

Liberty for Advanced Flows (Based on 2008.09 release)

Rajesh Kumar

November 6, 2008

AGENDA

- Motivation – Complete Design Flows
- Compact CCS Power
- Back-Bias Pins
- Customer PG Types
- Decap/Filler/Tap cell
- Critical Area Analysis (CAA) Table
- PLL Model
- Variation Aware Leakage

Motivation – Complete Design Flows

- Newest Liberty enhancements help flows:
 - Multi-corner multi-mode (MCMM)
 - via CCS timing
 - Low Power
 - Low power attributes and PG/bias pins
 - Additional power modeling using CCS, VA leakage.
 - DFM
 - Critical Area Analysis

Compact CCS

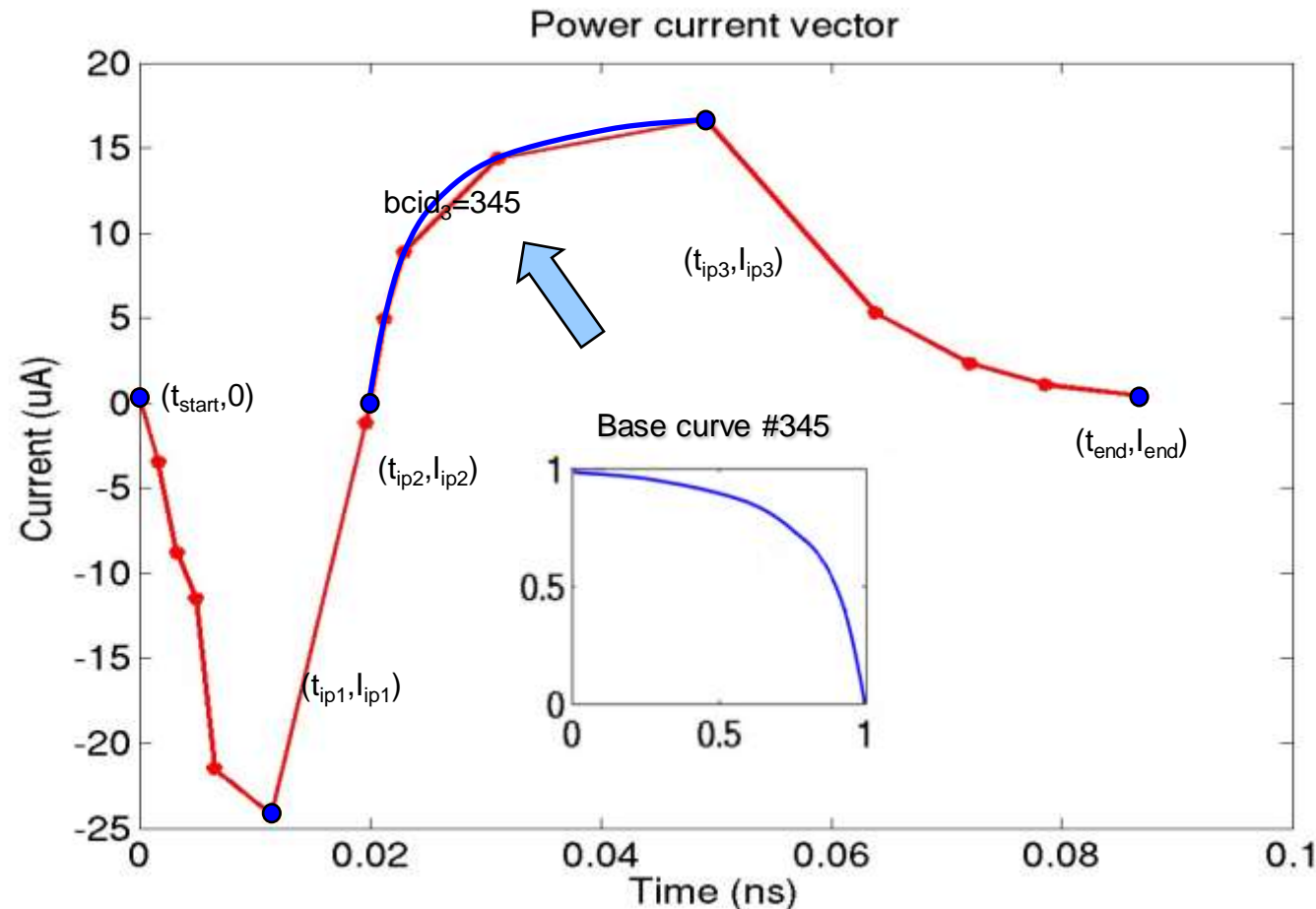
- With move to 45nm and below, there is a need to reduce library data size
 - Increasing number of cells, corners, $V_t(s)$
- Compact CCS modeling offers an efficient representation of current waveforms
 - Compact CCS timing introduced in late 2006 provides 3X smaller library data size
 - CCS power now supported in compact format to provide up to 4X smaller library data size

