This document contains suggested enhancements to the Advanced Library Format, using ALF 2.0 as baseline. The document serves as a worksheet rather than a formal proposal. The suggested enhancements are collected in no particular order. The idea is to keep track of evolving proposals here and then agree formally whether or not they should be part of the IEEE spec.

The following template is used throughout this document:

X.0 Item

relation to ALF 2.0 reference to ALF 2.0 chapter

relation to ALF IEEE reference to ALF IEEE chapter

History date of initial draft, date of revisions

X.1 Motivation

Explain reason for new feature

X.2 Proposal

Describe new feature

1.0 Level definition for Vector Expression Language

relation to ALF 2.0 5.3, 5.4, 11.3

relation to ALF IEEE

History initial draft April 16 2001 by Wolfgang
reviewed and rejected by Study Group April 16
rejection confirmed by Tim Ehrler May 1
changed title and closed May 4 by Wolfgang

1.1 Motivation

The vector expression language is a new concept which has almost no equivalent in legacy library model description languages. Currently there are EDA tools which support a subset of the vector expression language. Purpose of this proposal is to re-write the definitions in such a way that it is easy to identify subsets for different levels of support. For example: level0=basic subset, level1=intermediate subset, level2=full set in ALF 2.0, level3=full set in ALF 2.0 plus new proposed extensions.

1.2 Proposal

Level 0: single event, single event & boolean condition, two-event sequence

Level 1: N-event sequence, N-event sequence & boolean condition, alternative event sequence

Level 2: everything in ALF 2.0 (except if we decide to drop something fundamentally unpractical or un-implementable)

Level 3: new operators for repetition of sub-sequences

2.0 Metal Density

relation to ALF 2.0 9.2, 9.5

relation to ALF IEEE

History initial draft April 16 2001 by Wolfgang
reviewed and retained by Study Group April 16
o.k. as is by Tim Ehrler May 1

2.1 Motivation

Manufacturability in 130 nm technology and below requires so-called metal density rules. For a given routing layer, metal must cover a certain percentage of the total area within a lower and upper bound in order to ensure planarity. This percentage also depends on the total area under consideration, i.e., there are “local” and “global” metal density rules.

2.2 Proposal

Introduce new keyword DENSITY (or other word) for arithmetic model. Shall be non-negative number normalized between 0 and 1 (1 means 100%). Usable in context of LAYER (see ALF 2.0, chapter 9.5.1) with PURPOSE=routing (see ALF 2.0, chapter 9.5.2). Legal argument (i.e. HEADER) includes AREA, meaning the die area subjected to manufacturing of this layer.

Example:

```
LAYER metal1 {
   PURPOSE = routing;
   LIMIT {
      DENSITY {
         MIN {
            HEADER {
               AREA {
                  INTERPOLATION = floor;
                  TABLE { 0 100 1000 }
               }
            }
            TABLE { 0.2 0.3 0.4 }
         }
         MAX {
            HEADER {
               AREA {
                  INTERPOLATION = floor;
                  TABLE { 0 100 1000 }
               }
            }
            TABLE { 0.8 0.7 0.6 }
         }
      }
```

```
    }
}
```

Within an area of less than 100 units, the metal density must be between 20% and 80%. Within an area of 100 up to less than 1000 units, the metal density must be between 30% and 70%. Within an are of 1000 units or more, the metal density must be between 40% and 60%. The annotation INTERPOLATION=floor indicates that no interpolation is made for areas in-between, but the next lower value is used (see ALF 2.0, chapter 7.4.4).

3.0 Current

relation to ALF 2.0 8.1, 8.7, 8.15

relation to ALF IEEE

History initial draft April 162001 by Wolfgang
reviewed and retained by Study Group April 16
also reviewed by Tim Ehrler May 1
add text to clarify purpose by Wolfgang May 4

