

IEC TC93 WG3 meeting, October 24, 2002

# ALF tutorial



Wolfgang Roethig

Chairman, IEEE P1603 (ALF) Workgroup

Senior Engineering Manager, NEC Electronics

# Overview

- Motivation for ALF
- ALF support in the industry
- ALF standardization status
- ALF modeling concepts
- ALF modeling applications
- Conclusion and outlook

# Motivation for ALF

- Complexity of design flows and tools
  - Multiple views for increasing number of tools
- Expensive library preparation
  - Frequent version change of tool-specific libraries
- Advantages of standard library description
  - Reduced cost
  - Increased quality
  - Resource and time saving for library creation and validation
  - Facilitate tool interoperability
  - Leverage 3<sup>rd</sup> party library sources
  - Anticipate technology innovations

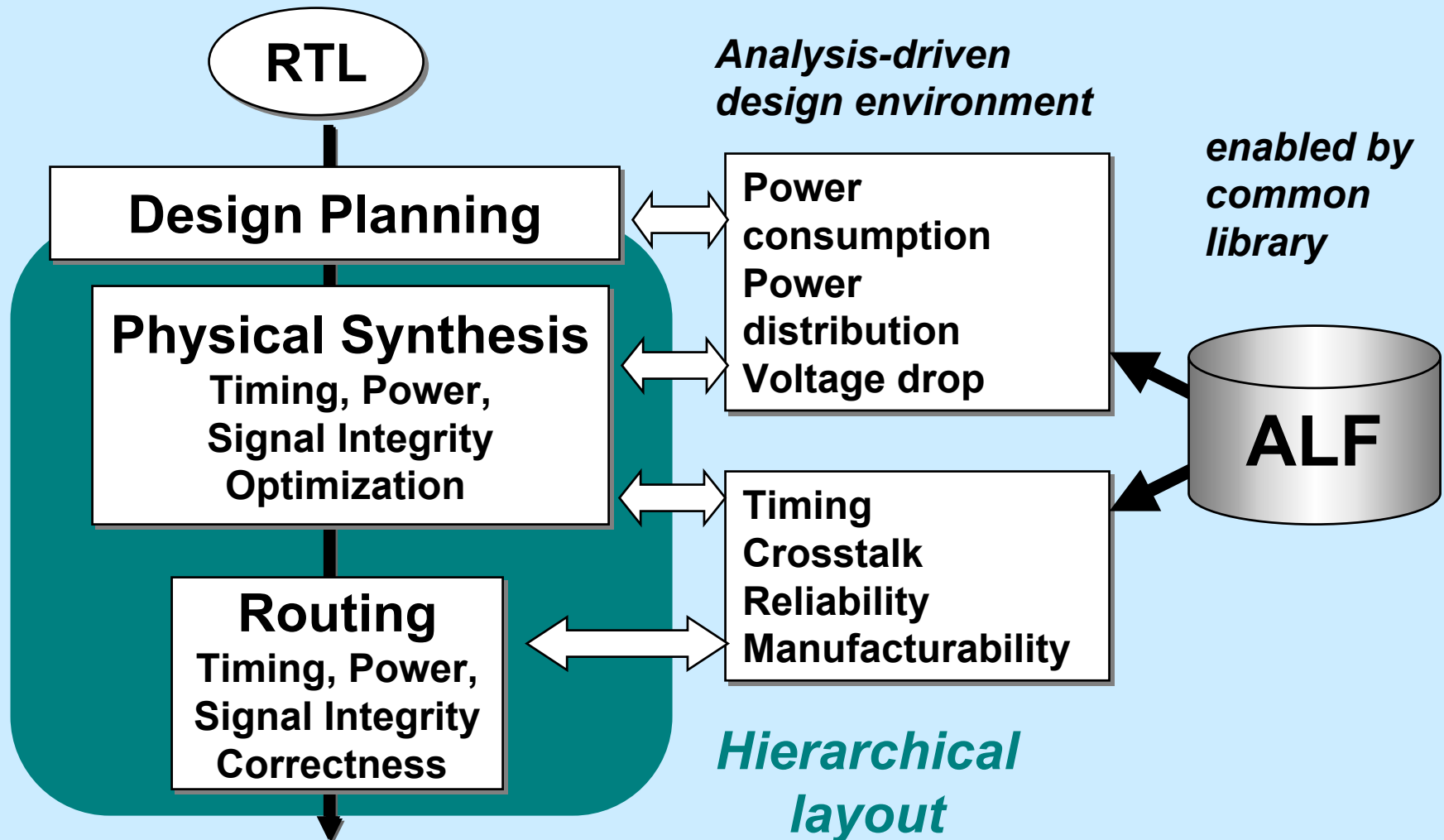
# ALF support for EDA tools today

Tool class	Provider
Behavioral Synthesis RTL prototype Power analysis Simulation, ATPG Physical synthesis Layout Static timing analysis Signal integrity Infrastructure, utilities	ASC Tera Systems Sequence V-cube Magma Avant!, Magma Sequence Sequence, Magma ASC, SynApps

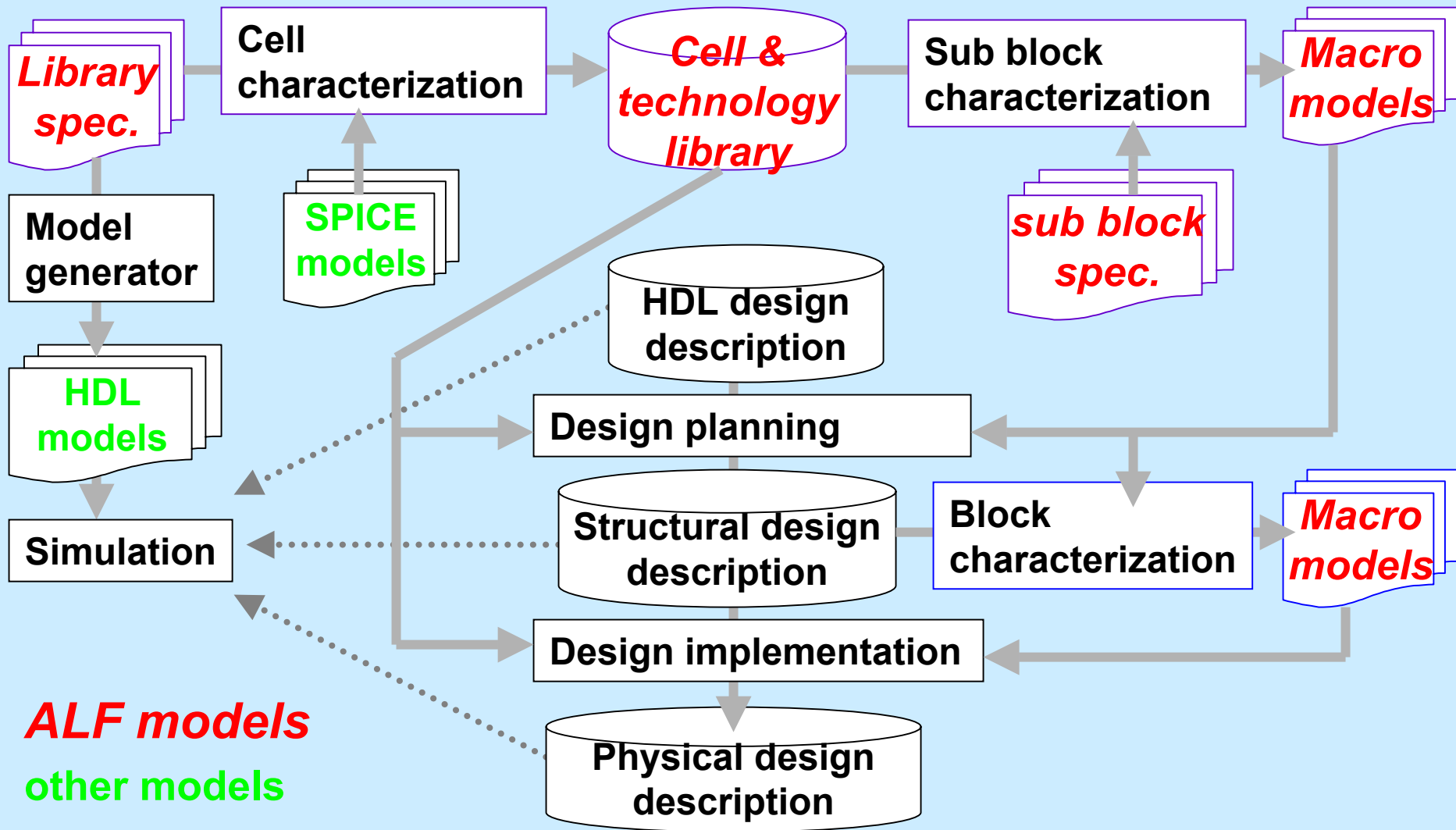
# ALF support for libraries today

Category	Provider
ASIC / ASSP	Infineon, NEC, Philips  Agere, Intel, Motorola
Fabless IP core	ARM, Artisan, NurLogic, Virtual Silicon
Characterization tool	LibTech, Silicon Metrics

# Emerging Design Environment



# Design flow with ALF



# ALF standardization status

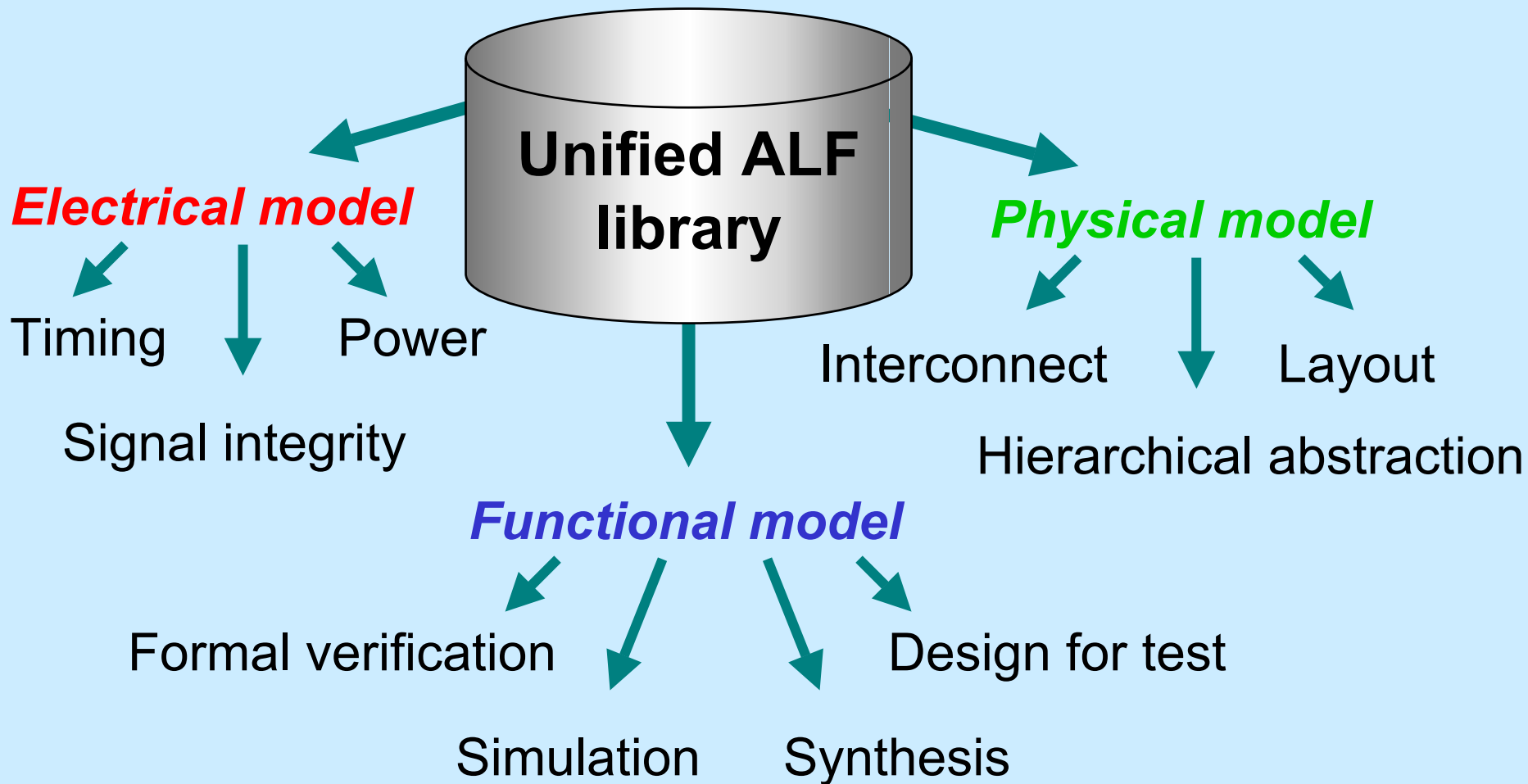
- Started as OVI workgroup in 1996
  - Initial members:  
Avant!, Cadence, LSI Logic, Mentor Graphics, ViewLogic, VLSI
- Version 1.0 approved as OVI standard e/o 1997
  - covers function, timing, power
- OVI successor organization Accellera endorsed ALF
- Version 2.0 approved as Accellera standard e/o 2000
  - added signal integrity, interconnect analysis and layout
- IEEE P1603 workgroup started in 2001
  - Today's members:  
ASC, Infineon, Magma, Mentor Graphics, Monterey, NEC, Philips, Sequence, Simplex, Sun Microsystems, Tera Systems
- IEEE P1603 ballot scheduled for 2H of 2002
  - IEC standardization planned



