EMD

(Electrical Module Description Specification)

Version 0.9 Draft 7
May 1, 2013

Ratified TBD

Contents

1   ELECTRICAL MODULE DESCRITION INTRODUCTION

The Electrical Module Description (EMD) format describes electrical interconnectivity between and within modules for multi-chip modules (MCM), interposers, connectors and cables. EMD files support connecting components described by IBIS files and also connecting to other EMD files. This format has more general modeling and connection capabilities than supported by the older Electrical Board Description (EBD) format documented within the IBIS Specification.

For example, the EMD format uses sub-circuits document by the IBIS Interconnect SPICE Sub-circuit (IBIS-ISS) Specification and/or Touchstone files to describe the electrical properties of interconnects. Consequently the EMD format can support coupling between paths and also describe broadband lossy, distributed, frequency-dependent interconnects. Furthermore, parameters values different files can be passed in to support corner analysis based on typical, slow, and fast settings. EMD follows a simplified tree structure as a variation of the Algorithmic Modeling Interface (AMI) format described within IBIS, and the EMD format is easy to implement and expand in the future.

Because the EBD syntax is restricted to uncoupled paths and does not support frequency-dependent losses, it has limited accuracy for some high speed applications. The EBD format also uses an older Fork/Endfork syntax to describe branches in board topology. This syntax is topologically limited and sometimes awkward to implement.

However, the EBD format might be an alternative for lower frequency applications where IBIS-ISS or Touchstone files are not available. L/R/C elements for discrete or distributed networks (for uncoupled transmission lines) are described directly within the format. This avoids management or references to IBIS-ISS or Touchstone files and the corresponding co-simulation or emulation involving a separate SPICE tool. Most EDA tools already support EBD, and some tools can generate EBD files directly from physical board databases.

The EMD format described in this document is applicable for higher frequency modeling applications and topological generality where the EBD format falls short..

2   STATEMENT OF INTENT

3   GENERAL SYNTAX RULES AND GUIDELINES

This section contains general syntax rules and guidelines for ASCII EMD files:

1) The content of the file is case sensitive.

2) To facilitate portability between operating systems, file names used in the EMD file must only have lower case characters. File names should have a base name of no more than forty (40) characters followed by a period ("."), followed by a file name extension of no more than three characters. The file name and extension must use characters from the set (space, " ", 0x20 is not included):

   i)   a b c d e f g h i j k l m n o p q r s t u v w x y z
   ii) 0 1 2 3 4 5 6 7 8 9 _ ^ $ ~ ! # % & - { }) (@ ' `

   b) The file name and extension are recommended to be lower case on systems that support such names.

3) The length of a line is not limited to a specific number of characters. A line is followed by a line termination sequence. The line termination sequence must be one of the following two sequences: a linefeed character or a carriage return followed by linefeed character.

4) Anything following the "|" (pipe) character is ignored and considered a comment on that line.

5) Valid scaling factors are:

   i)   T = tera   k = kilo   n = nano
   ii) G = giga   m = milli   p = pico
   iii) M = mega   u = micro   f = femto

   b) When no scaling factors are specified, the appropriate base units are assumed. (These are volts, amperes, ohms, farads, henries, and seconds.) The parser looks at only one alphabetic character after a numerical entry, therefore it is enough to use only the prefixes to scale the parameters. However, for clarity, it is allowed to use full abbreviations for the units, (e.g., pF, nH, mA, mOhm). In addition, scientific notation IS allowed (e.g., 1.2345e-12).

6) The use of tab characters is legal, but they should be avoided as much as possible. This is to eliminate possible complications that might arise in situations when tab characters are automatically converted to multiple spaces by text editing, file transferring and similar software..

7) Only ASCII characters, as defined in ANSI Standard X3.4-1986, may be used in an IBIS file. The use of characters with codes greater than hexadecimal 07E is not allowed. Also, ASCII control characters (those numerically less than hexadecimal 20) are not allowed, except for tabs or in a line termination sequence. As mentioned in item 6) above10 above, the use of tab characters is discouraged.

