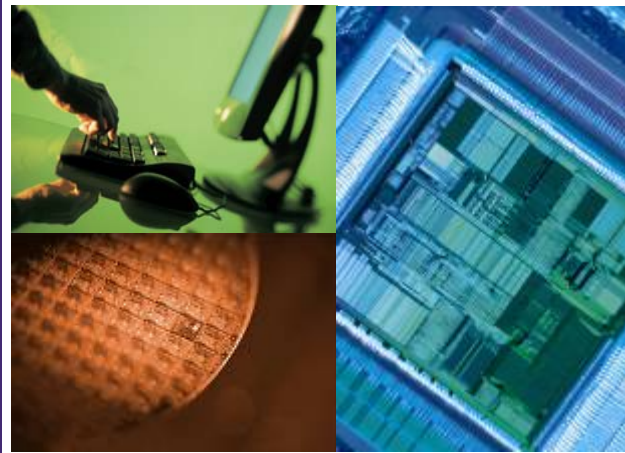


Liberty 2007 Update

Richard Trihy



Agenda

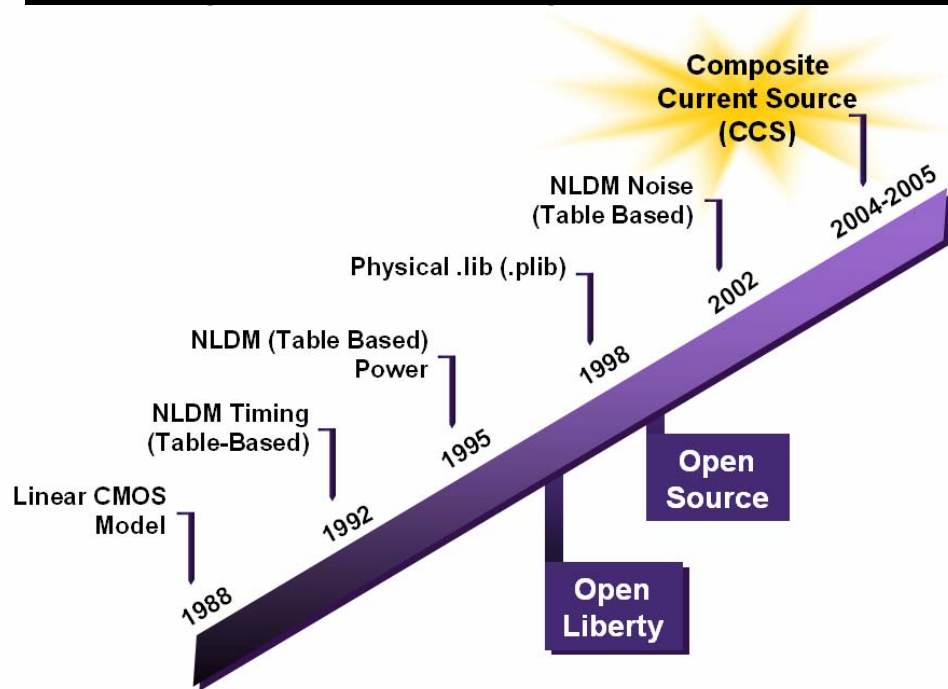
- **Recent LTAB Extensions**
- Low Power Modeling Support in Liberty
- Library Delivery System

Liberty™ Technical Advisory Board

Building on Composite Current Source (CCS)
Technology to Address Industry Needs



Liberty: A History of Innovation



TAB Member Companies

- AMD
- Apache
- ARM
- Infineon
- Intel
- Sequence
- STARC
- ST
- Sun
- Synopsys
- TI
- Virage Logic

SI2 Liberty TAB

- Technical Advisory Board to steer future evolution of Liberty, meets ~4 times per year
- Enhancements and extensions to Liberty Format
 - Reviewed and voted on by Liberty TAB
- Proposals focus on new challenges and technology
- Specifications available on Synopsys and SI2 webpage
 - <http://www.synopsys.com/cgi-bin/tapin/login1.cgi>
- Next meeting: DAC 2007

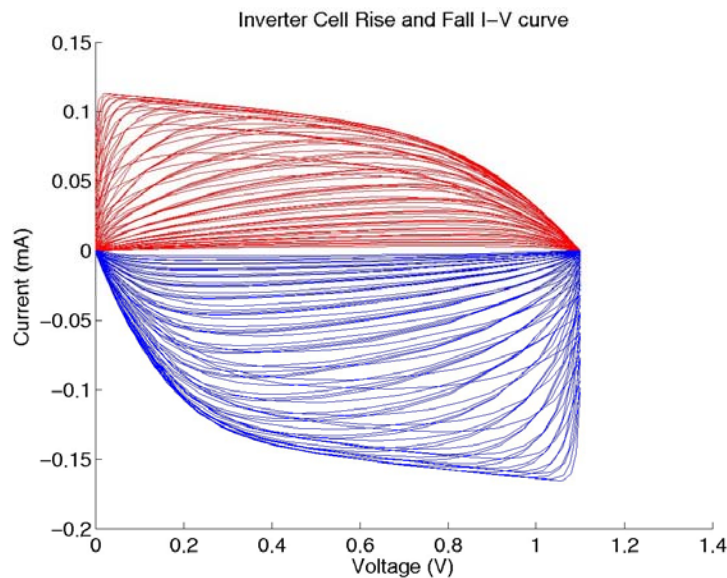
Latest Additions to Liberty

Approved by LTAB in December '06

- Base curve technology
- Variation aware modeling
- Retention cell modeling
- Level shifter modeling
- Switch cell modeling
- Liberty sensitization language

Data Compaction Using Base Curves

- A typical CCS library could have 1M switching $I(t)$ curves
 - All corresponding $I(V)$ curves look very similar
- Employ a shared base curve data base
 - Describes curve shapes efficiently
 - Avoids storing repeated information