Compact CCS Power

- Apply Base Curve Technology in Liberty dynamic current waveform format
- Each dynamic current waveform is represented by 2 or more base curves
- Can share base curves with compact CCS timing
- Testing shows up to 4X library file size reduction

Apply Based Curves to CCS Power

- Segment the waveform
- model the shape of each segment using a base curve



Compact CCS Power: Liberty Syntax (1 of 2)

```
library ( <library_name> ) {  
  compact_lut_template ( <template_name> ) {  
    base_curves_group : <bc_name>;  
    variable_1 : input_net_transition | total_output_net_capacitance;  
    variable_2 : input_net_transition | total_output_net_capacitance;  
    variable_3 : input_net_transition | total_output_net_capacitance;  
    variable_4 : curve_parameters ;  
    index_1 ("float, ..., float");  
    index_2 ("float, ..., float");  
    index_3 ("float, ..., float");  
    index_4 ("init_time, init_current, bc_id1, point_time1,  
            point_current1, bc_id2, [point_time2, point_current2,  
            bc_id3, ...], end_time, end_current");  
  }  
}
```

Compact CCS Power: Liberty Syntax (2 of 2)

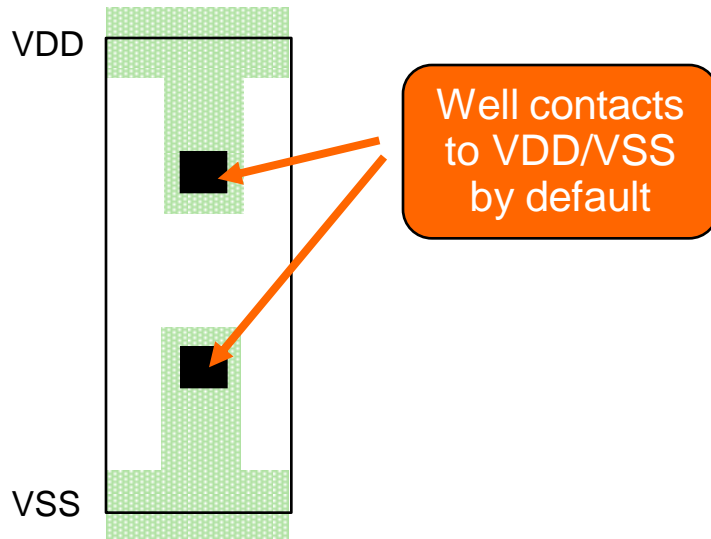
```
base_curves( <template_name> ) {
  base_curve_type : ccs_half_curve | ccs_timing_half_curve ;
  curve_x("float, ..., float");
  curve_y("integer_index, float, ..., float"); /*base curve #1*/
  curve_y("integer_index, float, ..., float"); /*base curve #2*/
}
cell ( <cell_name> ) {
  dynamic_current() {
    switching_group() {
      pg_current ( <pg_pin_name> ) {
        compact_ccs_power ( <template_name> ) {
          values("t_start, I_start, bcid1, tip1, Iip1, bcid2, ..., t_end, I_end",
            "t_start, I_start, bcid1, tip1, Iip1, bcid2, tip2, Iip2, bcid3, ..., t_end, I_end",
            ... ) ;
          }}}}}}}}}
        }
      }
    }
  }
}
```