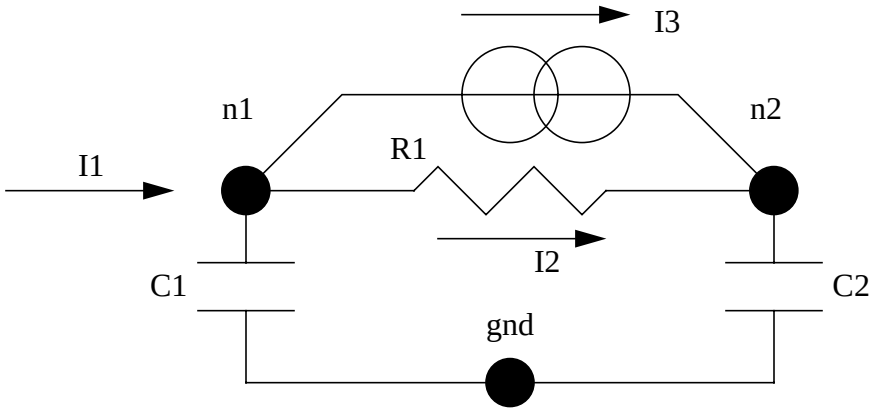
3.1 Motivation

CURRENT needs PIN annotation indicating the target point where the current is flowing into. Cannot define a branch of an electrical network where the current flows through.

Therefore there will be 3 types of CURRENT specification:

1. Current into PIN from unspecified source
2. Current through a COMPONENT with two terminal nodes
3. Current through an independent current source connected between two NODEs

see I1, I2, I3 in illustration



3.2 Proposal

In the context of WIRE, the following annotations for CURRENT shall be legal:

```
PIN = pin_identifier ;
```

Current flows from unknown source into the pin (already supported).

```
COMPONENT = component_identifier ;
```

Current flows through the component. The component must be a declared two-terminal electrical component in the context of the WIRE, i.e. a RESISTANCE, CAPACITANCE, VOLTAGE or INDUCTANCE (excluding mutual inductance, which has 4 terminals). The direction of the current flow is given by the order of node identifiers in the NODE annotation for that component (see ALF 2.0, chapter 8.15.3, 8.15.4).

```
NODE { 1st_node_identifier 2nd_node_identifier }
```

Current flows through a current source connected between the nodes. The direction of the current flow is given by the order of node identifiers in this NODE annotation.

Example:

```
WIRE interconnect_analysis_model_1 {
   CAPACITANCE C1 { NODE { n1 gnd } }
   CAPACITANCE C2 { NODE { n2 gnd } }
   RESISTANCE  R1 { NODE { n1 n2  } }
   CURRENT I1 { PIN = n1; }
   CURRENT I2 { COMPONENT = R1; }
   CURRENT I3 { NODE { n1 n2 } }
}
```

This example corresponds exactly to the illustration shown above.

4.0 Noise

relation to ALF 2.0 8.1, 8.14

relation to ALF IEEE

History initial draft April 16 2001 by Wolfgang
o.k by Tim Ehrler May 1
updated by Wolfgang May 4

4.1 Motivation

NOISE_MARGIN defines a normalized voltage difference between nominal signal level and tolerated signal level. If violated, the correct signal level can not be determined. In order to check against noise margin, actual noise must be calculated. Currently VOLTAGE is used for noise calculations. However, since noise margin is normalized to signal voltage swing, it would be convenient, if the actual noise could also be represented in a normalized way. In CMOS, actual noise and noise margin tend to scale with supply voltage. A non-normalized model requires supply voltage as a parameter, if the supply voltage is subject to variation. A normalized model would to a 1st order degree approximate the voltage scaling effect already and therefore eliminate the supply voltage as a model parameter.

4.2 Proposal

Introduce new keyword NOISE, representing a normalized voltage difference between nominal signal level and actual signal level. Same measurement definition as for noise margin (see ALF 2.0, chapter 8.14). Noise margin is violated, if noise is larger than noise margin.

Example: use examples in ALF 2.0, chapter 8.14, replacing VOLTAGE with NOISE and eliminating annotations MEASUREMENT=peak and CALCULATION=incremental.

Context-specific meaning of NOISE:

Context is output or bidirectional PIN: NOISE specifies maximum amount of noise at output pin, when any input pin is subjected to the amount of noise specified by NOISE_MARGIN. NOISE may have submodel HIGH and LOW.

Context is VECTOR with vector_expression: NOISE needs PIN annotation. NOISE specifies peak noise while pin is in “*” state. NOISE may only have submodel HIGH and LOW, if “?” state as opposed to “0” or “1” state is specified in vector_expression.

Context is CELL, SUBLIBRARY, or LIBRARY: no PIN annotation. NOISE specifies maximum amount of noise at any output or bidirectional pin within scope, unless this specification is overwritten locally.

