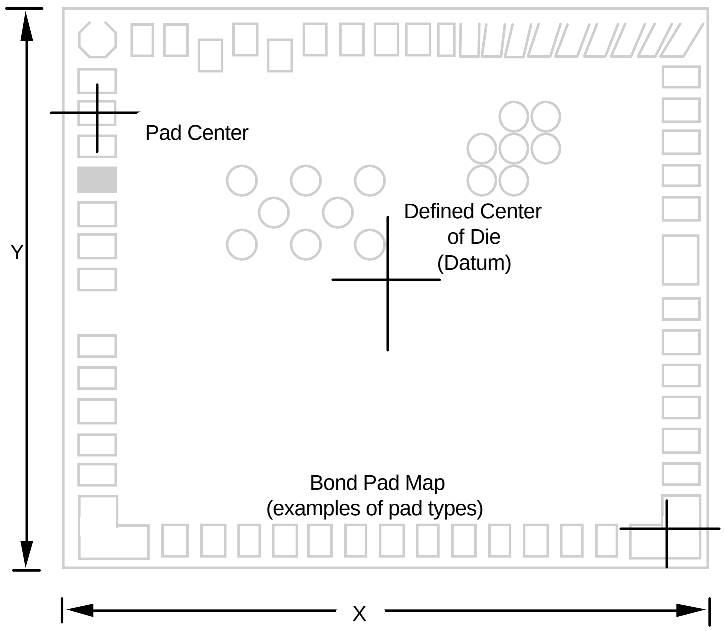


Figure 11: Die and Pad Datum (centers),
Pad outline and placement examples

<u>Example(s)</u>

```
pad_geom_shape circle 4um;
pad_geom_shape rectangle 4 6;
pad_geom_shape polygon 5 -2 2 2 2 2 -2 -2 -2 -2 2 ;
```

<u>See Also</u>

pad_geom_metal_extent

pad_geom_passivation_extent

5.83. pad_geom_tolerance

Section:	**Pad_geom**	Level:	❶
Pad geometry tolerance for sizes			

Syntax

<pad_geom_tolerance> ::= **PAD_GEOM_TOLERANCE** <tolerance> [<tolerance>] ';'

The tolerance is presented in detail in the attribute section of this document.

Model

The pad geometry tolerance represents the tolerance for the overall shape described by the **pad_geom_shape**, **pad_geom_metal_extent**, and **pad_geom_passivation_extent** settings. See the basic model description for more details.

When only a single tolerance is specified where two are allowed, the single tolerance is for the X and Y dimensions (or width and height, if elaborating a rectangular definition). When two tolerances are specified, each specifies a separate tolerance for the X and Y (width and height; respectively) dimensions. The X (width) tolerance always comes first if both are present.

For a setting elaborating the tolerance for a circle, and when two tolerances are specified, the diameter should be given a tolerance equal to the average of the two tolerance values.

Example(s)

```
pad_geom_tolerance .25 ;
```

See Also

pad_geom_shape

pad_geom_metal_extent

pad_geom_passivation_extent

5.84. pad_supply_current_max

Section:	**pad_supply**	Level:	⓿
Maximum current of a supply pad.			

Syntax

```
<pad_supply_current_max> ::= PAD_SUPPLY_CURRENT_MAX <current> ';'
<current> ::= <numeric_value>
```

Current is a positive numeric value. The default and only allowed unit is Ampere.

Model

The setting indicates the absolute maximum amount of current which should flow through a supply pad. Steady state current beyond this maximum value may permanently damage the part.

Note:

The setting is provided as a guide to the connection of supply pads. When the first level of interconnect beyond the die has current limits less than this setting specification, then multiple bonds may be necessary for reliability.

Example(s)

```
pad_supply_current_max 70mA;
```

See Also

pad_supply_name

5.85. pad_supply_name

Section:	**pad_supply**	Level:	⓿
Name of a power supply section.			pri

Syntax

```
<pad_supply_name> ::= PAD_SUPPLY_NAME <name_value> ';'
```

The name value is described further in the lexical analysis section of the document.

Model

The pad supply name is an identifier used to uniquely identify the **pad_supply** section within a block. The name can be anything meaningful to the information provider. The name must be unique to all other **pad_supply** names in the DIE Block to avoid confusion.