8) Definition of a Parameter Tree

   a) A Parameter Tree is a data structure consisting of one Root Node, Branch Nodes and Leaf Nodes, and relationships between the Nodes. A Root Node does not have any Parent Nodes. Branch and Leaf Nodes may have only one Parent Node. A Leaf Node does not have any Child Nodes. Leaf Nodes must have Values (either a single Value or a List of Values. Root Nodes and Branch Nodes may not have Values.

   b) An EMD file implements a Parameter Tree as an ASCII file with a specific syntax and a specific context.

    c) EMD Parameter Tree Syntax

       i) The EMD File is an ASCII file that contains Tokens delimited by the characters "(", ")", and "White Space". White Space is a sequence of characters containing the "<space>" character, "<tab>" character, and "<end of line sequence>" characters. If a Token contains any of the above delimiters, the token must be surrounded by the " character. The " character is an invalid character inside of a Token.

       ii) A Token Preceded by a "(" is a Node. The first Node in an EMD file is the Root Node. There may be no Tokens in the file before the Root Node. A Node must be followed by either an "(" or a Token. If followed by a "(" then it is a Parent Node and the Token is a Daughter Node; if followed by a Token then it is a Leaf Node and the Token is a Value of the Leaf Node. A Value may be followed by either another Value Token or a ")". If followed by a Value Token, that Value is added to the list of values of the Leaf Node. If followed by a ")" there will be no more values added to this Leaf Node, and the next non-white space character must either be a ")" or a "(". If it is a "(" the next Token will be a Daughter Node of the previous Parent Node. If it is a ")", then there will be no more Daughter Nodes of the previous Parent Node, and that Parent Nodes Parent will be the parent of subsequent Daughter Nodes until there is a matching ")", and so on until the Parameter Tree is concluded when there is a ")" that matches the first "(" of the Parameter Tree.

    d) EMD Parameter Tree Context

       i) The rules for naming Nodes and the rules for Values of Leaf Nodes are described in section 4 of this document.

9) From the AMI section of 5.1

    a) The content of the parameter definition file is case sensitive.

    b) Only the pipe ("|") character is acceptable as a comment character regardless of what the calling IBIS file uses for the comment character.

    c) The line length of the parameter definition file is not limited to a specific number of characters.

    d) The root name in the file may contain an arbitrary string and does not need to match the file name.

    e) A white space in the parameter definition file may be one or more space, tab, and/or line termination characters.

    f) Scaling factors or suffixes, such as p, n, etc., are not permitted in the Parameter definition file.

    g) Scientific and floating point notation is permitted.

    h) Note:

       i) Throughout this section, text strings inside the symbols "<" and ">" should be considered to be supplied or substituted by the model maker. Text strings inside "<" and ">" are not reserved and can be replaced.

4   ELECTRICAL MODULE DESCRIPTION

INTRODUCTION:

A "module level component" is the generic term to be used to describe an electronic module which can contain components, modules, and which can connect to another module through a set of external pins. The electrical connectivity of such a module level component is referred to as an "Electrical Module Description". An electrical module description file (an .emd file) is defined to describe the connections of a module level component between the external pins and its exiternal pins, components and other modules. A module can be a package with a single component, a package with multiple components, a board with zero, one, or multiple components, a board with zero, one, or multiple components, an interposer, a connector, and a cable. The external pins can be pins, ball pads, or connector sockets or plugs.

A fundamental assumption regarding the electrical module description is that the interconnect between the external pins, components, and modules can be represented by IBIS-ISS subckts and Touchstone files directly. Also, this current description does allow one to describe electrical coupling between connections.

A component is represented by a .ibs file which may represent either a die and package, or a bare die.

A connection is represented by a list of its external pins, component pins and module pins that have a small insertion loss at Nyquist between all of the pins.

What is, and is not, included in an Electrical Module Description is defined by its boundaries. For the definition of the boundaries, see the Description section under the Interconnect Branch.

Usage Rules:

A .emd file is intended to be a stand-alone file, not referenced by or included in any .ibs or .pkg file. Electrical Module Descriptions are stored in a file whose name looks like <filename>.emd, where <filename> must conform to the naming rules given in Section 3 of this specification. The .emd extension is mandatory.