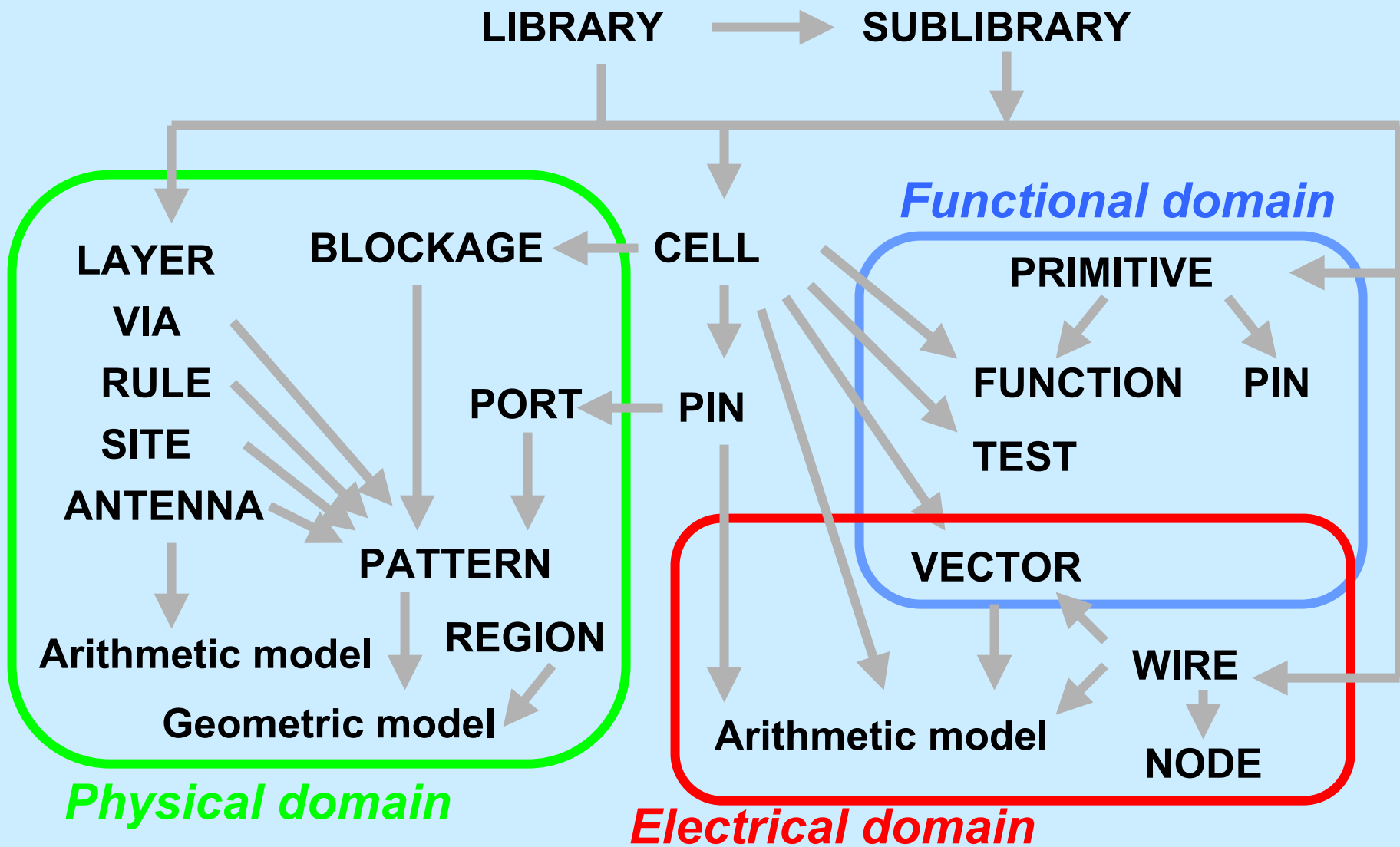
# ALF scope defined in IEEE PAR

- ALF shall serve as the data specification of library elements for design applications used to implement integrated circuits. The range of abstraction shall include from the register-transfer level (RTL) to the physical implementation level.
- The language shall model behavior, timing, power, signal integrity, physical abstraction and physical implementation rules of library elements.

# ALF scope illustrated





# ALF data model



# Example for CELL description

```
CELL myCell {
  PIN in1 { DIRECTION = input; }
  PIN in2 { DIRECTION = input; }
  PIN out1 { DIRECTION = output; }
  FUNCTION {
    BEHAVIOR { out1 = !( in1 & in2 ); }
  }
  VECTOR (01 in1 -> 10 out1) {
    DELAY { FROM { PIN = in1; } TO { PIN = out1; }
      HEADER {
        CAPACITANCE cload { PIN = out1; }
        SLEWRATE trise { PIN = in1; }
      } EQUATION { 0.3 + cload*(0.2 + 0.1*trise) }
    }
  } // put other models, e.g. ENERGY, NOISE etc.
}
```

# ALF modeling concepts

- Modeling foundation concepts  Covered by this tutorial
  - Arithmetic model concept
    - > Electrical and physical library data description
  - VECTOR concept
    - > Stimulus for function, timing, electrical characterization
- Higher-level modeling concepts  Covered by other tutorial [CICC2001]
  - FUNCTION, TEST
    - > Canonical description of functional behavior
    - > Interface between tester algorithm and DUT
  - TEMPLATE, GROUP
    - > Re-usable definitions
    - > Description of parametrizable IP blocks

# Arithmetic model concept

- Purpose of arithmetic model
  - Mathematical calculation of measurable quantities in library
- ALF supports rich set of predefined keywords
  - Timing, analog and physical modeling
- ALF is highly self-descriptive
  - Declaration of legal range or value set
  - Declaration of customized keywords
- Description methods
  - Lookup table
  - Analytical model
  - Calculation graph involving multiple models

# Predefined arithmetic models (1 of 2)

## Standard keywords for arithmetic model

### Timing

DELAY, RETAIN, SLEWRATE, SKEW, JITTER, SETUP, HOLD, RECOVERY, REMOVAL, PULSEWIDTH, PERIOD, ILLEGAL, NOCHANGE, THRESHOLD, NOISE, NOISE\_MARGIN

### Analog

VOLTAGE, CURRENT, TIME, FREQUENCY, CAPACITANCE, RESISTANCE, INDUCTANCE, ENERGY, POWER, FLUX, FLUENCE, TEMPERATURE

# Predefined arithmetic models (2 of 2)

Standard keywords for arithmetic model (cont.)	
Physical	LENGTH, WIDTH, HEIGHT, THICKNESS, AREA, PERIMETER, SIZE, EXTENSION, DISTANCE, OVERLAP
Misc.	PROCESS, DERATE_CASE, DRIVE_STRENGTH, SWITCHING_BITS, CONNECTIVITY



# Global arithmetic model definitions

*Declaration of legal value range*

CAPACITANCE { MIN = 0; }

TEMPERATURE { MIN = -273; }

VOLTAGE { MIN = -1000; MAX = 1000; }

*Declaration of discrete legal value set*

PROCESS { TABLE { best nominal worst } }

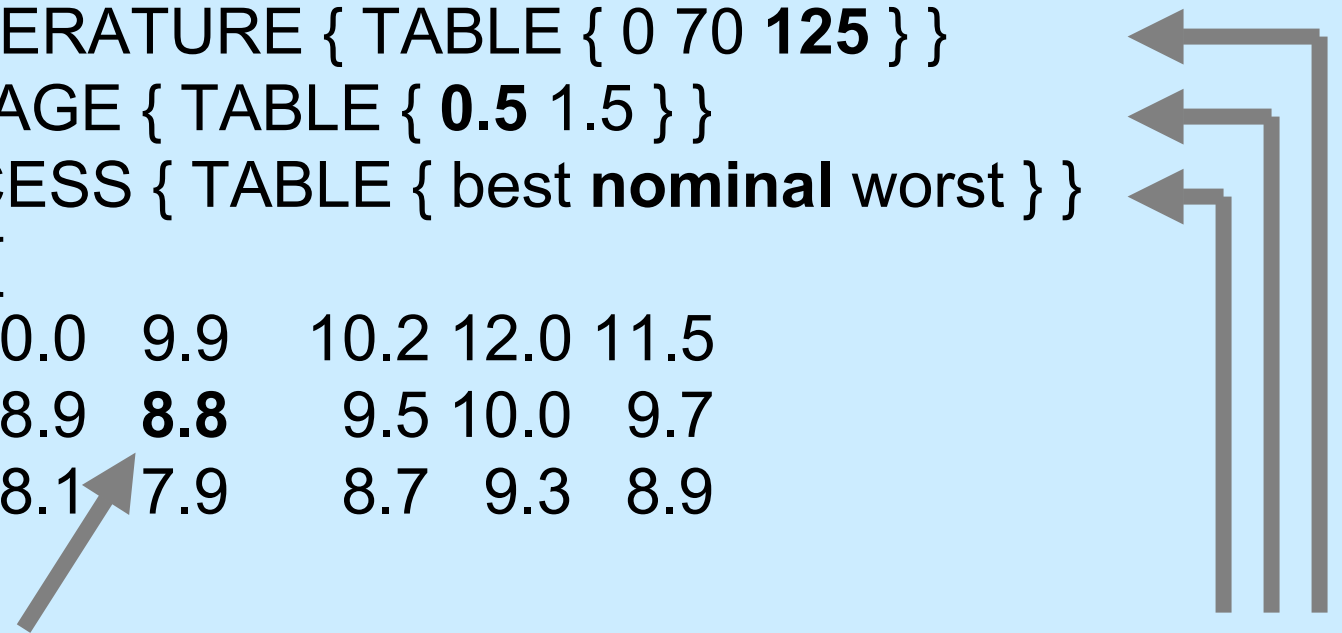
*Declaration of new keyword for arithmetic model*

KEYWORD NEW\_MODEL = arithmetic\_model {  
    VALUETYPE = number ; }

# Arithmetic model with TABLE

*Example for 3-D lookup table*

```
CAPACITANCE {  
  HEADER {  
    TEMPERATURE { TABLE { 0 70 125 } }  
    VOLTAGE { TABLE { 0.5 1.5 } }  
    PROCESS { TABLE { best nominal worst } }  
  } TABLE {  
    9.8 10.0 9.9 10.2 12.0 11.5  
    8.5 8.9 8.8 9.5 10.0 9.7  
    7.8 8.1 7.9 8.7 9.3 8.9  
  } }  
}
```



CAPACITANCE = **8.8**  
*applies for*

TEMPERATURE = **125**  
VOLTAGE = **0.5**  
PROCESS = **nominal**

*3<sup>rd</sup> point in 1<sup>st</sup> dimension  
1<sup>st</sup> point in 2<sup>nd</sup> dimension  
2<sup>nd</sup> point in 3<sup>rd</sup> dimension*

# Arithmetic model with EQUATION

*Example for 3-D analytical model*

```
CAPACITANCE {  
  HEADER {  
    TEMPERATURE Ta { /* no table */ }  
    VOLTAGE Vc { /* no table */ }  
    PROCESS { /* no table */ }  
  } EQUATION {  
    (PROCESS==best)? ( 10.0 + 0.01*(Vc + 0.2*Ta) ) :  
    (PROCESS==nominal)? ( 9.8 + 0.02*(Vc + 0.1*Ta) ) :  
    (PROCESS==worst)? ( 9.5 + 0.025*(Vc + 0.15*Ta) ) :  
    -1  
  }  
}
```