The **pad_supply_name** is referenced in the **die_pads** setting in a **die** section. It is used to associate the electrical settings defined in the **pad_supply** section to an actual supply pad in the **die** section.

Example(s)

```
pad_supply_name VCC_pad;
```

See Also

pad_supply_current_max

pad_supply_voltage

5.86. pad_supply_voltage

Section: **pad_supply**	Level: ⓿
Supply voltage specification.	

Syntax

```
<pad_supply_voltage> ::= PAD_SUPPLY_VOLTAGE <supply_spec> ';'
<supply_spec> ::= <fixed_supply> | <range_supply>
<fixed_supply> ::= FIXED <voltage_nom> <voltage_min> <voltage_max>
<range_supply> ::= RANGE <voltage_min> <voltage_max>
<voltage_nom> ::= <numeric_value>
<voltage_min> ::= <numeric_value>
<voltage_max> ::= <numeric_value>
```

The voltage attributes are numeric values. The default and only allowed unit is Volt.

Model

This setting specifies the voltage required of a DC voltage source which is connected to the supply pad(s) of the die. There are 2 types of specifications: fixed and variable voltage specifications.

In a fixed voltage specification, the nominal voltage of the DC voltage source applied to the supply pad is specified. The supply voltage has an operating tolerance range from voltage_min up to voltage_max.

In a range supply voltage specification, the supply source can have an operating range from voltage_min up to voltage_max. There is no additional tolerance or nominal value within the range specified.

All voltage values are specified with reference to the nominal ground supply.

When more than one supply pad instance references the same **pad_supply** model, these pads should be connected in such a manner as to minimize the voltage difference between them. If there is any exception to this requirement, the exception should be described in the **die_conditions_special** setting.

Note:

> Discrete operating points in a range are not allowed. The range specification is intended for devices which can operate at any voltage in the range. Devices designed at only discrete points of operating voltage cannot be accurately described here.

Example(s)

```
pad_supply_voltage fixed 5.0V 4.5V 5.5V;
pad_supply_voltage range 3V 15V;
```

See Also

pad_supply_current_max

pad_supply_name

die_conditions_special

References

The Reference document section defines informative references for further reading or additional identification information for normative references to other standards.

[ANSIY14.5M] ASME/ASE, **Dimensioning and Tolerancing: ANSI Y14.5M - 1982**, American Society of Mechanical Engineers, (Dec 1982)

[BNF1964]

[BSDL1149] JTAG Industry Group, **Boundary Scan Description Language**, Supplement (B) to Standard Test Access Port and Boundary-Scan Architecture: IEEE Std 1149.1-1990, Institute of Electronic and Electrical Engineers, NY, NY (Mar 1993).

[EIA548] EDIF Steering Committee, **Electronic Design Interchange Format: ANSI/EIA-548-1993**, Electronic Industries Association, Washington DC. (version 3 0 0, 1993)

[EIA567] Design Automation Division, **VHDL Commercial Component Model Specification: EIA-567-1990**, Electronic Industries Association, Washington DC

[IBISv1.1] IBIS Open Forum, **I/O Buffer Information Specification**, Intel Corporation, Hillsboro, OR (version 1.1, Aug. 1993)

[ICREQUIRE] Harr, R.E. et al, **IC Manufacturer Die Library Data Study**, Logic Modeling Corporation, Milpitas, CA (Aug. 1993)

[IEEE260] IEEE SCC 14, **American National Standard Letter Symbols for Units of Measurement (SI Units, Customary Inch-Pound Units, and Certain Other Units): ANSI/IEEE Std 260.1-1993**, Institute of Electrical and Electronics Engineers, NY, NY (Jan. 1994)

[ISO1000] ISO/TC 12, **ISO 1000 : 1992 (E) SI units and recommendations for the use of their multiples and of certain other units**, International Organization for Standardization, Geneva, Switzerland.