Liberty TAB Extensions

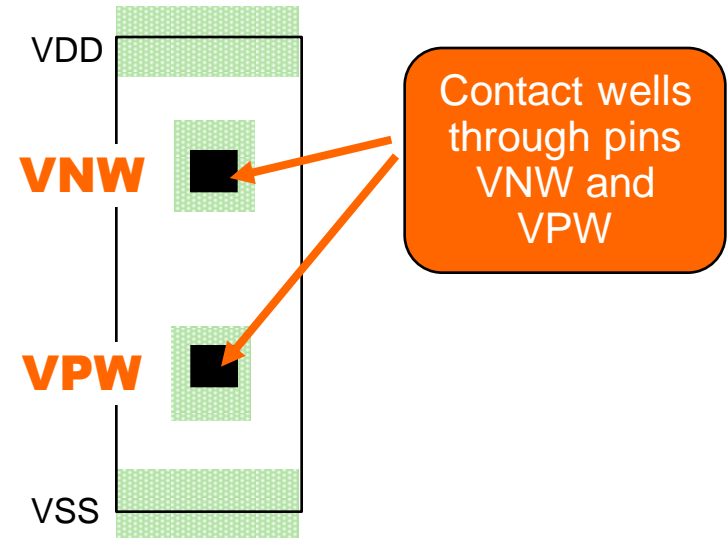
- Compact CCS Power
- **Back-Bias Pins**
- Customer PG Types
- Decap/Filler/Tap cell
- Critical Area Analysis (CAA) Table
- PLL Model
- Variation Aware Leakage

