AN INTRODUCTION TO MHDL

David L. Barton
Douglas D. Dunlop

Intermetrics, Inc.
McLean, Virginia

Abstract

The MIMIC Hardware Description Language (MHDL) was recently developed under the sponsorship of DARPA and is intended for use in describing microwave and analog hardware. We describe how MHDL addresses the key issues of (1) the representation of hardware hierarchical structure and support for multiple views and alternatives, (2) signal representation, including a mathematically-rigorous signal concept that allows a variety of signal formats such as time-based and frequency-based signals, (3) general capabilities for describing device behavior at multiple levels of abstraction, (4) compatibility with VHDL and support for mixed-mode simulation of hybrid circuits, and, (5) support for concurrent engineering and performance modeling in the development of microwave systems.

Hardware Description Languages

A hardware description language is a format used in describing hardware designs. The format gives a standard, textual form to the information in a document that describes a piece of microwave hardware. Thus, the information in a data sheet or an SCD may be represented in the hardware description language. The MIMIC program has extended the concept of hardware description languages into the microwave arena, and has produced the MIMIC Hardware Description Language (MHDL). A description written in MHDL is given meaning by the **semantics** of the language. The semantics of a language provide a correspondence between a description in the language and a mathematical model of that description. In the case of MHDL, the description model is also a mathematical model of the device being described. Various tools, such as simulators and visualization tools, can operate on this model to provide information to the design engineer.

Models and Structures

The primary organizing unit of an MHDL description is the **model**. A model is a representation of the device, or of some portion of the device that "stands on its own". Intuitively, a model is a section (or a subsection) in a design document that addresses a self-contained part of the design. This may correspond to an actual component, or to a part of the design that has no independent existence, but which is separated out for the sake of presenting the overall design in pieces small enough to be understood. Just as a document refers to other documents (indeed, summary documents may consist almost completely of references to other documents), so MHDL models may be derived from other models.

The backbone of a microwave system description is the hierarchical set of block diagrams that shows how the system is composed. A top level diagram may break down the system into subsystems, where more detailed diagrams may show actual components and component interconnections; at the more detailed level, layouts may take the place of block diagrams that show the geometrical relationships between the various components. In MHDL, the block diagram (and a layout) is represented textually by the **structure**. A block diagram has a straightforward representation as an MHDL structure that specifies what the blocks are and how they are put together. The diagram in Figure 1 is equivalent to the structure found in Figure 2:

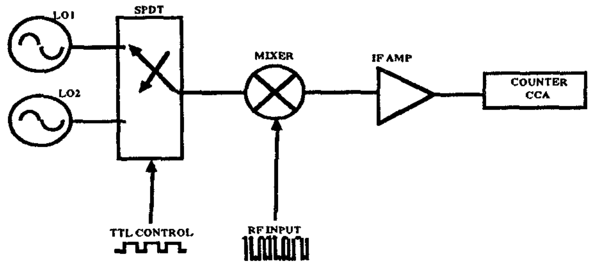

Figure 1. Sample block diagram

Several things should be noted about the two figures. For each block in the diagram, there is a component listed under the component section of the structure. The model is made up of these components, and the information in the model is provided primarily by the information in the components. Each component is mapped to a model. This name of this model is given by the name directly following the ": :" symbol. This name may be changed later, at the discretion of the designer, by a **configuration**; this will be described further later. Thus, models contain structures that are made up of components; in turn, these components are given meaning by other models. This process continues recursively.

The two lines originating outside the diagram are **connectors**; they represent the interface this model presents to those who would use it. The term "connector" is a neutral word that represents a port, terminal, or physical or electromagnetically coupled transfer of energy or information. The connectors of the structure and the connectors of the components are joined by connections (statements preceded by the keyword `connect`); these

```
structure electrical_breakdown
    connectors ttl_control, rf_input;
    definitions
        lo1.frequency = 8 GHz;
        lo2.frequency = 17 GHz;
    components
        lo1:: oscillator;
        lo2:: oscillator;
        spdt:: switch;
        mix:: mixer;
        if_amp:: amplifier;
        count:: counter;
    connection map
        connect lo1.osc_out, spdt.at1;
        connect lo2.osc_out, spdt.at2;
        connect lpdt.cont, ttl_control;
        connect spdt.out, mix.carrier;
        connect mix.out, if_amp.in;
        connect mix.in, rf_input;
        connect if_amp.out, counter.in;
end electrical_breakdown;
```

Figure 2. Structure describing block diagram

connections represent any kind of information flow between parts of the description.

Each of these parts of the description --- structures, components, and connectors --- may have information attached to it. The sum total of this information describes the model of the device. Two kinds of information may be attached to parts of the description. **Attributes** contain information such as device parameters. Frequency is a device parameter of the oscillators; this attribute is set by the statements "`lo1.frequency = 8 GHz`" in figure 2. **Signals** represent information that flows past, or occurs at, the point in the description to which the signal is attached.