# Arithmetic model with reference

*Example for calculation graph*

```
TEMPERATURE temp1 {  
  HEADER { NEW_MODEL { TABLE { ... } } }  
  TABLE { ... }  
CAPACITANCE {  
  HEADER {  
    TEMPERATURE { MODEL=temp1; TABLE { ... } }  
    VOLTAGE { TABLE { ... } }  
    PROCESS { TABLE { ... } }  
  } TABLE { ... } }
```

Primary input data

Primary input data

Primary input data

*Data for NEW\_MODEL*

*Data for VOLTAGE*

*Data for PROCESS*









*calculate TEMPERATURE*

*calculate CAPACITANCE*



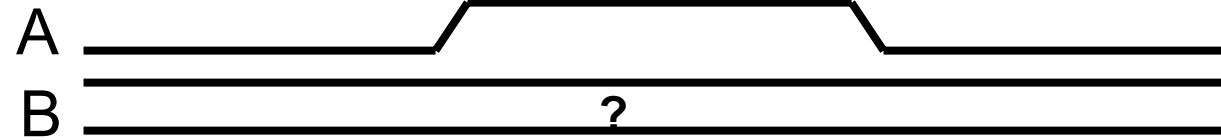

# VECTOR concept

- Purpose of Vector
  - Describe stimulus for electrical characterization
  - Describe functional waveform
  - Describe trigger for sequential behavior
- Description methods
  - Boolean expression for static state
  - Vector expression for temporal behavior

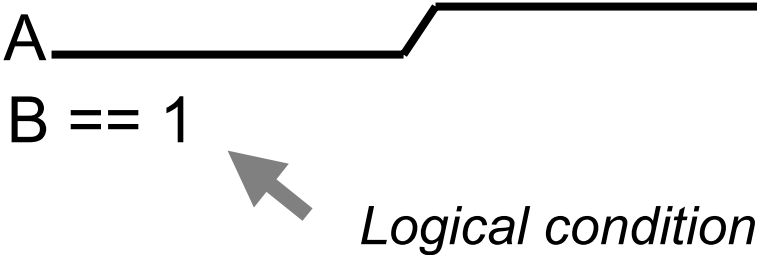
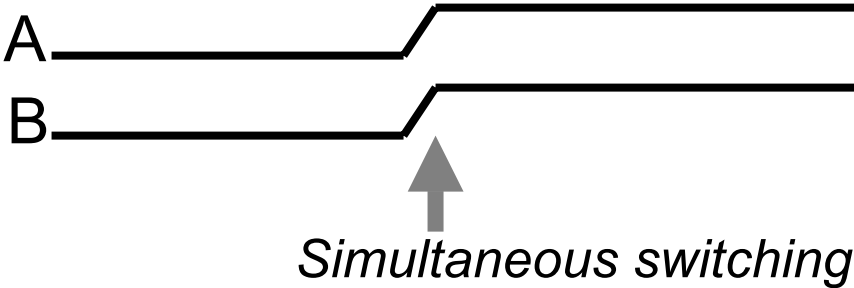
# Single-Event Vector Expressions

<i>Timing diagram for a signal A</i>	<i>Vector expression</i>
	(01 A)
	(0? A)
	(?1 A)
	(?! A)
	(0* A)
	(*1 A)
	(?* A)
	(*? A)

# Dual-Event Vector Expressions

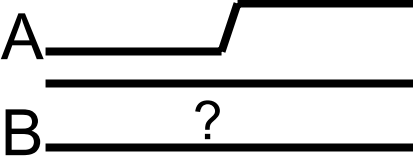
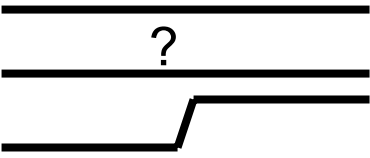
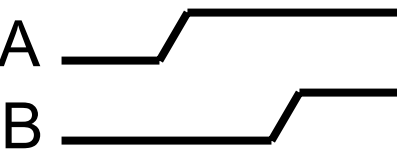
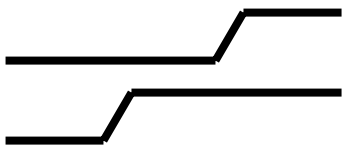
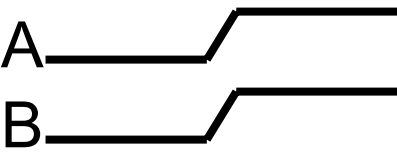
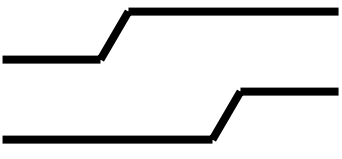
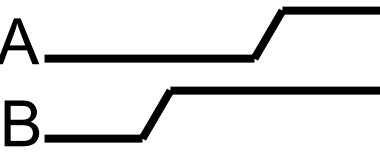
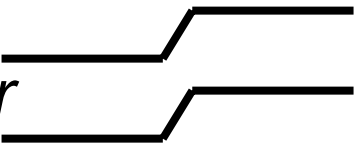
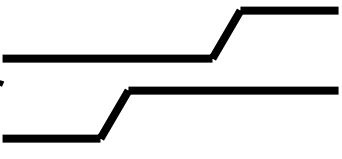
Timing diagram for two signals A and B	Vector expression
	(01 A -> 01 B)
	(01 A ~> 01 B)
	(01 A -> 10 A)
	(01 A ~> 10 A)

# Conditional Vector Expressions

<i>Timing diagram for two signals A and B</i>	<i>Vector expression</i>
 <p>A</p> <p>B == 1</p> <p>Logical condition</p>	<p>(01 A &amp; B)</p>
 <p>A</p> <p>B</p> <p>Simultaneous switching</p>	<p>(01 A &amp; 01 B)</p>



# Alternative Vector Expressions

Timing diagram for two signals A and B	Vector expression
 <p>or</p> 	(01 A   01 B)
 <p>or</p> 	(01 A <=> 01 B)
 <p>or</p> 	(01 A &> 01 B)
 <p>or</p>  <p>or</p> 	(01 A <&> 01 B)

# ALF Modeling applications

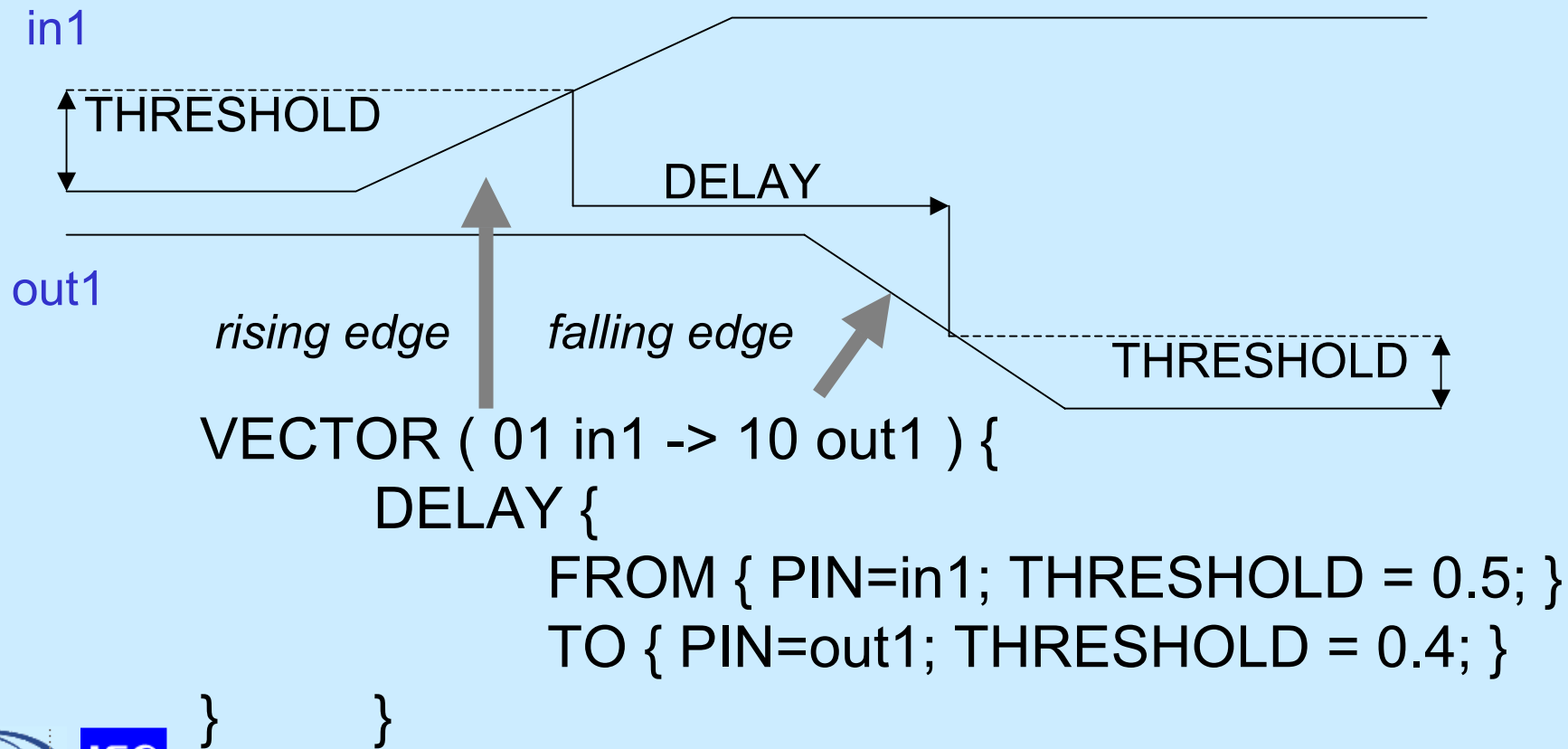
- Timing modeling
  - Cell delay, timing waveforms
  - Interconnect analysis, parasitics
- Power modeling
  - Power consumption
  - Voltage drop
- Signal integrity
  - Noise
- Reliability
  - Electromigration
- Manufacturability
  - Antenna

# Timing modeling

- ALF supports DELAY and SLEWRATE with THRESHOLD definition per timing arc
  - Optimal THRESHOLD can be chosen for characterization
  - Library data matches SPICE characterization data
- ALF supports driver RESISTANCE
  - Accurate waveform at driver output
  - Accurate calculation of effective capacitance
  - Better accuracy for cell and interconnect delay
- ALF supports standard timing checks
  - SETUP, HOLD, RECOVERY, REMOVAL, SKEW
  - MIN, MAX LIMIT for PULSEWIDTH, PERIOD

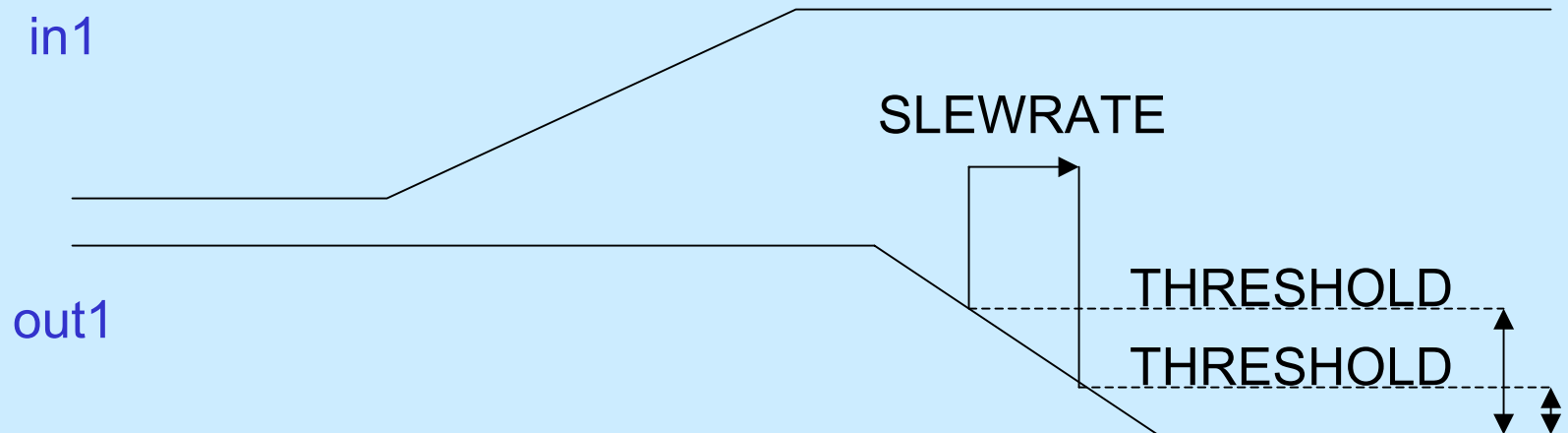
# DELAY

- Timing arc specification in VECTOR
- PIN and THRESHOLD definition in FROM, TO
- THRESHOLD per library, per pin, or per arc