Special Fill Cells – Not Just for Back-Biasing

Standard FILL_TIE

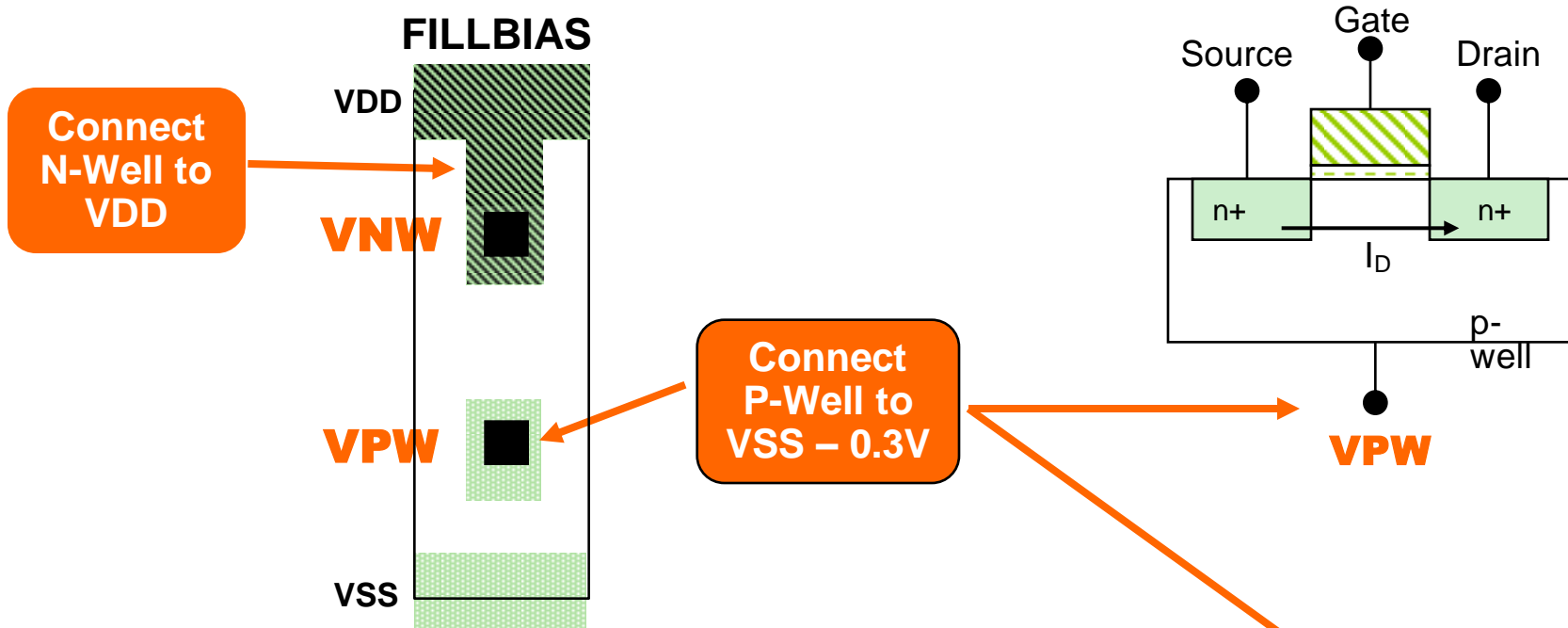


FILL_BIAS

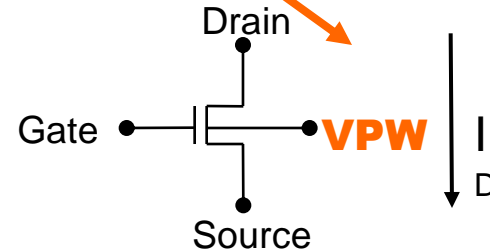


- Back biasing for performance/leakage optimization
 - N-well voltage different from VDD
 - Substrate or P-well (triple well process) voltage different from VSS
 - Bias voltage routed as signal pin or special power net

Back Bias Example with NMOS



- When **VPW** < Source
 - V_{th} increases
 - I_D decreases
- Leakage **decreases**



Back-Bias pin: Liberty Extension (1 of 4)

- Provide capability to model substrate bias pins and their relationship with PG pins and signal pins
- Substrate bias pins on stdcell can be connected thru
 - abutment on substrate layer; or
 - tap point on metal layer
- Substrate bias pins on macro can be
 - Connected thru tap point;
 - Connected thru abutment; or
 - Completely internal bias pin not accessible outside the macro

Back-Bias pin: Liberty Extension (2 of 4)

```
Cell (<cell_name>) {  
  pg_pin (<pg_pin_name>) {  
    direction : input | output | internal ;  
    pg_type : primary_power | primary_ground | backup_power | backup_ground |  
             internal_power | internal_ground | nwell | pwell | deepnwell |  
             deppwell> ; /* regular well; isolation well */  
    bias_connection : device_layer | routing_pin ;  
    related_bias_pin : “ <bias_pin_name> <bias_pin_name> ... “ ;  
    ...  
  }  
  pin (<signal_pin>) {  
    related_power_pin : <pg_pin_name> ;  
    related_ground_pin : <pg_pin_name> ;  
    related_bias_pin : “<bias_pin_name> <bias_pin_name> ... “ ;  
    ...  
  }  
}
```

Back-Bias pin: Liberty Extension (3 of 4)

- “True Routing Pin Modeling” (Std Cell)

```
pg_pin(<name>) {  
    pg_type : ... | pwell | nwell | deepnwell | deeppwell ;  
    direction : input ;  
    bias_connection : routing_pin ;  
}
```

- “Virtual Routing Pin Modeling” (Std Cell)

```
pg_pin(<name>) {  
    pg_type : ... | pwell | nwell | deepnwell | deeppwell ;  
    direction : input ;  
    bias_connection : device_layer ;  
}
```