[ISO8859] ISO, **ISO 8859-1 : 1987 (E) 8 -bit single-byte coded graphic character sets -- Part 1: Latin Alphabet No. 1,** International Organization for Standardization, Geneva, Switzerland

[JTAG1149] JTAG Industry Group, **IEEE Standard Test Access Port and Boundary-Scan Architecture**, IEEE Std 1149.1-1990, Institute of Electronic and Electrical Engineers, NY, NY (May 1990)

[REQUIRE] Harr, R.E. et al, **MCM Manufacturer and Designer Die Library Data Requirements**, Logic Modeling Corporation (June 1993)

[SGML8879] Goldfarb, C.F.,**The Standard Generalized Markup Language (ISO 8879)**, International Organization for Standards, Geneva (1986)

[TIMEASURE] Texas Instruments, **Enhanced Die Measure Specification Version 2.1**, Texas Instruments, Dallas, TX (Nov. 1993)

[VHDL1076] DASC VASG, **IEEE Standard VHDL Language Reference Manual: IEEE Std 1076-1987**, Institute of Electrical and Electronics Engineers, NY, NY (March 1988)

[WORKSHOP1] Chan, S et al, **Minutes from the Physical Die Information Workshop**, Logic Modeling Corporation, Milpitas, CA (June 1993)

[WORKSHOP2] Harr, R et al, **Minutes from the Second Physical Die Information Workshop**, Logic Modeling Corporation, Milpitas, CA (Sept. 1993)

[WORKSHOP3] Harr, R et al, **Minutes from the Third Physical Die Information Workshop**, Logic Modeling Corporation, Milpitas, CA (Feb. 1994)

Appendix A: Collected BNF

Contained below is the alphabetic ordered, BNF productions (non-terminal definitions) extracted from the main, reference manual document sections. This section is informative only and, if different than the main document, should be deemed in error.

<abs_max_temp> ::= <numeric_value>

<active_driver> ::= **OUTPUT** | **TRISTATE** | **OPEN_COLLECTOR** | **OPEN_EMITTER** |
OPEN_DRAIN | **OPEN_SOURCE**

<area> ::= <pad_ID> <diameter> <point>

<backside_finish> ::=
POLISHED | **LAPPED** | **METALLIZED** <material_list>

<base_name> ::= <ISO_graphic>

<bit_notation> ::= '**<**' <bit_range> { '**,**' <bit_range> } '**>**'

<bit_range> ::= <numeric_value> ['**:**' <numeric_value>]

<block> ::= <header_section> { <block_body> }1+ <trailer_section>

<block_body> ::= <pad_geom> | <pad_digital> | <pad_supply> |
<die> | <model> | <model_end>

<block_DIE_format_version> ::=
BLOCK_DIE_FORMAT_VERSION <version_attribute> '**;**'

<block_disclaimer> ::= **BLOCK_DISCLAIMER** [<text_string>] '**;**'

<block_level> ::= **BLOCK_LEVEL** <compliance_level> '**;**'

<block_notes> ::= **BLOCK_NOTES** [<text_string>] '**;**'

<block_source> ::= **BLOCK_SOURCE** <text_string> '**;**'

<block_version> ::= **BLOCK_VERSION** <version_attribute> '**;**'

<bonding_sequence> ::= '**(**' <pads_list> '**)**' {'**(**' <pads_list> '**)**'} |
<pads_list> {'**,**' <pads_list> }

<BSDL_entity_name> ::= <name_value>

<BSDL_package_decl> ::= <name_value>

<bump_type> ::= **DEPOSITED** | **REFLOWED**

<bump_height> ::= <dimension>

<bump_volume> ::= <numeric_value>

<cage_code> ::= { <ISO_digit> }4-5

<center> ::= <point>

<circle> ::= **CIRCLE** <diameter>

<circuit_type> ::= [**INPUT**] [<active_driver>] [<passive_driver>]

<comment> ::= '|' { <ISO_character> } <record_terminator>

<compliance_level> ::= '**0**' | '**1**' | '**2**'