Signals are a basic concept in MHDL. An MHDL signal is defined just as signals are mathematically: as functions from one or more independent variables (usually time) to a dependent variable. Such functions may be defined in a number of ways, including mathematical expressions and tables of measured data with an interpolation function. A simple definition of the oscillator assigns a sinusoidal signal of the proper frequency to the output connector; thus, the model for the oscillator in the example above might contain the statement "`out.value = sin(frequency * t)`". The symbol"t" is a **special name** in MHDL, and always represents time.

Thus, the MHDL model of a microwave design consists of:

- A **model**, that represents the device itself.
- A **structure**, that represents the block diagram.
- **Components**, or blocks on the diagram.
- **Connectors**, or points of information transfer.
- **Connections**, which join connectors together.
- **Attributes** that hold device parameters and information.
- **Signals**, that represent information flow across parts of the design.

These different language structures can represent microwave hardware to an arbitrary level of complexity. Frequently, however, more than one view is needed to completely describe a design; for example, an electrical view and a thermal view. Different views are represented in MHDL as different structures. A model can contain an arbitrary number of structures which represent different views of the microwave hardware. Components and connections in a structure refer to the connectors in structures of the same name (electrical or thermal), once the components are matched up with models. Thus, structures at different levels are matched up by name. This can be overridden by explicitly naming different structures if necessary; for example, to join an electrical and a thermal connector for purposes of reflecting thermal properties in an electrical model.

Modeling Microwave Behavior

Microwave behavior is thus represented by information (attributes and signals) attached to various parts of the description. Microwave behavior is modeled by stating mathematical relationships between the information on the different parts of that description. These mathematical relationships are stated using two language structures: the **value definition**, which defines information as flowing from an expression to the attribute or signal, and the **equation**, which states a relation between different attributes and signals in the form of a mathematical equation.

For example, the action of the oscillator is modeled by attaching the signal "`sin(frequency * t)`" to the out connector, and naming this signal "`value`". A very simple switch model can be defined by choosing the value of the out connector based upon the value of the TTL control line, which gives a value of 0 or 5 volts. Such a statement in MHDL would look like:

```
out.value(t) =
        if TTL_Control(t) == 0 volts
           then in1.value(t)
           else in2.value(t);
```

At any given time, this switch places the value of the `in1` connector on the `out` connector if the TTL control line is at zero volts; otherwise, the switch places the value of the `in2` connector. Including this value definition (information flows from the right hand side of the "=" symbol to the left hand side) defines the action of the SPDT switch (to a given level of accuracy).

Standard Connectors

Such a combination of features is obviously extremely flexible. MHDL defines a number of basic ways of modeling hardware: attaching information to connectors and stating mathematical relationships about that information. In particular, three different **connector types** are defined as standard MHDL concepts. Signal flow connectors define a direction of information flow, and the information on out signals as a function of the information on the in signals. Electrical connectors define the voltage and current of each connector, upon which conservation laws are defined to hold. Distributed connectors define information in terms of incident and reflected waves.

Signal flow connectors always name their signal "`value`". A given structure has only one connection of direction out; all other connections are of direction in. The model must define the signal on its out connector as a mathematical expression of the signals on its in connectors. An example is the oscillator and the switch given above. Each of these devices has a single out connector whose value is defined in terms of all the other in connectors.

Electrical connectors each define a voltage and an amperage signal. The definition of the electrical connectors automatically states that Kirchoff's current law applies to each electrical connector; i.e., the sum of the currents on a specific connector must be zero. Devices with electrical connectors define their behavior by stating equations on the voltage and amperage of its connectors. For example, a resistor with connectors "`R1`" and "`R2`" would contain the equation "`R2.voltage - R1.voltage == R2.current * resistance`", where "`resistance`" is a model parameter giving the resistance of the resistor in ohms. Similarly, the current is given by the equation "`R1.current == -R2.current`". These two equations, along with Kirchoff's current law, completely characterize the behavior of the resistor.

Distributed connectors are used in S-matrix and other high frequency applications. Each distributed connector has an incident and a reflected wave attached to it. The incident and reflected waves of the various connectors of the device are related to each other in an equation that contains and S-matrix multiplication. Thus, an N port device contains the equation "`incident == S_matrix |*| reflected`", where "`incident`" and "`reflected`" are vectors formed of the incident and reflected signals of each of the connectors, and "`S-matrix`" is the set of S parameters that define the behavior of the device.

Building Models

These three pre-defined kinds of connectors do not preclude the definition of other kinds of connectors, at the discretion of the engineer. For example, the electrical connector type could be used as a model for a thermal connector type that obeys conservation laws. Connector types are defined via an MHDL **package**. A package is a collection of related definitions that a model can use by naming the package at the beginning of the model. This is called **inheriting** the package. Packages may inherit other packages as well.

Packages may be used for any purpose that calls for grouping related definitions together. Packages may:

- Define special kinds of connectors by combining the definitions of the attributes, signals, and equations for those connectors.
- Create collections of functions and attributes used by specific tools such as simulators.
- Create type, attribute, and signal definitions that project management decides needs to be included with all models.

Models are thus created out of packages, which contain related groups of definitions, and other models, which create a base upon which to build. Models can thus be built out of small pieces. As an example, a power amplifier might be built out of:

- The electrical connector package.
- A project package that requires budgeted power consumption figures for each device in the design.
- A package of amplifier device parameters.
- A general amplifier model that contains connector definitions for all kinds of amplifiers.