Back-Bias pin: Liberty Extension (4 of 4)

- Macro with Pin Access for Power Net Connection (True | Virtual)

```
pg_pin(<name>) {  
    pg_type : ... | pwell | nwell | deepnwell | deeppwell ;  
    direction : input ;  
    bias_connection : routing_pin | device_layer;  
}
```

- Macro w/o Pin Access for Power Net Observability outside

```
pg_pin(<name>) {  
    pg_type : ... | pwell | nwell | deepnwell | deeppwell ;  
    direction : internal ;  
}
```

Liberty TAB Extensions

- Compact CCS Power
- Back-Bias Pins
- **Customer PG Types**
- Decap/Filler/Tap cell
- Critical Area Analysis (CAA) Table
- PLL Model
- Variation Aware Leakage

Custom PG Types

- Allow specification of arbitrary user-defined pg_type, on top of the enumerated pg_type's in Liberty
- Enumerated pg_type's have heavy semantics
 - Primary_power, primary_ground
 - Backup_power, backup_ground
 - Internal_power, internal_ground
 - pwell, nwell, deeppwell, deepnwell (proposed)
- User-defined pg_type takes arbitrary string with no semantics

Custom PG Type: Liberty extension

```
pg_pin (<pg_pin_name>) {  
    pg_type : primary_power | primary_ground |  
            backup_power | backup_ground |  
            internal_power | internal_ground |  
            nwell | pwell | deepnwell | deppwell ;  
    user_pg_type : <user_pg_type_name>;  
    ...  
}
```

Liberty TAB Extensions

- Compact CCS Power
- Back-Bias Pins
- Customer PG Types
- **Decap/Filler/Tap cell**
- Critical Area Analysis (CAA) Table
- PLL Model
- Variation Aware Leakage

Decoupling Cap/Filler/Tap Cell

- Explicit identifiers for decoupling cap cell, filler cell and tap cell
- Current Liberty model of all 3 cells look alike
 - They are all cells with only pg pins, but no signal pins.

Decoupling Cap/Filler/Tap Cell

- Decoupling Cap Cell
 - Capacitor between power and gnd
- Filler cell
 - Connect gaps after placement
- Tap Cells
 - Bias the N-wells or P-wells
- New attributes to differentiate these cells in Liberty