# SLEWRATE

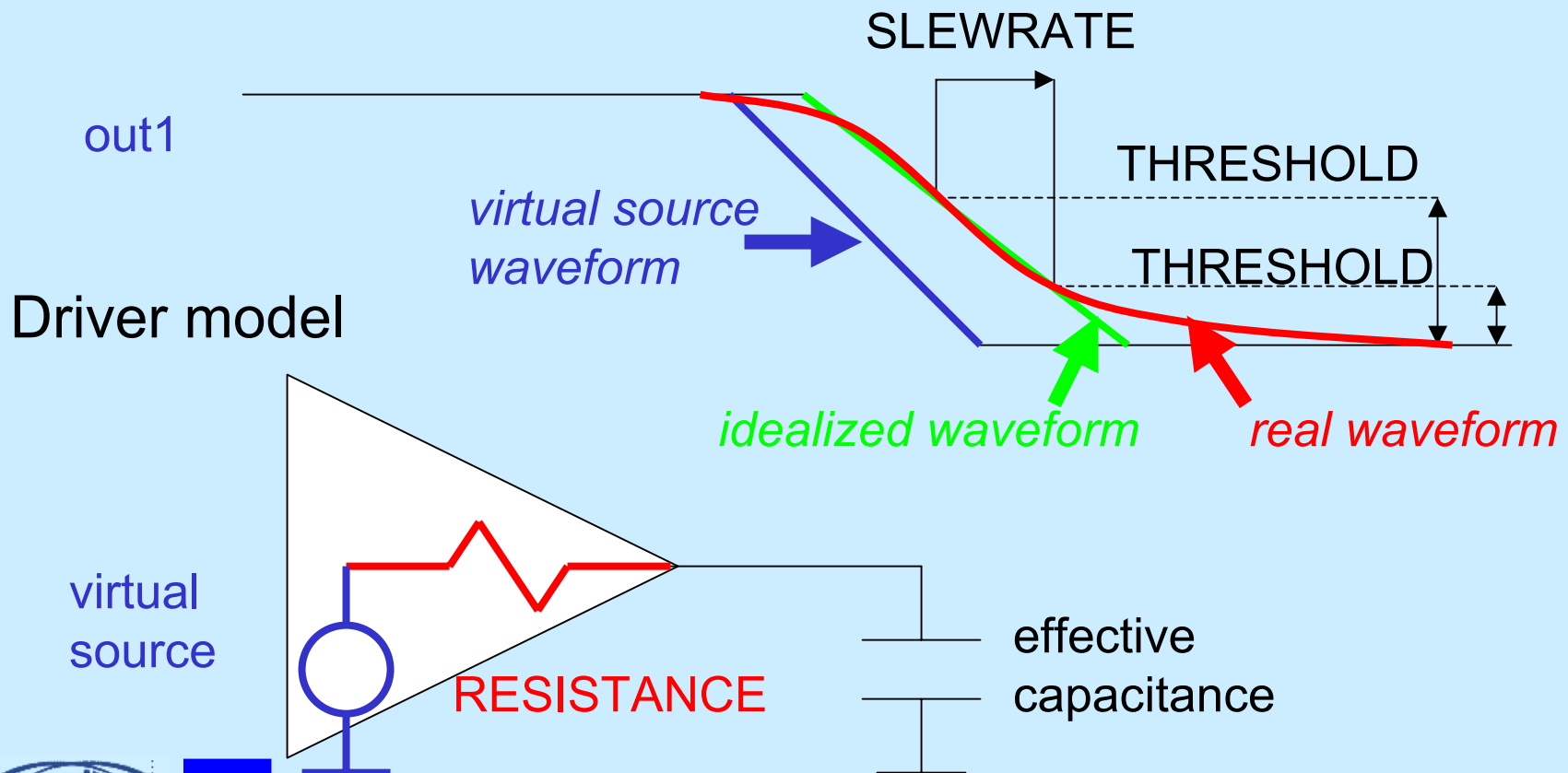
- Timing arc specification in VECTOR
- THRESHOLD definition in FROM, TO
- THRESHOLD per library or per arc



```
VECTOR ( 01 in1 -> 10 out1 ) {  
    SLEWRATE { PIN = out1;  
        FROM { THRESHOLD = 0.6; }  
        TO { THRESHOLD = 0.3; }  
    }  
}
```

# Driver RESISTANCE (1 of 2)

- Linear SLEWRATE not accurate
- Driver RESISTANCE for realistic waveform
- Driver model for calculation of effective capacitance



# Driver RESISTANCE (2 of 2)

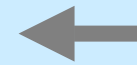
- Driver RESISTANCE can be associated with one specific timing arc or multiple timing arcs

```
VECTOR ( 01 in1 -> 10 out1 ) {  
    DELAY { ... }  
    SLEWRATE { ... }  
    RESISTANCE { PIN = out1; }  
}
```



*RESISTANCE applies for this arc involving in1 and out1*

```
VECTOR ( 10 out1 ) {  
    RESISTANCE { PIN = out1; }  
}
```

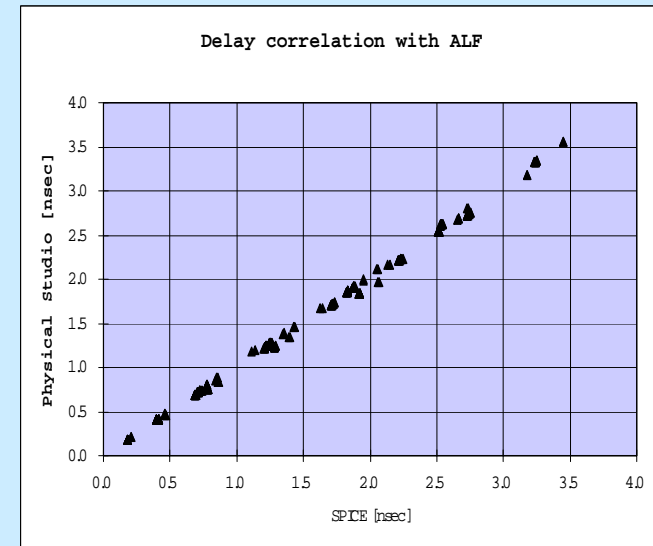
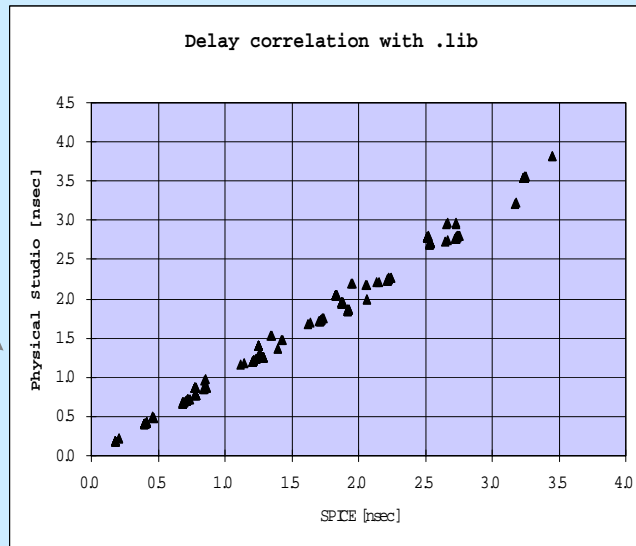


*RESISTANCE applies for all arcs involving out1*

# Timing accuracy

- ALF enables more accurate delay calculation
- Better correlation with SPICE

*Conventional  
timing library*



*ALF*



Error criterion	.lib	ALF
Average	+ 3.9 %	+ 0.5 %
Std deviation	+/- 5.0 %	+/- 2.2 %
Max - Min	17.4 %	11.1 %

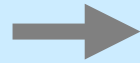


# Interconnect modeling

- ALF support distributed load
  - Characterize cell delay with R,C load
  - More accurate than lumped capacitance
- ALF supports boundary parasitics
  - Describe boundary parasitics as R, C
  - Can include coupling capacitance between pins
  - More accurate than lumped pin capacitance
  - Also in conjunction with “donut” model for complex block
- ALF supports interconnect analysis
  - Interconnect delay calculation
  - Interconnect noise calculation

# Distributed load

*Wire  
declaration*



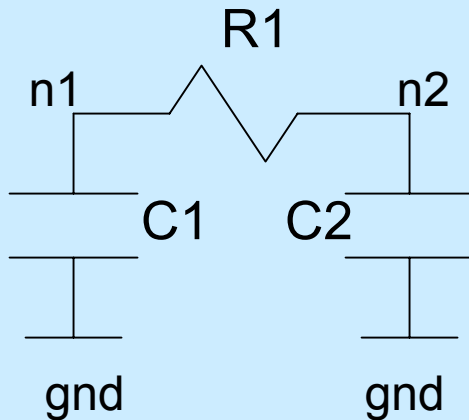
**WIRE pi\_load {**

```

    NODE n1 { NODETYPE=interconnect; }
    NODE gnd { NODETYPE=ground; }
    RESISTANCE R1 { NODE { n1 n2 } }
    CAPACITANCE C1 { NODE { n1 gnd } }
    CAPACITANCE C2 { NODE { n2 gnd } }

```

**}**



*Wire  
instantiation*



**DELAY { FROM { PIN=pin1; } TO { PIN=pin0; } }**

**pi\_load w1 { n1 = pin0; }**

**HEADER {**

**CAPACITANCE c\_near { MODEL = w1.C1; }**

**CAPACITANCE c\_far { MODEL = w1.C2; }**

**RESISTANCE r\_wire { MODEL = w1.R1; }**

**} EQUATION { ... } }**

# Boundary parasitics

*Cell  
declaration*



```
CELL myCell { ... }
```

```
CELL myBlock {
```

```
  PIN myPin {
```

```
    PORT p1 { CONNECT_TYPE=external; }
```

```
    PORT p2 { CONNECT_TYPE=internal; }
```

```
  }
```

```
  FUNCTION { STRUCTURE {
```

```
    myCell u1 { pin1 = myPin.p2; }
```

```
  }
```

```
  }
```



*CELL  
instantiation*

```
WIRE boundary {
```

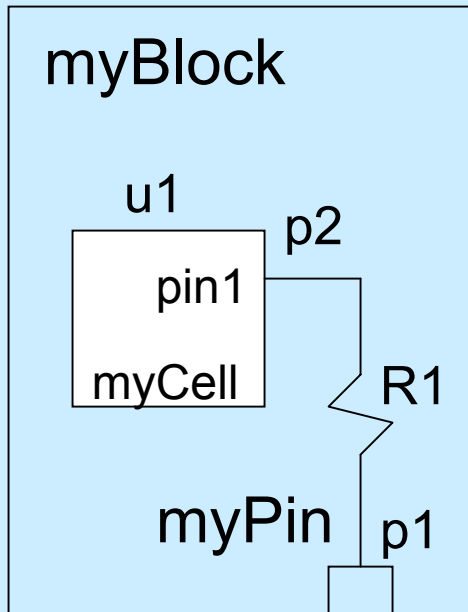
```
  RESISTANCE R1 {
```

```
    node { myPin.p1 myPin.p2 } }
```

```
}
```

```
}
```