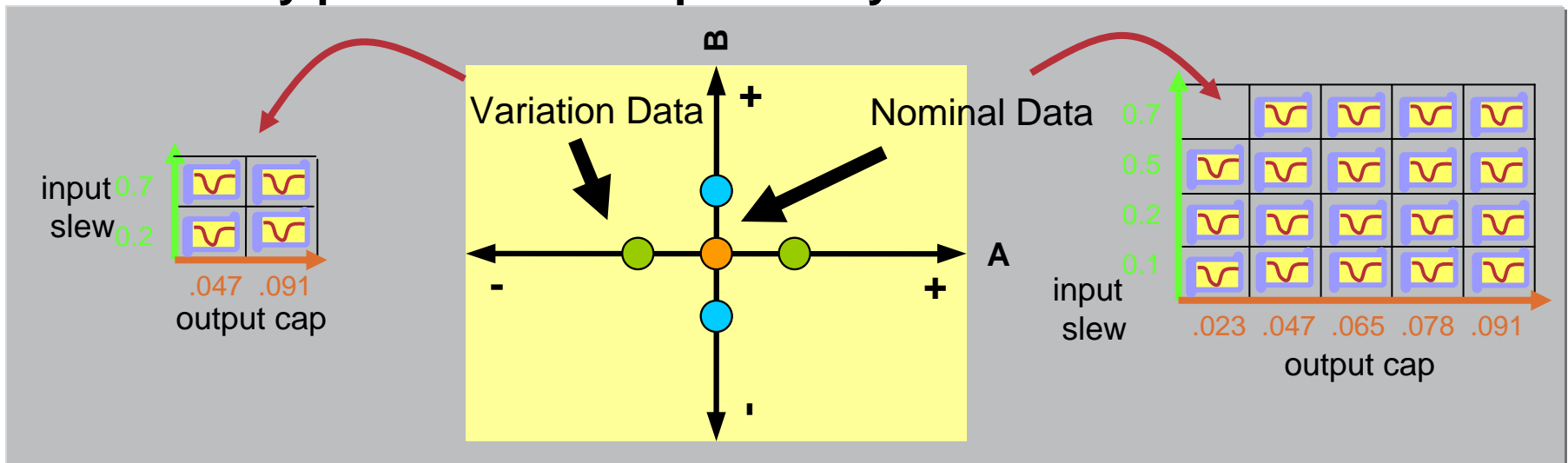


450 Inverter cell rise and cell fall I-V curves for 15 loads and 15 slews

Variation-Aware Modeling in CCS

Concepts

- Generate models for each device variation parameter
 - Vary parameters independently and re-characterize



- **Unified Library representation**

- Leverage Base Curve Technology
- Regular Nominal Data
- Added Process parameter variation data

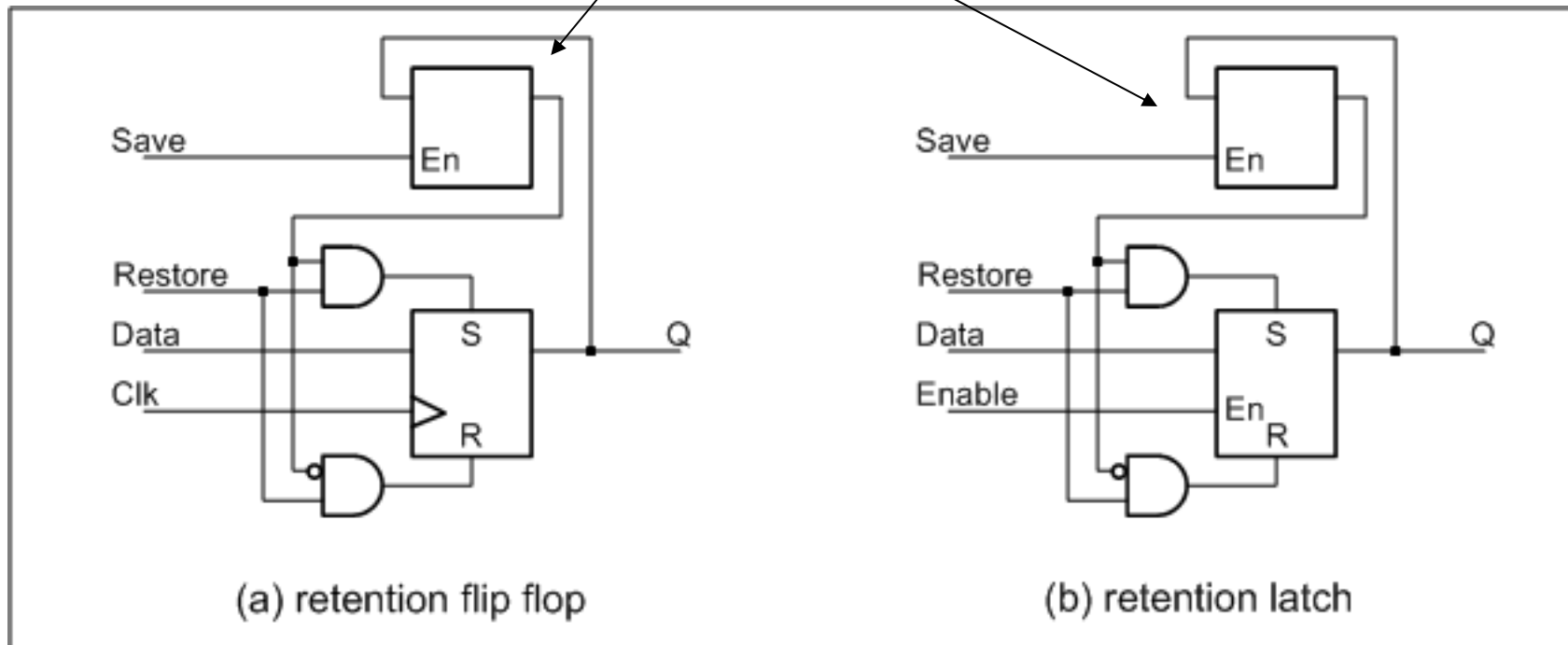
- **Variation Data**

- Reduced Input Slew/Output Load grid size
- Adaptive Model Reduction per cell and per variation while satisfying accuracy goals