The Parameter Tree shall contain a Root, Branches and Leaves. The Root, Branch and Leaf names shall be "token strings" that may contain the characters " ", "(", or ")". They may not contain the tab or " character. If the token string contains " ", "(", or ")", the token must be surrounded by the "s. A Leaf may contain one or more values. Values shall be token strings. The rules for values token strings shall be the same as the rules for Root, Branch and Leaf token strings (may be contextual specific).

Contents:

A .emd file is structured using a Parameter Tree Structure. It must contain a descriptive Root, and the following Branches listed below:

      Header_Information

      External Pin ListReference_Designators) Connections

      Interconnect

The Root must contain either one or both of the following Branches

      External Pin List

      Reference_Designators

The Root may contain the following Branches

      External_Designators

The Header_InformationBranch must contain the following Leaves:

      EMD_Version

      File_Name

      File_Rev

The Header_InformationBranch may also contain the following optional Leaves:

      Manufacturer

      Date

      Source

      Notes

      Disclaimer

      Copyright

The actual module description is contained in the following Branches:

      External Pin ListExternal Pin List

      Reference_Designators Reference_Designators

      External_Designators

      Connections

      Interconnect

ROOT DEFINITION

Root: **<Root Name>**

Required: Yes

Description: Marks the beginning of an Electrical Module Description.

Usage Rules: The root name of the module level component.

Example:

`(16Meg_X_8_SIMM_Module`

BRANCH DEFINITIONS

Branch: Header_Information

Required: Yes

Description: Branch contains Leaves containing requires and optional Leaves

Usage Rules: This Branch contains the required leaves EMD_Version, File_Name, File_Rev and may contain the optional Leaves Manufacturer, Date, Source, Notes, Disclaimer, Copyright

Example:

```
(16Meg_X_8_SIMM_Module
     (Header_Information
```

Header_Information**LEAF DEFINITIONS:**

Leaf: **EMD_Version**

Required: Yes

Description: Declares the version .emd file.

Usage Rules: Following the leaf is the emd version.

Example:

```
(16Meg_X_8_SIMM_Module
      (Header_Information
            (EMD_Version 1.0)
```

Leaf: **File_Name**

...

Leaf: **File_Rev**

…

Leaf: **Manufacturer**

Required: No

Description: Declares the manufacturer of the components(s) that use this .emd file.

Usage Rules: Following the leaf is the manufacturer's name. It must not exceed 40 characters, and can include blank characters. Each manufacturer must use a consistent name in all .emd files.

Example:

```
(16Meg_X_8_SIMM_Module
      (Header_Information
            (Manufacturer "Quality SIMM Corp.")
```

Leaf: **Date**

Leaf: **Source**

Leaf: **Notes**
Leaf: **Disclaimer**
Leaf: **Copyright**

Branch: **External Pin List**

Required: Yes

Description: Branch contains Leaves containing module external pins.

Usage Rules: This Branch contains leaves for each module external pin. Each external pin leaf shall consist of a pin_name and a connection_name. If there is a non-blank external module reference designator, the module pin name shall be of the form <designator>.<pin>.

Example:

```
(16Meg_X_8_SIMM_Module
     (External Pin List
          (A4 PWR1)
          (A5 PWR1)
          (A1 DQ1)
          (A2 DQ2)
          (A3 DQ3)    | Connector pin_names might be A.A3, Bside.A3
                      | Will we need a list of Connectors?
     )  | Supply pins and connection are included (pullup)
```

Branch: **Reference_Designators**

Required: Yes

Description: Branch contains Leaves containing modules and components in the module.

Usage Rules: This Branch contains leaves for each module and component in the module. Each module and component leaf shall consist of a reference designator. If the leaf is a component its values are an IBIS file name and component. If the leaf is a module its value is the EMD file name. All designator names must be unique.

```
(16Meg_X_8_SIMM_Module
     (Reference_Designators
          (u23 (Type IBIS) (File pp100.ibs) (Component Processor))
          (u24 (Type EMD) (File simm.emd))
          (u25 (Type IBIS) (File ls244.ibs) (Component 74LS244a))
          (r26 (Type IBIS) (File r10K.ibs) (Component 10K_Pullup))
     )
```

Branch: **External_Designators**

Required: Yes

Description: Branch contains Leaves containing designators of the modules external connections.