*Parasitics description*



# Interconnect delay calculation

WIRE lumpedRLC {

NODE n0 { NODETYPE = **source**; }

NODE n1 { NODETYPE = **driver**; }

VOLTAGE **V0** { NODE { n0 gnd } }

RESISTANCE **R0** { NODE { n0 n1 } }

RESISTANCE R1 { NODE { n1 n3 } }

INDUCTANCE L1 { NODE { n2 n3 } }

CAPACITANCE C1 { NODE { n1 gnd } }

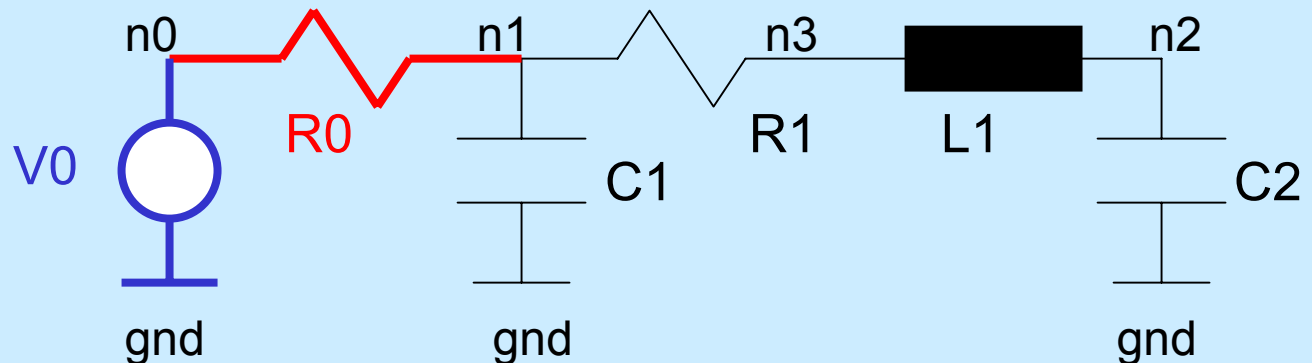
CAPACITANCE C2 { NODE { n2 gnd } }

DELAY { FROM { PIN=n1; } TO { PIN=n2; } ... }

}

← Identification  
← of driver model

DELAY =  
 $f(R0, R1, L1, C1, C2)$



# Interconnect noise calculation

WIRE lumpedRLC {

NODE n0 { NODETYPE = **source**; }

NODE n1 { NODETYPE = **driver**; }

NODE n2 { NODETYPE = **receiver**; }

VOLTAGE **V0** { NODE { n0 gnd } }

CAPACITANCE C1 { NODE { n0 n1 } }

RESISTANCE R1 { NODE { n1 gnd } }

RESISTANCE R2 { NODE { n1 n2 } }

CAPACITANCE C2 { NODE { n2 gnd } }

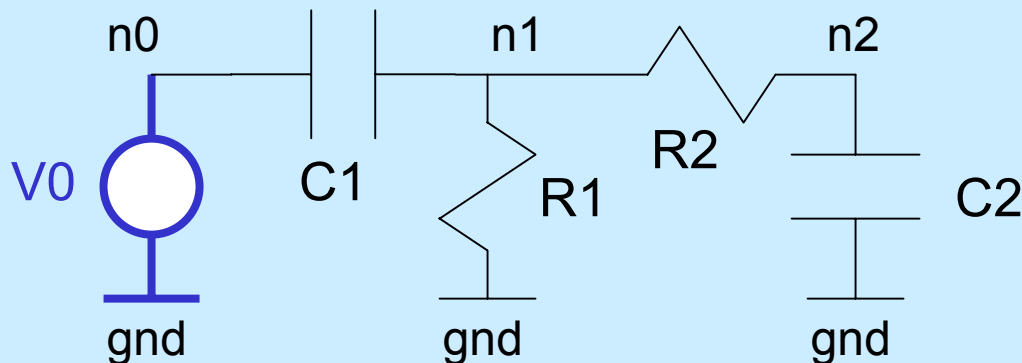
NOISE { PIN=n2; ... }

}

← Aggressor

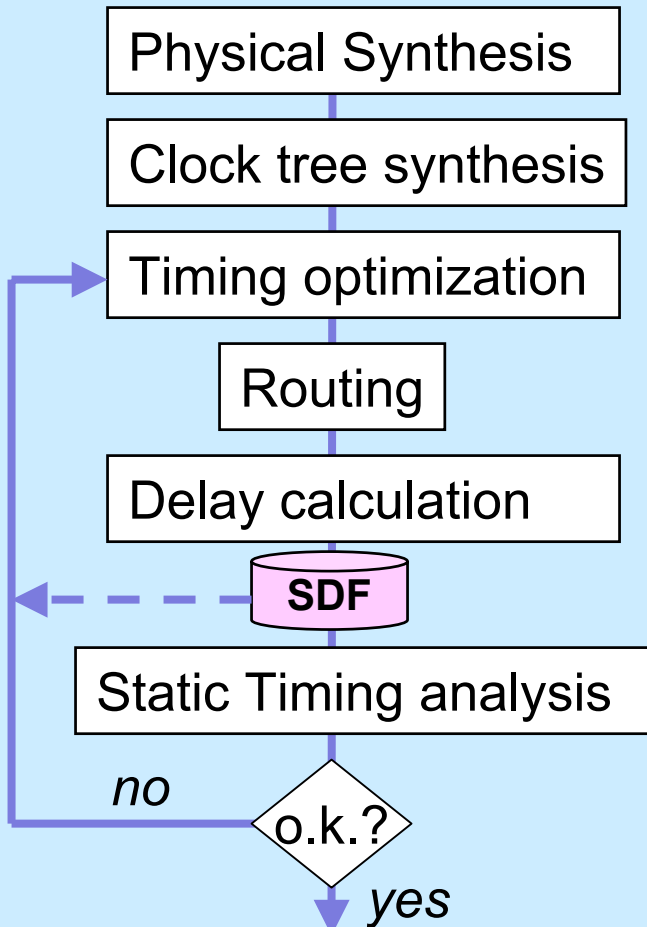
← Victim

←  $\text{NOISE} = f(V0, C1, R1, R2, C2)$

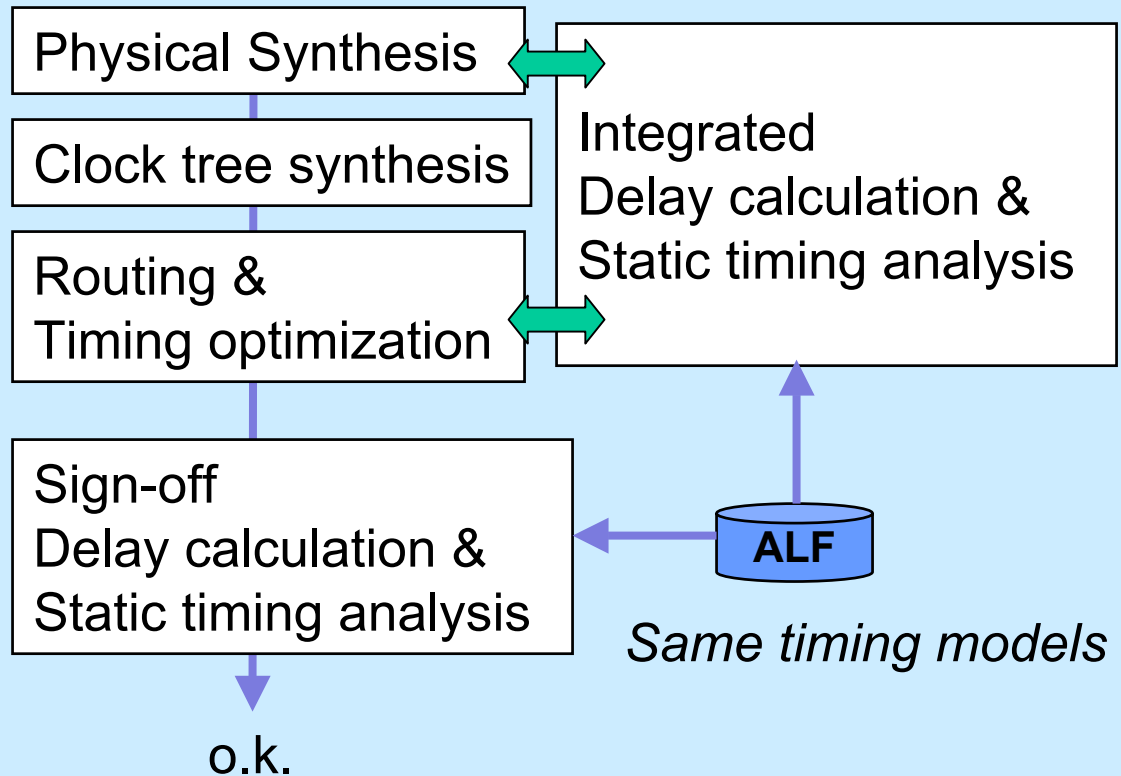


# Timing closure flow

without ALF



with ALF



# Power modeling

- ALF supports VECTOR-specific ENERGY & POWER
  - Most flexible modeling approach
  - Allows tradeoff between VECTOR set and accuracy
- ALF is complemented by Global Activity File (GAF)
  - GAF annotates design-specific VECTOR activity
  - GAF is an emerging industry standard
- ALF supports multiple voltage domains
  - Association between power supply pin and power rail system
  - Association between energy and power rail system
- ALF supports transient voltage drop analysis
  - Can describe pre-characterized current waveforms

# ENERGY and POWER

- ENERGY associated with transient VECTOR
- POWER associated with static VECTOR

*Event sequence*                      *Logical condition*

↓    ↓

```
VECTOR ( ( 01 in1 -> 10 out1 ) && ( ! in2 ) ) {  
    ENERGY { ... }  
}
```

← *Transient energy*

*Logical condition*

↓

```
VECTOR ( ! in1 && ! in2 ) {  
    POWER { ... }  
}
```

← *Static power*



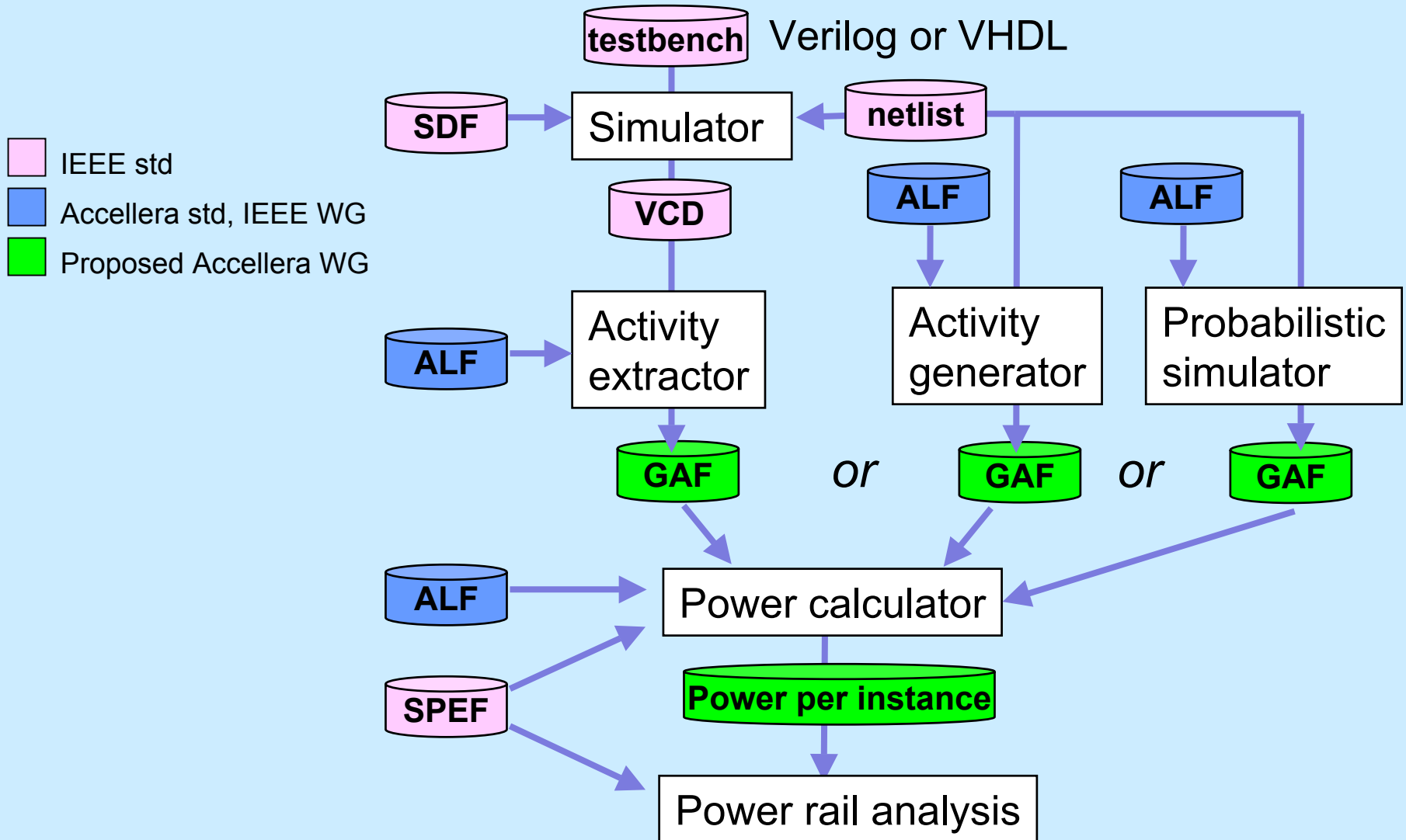
# Power analysis flow (1 of 2)