- *Single file representation*

Retention Cells

Slow cell with persistent power



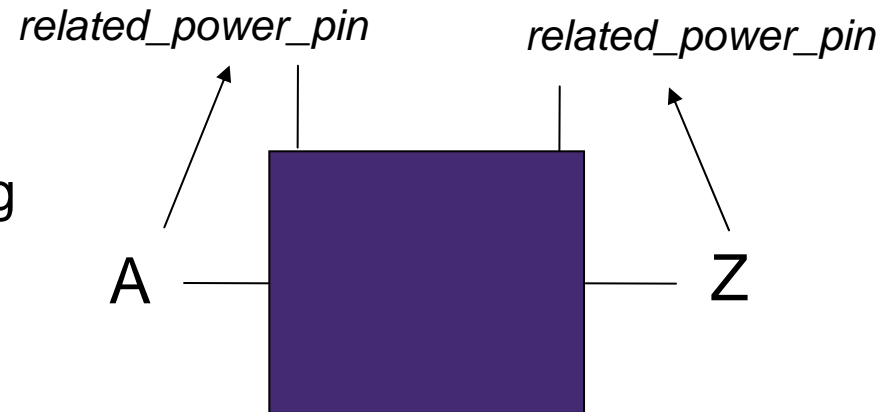
- Sequential cells that hold state when power is shut down.

Liberty Attributes

- New Attributes
 - *Retention_cell*
 - String attribute to identify the retention cell.
 - *Retention_pin* (<pin_class>, <disable_value>)
 - Pin_class
 - *Restore*
 - *Save*
 - *Save_restore*
 - Disable_value
 - Value of retention pin in normal mode

Level Shifter

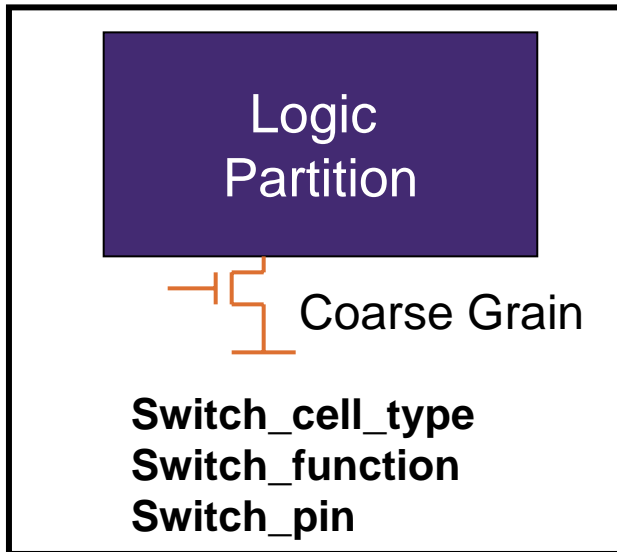
- *input_voltage_level_high/low*:
 - Actual input voltage swing
 - May differ from rail to rail swing
- *related_power_pin*
 - Pin group attribute
 - Identifies rail voltage for the cell pin



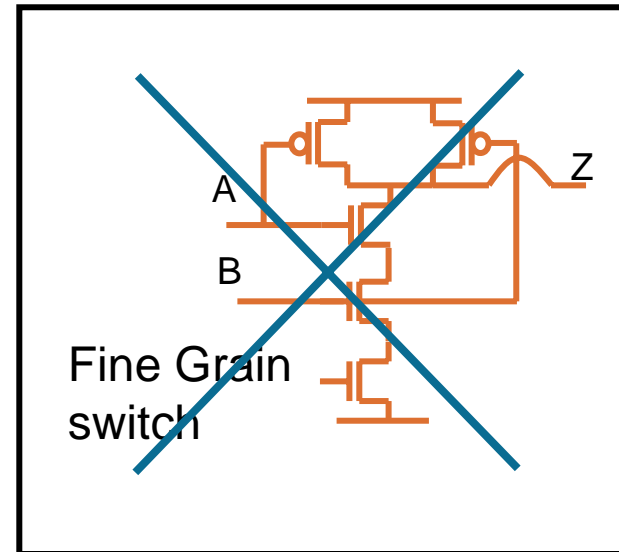
- New Attributes:
 - *level_shifter_type*: *HL | LH | HL_LH*
 - *input_voltage_range*: (<float>, <float>)
 - *output_voltage_range*: (<float>, <float>)
 - *std_cell_main_rail*: *true | false*

Switch Cell Support

- Power down logic to save power
- Part of Liberty 2006.06
- Latest incremental change: Remove fine grain support



Coarse grain (per block)
Remains part of Liberty



Coarse grain (per cell)
Removed from Liberty

Sensitization Language

- Extends Liberty Syntax to support sensitization descriptions
 - Describes the active (and side pin) setup sequence
- Stores pin transitions and initial conditions
 - Enables unambiguous correlation and re-characterization.

Sensitization Example

```
library(my_library) {  
  ...  
  sensitization(sensi_2in_1out) {  
    pin_names (IN1, IN2, OUT);  
    vector (0, "0 1 1");  
    vector (1, "1 0 0");  
    vector (2, "1 1 0");  
    vector (3, "1 1 1");  
  }  
  cell (my_nand2) {  
    sensitization_master : sensi_2in_1out;  
    pin_name_map (A, B, Z);  
    ...  
    pin(A) {  
      ...  
    }  
    Pin(B) {  
      ...  
    }  
  }  
}
```

```
pin(Z) {  
  ...  
  timing() {  
    related_pin : "A";  
    wave_rise (2, 0);  
    wave_fall (0, 2);  
    ...  
  }  
  timing() {  
    related_oin : "B";  
    wave_rise (3, 1);  
    wave_fall (1, 3);  
    ...  
  }  
} /* end pin(Z)*/  
} /* end cell(my_nand2) */  
...  
} /* end library */
```

Sensitization Group Reference

Sensitization Example

```
library(my_library) {  
  ...  
  sensitization(sensi_2in_1out) {  
    pin_names (IN1, IN2, OUT);  
    vector (0, "0 1 1");  
    vector (1, "1 0 0");  
    vector (2, "1 1 0");  
    vector (3, "1 1 1");  
  }  
  cell (my_nand2) {  
    sensitization_master : sensi_2in_1out;  
    pin_name_map (A, B, Z);  
    ...  
    pin(A) {  
      ...  
    }  
    Pin(B) {  
      ...  
    }  
  }  
}
```

```
pin(Z) {  
  ...  
  timing() {  
    related_pin : "A";  
    wave_rise (2, 0);  
    wave_fall (0, 2);  
    ...  
  }  
  timing() {  
    related_oin : "B";  
    wave_rise (3, 1);  
    wave_fall (1, 3);  
    ...  
  }  
} /* end pin(Z)*/  
} /* end cell(my_nand2) */  
...  
} /* end library */
```