Usage Rules: External connections are module pins, balls, or connector sockets or plugs. A module may have no external connections. This branch is required if the module has two or more external connection designators. The designator may have a Pin_Depth branch if the external pins are physical pins that penetrate a board or substrate via. The Pin_Depth branch chall contain a Value leaf and a Units leaf. The value of the Units leaf must be either Inch, Mil, mm, m, cm or microns.

```
(16Meg_X_8_SIMM_Module
     (External_Designators
          (J1)
          (P1 (Pin_Depth(Value .1) (Units Inch))
     )
)
```

Branch: **Connections**

Required: Yes

Description: Branch contains Leaves containing connections.

Usage Rules: This Branch contains leaves for each connection. Each connection shall consist of a connection_name followed by the external pins and component and modules pins that are connected.

(Connections are also known as Extended Nets in some EDA tools.)

A Pin can occur in one and only one connection.

An unconnected pin would not be in any connection, or may be in a connection by itself.

Example:
```
(16Meg_X_8_SIMM_Module
     (Connections
          (DQ1 A1 U1.17 U2.17 U3.17)
          (DQ2 A2 U1.18 U2.18 U3.18)
          (DQ3 A3 U1.19 U2.19 U3.19)
          (DQx U1.20 U2.20 U3.20)
          (PWR1 A4 A5 U1.30 U2.30 U3.30 U3.31)
          (UN1 A16)
          |    (F12  A.F12 B.F12)   Connector connection

     )
```

Branch: **Interconnect**

Required: Yes

Description: Branch contains Branches containing Model Interconnect Protocols for IBIS-ISS subckts or Touchstone files.

Usage Rules: This Branch contains branches containing Model Interconnect Protocols (MIP) for IBIS-ISS subckts and Touchstone files.. Each MIP shall contain leaves (or Branches) defining the IBIS-ISS file and subckt name or Touchstone file. Each MIP shall contain a Branch defining the ports of the IBIS-ISS subckt or Touchstone file.

If the MIP contains an IBIS-ISS Branch it may also contain a Parameter Branch, which shall contain Branches or Leaves for each Parameter that can be passed as parameters on the IBIS-ISS instance. If the parameter is a leaf, then it shall contain a single value that is a legal IBIS-ISS parameter value. If the parameter is a Branch, it may contain a single Corner leaf with Typ, Min, and Max values.

Note that an Interconnect might be for a single connection, a differential connections, a group of coupled signal connections, a supply connection, a group of supply connections, a group of supply and signal connections. A connection may appear in one or more Interconnects.

All pins are not necessarily in an Interconnect.

In a Interconnect has a connection, all pins in the connection must be included (does this make sense for supply?)

Example:
```
(16Meg_X_8_SIMM_Module
     (Interconnect
         (DQ1
              (IBIS_ISS_File xyz.iss)
              (IBIS_ISS_Circuit (Corner dq1_typ dq1_slow dq1_fast)
              (Parameters
                   (Impedance 50ohms)
                   (Delay    (Corner 30ps 40ps 20ps)
                   (Tstonefile 'xyz_dqx.s2p')
              )
              (Ports
                   (1 A1)
                   (2 U1.17)
                   (3 U2.17)
                   (4 U3.17) | Ports number not monotonic
              )
         )
         (DQ2
              (Tstonefile (Corner DQ2_Typ.s4p DQ2_Slow.s4p
                                  DQ2_Fast.s4p))
```

```
            (Ports
                (1 A2)
                (2 U1.18)
                (3 U2.18)
                (4 U3.18)
            )
        )
        (DQ1_DQ2
            (Tstonefile DQ1_DQ2.s128p)
            | Radek: Unused port terminations
            (Ports
                (21 A1)
                (22 U1.17)
                (23 U2.17)
                (24 U3.17)
                (15 A2)
                (16 U1.18)
                (17 U2.18)
                (18 U3.18)
            )
        )
    )
)
```

Branch: **<MIP>**

Required: Yes

Description: A Branch of the **Interconnect** Branch.

Usage Rules: This Branch contains a Model Interconnect Protocols (MIP) for IBIS-ISS subckts or Touchstone files.. Each MIP shall contain leaves (or Branches) defining the IBIS-ISS file and subckt name or Touchstone file. Each MIP shall contain a Branch defining the ports of the IBIS-ISS subckt or Touchstone file.