<condition> ::= <text_string>

<connected> ::= **MUST_CONNECT** <pad_electrical_name>

<connection_type> ::= <isolated> | <optional> | <connected>

<current> ::= <numeric_value>

```
<date_value> ::== <day> '/' <month> '/' <year>

<diameter> ::= <dimension>

<die> ::= '[' DIE ']' { <die_level_0_setting> |
                        <die_level_1_setting> |
                        <die_level_2_setting> }

<die_backside_finish> ::= DIE_BACKSIDE_FINISH  <backside_finish> ';'

<DIE_Block> ::= '[' DIE_BLOCK ']' <block_die_format_version> <block_level>
                        { <other_header_setting> }

<DIE_Block_end> ::= '[' DIE_BLOCK_END ']'

<die_bonding_diagram> ::= DIE_BONDING_DIAGRAM <text_string> ';'

<die_bonding_sequence> ::=
            DIE_BONDING_SEQUENCE <bonding_sequence> ';'

<die_BSDL> ::= DIE_BSDL
                  <model_ref> <BSDL_entity_name> <BSDL_package_decl> ';'

<die_conditions_bonding> ::= DIE_CONDITIONS_BONDING <text_string> ';'

<die_conditions_process> ::= DIE_CONDITIONS_PROCESS <text_string> ';'

<die_conditions_sealing> ::= DIE_CONDITIONS_SEALING <text_string> ';'

<die_conditions_special> ::= DIE_CONDITIONS_SPECIAL <text_string> ';'

<die_conditions_storage> ::= DIE_CONDITIONS_STORAGE <text_string> ';'

<die_description> ::= DIE_DESCRIPTION <text_string> ';'

<die_fiducials> ::= DIE_FIDUCIALS  <polygonal_area_list> ';'

<die_flatness> ::= DIE_FLATNESS <flatness> ';'

<die_IBIS> ::= DIE_IBIS <model_ref> <IBIS_component_name> ';'

<die_junction_temperature> ::= DIE_JUNCTION_TEMPERATURE
            <min_temp> <max_temp> [ <abs_max_temp> ] ';'

<die_lead_frame> ::= DIE_LEAD_FRAME <text_string> ';'

<die_level_0_setting> ::=
            <die_junction_temperature> |
            <die_lead_frame> |
            <die_manufacturer> |
            <die_mask_version> |
            <die_name> |
            <die_notes> |
            <die_packaged_part_name> |
            <die_pads> |
            <die_power_max> |
            <die_power_nom> |
            <die_section_version> |
            <die_size> |
            <die_solder_bump> |
            <die_source> |
            <die_substrate_connection> |
            <die_substrate_material> |
            <die_technology> |
            <die_thickness> |
            <die_type>

<die_level_1_setting> ::=
            <die_backside_finish> |
            <die_bonding_diagram> |
```

```
            <die_BSDL> |
            <die_conditions_storage> |
            <die_fiducials> |
            <die_IBIS> |
            <die_packaged_part_bonding> |
            <die_pad_pitch> |
            <die_pads_supply_grouping> |
            <die_pads_tolerance> |
            <die_pads_VHDL_map> |
            <die_passivation_material> |
            <die_size_tolerance> |
            <die_specific_heat_capacity> |
            <die_thermal_conductivity> |
            <die_thickness_tolerance> |
            <die_VHDL> |
            <die_wafer_scribe_line_width> |
            <die_wafer_size> |
            <die_wafer_step_and_repeat_size>
```

```
<die_level_2_setting> ::=
            <die_bonding_sequence> |
            <die_conditions_bonding> |
            <die_conditions_process> |
            <die_conditions_sealing> |
            <die_conditions_special> |
            <die_description> |
            <die_fiducials> |
            <die_flatness> |
            <die_load_factor> |
            <die_lot> |
            <die_manufacturer_cage> |
            <die_measure> |
            <die_military_spec> |
            <die_packaged_part_attach> |
            <die_pad_dielectric> |
            <die_pad_metal> |
            <die_pads_jumpers> |
            <die_pads_noncontact_area> |
            <die_passivation_extent_size> |
            <die_power_capacitance> |
            <die_power_cell> |
            <die_quiescent_current> |
            <die_saw_step_error> |
            <die_saw_width> |
            <die_separation>
```

<die_load_factor> ::= **DIE_LOAD_FACTOR** <factor> ';'

<die_lot> ::= **DIE_LOT** <text_string> ';'

<die_manufacturer> ::= **DIE_MANUFACTURER** <text_string> ';'

<die_manufacturer_cage> ::= **DIE_MANUFACTURER_CAGE** <cage_code> ';'

<die_mask_version> ::= **DIE_MASK_VERSION** <version_attribute> ';'

<die_measure> ::= **DIE_MEASURE** <model_ref> { <model_ref> } ';'

