Library Harmonization for Timing

Version	Date
0.0	11/17/03

Template for liberty/ALF xref examples

```
/* liberty */                    /* ALF */
```

1.0 Basic description of timing arcs

1.1 Overview timing arcs

Timing arcs are defined not only by standalone statements but also by the context in which the statements appear. This is shown in Figure 1 on page 2.

FIGURE 1. Basic timing arc description in liberty and ALF

```
/* liberty */                          /* ALF */
cell (CellName) {                      CELL CellName {
  pin(FromPin) {                         PIN FromPin {
    direction : input;                     DIRECTION = input;
  }                                      }
  pin(ToPin) {                           PIN ToPin {
    direction : output;                    DIRECTION = output;
    timing() {                           }
      timing_type : timing_type;         VECTOR (vector_expression) {
      timing_sense : timing_sense;         ALF_TimingModel
      lib_TimingModel                    }
    }                                    }
  }
}
```

In both liberty and ALF, a timing arc is defined in the context of a CELL identified by a **CellName**. A declaration of each PIN involved in the timing arc is required, refered herein as the **FromPin** and the **ToPin**.

In liberty, the timing model is further defined inside the declaration of the **ToPin**. In ALF, the timing model is defined by the declaration of a VECTOR, separate from the declaration of each PIN.

The occuring edge combinations are defined in liberty by ***timing_type*** and ***timing_sense***. In ALF, the edge combinations are defined by a ***vector_expression***.

In ALF, there is no dependency between the ***vector_expression*** and the ***ALF_TimingModel***. In liberty, there is a dependency between the ***timing_type*** and the ***lib_TimingModel***, as shown below.

TABLE 1.

Liberty keyword		**ALF keyword**
timing_type	***lib_TimingModel***	***ALF_TimingModel***
combinational	cell_rise, cell_fall	DELAY
	rise_transition, fall_transition	SLEWRATE
three_state_enable	cell_rise, cell_fall ?	DELAY
three_state_disable	cell_rise, cell_fall ?	DELAY
rising_edge	cell_rise, cell_fall	DELAY
	rise_transition, fall_transition	SLEWRATE
falling_edge	cell_rise, cell_fall	DELAY
	rise_transition, fall_transition	SLEWRATE
preset	cell_rise	DELAY
	rise_transition	SLEWRATE

TABLE 1.

Liberty keyword		**ALF keyword**
timing_type	***lib_TimingModel***	***ALF_TimingModel***
clear	cell_fall	DELAY
	fall_transition	SLEWRATE
setup_rising	rise_constraint, fall_constraint	SETUP
setup_falling	rise_constraint, fall_constraint	SETUP
hold_rising	rise_constraint, fall_constraint	HOLD
hold_falling	rise_constraint, fall_constraint	HOLD
recovery_rising	intrinsic_rise, intrinsic_fall	RECOVERY
recovery_falling	intrinsic_rise, intrinsic_fall	RECOVERY
removal_rising	intrinsic_rise, intrinsic_fall	REMOVAL
removal_falling	intrinsic_rise, intrinsic_fall	REMOVAL
skew_rising	intrinsic_rise, intrinsic_fall	LIMIT.SKEW.MAX
skew_falling	intrinsic_rise, intrinsic_fall	LIMIT.SKEW.MAX
non_seq_setup_rising	intrinsic_rise, intrinsic_fall	SETUP
non_seq_setup_falling	intrinsic_rise, intrinsic_fall	SETUP
non_seq_hold_rising	intrinsic_rise, intrinsic_fall	HOLD
non_seq_hold_falling	intrinsic_rise, intrinsic_fall	HOLD
nochange_high_high	rise_constraint	SETUP
	fall_constraint	HOLD
nochange_high_low	rise_constraint	SETUP
	fall_constraint	HOLD
nochange_low_high	rise_constraint	SETUP
	fall_constraint	HOLD
nochange_low_low	rise_constraint	SETUP
	fall_constraint	HOLD

1.2 Delay and Slew

A timing model involving delay and slew measurements is declared as follows:

FIGURE 2. Timing model declarations for delay and slew in liberty and ALF

```
/* liberty */                          /* ALF */
pin(ToPin) {                           VECTOR (vector_expression) {
  timing() {                             DELAY {
    timing_type : timing_type;             FROM { PIN = FromPin; }
    timing_sense : timing_sense;           TO { PIN = ToPin; }
    related_pin : "FromPin";               ALF_CalcData
    DelayKeyword (lib_CalcType) {        }
      lib_CalcData                       SLEWRATE { PIN = ToPin;
    }                                      ALF_CalcData
    SlewKeyword (lib_CalcType) {         }
      lib_CalcData                     }
    }
  }
}
```

In liberty, the timing model declaration can be optionally qualified by the ***timing_type*** **combinational**. The combination of edges is defined by the combination of ***timing_sense***, ***DelayKeyword*** and ***SlewKeyword***. The mapping of these liberty constructs into a ***vector_expression*** in ALF is shown in Table 2 on page 4.

TABLE 2. Mapping of liberty and ALF constructs for Figure 2, “Timing model declarations for delay and slew in liberty and ALF,” on page 4)

liberty construct				ALF construct
timing_type	***timing_sense***	***DelayKeyword***	***SlewKeyword***	***vector_expression***
combinational	positive_unate	cell_rise	rise_transition	01 FromPin -> 01 ToPin
		cell_fall	fall_transition	10 FromPin -> 10 ToPin
	negative_unate	cell_rise	rise_transition	10 FromPin -> 01 ToPin
		cell_fall	fall_transition	01 FromPin -> 10 ToPin
	non_unate	cell_rise	rise_transition	?! FromPin -> 01 ToPin
		cell_fall	fall_transition	?! FromPin -> 10 ToPin
rising_edge	?	cell_rise	rise_transition	01 FromPin -> 01 ToPin
	?	cell_fall	fall_transition	01 FromPin -> 10 ToPin
falling_edge	?	cell_rise	rise_transition	10 FromPin -> 01 ToPin
	?	cell_fall	fall_transition	10 FromPin -> 10 ToPin

Note: The representation of the actual calculation data in liberty (*lib_CalcType*, *lib_CalcData*) and ALF (*ALF_CalcData*) is independent of the physical nature of the data, i.e., timing data or power data or other data. The mapping between those liberty and ALF constructs is shown [insert reference].

1.3 Threshold definitions

The thresholds for delay and slew measurements in liberty are normalized values between 0 and 100, to be interpreted as percentage values. The corresponding thresholds in ALF are normalized values between 0 and 1.

FIGURE 3. ALF template for liberty threshold definitions

```
DELAY {
  FROM {
    THRESHOLD {
      RISE = input_threshold_pct_rise ;
      FALL = input_threshold_pct_fall ;
    }
  }
  TO {
    THRESHOLD {
      RISE = output_threshold_pct_rise ;
      FALL = output_threshold_pct_fall ;
    }
  }
}
SLEWRATE {
  FROM {
    THRESHOLD {
      RISE = slew_lower_threshold_pct_rise ;
      FALL = slew_upper_threshold_pct_fall ;
    }
  }
  TO {
    THRESHOLD {
      RISE = slew_upper_threshold_pct_rise ;
      FALL = slew_lower_threshold_pct_fall ;
    }
  }
}
```

1.4 Tristate delay

2.0 Description of conditional timing arcs

2.1 Description of an existence condition

2.2 Description of a value condition