Library Harmonization for Function

Version	Date
0.0	2004/10/09

Table of contents

1.0 Basic structure of a library model

1.1 Hierarchy of library objects

This section describes how objects in a library relate to each other:

Library, cell, pin

The definition of a basic structure of objects and relationship between objects is a prerequisite for library description. For example, a collection of cells is described in the context of a libray, a collection of pins is described in the context of a cell.

FIGURE 1. Library hierarchy in liberty

```
/* Liberty */
library (LibraryName) {
  cell (CellName) {
    pin (PinName) {
    }
  }
}
```

FIGURE 2. Library hierarchy in ALF

```
/* ALF */
LIBRARY LibraryName {
  CELL CellName {
    PIN PinName {
    }
  }
}
```

1.2 Definition of data related to library objects

Definitions related to objects in a library (e.g. unit, scope, pintype etc.) are also important for semantic understanding of the library contents.

2.0 Description of functionality

The functional specification of a cell (e.g. Nand, flipflop, memory etc.) has a strong impact on characterization. for example, characterization conditions for a timing arc are infered from the functional specification.

Some cells with special functionality (e.g. integrated-clock-gating-cell) have special requirements for characterization and modeling.

2.1 Combinatorial logic function

A combinatorial logic function is represented as a boolean expression associated with a pin. The boolean expression involves one or more other pins.

FIGURE 3. Combinatorial function in liberty

```
/* Liberty */
  cell (CellName) {
    pin (OutputPinName) {
      direction : output ;
      function : "BooleanExpression" ;
    }
    pin (InputPinName) {
      direction : input ;
    }
  }
```

FIGURE 4. Combinatorial function in ALF

```
/* ALF */
  CELL CellName {
    PIN OutputPinName {
      DIRECTION = output ;
    }
    PIN InputPinName {
      DIRECTION = input ;
    }
    FUNCTION {
      BEHAVIOR {
        OutputPinName = BooleanExpression ;
      }
    }
  }
```

TABLE 1. Operators involved in a boolean expression in order of precedence

operator	liberty symbol	ALF symbol
logical NOT (highest)	‘ or !	! or ~
exclusive OR		
logical AND	* or &	& or &&
logical OR (lowest)	+ or |	| or ||

2.2 Latch and flipflop

A latch or a flipflop is expressed in liberty as a predefined primitive. Pins of the cell are associated with the terminals of the primitive.

ALF has two modeling styles for sequential functions: a purely behavioral style, comparable to Verilog or VHDL, and a style using predefined primitives. The second style is used to establish a one-to-one correspondence between liberty and ALF.

The behavior of a predefined latch primitive or a predefined flipflop primitive is explicitely described in the IEEE 1603-2003 standard. Liberty implicitely assumes such a behavior when refering to a latch or a flipflop.

FIGURE 5. Latch description in liberty

```
/* Liberty */
  cell (CellName) {
    latch ( NonInvertedOutput , InvertedOutput ) {
      LatchVariable : "BooleanExpression" ;
      /* put more latch variables here */
    }
    /* put pin declarations here */
  }
```

FIGURE 6. Latch description in ALF

```
/* ALF */
  CELL CellName {
    /* put pin declarations here */
    FUNCTION {
      BEHAVIOR {
        /* put combinatorial statements here */
        ALF_LATCH {
          Q = NonInvertedOutput ;
          QN = InvertedOutput ;
          LatchVariable = BooleanExpression ;
          /* put more latch variables here */
        }
      }
    }
  }
```

FIGURE 7. Flipflop description in liberty

```
/* Liberty */
  cell (CellName) {
    ff ( NonInvertedOutput , InvertedOutput ) {
      FlipFlopVariable : "BooleanExpression" ;
      /* put more flipflop variables here */
    }
    /* put pin declarations here */
  }
```

FIGURE 8. Flipflop description in ALF

```
/* ALF */
  CELL CellName {
    /* put pin declarations here */
    FUNCTION {
      BEHAVIOR {
        /* put combinatorial statements here */
        ALF_FLIPFLOP {
          Q = NonInvertedOutput ;
          QN = InvertedOutput ;
          FlipFlopVariable = BooleanExpression ;
          /* put more flipflop variables here */
        }
      }
    }
  }
```