<die_military_spec> ::= **DIE_MILITARY_SPEC** <text_string> ';'

<die_name> ::= **DIE_NAME** <text_string> ';'

<die_notes> ::= **DIE_NOTES** [<text_string>] ';'

<die_packaged_part_attach> ::= **DIE_PACKAGED_PART_ATTACH** <text_string> ';'

```
<die_packaged_part_bonding> ::= DIE_PACKAGED_PART_BONDING <text_string> ';'

<die_packaged_part_name> ::= DIE_PACKAGED_PART_NAME <text_string> ';'

<die_pad_dielectric> ::= DIE_PAD_DIELECTRIC <material_list> ';'

<die_pad_metal> ::= DIE_PAD_METAL  <material_list> ';'

<die_pad_pitch> ::= DIE_PAD_PITCH <dimension> ';'

<die_pads> ::= DIE_PADS <num_pads> <pad_descr_list> ';'

<die_pads_jumpers> ::=  DIE_PADS_JUMPERS <interconnect_lists> ';'

<die_pads_noncontact_area> ::=
      DIE_PADS_NONCONTACT_AREA <num_areas> <area> { <area> } ';'

<die_pads_supply_grouping> ::=
            DIE_PADS_SUPPLY_GROUPING  <supply_group_list> ';'

<die_pads_tolerance> ::= DIE_PADS_TOLERANCE <tolerance> [ <tolerance> ] ';'

<die_pads_VHDL_map> ::=
            DIE_PADS_VHDL_MAP  <VHDL_map_pair> { <VHDL_map_pair> } ';'

<die_passivation_extent_size> ::=
            DIE_PASSIVATION_EXTENT_SIZE  <width> <height> <point> ';'

<die_passivation_material> ::= DIE_PASSIVATION_MATERIAL <material_list> ';'

<die_power_capacitance> ::= DIE_POWER_CAPACITANCE <power_capacitance> ';'

<die_power_cell> ::= DIE_POWER_CELL <power_area_list> ';'

<die_power_max> ::= DIE_POWER_MAX <power> <condition> ';'

<die_power_nom> ::= DIE_POWER_NOM <power> <condition> ';'

<die_quiescent_current> ::=
            DIE_QUIESCENT_CURRENT <current> [ <condition> ] ';'

<die_saw_step_error> ::= DIE_SAW_STEP_ERROR  <dimension> ';'

<die_saw_width> ::= DIE_SAW_WIDTH  <dimension> ';'

<die_section_version> ::= DIE_SECTION_VERSION <version_attribute> ';'

<die_separation> ::= DIE_SEPARATION <sep_type> ';'

<die_size> ::= DIE_SIZE <width> <height> ';'

<die_size_tolerance> ::= DIE_SIZE_TOLERANCE <tolerance> [ <tolerance> ] ';'

<die_solder_bump> ::= die_solder_bump <bump_type>
            <bump_height> <bump_volume> <material_name> ';'

<die_source> ::= DIE_SOURCE <text_string> ';'

<die_specific_heat_capacity> ::=
                  DIE_SPECIFIC_HEAT_CAPACITY <specific_heat_capacity> ';'

<die_substrate_connection> ::= DIE_SUBSTRATE_CONNECTION <connection_type> ';'

<die_substrate_material> ::=
            DIE_SUBSTRATE_MATERIAL <material_list> ';'

<die_technology> ::= DIE_TECHNOLOGY <text_string> ';'

<die_thermal_conductivity> ::=
                  DIE_THERMAL_CONDUCTIVITY <thermal_conductivity> ';'

<die_thickness> ::= DIE_THICKNESS <thickness> ';'
```

<die_thickness_tolerance> ::= **DIE_THICKNESS_TOLERANCE** <tolerance> '**;**'

<die_type> ::= **DIE_TYPE** <kind> '**;**'

<die_VHDL> ::= **DIE_VHDL** <model_ref> { <model_ref> } '**,**'
<VHDL_entity_name> { <VHDL_entity_name> } '**;**'

<die_wafer_scribe_line> ::=
DIE_WAFER_SCRIBE_LINE <width> [<width>] '**;**'

<die_wafer_size> ::= **DIE_WAFER_SIZE** <size> [<text_string>] '**;**'