Pin Mapping

Sensitization Example

```
library(my_library) {  
  ...  
  sensitization(sensi_2in_1out) {  
    pin_names (IN1, IN2, OUT);  
    vector (0, "0 1 1");  
    vector (1, "1 0 0");  
    vector (2, "1 1 0");  
    vector (3, "1 1 1");  
  }  
  cell (my_nand2) {  
    sensitization_master : sensi_2in_1out;  
    pin_name_map (A, B, Z);  
    ...  
    pin(A) {  
      ...  
    }  
    Pin(B) {  
      ...  
    }  
  }  
}
```

```
pin(Z) {  
  ...  
  timing() {  
    related_pin : "A";  
    wave_rise (2, 0);  
    wave_fall (0, 2);  
    ...  
  }  
  timing() {  
    related_oin : "B";  
    wave_rise (3, 1);  
    wave_fall (1, 3);  
    ...  
  }  
} /* end pin(Z)*/  
} /* end cell(my_nand2) */  
...  
} /* end library */
```

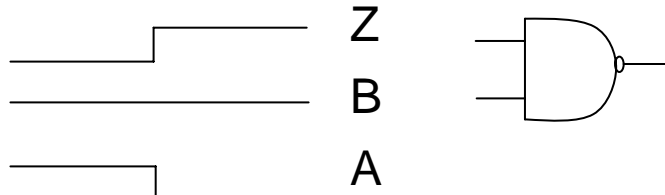
Arc sensitization --- Z rises when A falls

Sensitization Example

```

library(my_library) {
...
sensitization(sensi_2in_1out) {
pin_names (IN1, IN2, OUT);
vector (0, "0 1 1");
vector (1, "1 0 0");
vector (2, "1 1 0");
vector (3, "1 1 1");
}
cell (my_nand2) {
sensitization_master : sensi_2in_1out;
pin_name_map (A, B, Z);
...
pin(A) {
...
}
Pin(B) {
...
}
}

```



```

pin(Z) {
...
timing() {
related_pin : "A";
wave_rise (2, 0);
wave_fall (0, 2);
...
}
timing() {
related_oin : "B";
wave_rise (3, 1);
wave_fall (1, 3);
...
}
} /* end pin(Z)*/
} /* end cell(my_nand2) */
...
} /* end library */

```

Arc sensitization --- Z rises when A falls

Sensitization Attributes

- *Sensitization Master*
 - Defines the arc sensitization group
 - Pin input sequence
- *wave_rise, wave_fall*
 - Defines vector sequence for sensitization
- *Pin_name_map*
 - Name mapping from sensitization master to cell pin names

Sensitization Language Applications

- Stage or Path Correlation/Simulation
 - Accurate correlation impossible for many library cells today.
 - Incomplete setup information in library.
- Library Qualification
 - Cell based correlation limited to combinational cells
- Library (Re-)Characterization
 - Provide unambiguous arc sensitization information

Liberty™ Modeling Roadmap

	2006.06	2007.03	2007.12	Beyond
CCS	<ul style="list-style-type: none"> • CCS noise • CCS power • VA modeling (base curves) 	<ul style="list-style-type: none"> • Interdependent Setup and Hold 		
Low Power	<ul style="list-style-type: none"> • PG pins • Level Shifter & Isolation cell 	<ul style="list-style-type: none"> • Retention cell • Switch cell 		<ul style="list-style-type: none"> • Statistical Leakage Power
Physical		<ul style="list-style-type: none"> • New routing rules for 65nm & below 		
Characterization	<ul style="list-style-type: none"> • Characterization waveform shape 	<ul style="list-style-type: none"> • Acquisition Sensitization language 		

Agenda

- Recent LTAB Extensions
- **Low Power Modeling Support in Liberty**
- Library Delivery System

Liberty – Low Power Library Modeling

- Dynamic and leakage power modeling for library cells
- Multiple power supply modeling
 - Multiple rail modeling
 - PG pin cell library
- CCS Power
- Special cells for Multi-voltage design
 - Level shifters
 - Retention cells, Power gating cells
 - Isolation cells
- MTCMOS support

PG pin library

```

library(sample) {
  voltage_map(VDD1, 3.0);
  voltage_map(VDD2, 3.1);
  voltage_map(GND1, 0.3);
  voltage_map(GND2, 0.0);
  ...
  cell(test) {
    pg_pin(P1) {
      voltage_name : VDD1;
      pg_type : primary_power;
    }
    ...
    pg_pin(G2) {
      voltage_name : GND2;
      pg_type : primary_ground;
    }
    ...
    leakage_power() {
      when : "!A";
      value : 1.5;
      related_pg_pin : P1;
    }
  }
}

```

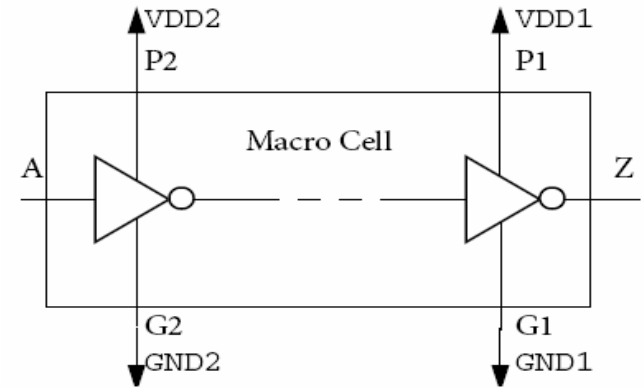
```

pin(A) {
  direction : input;
  related_power_pin : P2;
  related_ground_pin : G2;
  input_signal_swing() {
    low : 2.0;
    high : 2.8;
  }
  ...
}

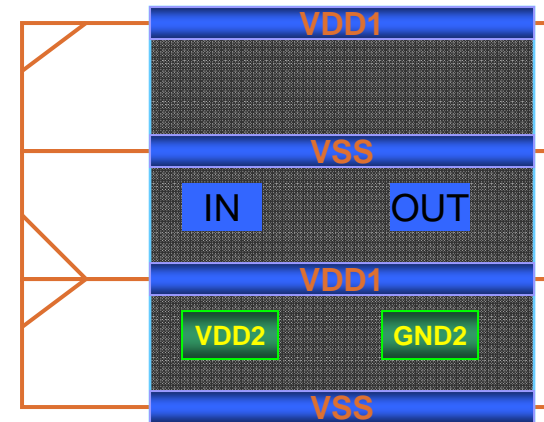
pin(Z) {
  direction : output;
  related_power_pin : P1;
  related_ground_pin : G1;
  output_signal_swing() {
    low : 2.1;
    high : 2.95;
  }
}

timing() {
  ...
}