5.0 Non-scan cell

relation to ALF 2.0 6.2, 11.2

relation to ALF IEEE

History initial draft April 16 2001 by Wolfgang
o.k. by Tim Ehrler May 1

5.1 Motivation

Non-scan cell defines the mapping between the pins of a non-scan cell (left-hand side) and the pins of a scan cell (right-hand side). The scan cells has always certain pins which do not exist in the non-scan cell. In some cases, the non-scan cell might have certain pins which do not exist in the scan cell (In such a case, the scan replacement can only be done, if the pin in question was tied to an inactive level in the non-scan cell in the first place).

Currently, the non-scan cell statement supports definition of LHS or RHS constants which specify the logic level to which the non-corresponding pins should be tied to. However, this definition is redundant, because every relevant pin in a cell model must have annotations for SIGNALTYPE and POLARITY in order to be usable for DFT tools. These annotations specify already the logic level to which non-corresponding pins must be tied.

5.2 Proposal

Reduce syntax for pin_assignment (see ALF 2.0, chapter 11.2) to the following:

```
pin_assignment ::=
   pin_identifier [ index ] = pin_identifier [ index ] ;
|  pin_identifier [ index ] = logic_constant ;
```

Only “ `pin_identifier [ index ] = pin_identifier [ index ] ;` “ will actually be used for non-scan cell. Since POLARITY defines the active signal level, the pin should be tied to the opposite level. For pins without POLARITY, the level does not matter (e.g. scan input for scan flipflop in non-scan mode).

Example (taken from ALF 2.0, chapter 6.2):

```
CELL my_flipflop {
   PIN q   { DIRECTION=output; } // SIGNALTYPE defaults to “data”
   PIN d   { DIRECTION=input; } // SIGNALTYPE defaults to “data”
   PIN clk { DIRECTION=input; SIGNALTYPE=clock; POLARITY=rising_edge; }
   PIN clear { DIRECTION=input; SIGNALTYPE=clear; POLARITY=low; }
}
CELL my_scan_flipflop {
   PIN data_out { DIRECTION=output; } // SIGNALTYPE defaults to “data”
   PIN data_in  { DIRECTION=input; } // SIGNALTYPE defaults to “data”
   PIN scan_in  { DIRECTION=input; SIGNALTYPE=scan_data; }
   PIN scan_sel { DIRECTION=input; SIGNALTYPE=scan_control;
```

```
      POLARITY { SCAN=high; } } // scan mode when 1, non-scan mode when 0
   PIN clock {DIRECTION=input; SIGNALTYPE=clock; POLARITY=rising_edge;}
   NON_SCAN_CELL {
      my_flip_flop {
         clk = clock;
         d   = data_in;
         q   = data_out;
      }
   }
}
```

The scan replacement works only, if the `clear` pin of `my_flip_flop` is tied high (active level is low). Note: This is an exceptional case and only shown because it might happen eventually. Normally, the pins of the scan cell represent a superset of the pins of the non-scan cell.

In order to simulate the non-scan mode, when the non-scan cell is replaced by the scan cell, the `scan_sel` pin of `my_scan_flipflop` must be tied low (scan mode level is high). The `scan_in` pin can be tied to either high or low.

This example shows that the constant logic levels need not be defined in the non-scan cell statements, because they can be completely infered from the POLARITY statements. The POLARITY statements are mandatory for DFT tools anyway.

6.0 VIOLATION in context of LIMIT

relation to ALF 2.0 7.5, 7.6, 8.4

relation to ALF IEEE

History Proposal May 1 by Tim Ehrler
written in doc May 4 by Wolfgang

6.1 Motivation

Want to specify level of severity, if a LIMIT is violated. Target is appropriate error report from tool.

6.2 Proposal

The VIOLATION statement may appear within the context of a LIMIT or an arithmetic model within LIMIT or an arithmetic submodel within LIMIT.

In this context, a MESSAGE_TYPE annotation or a MESSAGE annotation or both shall be legal within VIOLATION. A BEHAVIOR statement within VIOLATION shall only be legal if the LIMIT is within the context of a VECTOR. In that case, the vector_expression or boolean_expression shall define the triggering condition for the behavior described in the BEHAVIOR statement.