<die_wafer_step_and_repeat> ::=
DIE_WAFER_STEP_AND_REPEAT <step> [<step>] '**;**'

<dimension> ::= <numeric_value>

<factor> ::= <numeric_value>

<fixed_supply> ::= **FIXED** <voltage_nom> <voltage_min> <voltage_max>

<flatness> ::= <dimension>

<ground_ID> ::= <ID_value>

<header_section> ::= <DIE_Block>

<height> ::= <dimension>

<IBIS_component_name> ::= <name_value>

<IBIS_model_name> ::= <name_value>

<ID_value> ::= <ISO_digit> { <ISO_digit> }

<interconnect_lists> ::=
'**(**' <jumper_list> '**)**' {'**(**' <jumper_list> '**)**'} |
<jumper_list> {'**,**' <jumper_list> }

<isolated> ::= **ISOLATED**

<jumper_list> ::= <pad_ID> <pad_ID> { <pad_ID> }

<kind> ::= **BARE** | **SOLDER_BUMP** | **LEAD_FRAME**

<layer> ::= <material_name> <thickness>

<logic_high> ::= <threshold_voltage> <max_current>

<logic_low> ::= <threshold_voltage> <max_current>

<material_list> ::= <single_layer> | <multiple_layers>

<material_name> ::= <name_value> | <string_value>

<max_current> ::= <numeric_value>

<max_temp> ::= <numeric_value>

<min_temp> ::= <numeric_value>

<mirror> ::= **H** | **V**

<model> ::= '**[**' **MODEL** '**]**' <model_name> <model_type>
{ <ISO_character> }

<model_end> ::= '**[**' **MODEL_END** '**]**'

<model_kind> ::= **VHDL** | **BSDL** | **IBIS** | **MEASURE**

<model_name> ::= **MODEL_NAME** <model_section_name> '**;**'