Decoupling Cap/Filler/Tap Cell: Liberty Extension

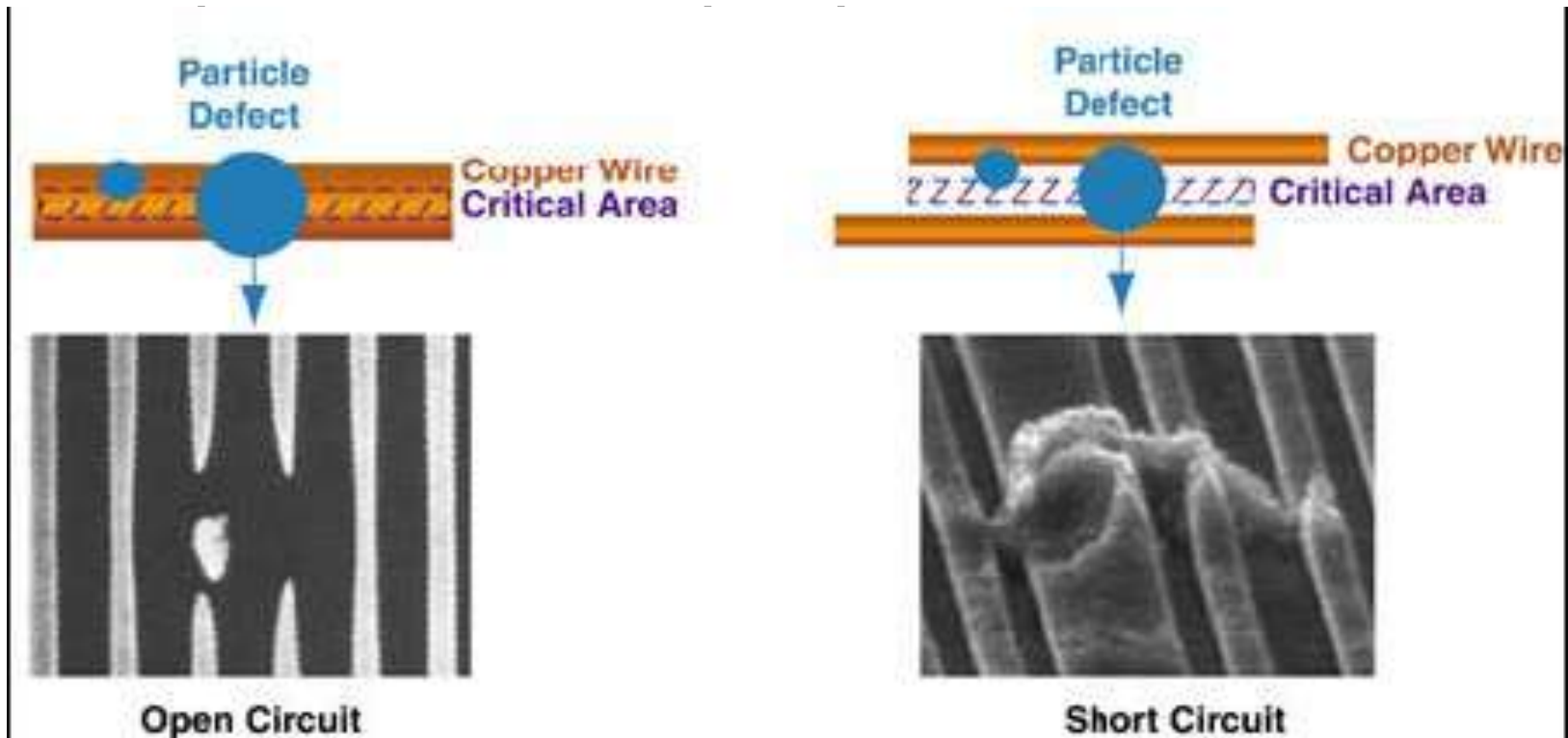
- New cell group attributes
 - Is_decap_cell : true | false
 - Is_filler_cell : true | false
 - Is_tap_cell : true | false

Liberty TAB Extensions

- Compact CCS Power
- Back-Bias Pins
- Customer PG Types
- Decap/Filler/Tap cell
- **Critical Area Analysis (CAA) Table**
- PLL Model
- Variation Aware Leakage

Critical Area

- The region of a library cell where the center of the particle must fall to create



Critical Area: Liberty Extension (1 of 2)

```
library(my_library) {  
  distance_unit : um | mm;           /* distance unit */  
  dist_conversion_factor : <integer>; /* distance resolution */  
  critical_area_lut_template (<template_name>) {  
    variable_1 : defect_size_diameter;  
    index_1 ("float...float");  
  }  
}
```

```
/* declare layers */  
device_layer(<string>) {}           /* e.g. diffusion layer OD */  
poly_layer(<string>) {}             /* e.g. poly layer */  
routing_layer(<string>) {}         /* e.g. Metal1, Metal2 */  
cont_layer(<string>){}             /* via layer, such as VIA */
```