```

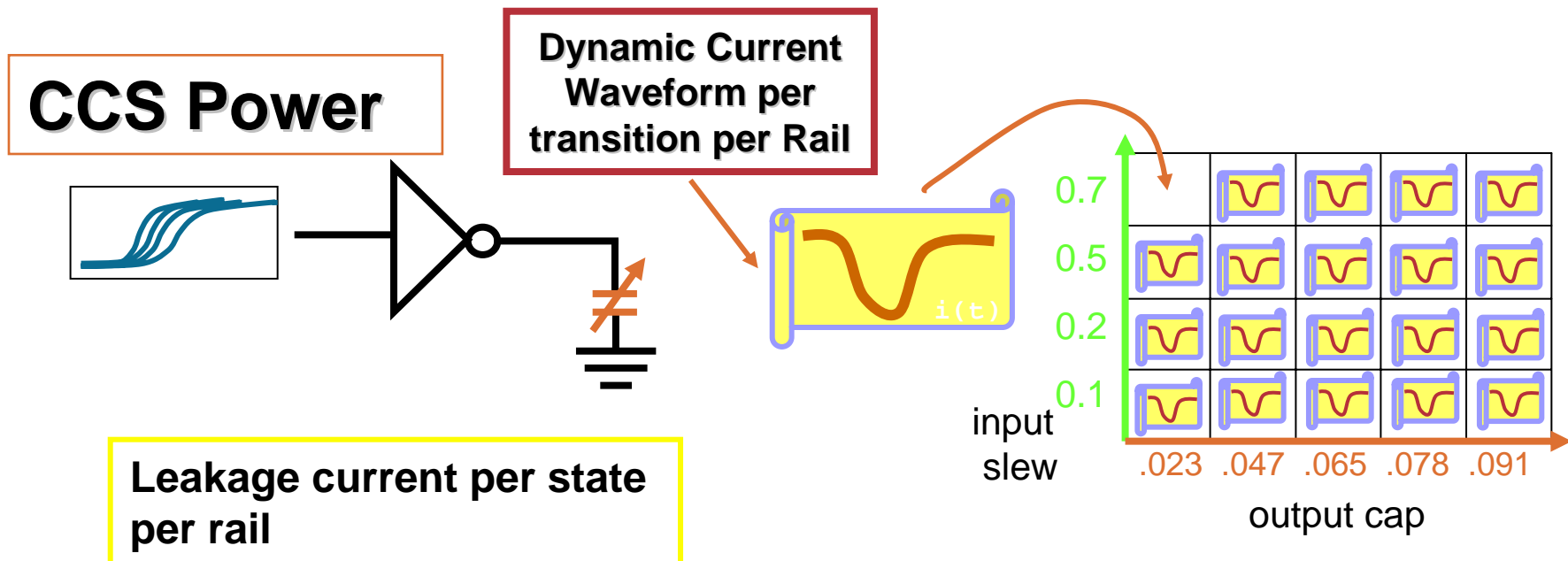


Example Level Shifter Cell



CCS Library for Power

- Unified library model for power optimization, power analysis, rail analysis
- Higher accuracy that is necessary for rail analysis
 - Time-based waveform
- Full multi-voltage support library
 - Allows current lookups and power calculation on all power and ground pins.

