Liberty™ Release Notes
Version 2009.06

September, 2009

These release notes present the latest information about the 2009.06 Liberty syntax extensions. The 2009.06 syntax extensions are described in the following sections:

Synchronous Pin Type in Latch Cells

The data pins of latches can have different categories, such as data, synchronous set, synchronous reset, and synchronous enable pins. Beginning with the 2009.06 release, Liberty syntax provides the `data_in_type` attribute to identify the type of data for a latch cell. The `data_in_type` attribute can be defined in the .lib file and specified on the pins in the library cells. The valid values for the `data_in_type` attribute are `data`, `preset`, `clear` and `load`.

Syntax

The syntax for the `data_in_type` attribute is as follows:

```
cell (cell_name) {
 …
pin (name) {
 …
data_in_type: enum(data, preset, clear, load) ;
}
 …
} /* End of cell group */
```

The arguments are as follows:

`data`

Identifies the pin as a synchronous data pin.

`preset`

Identifies the pin as a synchronous preset pin.

`clear`

Identifies the pin as a synchronous clear pin.

`load`

Identifies the pin as a synchronous load pin.

Example

The following example shows the `data_in_type` attribute set to `data` and `preset`, respectively.

```
library(sync_latch_pin_sample) {

  delay_model : table_lookup;

voltage_map( VDD, 0.8);  /* primary power */
voltage_map(VSS, 0.0);   /* primary ground */
…

cell(latch_with_sync_set) {
  cell_footprint : inv;
  area : 1.0;
pg_pin(VDD) {
…
}
pg_pin(VSS) {
…
}

  latch(IQ,IQN) {
    data_in : "D + S";
    enable : "CK";
  }

  pin(D) {
    direction : input;
    capacitance : 1.0;
    data_in_type : data;
    timing() {
…
  }

  pin(S) {
    direction : input;
    capacitance : 1.0;
    data_in_type : preset;
    timing() {
…
    }
  }

  pin(Y) {
    direction : output;
    function : "IQ";
    power_down_function : "!VDD + VSS";
    timing() {
…
```

```
    }/* End timing group */
  } /* End pin group */
 }/* End cell group */
}/* End library group */
```

Power Down Functionality For Sequential Cells

You can use the `power_down_function` string attribute to specify the Boolean condition under which the cell's output pin is switched off by the state of the power and ground pins (when the cell is in off mode due to the external power pin states). Beginning with the 2009.06 release, you can specify the `power_down_function` attribute for combinational *and* sequential cells. For simple and complex sequential cells, `power_down_function` also determines the condition of the cell's internal state.

Syntax

Liberty provides `power_down_function` support for the following types of sequential cells: flip-flops, latches, and state tables. These cells can be described in the `ff`, `latch`, and `statetable` groups. The following sections provide the syntax for the `ff`, `latch`, and `statetable` groups, respectively.

power_down_function Syntax For Flip-Flops

```
library (name) {
   cell (name) {
      ff (variable1,variable2) {
         //...flip-flop description...
         clear : "Boolean expression" ;
         clear_preset_var1 : L | H | N | T | X ;
         clear_preset_var2 : L | H | N | T | X ;
         clocked_on : "Boolean expression" ;
         clocked_on_also : "Boolean expression" ;
         next_state : "Boolean expression" ;
         preset : "Boolean expression" ;
         power_down_function : "Boolean expression" ;
      }
   …
   }
…
}
```

power_down_function Syntax For Latch Cells

```
library (name) {
   cell (name) {
      latch (variable1,variable2) {
         //...latch description...
         clear : "Boolean expression" ;
         clear_preset_var1 : L | H | N | T | X ;
         clear_preset_var2 : L | H | N | T | X ;
         data_in : "Boolean expression" ;
         enable : "Boolean expression" ;
         preset : "Boolean expression" ;
         power_down_function : "Boolean expression" ;
      }
   …
   }
…
}
```

power_down_function Syntax for State Tables

```
statetable( "input node names", "internal node names" ){
   table : " input node values : current internal values :\
           next internal values,\
           input node values : current internal values: \
           next internal values" ;
   power_down_function : "Boolean expression" ;
}
```

Example

The following example shows the `power_down_function` attribute specified in the `ff` group.

```
library ("low_power_cells") {
   cell ("retention_dff") {
      pg_pin(VDD) {
         voltage_name : VDD;
         pg_type : primary_power;
      }
      pg_pin(VSS) {
         voltage_name : VSS;
         pg_type : primary_ground;
      }
      pin ("D") {
         direction : "input";
      }
      pin ("CP") {
         direction : "input";
      }
      ff(IQ,IQN) {
         next_state : "D" ;
         clocked_on : "CP" ;
         power_down_function : "!VDD + VSS" ;
      }
      pin ("Q") {
         function : " IQ ";
         direction : "output";
         power_down_function : "!VDD + VSS";
      }
   …
   }
…
}
```