- For each cell instance in design:
  - Calculate ENERGY or POWER for each VECTOR
  - Get frequency or probability of each VECTOR
- Global Activity File (GAF) contains instance-specific frequency or probability for VECTOR
  - More accurate than frequency and probability per net
  - Logical correlations are preserved
  - Exact power results in conjunction with ALF library

$$\begin{aligned} \text{Total power} = & \sum_{\text{All transient vectors}} \text{ENERGY(VECTOR)} * \text{frequency(VECTOR)} \\ & + \sum_{\text{All static vectors}} \text{POWER(VECTOR)} * \text{probability(VECTOR)} \end{aligned}$$

ALF                      GAF

# Power analysis flow (2 of 2)



# Multiple voltage domains (1 of 2)

- Define a CLASS for a power supply system
- Define another CLASS for a power rail
- Power rail refers to power supply system

```
CLASS supply1 { USAGE = SUPPLY_CLASS; }  
CLASS supply2 { USAGE = SUPPLY_CLASS; }
```

```
CLASS vdd1 { SUPPLY_CLASS = supply1;  
             SUPPLYTYPE = power; VOLTAGE = 1.5; }  
CLASS vdd2 { SUPPLY_CLASS = supply2;  
             SUPPLYTYPE = power; VOLTAGE = 1.0; }  
CLASS vss { SUPPLY_CLASS { supply1 supply2 }  
            SUPPLYTYPE = ground; }
```



*Common ground  
for both supplies*

# Multiple voltage domains (2 of 2)

- Power/ground pin is connected to power rail
- Signal pin refers to power supply system
- Energy consumption refers to power supply system

CELL LevelShifter {

PIN vdd\_15 { CONNECT\_CLASS = **vdd1** ; }

PIN vdd\_10 { CONNECT\_CLASS = **vdd2** ; }

PIN vss { CONNECT\_CLASS = **vss** ; }

PIN in { DIRECTION=input; SUPPLY\_CLASS=**supply2**; }

PIN out { DIRECTION=output; SUPPLY\_CLASS=**supply1**; }

VECTOR (?! in -> ?! out ) {

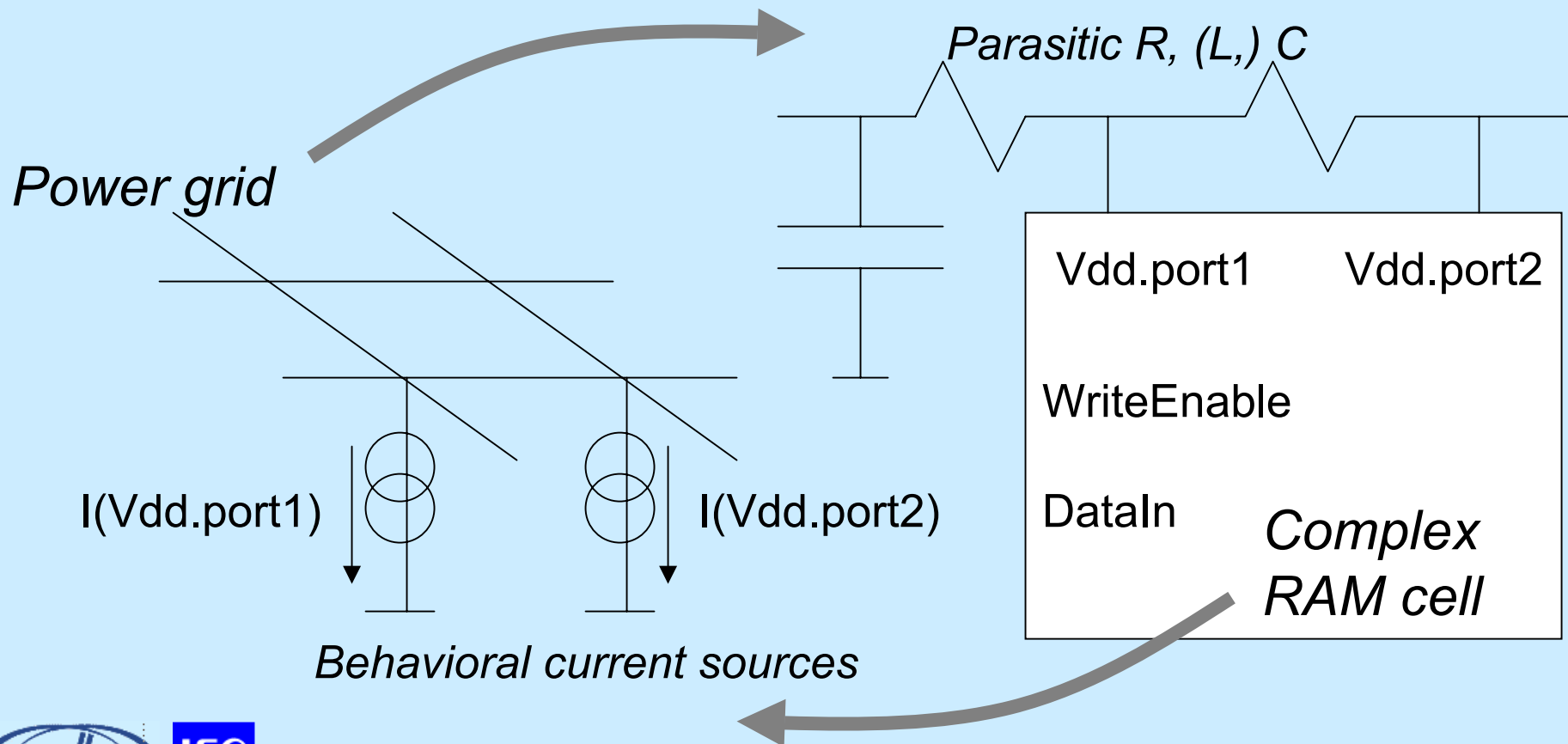
ENERGY = 0.8 { SUPPLY\_CLASS=**supply1**; }

ENERGY = 0.3 { SUPPLY\_CLASS=**supply2**; }

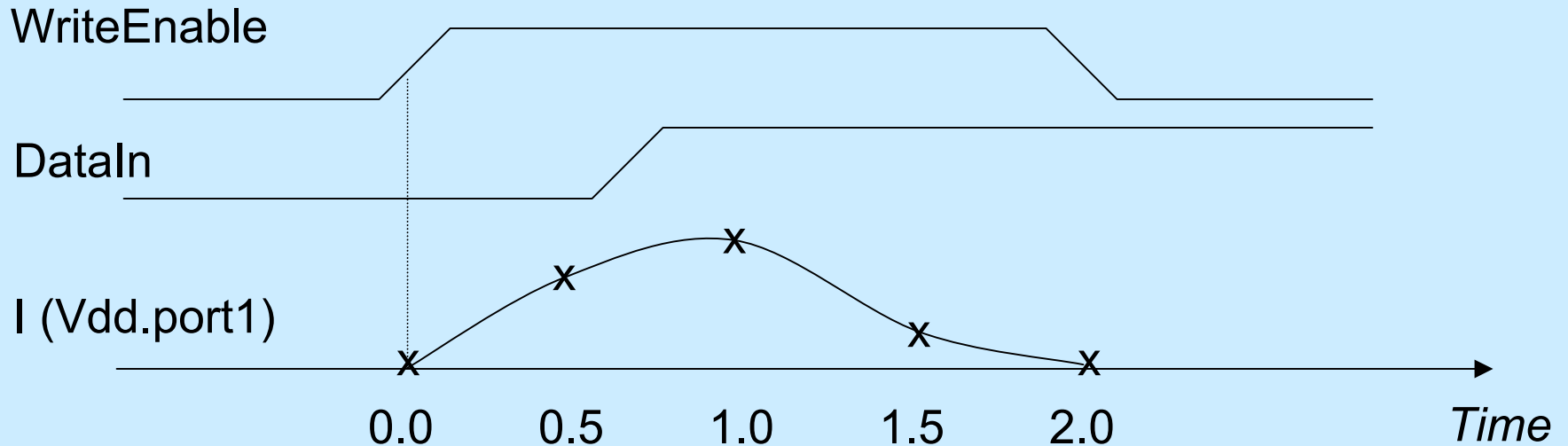
} }

# Transient voltage drop analysis (1 of 2)

- Transient current sources associated with cell
  - Temporal granularity: current per VECTOR
  - Spatial granularity: current per PIN or per PIN.PORT



# Transient voltage drop analysis (2 of 2)



```
VECTOR ( 01 WriteEnable -> 01 DataIn -> 10 WriteEnable ) {
  CURRENT { PIN = Vdd.port1; MEASUREMENT = transient;
    HEADER {
      TIME { FROM { PIN=WriteEnable; EDGE_NUMBER=0; }
        TABLE { 0.0 0.5 1.0 1.5 2.0 }
      } } TABLE { 0.0 0.4 0.5 0.2 0.0 }
    } } }
```

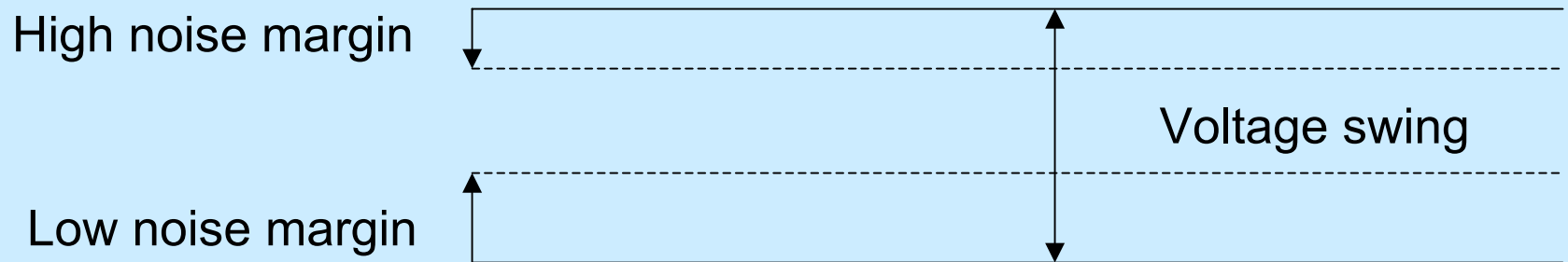
Time scale relative to event in VECTOR

# Advanced technology modeling

- ALF supports signal integrity
  - Static NOISE MARGIN
  - Event-sensitive NOISE MARGIN
  - Transient NOISE MARGIN
  - NOISE propagation
- ALF supports reliability
  - Signal and power electromigration
  - LIMIT for VECTOR-specific FREQUENCY
- ALF supports manufacturability
  - ANTENNA rules for technology
  - Artwork abstraction for hierarchical ANTENNA check

# Static NOISE MARGIN

- Static NOISE MARGIN in context of a PIN
- Can be specified as LOW and HIGH
- NOISE MARGIN is normalized to voltage swing