The **Interconnect** Branch may contain one or more MIP Branches. If the **Interconnect** Branch contains more than one MIP Branch, the MIP Interconnect branches must have unique names.

Branch: **Ports**

Required: Yes

Description: A Branch of an MIP Branch

Usage Rules: This Branch contains **Port** Branches for the Ports of an IBIS-ISS subckt or a Touchstone file.

Branch: **<Port>**

Required: Yes

Description: A Branch of a **Ports** Branch

Usage Rules: This Branch contains two fields, Port Number and Port Pin. The Port Number shall be an integer number between 1 and the number of ports of the IBIS ISS subckt or the Touchstone File. All of the Port Numbers shall be unique. The Port Pin shall either be an External EMD Pin (from the **External Pin List**) or an Internal EMD Pin. An Internal EMD Pin shall be of the form <designator>.<pin>. <designator> shall be a Branch of **Reference_Designators**. If <designator is an EMD then <pin> shall be a Branch of the <designator>'s **External Pin List.** If <designator> is a .ibs component then <pin> is a [Pin] of the .ibs [Component]. All of the Port Names shall be unique

Branch: **Tstonefile**

Required: No

Description: A Branch of an **MIP** Branch

Usage Rules: This Branch contains the name of a Touchstone file. The Touchstone file must be located in the same directory of the EMD file. An MIP must contain either a Tstonefile Branch or both and IBIS_ISS_File and IBIS_ISS_Circuit Branch.

Branch: **IBIS_ISS_File**

Required: No

Description: A Branch of an **MIP** Branch

Usage Rules: This Branch contains the name of a file containing the IBIS_ISS_Circuit. The IBIS_ISS file must be located in the same directory of the EMD file. An MIP must contain either a Tstonefile Branch or both and IBIS_ISS_File and IBIS_ISS_Circuit Branch.

Branch: **IBIS_ISS_Circuit**

Required: No

Description: A Branch of an **MIP** Branch

Usage Rules: This Branch contains the name of a subcircuit containing in the IBIS_ISS_File. The An MIP must contain either a Tstonefile Branch or both and IBIS_ISS_File and IBIS_ISS_Circuit Branch.

Branch: **Parameters**

Required: No

Description: A Branch of an **MIP** Branch

Usage Rules: This Branch contains the names and values of parameters that should be included on instance of an IBIS-ISS subcircuit. This Branch is not allowed in MIP Branches that contain the Tstonefile Branch. The Branches of the Parameters Branch shall either be leaves containing a single value for the parameter, or a Corner Branch containing a Typ Min and Max value. The Typ Min and Max value be an integer, float or string. For integer or float there is not rule that min<=typ<=max.

Branch: **Supply**

Required: No

Description: A Branch of the **Root** Branch

Usage Rules: This Branch contains Branches containing the names and voltage values of supply connections. Each Branch contains the name of the supply connection and either a voltage value or a Corner leaf containing the Typ Min Max voltage values. The EDA tool may either use this value as the voltage supplied to the EMD, or may connect the External Pins of the supply connection to an external power distribution circuit.

Example:
```
(16Meg_X_8_SIMM_Module
     (Supply
          (VDD          3.2)
          (VQQ          (Corner 3.2 2.8 3.5))
     )
```

Notes

Overlapping MIP

Multiple MIPs can contain the same pins. For example, an EMD that has 12 external pins may have an MIP with an s24p Tstonefile and one s2p for each of the 12 Connections. If the User requests a simulation of a single Connection either the User or EDA tool can choose to use the s2p for that Connection or 2 of the ports of the s24p MIP. If the latter, EDA tool can choose to extract an s2p from the s24p or use just two of the ports of the s24p (being careful to terminate the unused ports of the s24p (Radek to expound on this)).