Each of these packages or models might consume one or two pages of MHDL text. Each may be understood as a self-contained unit, and may serve as a piece of the overall documentation. MHDL thus encourages breaking up the overall description into pieces small enough to be understood separately, but that combined together, provide a complete definition of a device.

Creating and Particularizing General Models

General models can be defined that are then particularized according to the parameters of the device involved. MHDL provides several specific features to assist in this particularization.

The first is the **configuration**. The configuration allows the user to select the models that are matched with the components in a structure. Thus, a structure such as shown in Figure 2 can be analyzed either as a signal flow description or as an electrical description by matching the components with models written as signal flow descriptions or as electrical descriptions. Models to be matched with a component can be made contingent upon attributes; for example, a low frequency or high frequency amplifier can be chosen depending upon the frequency range of the device that uses the amplifiers, as follows:

```
configuration all_frequency
   for op_amp use
      high_frequency_amp when freq>10 GHz,
      low_frequency_amp when freq<1 GHz,
      mid_frequency_amp;
   end op_amp;
end all_frequency;
```

This configuration selects one of three op amps based upon the frequency at which the op amp is to operate.

Another feature that encourages the writing of general models is the **generate** statement. The generate statement allows the model writer to create repetitive matrices of structures, such as a phased array radar from a variable number of T/R modules. An alternative form of the generate statement allows the selection of a number of definitions depending upon attributes; for example, a chain of amplifiers may be lengthened or shortened based upon the noise figure and required gain of the entire chain.

More general than either of these options is the library construct. A library is a collection of related models. Libraries are mapped to the file system in an implementation dependent manner. This can be used to substitute one collection of models for another; for example, a collection of electrical models may be subsititued for a collection of distributed S-matrix models.

External Tools and Functions

MHDL descriptions will be used by external tools, and may invoke models and tools written in other languages. MHDL uses the concept of the **primitive** function (primitive in the sense that it is not defined in MHDL) to accomplish this. Primitive values in MHDL include anything that is provided by the operating system or the design environment. This includes: external files (such as files of S-parameter figures), external simulators, and routines written in existing languages such as C and Fortran.

Together with packages written to work with specific tools, primitive values provide a general capability for communication with external packages. For example, a package might have the task of: assembling equations from all the electrical components, invoking a conservation law oriented simulator such as Saber(tm), and distributing the answers from the simulator to the individual components of the description. All of this can be implemented without the knowledge or attention of the model writer; thus, general models can be written without orienting the format of equations and values to a specific tool.

A primary example of this external interface is the predefined connection to digital models written in VHDL. A VHDL model consists of digital (VHDL) ports, processes, and (VHDL) signals. A correspondence between VHDL ports and connectors can be made so that the action of a digital model can be defined as a primitive function. This, together with the definition of VHDL signals in terms of MHDL discrete signals, allows the MHDL description to include digital components written in VHDL; alternatively, an MHDL description can work within a VHDL description by describing the interface between the two and defining the action of the analog hardware as a function between two VHDL signals.

Performance Modeling

The features of MHDL that allow the description of complex signals and behaviors also encourage the specification of performance measurements that can be verified via simulation or physical measurements. At the most primitive level, this process can consist of a comparison between measured and simulated performance. MHDL allows an arbitrary amount of information to be attached to the parts of the description. Each connector, for example, can be provided with a measured and a simulated incident and reflected wave. Where these two figures differ by more than a specific amount, this condition can be flagged.

This automatic flagging is provided by the **constraint**, a basic MHDL language feature. A constraint is a boolean expression with an accompanying message that is transmitted to the user when a constraint is violated. The MHDL language definition allows considerable latitude concerning when a constraint is applied and measured. An implementation may, at its option, test constraints continually, or when specifically indicated by the engineer.

Constraints can be used to check arbitrarily complex conditions. For example, a modulated signal could be submitted to a filter function and the results compared to the input information. If the two differ beyond a certain level, an error can be signaled.

MHDL allows noise signals to be represented either as normal signals (which can then be manipulated mathematically as any other signal), or as special data structures that refer to statistical distributions. The former representation has definite limitations; a given signal can, during a calculation, represent only one of an infinite number of noise signals. Extraction of noise from calculated signals is a topic of future research in MHDL; it is not clear how this important issue is best represented within the features of the language. Various tools take widely differing solutions, and a standard approach may be difficult to find.

Conclusion

MHDL provides a flexible notation for creating general models, particularizing those models based upon device parameters, and building microwave systems descriptions based upon those models. The flexibility of the language provides capability, but also the obligation to decide how to use this capability. Much further research is needed concerning the best way of writing general, tool-independent models. The mathematics of this is well-understood; the practicalities are not. Modeling guidelines, standard libraries, and calculational packages are needed before MHDL can realize its potential as a language for microwave models and design.

In particular, standard approaches to modeling important quantities such as noise and standard interpolation functions remain to be determined. There is no lack of mathematical capability to represent these issues; a common approach within the industry needs to be determined before MHDL can provide a standard representation.