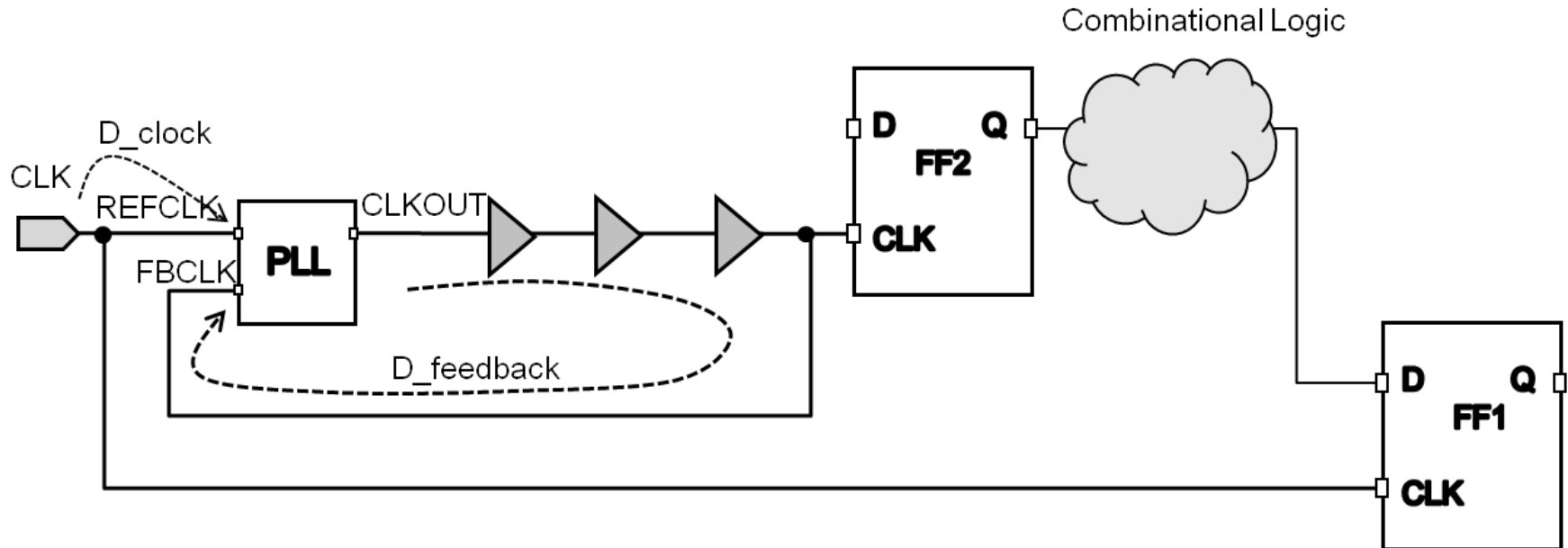
Critical Area: Liberty Extension (2 of 2)

```
cell (my_cell) {  
  functional_yield_metric() {  
    critical_area_table (<template_name>) {  
      defect_type : open | short | open_and_short ;  
      related_layer : <string>;  
      index_1 ("float...float"); /* diameter of the particles */  
      values ("float...float"); /* total critical area for given size of  
particles */  
    }  
    ...  
  } } }
```

Liberty TAB Extensions

- Compact CCS Power
- Back-Bias Pins
- Customer PG Types
- Decap/Filler/Tap cell
- Critical Area Analysis (CAA) Table
- **PLL Model**
- Variation Aware Leakage

PLL Model



Add attributes to identify PLL cell and pins

PLL Model: Liberty extension

- Add attributes to identify PLL cells and behavior
- New PLL related Attributes
 - Is_pll_cell in cell group
 - Identifies Cell as a Phased Locked Loop
 - Is_pll_reference_pin in pin group
 - PLL reference input pin
 - Is_pll_feedback_pin in pin group
 - PLL Feedback input pin
 - Is_pll_output_pin in pin group
 - PLL Output pin

Liberty TAB Extensions

- Compact CCS Power
- Back-Bias Pins
- Customer PG Types
- Decap/Filler/Tap cell
- Critical Area Analysis (CAA) Table
- PLL Model
- **Variation Aware Leakage**