<model_ref> ::= <name_value>

<model_section_name> ::= <name_value>

```
<model_type> ::= MODEL_TYPE <model_kind> ';'

<multiple_layers> ::= <number_of_layers> { <layer> }n

<name_value> ::= <base_name> [ <bit_notation> ]

<num_areas> ::= <numeric_value>

<num_def> ::= <numeric_value>

<num_pads> ::= <numeric_value>

<num_points> ::= <numeric_value>

<number_of_layers> ::= <numeric_value>

<numeric_value> ::==  [ '+' | '-' ] <ISO_digit> { <ISO_digit> }
                              [ '.' <ISO_digit> { <ISO_digit> } ]
                              [ [ <multiplier> ] <unit> ]

<optional> ::= OPTIONAL <pad_electrical_name>

<other_header_setting> ::= <block_disclaimer> | <block_notes> |
                              <block_source> | <block_version>

<pad_common_name> ::= <name_value>

<pad_descr> ::= <pad_ID> <pad_geom_inst> <pad_elec_inst> <pad_info>

<pad_descr_list> ::=  <pad_descr>     { ',' <pad_descr>     } |
                  '(' <pad_descr> ')' { '(' <pad_descr> ')' }

<pad_digital> ::= '[' PAD_DIGITAL ']'
                  { <pad_digital_level_0_setting> | <pad_digital_IBIS_model> }

<pad_digital_circuit> ::= PAD_DIGITAL_CIRCUIT <technology> <circuit_type> ';'

<pad_digital_IBIS_model> ::=
            PAD_DIGITAL_IBIS_MODEL <model_ref> <IBIS_model_name> ';'

<pad_digital_level_0_setting> ::=
            <pad_digital_circuit> |
            <pad_digital_name> |
            <pad_digital_pull_down> |
            <pad_digital_pull_up> |
            <pad_digital_threshold>

<pad_digital_name> ::= PAD_DIGITAL_NAME <name_value> ';'

<pad_digital_pull_down> ::=
            PAD_DIGITAL_PULL_DOWN <pull_down_list> ';'

<pad_digital_pull_up> ::=
            PAD_DIGITAL_PULL_UP <pull_up_list> ';'

<pad_digital_threshold> ::= PAD_DIGITAL_THRESHOLD <threshold_list> ';'

<pad_elec_inst> ::= <pad_type> [ <pad_elec_ref> ]

<pad_elec_ref> ::= <name_value>

<pad_electrical_name> ::= <name_value>

<pad_geom> ::= '[' PAD_GEOM ']' {
            <pad_geom_level_0_setting> |
            <pad_geom_level_1_setting> |
            <pad_geom_level_2_setting> }

<pad_geom_bond_sites> ::=
            PAD_GEOM_BOND_SITES <num_points> { <point> } ';'

<pad_geom_inst> ::= <pad_geom_ref> <point> <rotmir>
```

```
<pad_geom_level_0_setting> ::= <pad_geom_name> | <pad_geom_shape>

<pad_geom_level_1_setting> ::=
            <pad_geom_metal_extent> |
            <pad_geom_passivation_extent> |
            <pad_geom_tolerance>

<pad_geom_level_2_setting> ::=
            <pad_geom_bond_sites>

<pad_geom_metal_extent> ::= PAD_GEOM_METAL_EXTENT  <center> <shape> ';'

<pad_geom_name> ::= PAD_GEOM_NAME <name_value> ';'

<pad_geom_passivation_extent> ::=
            PAD_GEOM_PASSIVATION_EXTENT  <center> <shape> ';'

<pad_geom_ref> ::= <name_value>

<pad_geom_shape> ::= PAD_GEOM_SHAPE <shape> ';'

<pad_geom_tolerance> ::= PAD_GEOM_TOLERANCE <tolerance> [ <tolerance> ] ';'

<pad_ID> ::= <ID_value>

<pad_info> ::= [ <pad_common_name> [ <pad_swap_code> ] ]

<pad_supply> ::= '[' PAD_SUPPLY ']' { <pad_supply_level_0_setting> }

<pad_supply_current_max> ::= PAD_SUPPLY_CURRENT_MAX <current> ';'

<pad_supply_level_0_setting> ::=
            <pad_supply_name> |
            <pad_supply_current_max> |
            <pad_supply_voltage>

<pad_supply_name> ::= PAD_SUPPLY_NAME <name_value> ';'

<pad_supply_voltage> ::= PAD_SUPPLY_VOLTAGE <supply_spec> ';'

<pad_swap_code> ::= <swap_function> <pin_type> <pin_group>

<pad_type> ::= SUPPLY_POWER | SUPPLY_GROUND | SIGNAL_DIGITAL |
                  SIGNAL_ANALOG | TEST_POINT | NO_CONNECT | NOT_DEFINED

<pads_list> ::= <pad_ID> { <pad_ID> }

<passive_driver> ::= PULL_UP | PULL_DOWN

<pin_group> ::= <numeric_value>

<pin_type> ::= <numeric_value>

<point> ::= <x_coord> <y_coord>

<polygon>   ::= POLYGON <polygonal_area>

<polygonal_area> ::= <num_points> { <point> }4+

<polygonal_area_list> ::=
            '(' <polygonal_area> ')' {'(' <polygonal_area> ')'}  |
                <polygonal_area>     {',' <polygonal_area>    } ';'

<power> ::= <numeric_value>

<power_area> ::= <power> <polygonal_area>

<power_area_list> ::=
             '(' <power_area> ')' {'(' <power_area> ')'}  |
                 <power_area>     {',' <power_area>    }

<power_capacitance> ::= <numeric_value>

<power_ID> ::= <ID_value>
```

```
<pull_down> ::= <supply_voltage> <current> <voltage>

<pull_down_list> ::=  <pull_down>     {',' <pull_down>    } |
                  '(' <pull_down> ')' {'(' <pull_down> ')'}

<pull_up> ::= <supply_voltage> <current> <voltage>

<pull_up_list> ::=  <pull_up>     {',' <pull_up>    } |
                '(' <pull_up> ')' {'(' <pull_up> ')'}

<range_supply> ::= RANGE <voltage_min> <voltage_max>

<record_terminator> ::= <new_line> | <carriage_return> | <line_feed>

<rectangle> ::= RECTANGLE <width> <height>

<revision_mark> ::== <ISO_alphanumeric> { <ISO_alphanumeric> }

<revision_value> ::== <revision_mark> { <revsep> <revision_mark> }

<revsep> ::= '.' | ':' | '-'

<rotation> ::= 0 | 90 | 180 | 270

<rotmir> ::= <rotation> [ <mirror> ]

<sep_type> ::= SAWED | SCRIBE_AND_BREAK

<shape> ::= <circle> | <rectangle> | <polygon>

<single_layer> ::= <material_name> [ <thickness> ]

<size> ::= <dimension>

<special_characters> ::= ';' | ',' | '(' | ')' | '[' | ']' | '|'

<specific_heat_capacity> ::= <numeric_value>

<step> ::= <dimension>

<string_value> ::= '"' { <ISO_character> } '"'

<supply_descr> ::= <pad_ID> <power_ID> <ground_ID>

<supply_group_list> ::=
            '(' <supply_descr> ')' { '(' <supply_descr> ')' }  |
                <supply_descr>     { ',' <supply_descr>     }

<supply_spec> ::= <fixed_supply> | <range_supply>

<supply_voltage> ::= <numeric_value>

<swap_function> ::= <numeric_value>

<technology> ::= ECL | TTL | CMOS | NMOS | PMOS

<text_string> ::= <string_value> | { <ISO_character> }

<thermal_conductivity> ::= <numeric_value>

<thickness> ::= <dimension>

<threshold> ::= <supply_voltage> <logic_low> <logic_high>

<threshold_list> ::=  <threshold>      { ',' <threshold> }  |
                  '(' <threshold> ')' { '(' <threshold> ')' }

<threshold_voltage> ::= <numeric_value>

<time_value> ::== <hour> ':' <minute> [':' <second> ] [ AM | PM ]

<tolerance> ::= <numeric_value>

<trailer_section> ::= <DIE_Block_end>

<version_attribute> ::=  <revision_value> [ <date_value> [ <time_value> ]]
```

```
<VHDL_entity_name> ::= <name_value>

<VHDL_map_pair> ::= <VHDL_entity_name> <BSDL_package_decl>

<voltage> ::= <numeric_value>

<voltage_max> ::= <numeric_value>

<voltage_min> ::= <numeric_value>

<voltage_nom> ::= <numeric_value>

<white_space> ::= <new_line> | <carriage_return> | <line_feed> |
                  <form_feed> | <space> | <htab>

<width> ::= <dimension>

<x_coord> ::= <numeric_value>

<y_coord> ::= <numeric_value>
```