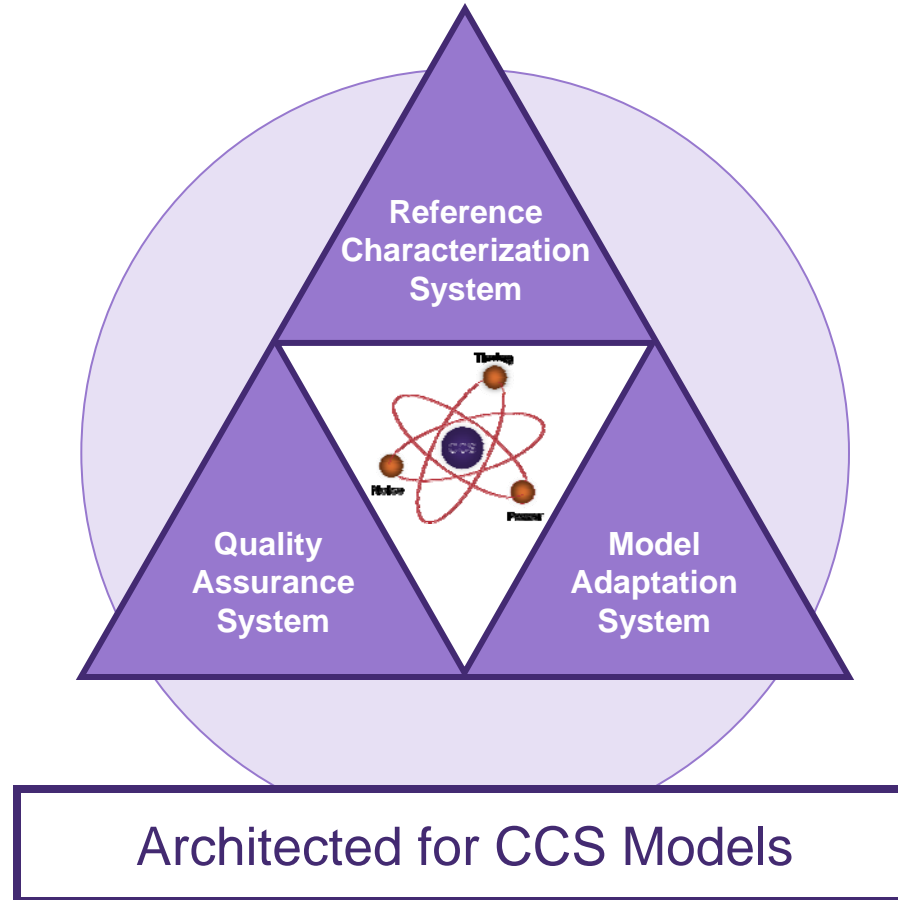


Agenda

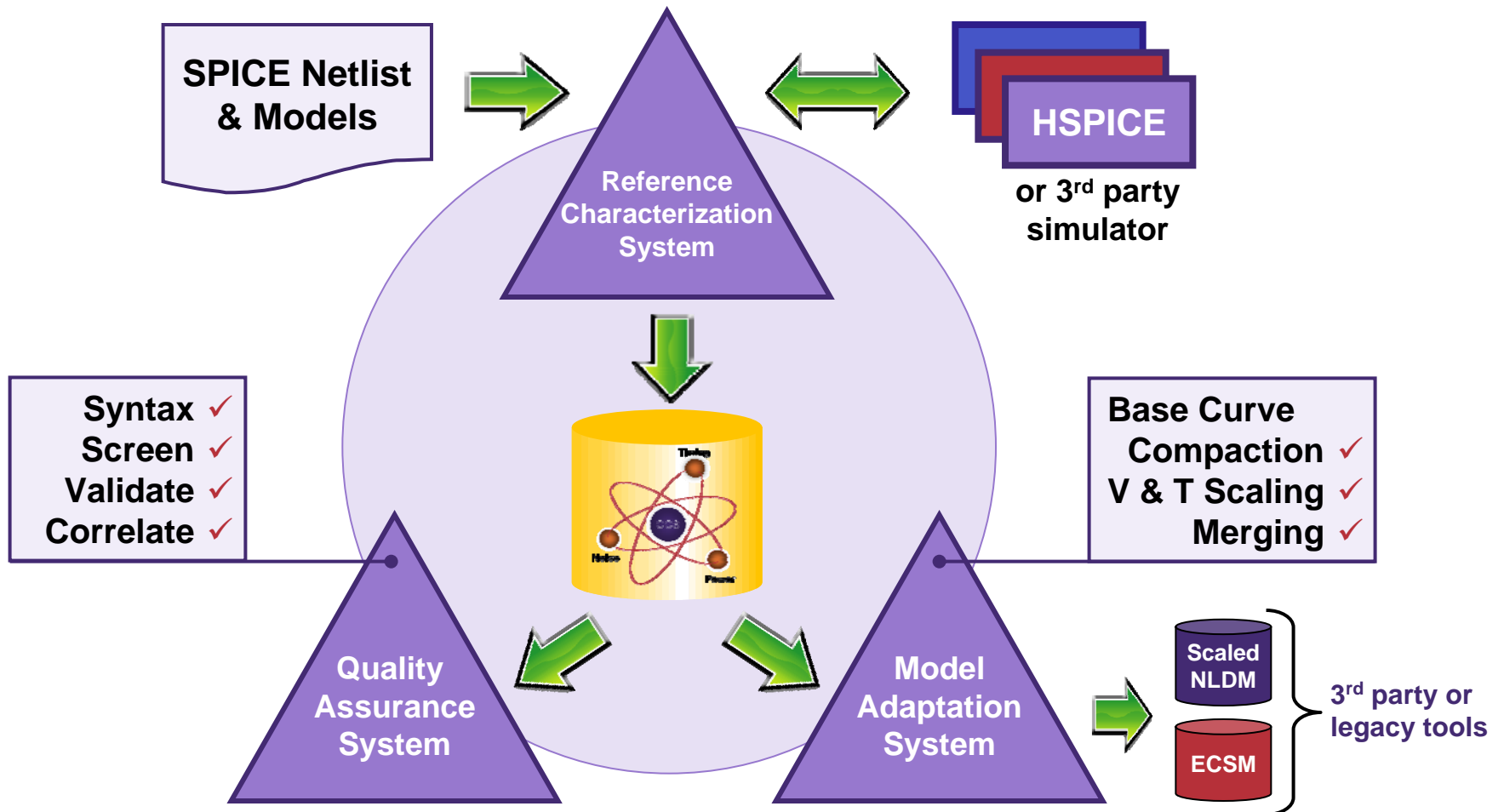
- Recent LTAB Extensions
- Power Modeling support in Liberty
- **Library Delivery System**