Terminator Connections

A Connection may have a termination circuit within an EMD (e.g. Pullup resistor). There are several ways that this can be accomplished. Consider the following example:

```
(16Meg_X_8_SIMM_Module
     (External Pin List
          (1 VDD)
          (2 Vref)
          (3 GND)
          (4 DQ1)
     )
     (Reference_Designators
          (u1        pp100.ibs   Processor)
     )
     (Interconnect
          (DQ1_A
               (IBIS_ISS_File dq_a.iss)
               (IBIS_ISS_Circuit dq_a)
               (Ports
                    (1 DQ1)
                    (2 U1.17)
               )
          )
          (DQ1_B
               (IBIS_ISS_File dq_b.iss)
               (IBIS_ISS_Circuit dq_b)
               (Ports
                    (1 DQ1)
                    (2 U1.17)
                    (3 1)
               )
```

```
            )
            (DQ1_C
                  (IBIS_ISS_File dq_c.iss)
                  (IBIS_ISS_Circuit dq_c)
                  (Ports
                        (1 DQ1)
                        (2 U1.17)
                        (3 2)
                  )
            )
            (DQ1_D
                  (IBIS_ISS_File dq_d.iss)
                  (IBIS_ISS_Circuit dq_d)
                  (Ports
                        (1 DQ1)
                        (2 U1.17)
                        (3 X)
                  )
            )
            (VDD
                  (IBIS_ISS_File vdd.iss)
                  (IBIS_ISS_Circuit vdd)
                  (Ports
                        (1 1)
                        (2 U1.1)
                        (3 X)
                  )
            )

      )
)

.subckt dq_a 1 2
W1 1 0 3 0 N=1 L=1.1 rlgcmodel=rlgc
W2 3 0 2 0 N=1 L=.1 rlgcmodel=rlgc
R1 3 4 1K
V2 4 0 DC=3.3
.ends dq_a

.subckt dq_b 1 2 3
W1 1 0 I1 0 N=1 L=1.1 rlgcmodel=rlgc
W2 I1 0 2 0 N=1 L=.1 rlgcmodel=rlgc
R1 I2 3 1K
.ends dq_b

.subckt dq_c 1 2 3
W1 1 0 I1 0 N=1 L=1.1 rlgcmodel=rlgc
```

```
W2 I1 0 2 0 N=1 L=.1 rlgcmodel=rlgc
R1 I2 3 1K
.ends dq_c

.subckt dq_d 1 2 3
W1 1 0 I1 0 N=1 L=1.1 rlgcmodel=rlgc
W2 I1 0 2 0 N=1 L=.1 rlgcmodel=rlgc
R1 I2 3 1K
.ends dq_d

.subckt vdd 1 2 3
R1 1 X .1
R2 X 2 .1
R1 X 3 1K
.ends vdd
```

Now consider the simulation deck for case _d showing power distribution and signal distribution:

```
* VRM (Voltage Regulator Module)
XVRM VDD_Source VDD_Sense VRM
* Board connection for VRM sense node
Rsense VDD_Sense EMD.1 .1
* Board connection for VDD_Source node
Rboard VDD_Source EMD.1 .1
Cboard EMD.1 0 1pF
* EMD Power distribution subckt Xvdd
X1 EMD.1 U1.1 Xvdd
* EMD signal subckt dq_d
X2 EMD.4 U1.17 dq_d
* IBIS B element
* <instance name> <Pullup> <pulldown> <pad> <Stimulus> …
B1 U1.1 0 U1.17 Stimulus
* Stimulus source
Vstim Stimulus 0 PWL=(…)* Chip on-die power distribution circuit
(simple Cdie)
Cdie U1.1 0 1pF
```

   1. The VRM generates VDD_Source based on sense voltage VDD_Sense
   2. Sense point is EMD.1 Power pin of EMD
   3. Rboard and Cboard is Power distribution from VRM to EMD
   4. X1 is EMD Power distribution from EMD Power pin to IBIS Power pin
   5. X2 is signal connection from EMD signal pin to IBIS signal pin
   6. B1 is instance of IBIS Output [Model]
   7. Cdie in on-die Power distribution circuit
   8. The IBIS component is a bare die component, so no package model

  9. The connection from the EMD signal pin to the rest of the signal
     interconnect and other IBIS component is not shown

The B1 element draws current from the node U1.1 at transitions (Composit
Current). This causes fluctuations in the nodes U1.1, EMD.1, VDD_Source and
VDD_Sense. The VRM responds to changes in VDD_Sense and adjusts VDD_Source to
maintain VDD_Sense at some target voltage. The variations in U1.1 causes
changes to the output of the B element (U1.17), ultimately changing the voltage
at the Input end(s) of the of the DQ signal.