Appendix B: Keyword List

The keywords of the DIE Format (Section and Setting identification names) are collected here for convenience. This document section is informative and deemed incorrect if different from the main, reference manual sections.

block_DIE_format_version
block_disclaimer
block_level
block_notes
block_source
block_version
die
die_backside_finish
DIE_block
DIE_block_end
die_bonding_diagram
die_bonding_sequence
die_BSDL
die_conditions_bonding
die_conditions_process
die_conditions_sealing
die_conditions_special
die_conditions_storage
die_description
die_fiducials
die_flatness
die_IBIS
die_junction_temperature
die_lead_frame
die_load_factor
die_lot
die_manufacturer
die_manufacturer_cage
die_mask_version
die_measure
die_military_spec
die_name
die_notes
die_packaged_part_attach
die_packaged_part_bonding
die_packaged_part_name
die_pad_dielectric
die_pad_metal
die_pad_pitch
die_pads
die_pads_jumpers
die_pads_noncontact_area
die_pads_supply_grouping
die_pads_tolerance
die_pads_VHDL_map
die_passivation_extent_size
die_passivation_material
die_power_capacitance
die_power_cell
die_power_max
die_power_nom
die_quiescent_current
die_saw_step_error
die_saw_width
die_section_version
die_separation
die_size
die_size_tolerance
die_solder_bump
die_source
die_specific_heat_capacity
die_substrate_connection
die_substrate_material
die_technology
die_thermal_conductivity
die_thickness
die_thickness_tolerance
die_type
die_VHDL
die_wafer_scribe_line_width
die_wafer_size
die_wafer_step_and_repeat
model
model_end
model_name
model_type
pad_digital
pad_digital_circuit
pad_digital_IBIS_model
pad_digital_name
pad_digital_pull_down
pad_digital_pull_up
pad_digital_threshold
pad_geom
pad_geom_bond_sites
pad_geom_metal_extent
pad_geom_name
pad_geom_passivation_extent
pad_geom_shape
pad_geom_tolerance
pad_supply
pad_supply_current_max
pad_supply_name
pad_supply_voltage