Introducing Liberty™ NCX

A Complete Library Delivery System

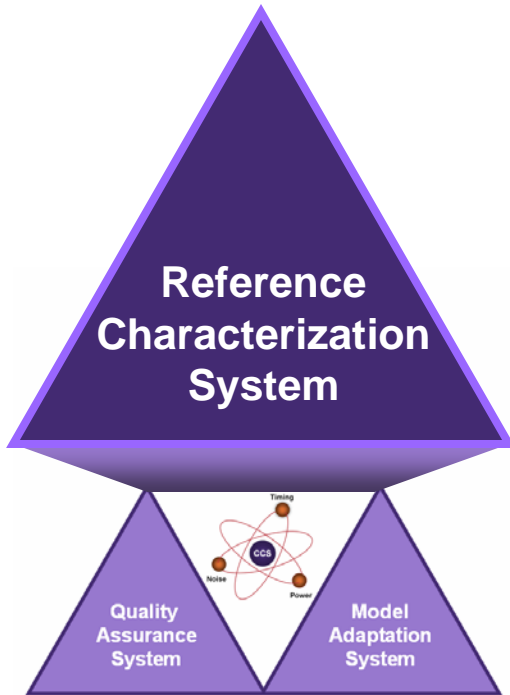


Liberty™ NCX Flow



Introducing Liberty™ NCX

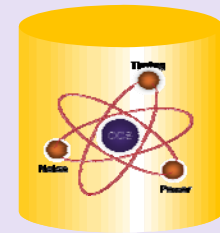
Single Pass CCS Model Generation



Characterization tool for every design team



Simultaneous characterization & verification

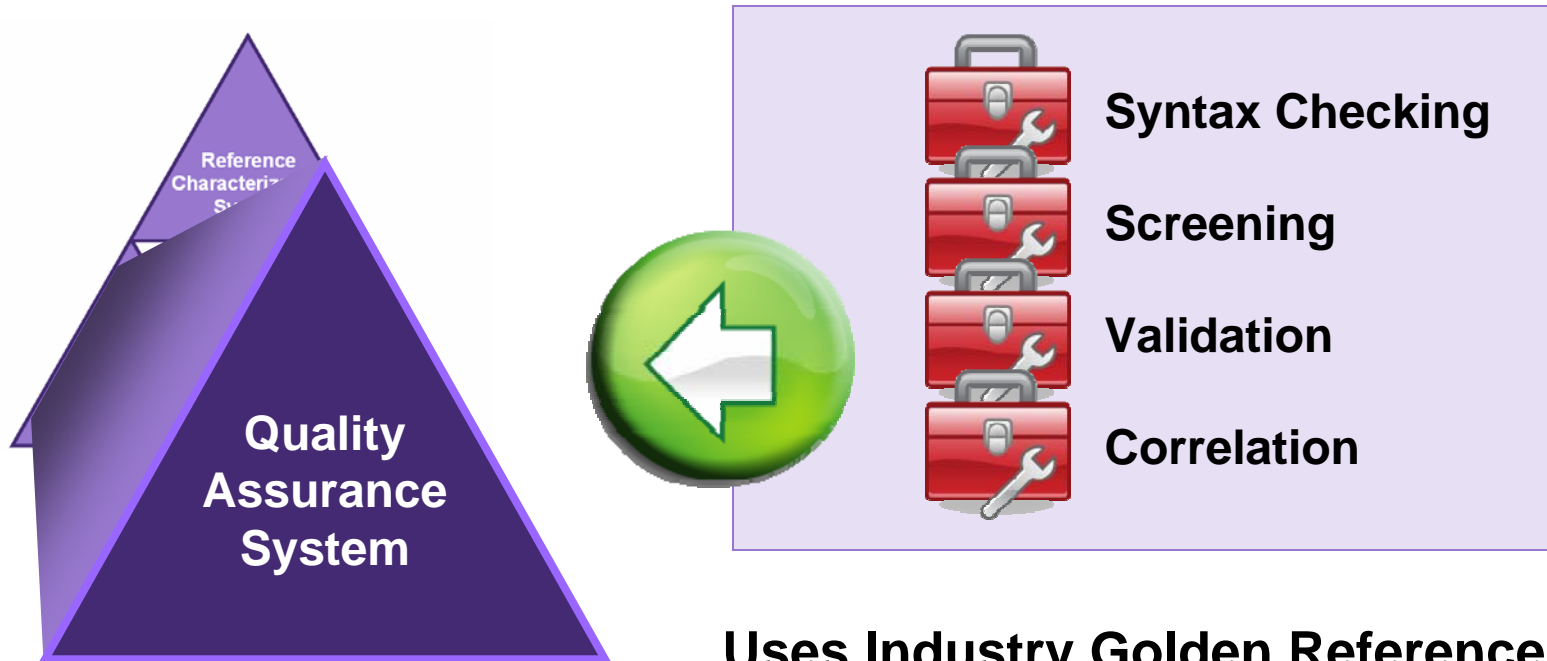


Generates reference CCS libraries

Typical 600 cell, 90nm Library in 33 CPU-Hrs*
800 Cell 65nm Library in 50 CPU-Hrs*

*Runtime for CCS timing + NLDM timing characterization

Introducing Liberty™ NCX



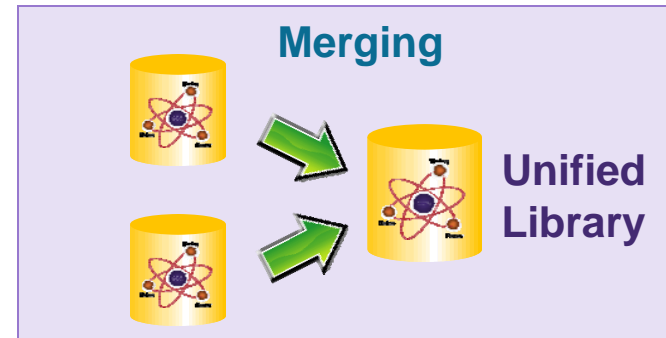
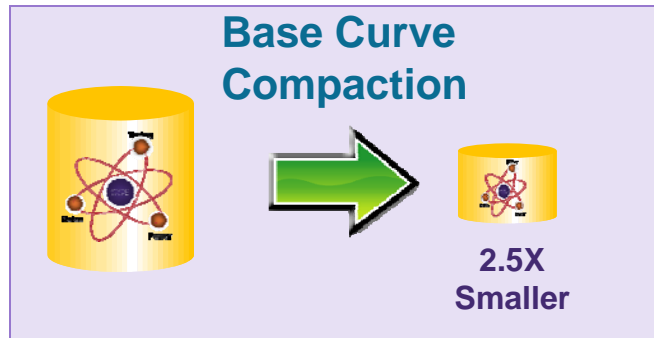
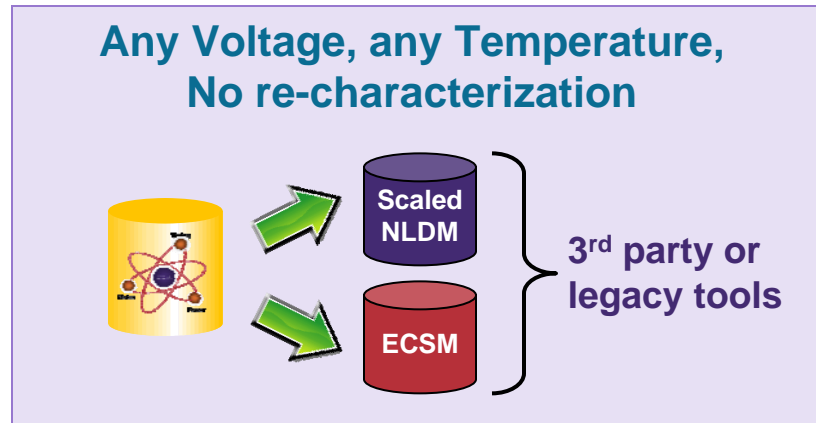
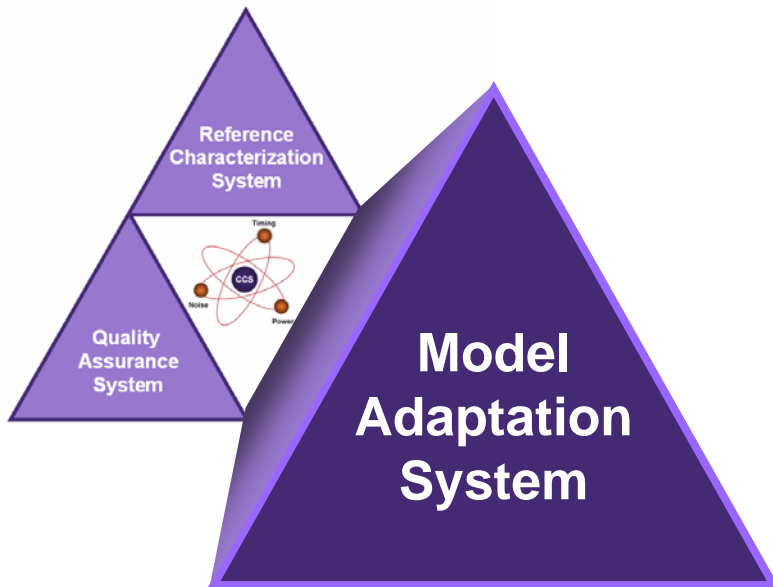
Uses Industry Golden Reference Tools:

- **PrimeTime & HSPICE**

Proven Technology @ ARM, Virage Logic, UMC, Altos

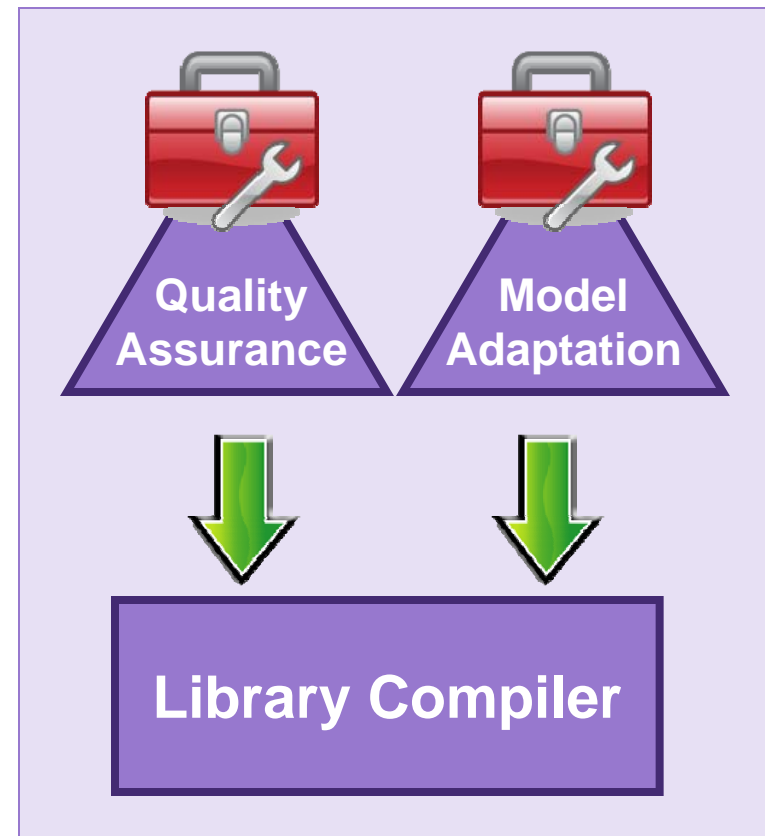
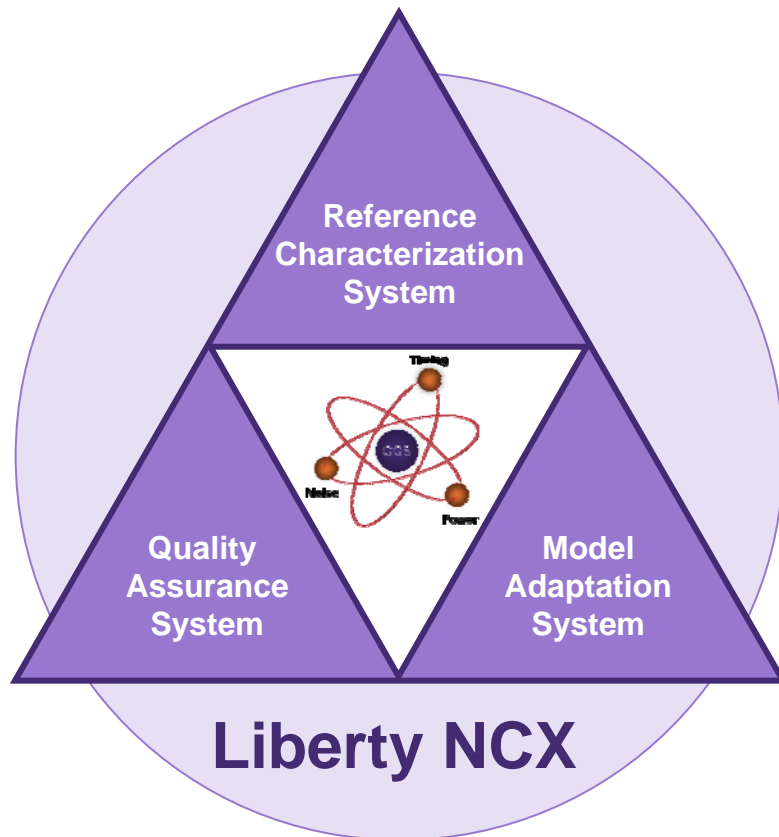
Introducing Liberty™ NCX

Minimum Characterization, Go Anywhere



Enabling Widespread CCS Adoption

All inclusive Package



LC Install Base: 250+ Companies

Summary

- Liberty TAB continues to enhance Liberty Standard to meet emerging needs.
- Liberty has a rich set of constructs for power modeling.
- Liberty NCX : A Complete Library Delivery System.

SYNOPSYS[®]

Predictable Success