TABLE 2. Latch and flipflop variables in liberty and ALF

variable	liberty keyword	ALF keyword
trigger signal for latch	enable	ENABLE
trigger signal for flipflop	clocked_on	CLOCK
data input signal for latch	data_in	D
data input signal for flipflop	next_state	D
set to zero signal	clear	CLEAR
set to one signal	preset	SET
data output signal	*first argument of latch or flipflop function*	Q
inverted data output signal	*second argument of latch or flipflop function*	QN
value of data output when *set to zero* and *set to one* both active	clear_preset_var1	Q_CONFLICT
value of inverted data output when *set to zero* and *set to one* both active	clear_preset_var2	QN_CONFLICT

2.3 Scan flipflop

The description of a scan flipflop involves the following aspects:

1. The behavior in any mode

2. The behavior in normal (mission) mode only
3. The behavior in scan mode only

The behavior of a scan mode in any mode is described in the same way as a conventional flipflop (see Section 2.2 on page 6), only the boolean expression describing the next state is more involved. However, from this description it can not be infered which logic state corresponds to the normal mode and wwhich logic state corresponds to the scan mode.

To describe the normal mode, the scan flipflop is reduced to a conventional flipflop, and the extra pins necessary for scan mode are omitted. To describe the scan mode, the functionality and active state of the extra pins are declared.

FIGURE 9. Scan flipflop description in liberty

```
/* Liberty */
  cell (CellName) {
    /* put flipflop declaration here */
    /* put pin declarations here */
    test_cell () {
      /* put flipflop declaration without scan here */
      /* put pin declarations with scan functionality here */
    }
  }
```

FIGURE 10. Scan flipflop description in ALF

```
/* ALF */
  CELL CellName {
    /* put pin declarations here */
    /* put function declaration here */
    NON_SCAN_CELL {
      NonScanCellName {
        /* put pin map here */
      }
    }
  }
```

2.4 State table

3.0 Characterization data and calculation methods

3.1 Characterization dimensions

Timing data (and to a slightly lesser extend) power, signal integrity and reliability data should be characterized as a function of supply voltage(s) and/or environmental temperature. This leads typically to 4 or more characterization dimensions (input slew and output load cap being the two basic characterization dimensions).

Another "characterization dimension", albeit without interpolation or curve fitting, is the process corner.

Selection of significant characterization points and appropriate fitting into polynomials or accurate interpolation are important aspects in this context.

Other parameters: related to vt, back bias ...

On-chip process variation parameters: geometry ...

The semantics of characterization data is primarily defined by the physical meaning of the measurements used to generate the data. Describing or recommending a particular method of measurement (e.g. using SPICE, making silicon measurements, scaling available measurement data from other technologies etc.) is outside the scope of this project. Any measurement method that is deemed adequate and sufficiently accuracy can be used to generate the data.

On the other hand, the representation of the characterization data is semantically related to the calculation method used by the application tool to evaluate the data. The representation of data (e.g. look-up table, polynom etc.) implies that an application tool can evaluate a mathematical expression or interpolate/extrapolate a table.

One of the expected outcomes of this project is to define a standard for consistent representation of the same characterization data in Liberty and ALF, without hidden semantics related to either the characterization process or the application tool.

3.2 Lookup table

3.3 Equation

3.4 Other methods

4.0 Design rules, electrical constraints

Electrical design rules, such as max capacitance, max fanout, max slewrate on cell pins will be covered. Some of the rules, espacially fanout rules, have a legacy in their purpose. This legacy is important to understand in order to define their semantics correctly. Most of the rules are related to timing (see Section 2.2 on page 5) and can be covered as a subtopic therein.

5.0 Format extensibility

The purpose of format extensibility is two-fold. On one hand, it allows to embed information with proprietary semantics in the standard library. On the other hand, it allows experimental exploration or prototyping of new constructs as candidates for future standard contructs.

Timing data (and to a slightly lesser extend) power, signal integrity and reliability data should be characterized as a function of supply voltage(s) and/or environmental temperature. This leads typically to 4 or more characterization dimensions (input slew and output load cap being the two basic characterization dimensions).