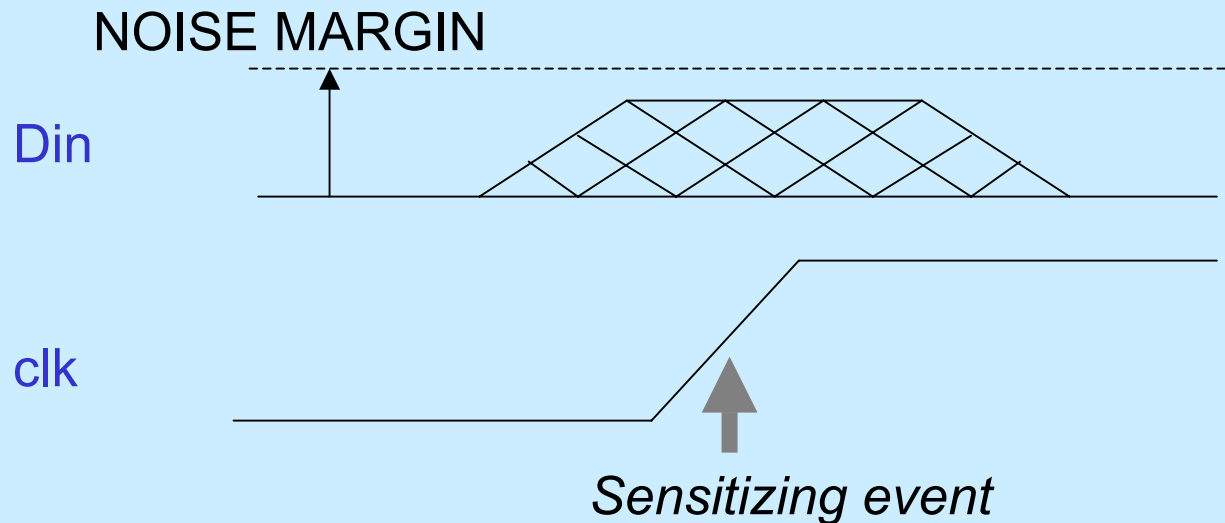


```
CELL FlipFlop {  
    PIN clk {DIRECTION = input; SIGNALTYPE = clock;  
            NOISE_MARGIN { LOW=0.4; HIGH=0.3; } }  
    PIN Din { DIRECTION = input; SIGNALTYPE = data; }  
    PIN Dout { DIRECTION = output; SIGNALTYPE = data; }  
}
```



# Event-sensitive NOISE MARGIN

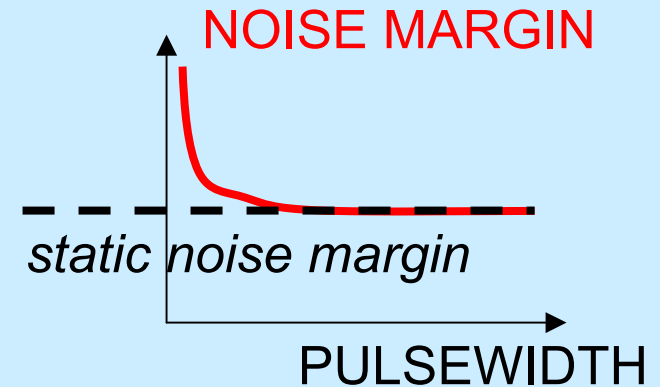
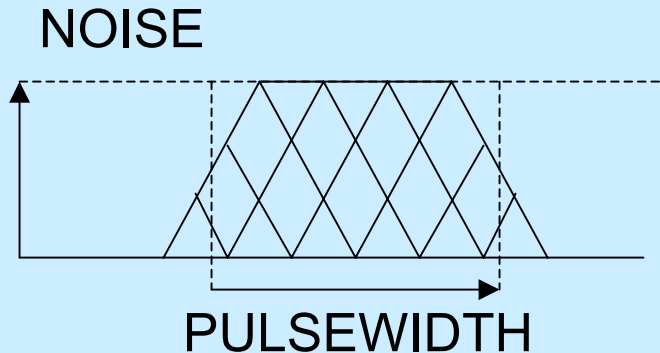
- Event-sensitive NOISE MARGIN in context of a VECTOR
- Event is described in VECTOR
- Example: noise on data pin during triggering clock edge



```
VECTOR ( 0* Din -> 01 clk -> *0 Din ) {  
    NOISE_MARGIN = 0.4 { PIN = Din; }  
}
```

# Transient NOISE MARGIN

- Transient NOISE MARGIN in context of a VECTOR
- Depends on PULSEWIDTH of noise waveform



*Noise event*

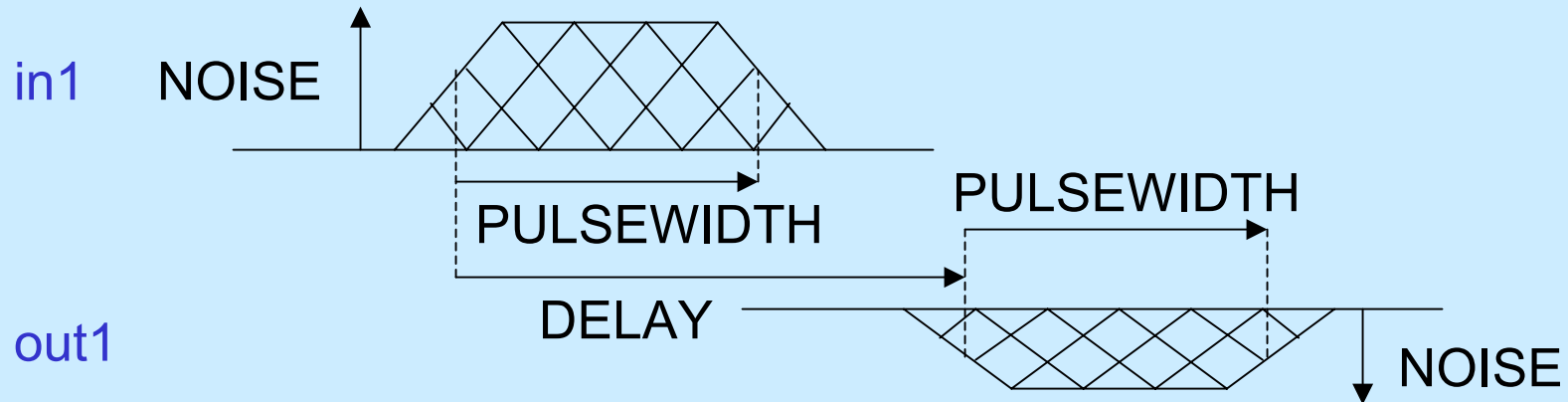
*Logical condition*

```

VECTOR ( ( 0* clk -> *0 clk ) && ( Din != Dout ) ) {
  NOISE_MARGIN { PIN = clk;
    HEADER {
      PULSEWIDTH { PIN=clk; TABLE { ... } } }
    TABLE { ... } } }
  
```

# NOISE propagation

- NOISE at output pin depends on NOISE at input pin
- NOISE propagation arc in context of VECTOR

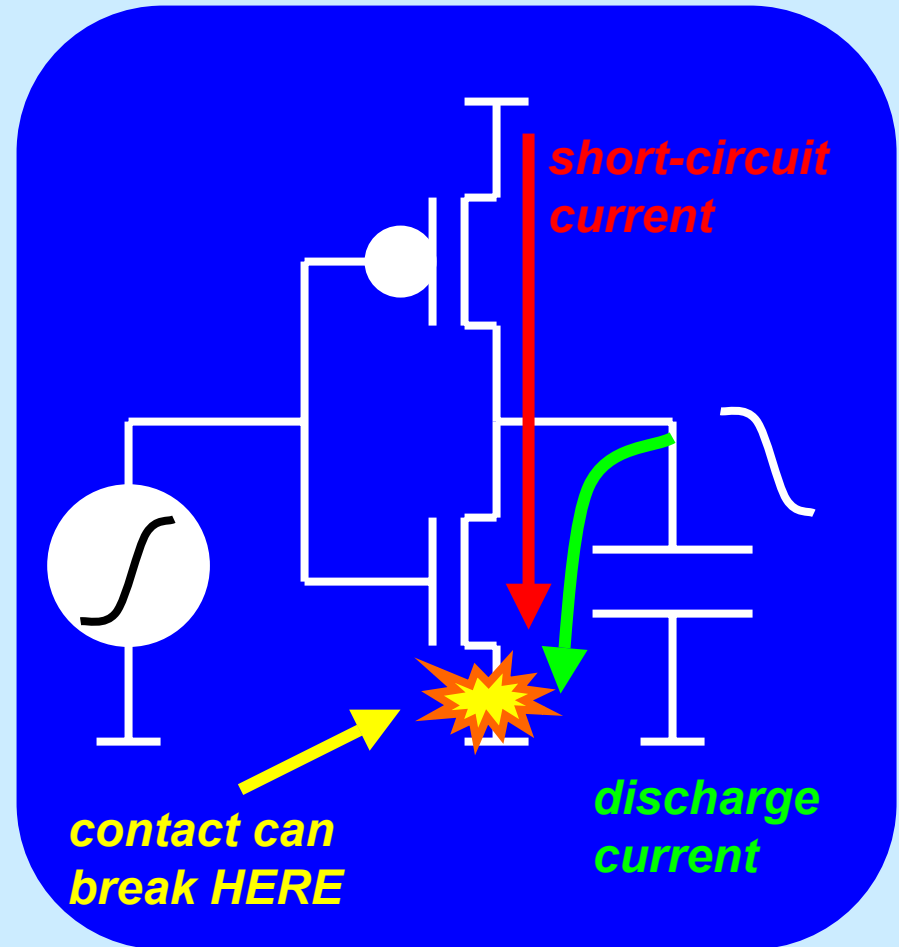
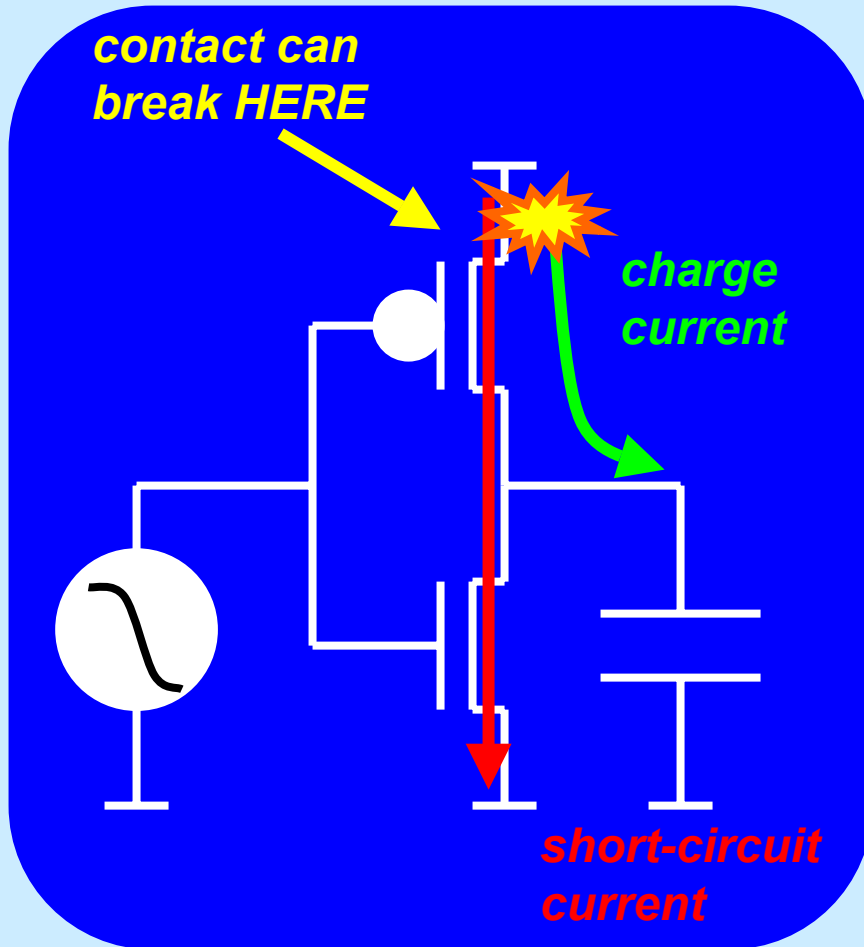


```

VECTOR ( 0* in1 -> *0 in1 <&> 1* out1 -> *1 out1 ) {
    NOISE { PIN = out1;
        HEADER {
            NOISE { PIN = in1; TABLE { ... } }
            PULSEWIDTH { PIN = in1; TABLE { ... } }
            CAPACITANCE { PIN = out1; TABLE { ... } }
        } TABLE { ... } } }
    
```

# Electromigration (EM) illustration

- Excessive current density leads to metal displacement
- Contacts or wire segments can break



# EM rules for technology (1 of 2)

- LIMIT for CURRENT described in context of LAYER
- Average measurement for DC (power route)
- Absolute average measurement for AC (signal route)
- Peak and RMS measurement also supported

```
LAYER metal1 {  
    LIMIT {
```

```
        CURRENT i_dc { MAX { ... }  
            MEASUREMENT = average; }  
        CURRENT i_ac { MAX { ... }  
            MEASUREMENT = absolute_average; }  
        CURRENT i_peak { MAX { ... }  
            MEASUREMENT = peak; }  
    }  
}
```