Variation Aware Leakage

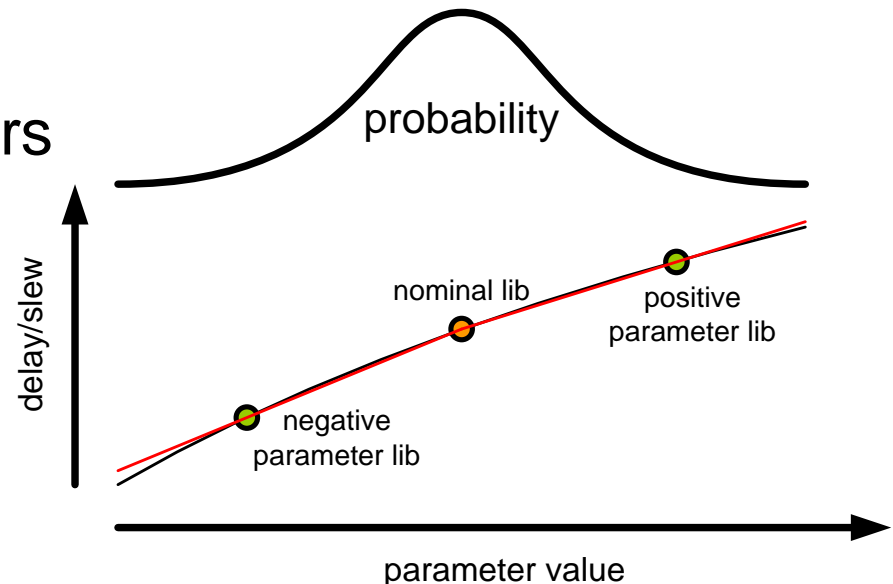
- Process Corner Too Pessimistic
 - 99th leakage percentile is less than corner
- Yield Trade-offs Possible
- Example Small Testcase (317 cells)

What	Source	Value
FF Corner Leakage	HSPICE	255.8uW
99.866% Percentile	HSPICE Monte-Carlo	215.4uW
97.72% Percentile	HSPICE Monte-Carlo	134.6uW
87.14% Percentile	HSPICE Monte-Carlo	92.8uW

CORNER VALUE IS > 20% HIGHER THAN 99.9% LEAKAGE PERCENTILE

Variation Aware CCS Leakage Power

- Variation aware analysis
 - Model impact of process variation parameters
 - Similar approach to existing VA syntax
 - Capture CCS power leakage at $2N+1$ samples
 - N variation parameters



Sample Syntax

```
library (<library_name>) {  
  va_parameters(<string> , ... );  
  cell (<design_name>) {  
    leakage_current() { ... } /* nominal CCS power leakage current */  
    ...  
    cell_based_variation() {  
      va_parameters(<string> , ... );  
      nominal_va_values(<float>,...);  
      va_leakage_current () {  
        va_values(<float> , ...);  
        when : <boolean expression>;  
        pg_current(<pg_pin_name>) {  
          value : <float>;  
        }  
      }  
      ...  
      gate_leakage(<an input pin name>) {  
        input_low_value : <float>;  
        input_high_value : <float>;  
      }  
    }  
  }  
}
```

For More Liberty Information

- <http://opensource.liberty.org/>



LIBERTY

The semiconductor industry's most widely used library modeling standard.

News

- Press Releases
- Articles

Resources

- Format Specs
- Technical Papers
- CCS Accuracy Papers

Contact Us

Welcome to Open Source Liberty!

Open Source Liberty is a comprehensive online resource for the Liberty library modeling standard. In addition to providing up to date Liberty format specifications for download, this site hosts the Liberty Discussion Forum, where members of the semiconductor community can interact with each other and discuss topics relating to Liberty and the Composite Current Source (CCS) modeling technology. This site also provides the latest news and information regarding Liberty and the Liberty Technical Advisory Board (LTAB).

Featured Article



Synopsys boosts CCS with library tools

An updated version of the Library Compiler, used by virtually every Synopsys user to generate binaries for Synopsys tools, adds new support for that company's Composite Current Source (CCS) models.