*for power route* →

*for signal route* →

# EM rules for technology (2 of 2)

- Current limit can be temperature-dependent
- Current limit can be width-dependent for routing layer
- Current limit can be area-dependent for cut layer

```
LIMIT {  
    CURRENT i_dc {  
        MAX {  
            HEADER {  
                WIDTH { TABLE { ... } }  
                TEMPERATURE { TABLE { ... } }  
            }  
            TABLE { ... }  
        }  
    }  
}
```

# EM rules for interconnect

- Models for peak and RMS current can be precharacterized
- Simple example: 1<sup>st</sup> order interconnect model



```
CAPACITANCE C1 { NODE { n1 gnd } }
```

```
RESISTANCE R1 { NODE { n0 n1 } }
```

```
CURRENT { COMPONENT = R1 ; MEASUREMENT = peak;
```

```
  HEADER {
```

```
    RESISTANCE { MODEL = R1; }
```

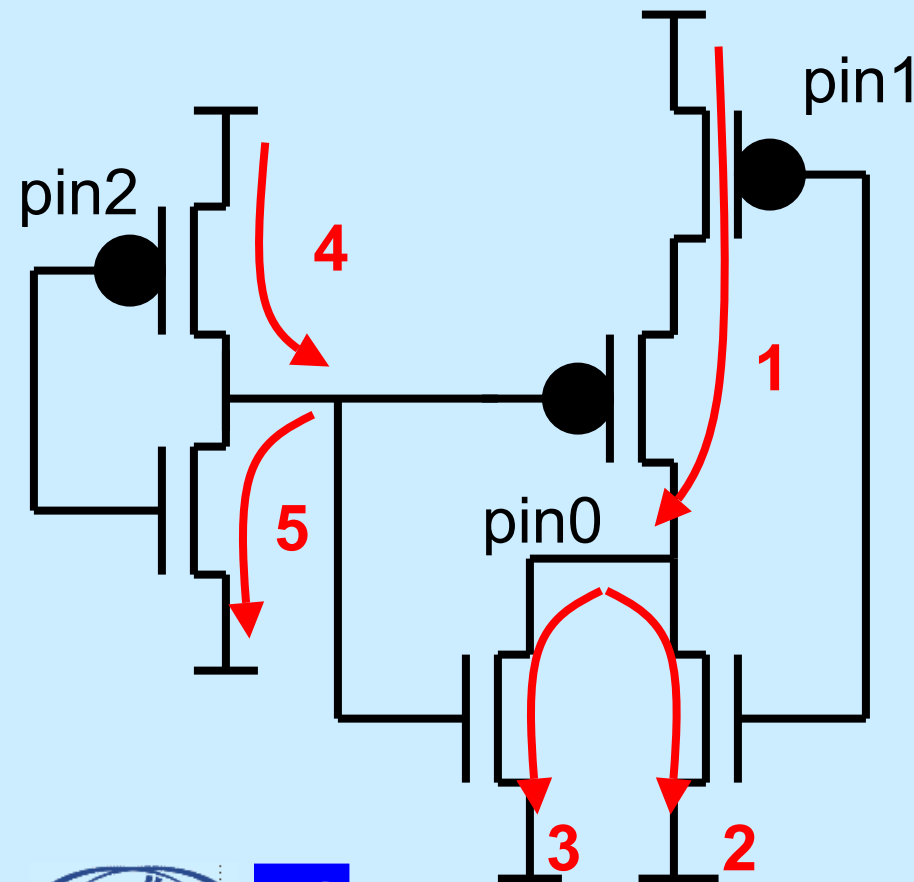
```
    CAPACITANCE { MODEL = C1; }
```

```
    SLEWRATE { PIN = n0; }
```

```
  } TABLE { ... } }
```

# EM rules for cells (1 of 2)

- Idea: Identify paths inside cell subjected to EM
- Define orthogonal VECTOR set for activating all paths
- Abstract EM rule into LIMIT for VECTOR FREQUENCY



VECTOR (01 pin0)  
{ /\* path 1 \*/ }

VECTOR (01 pin1 -> 10 pin0)  
{ /\* path 2 \*/ }

VECTOR (10 pin2 -> 10 pin0)  
{ /\* path 3 \*/ }

VECTOR (10 pin2)  
{ /\* path 4 \*/ }

VECTOR (01 pin2)  
{ /\* path 5 \*/ }



# EM rules for cells (2 of 2)

- Actual current = f (load CAPACITANCE, FREQUENCY)
- Maximum allowed current = f (TEMPERATURE)
- Maximum allowed FREQUENCY  
= f (load CAPACITANCE, SLEWRATE, TEMPERATURE)

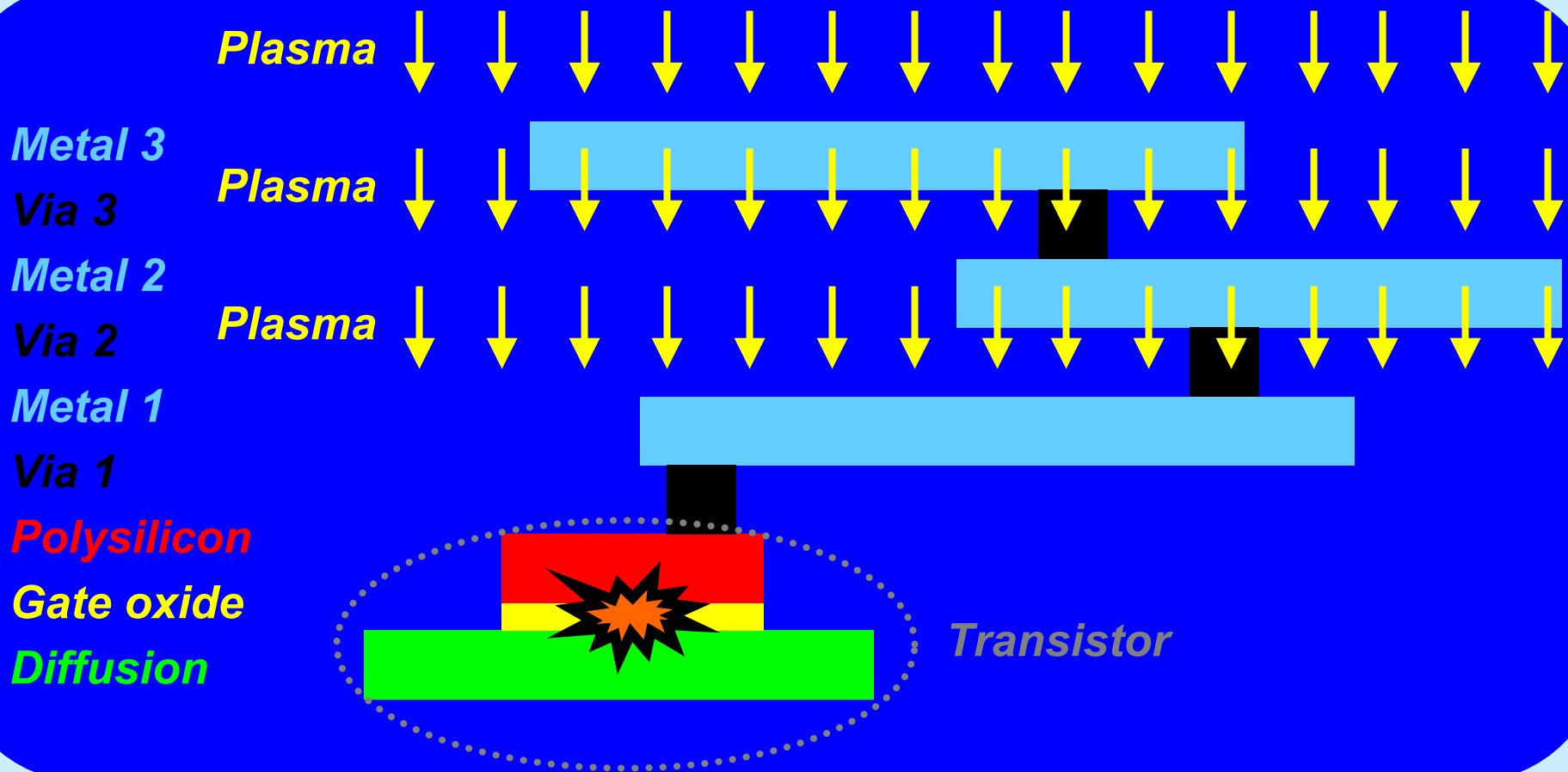
```
VECTOR (01 pin1 -> 10 pin0) {  
    LIMIT { FREQUENCY { MAX {  
        HEADER {  
            SLEWRATE { PIN=pin1; TABLE { ... } }  
            CAPACITANCE { PIN=pin0; TABLE { ... } }  
            TEMPERATURE { TABLE { ... } }  
        } TABLE { ... }  
    } } } }
```

# EM summary

- ALF supports comprehensive technology EM rules
  - ALF supports models for signal current calculation
  - Calculated current must be checked against EM rules
- ALF supports abstract models for cell EM rules
  - LIMIT for VECTOR-specific FREQUENCY
  - Can be dependent on SLEWRATE, load CAPACITANCE, TEMPERATURE
  - Can incorporate other lifetime-impacting effects, such as Hot Carrier, Thermal Instability
- Modeling approach scalable to complex cores
  - VECTOR paradigm same as for power analysis
  - Global Activity File (GAF) also usable in EM flow

# ANTENNA illustration

- Transistor collects charge during etching of metal structures
- Cumulative effect can destroy the transistor



# ANTENNA rules for technology (1 of 2)

- Prerequisite for ANTENNA rule description:
  - Each LAYER must be declared in LIBRARY
  - Order of LAYER declaration must be manufacturing order
  - Declaration from bottom to top

```
LIBRARY myTechnology {  
    LAYER diffusion { LAYERTYPE=reserved; }  
    LAYER poly { LAYERTYPE=reserved; }  
    LAYER cut0 { LAYERTYPE=cut; }  
    LAYER metal1 { LAYERTYPE=routing; }  
    LAYER cut1 { LAYERTYPE=cut; }  
    LAYER metal2 { LAYERTYPE=routing; }  
    // etc.
```

```
}
```

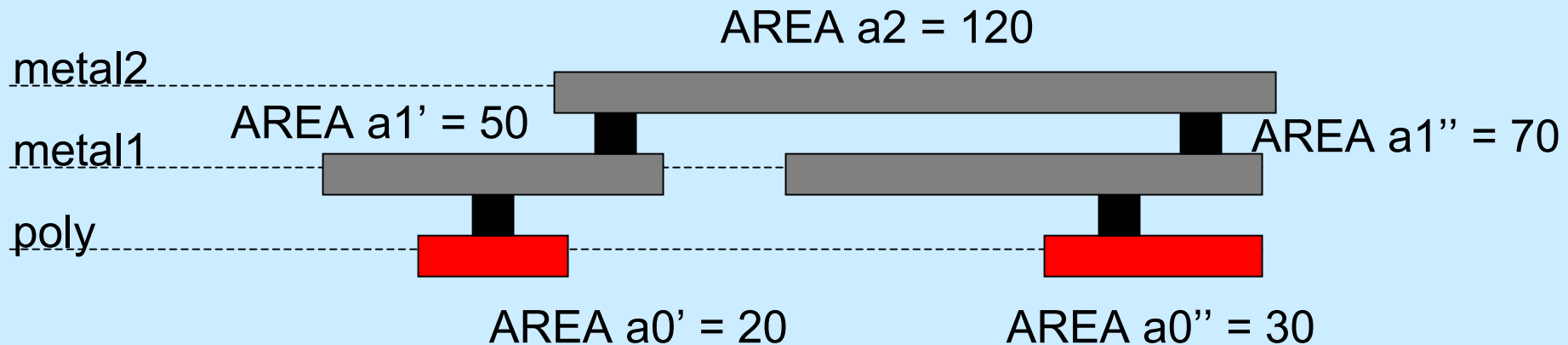
# ANTENNA rules for technology (2 of 2)

- Charge density depends on ratio between metal and transistor
- The greater the metal area and the smaller the transistor area, the greater the damage
- Diffusion alleviates antenna problem by diverting charge

```
ANTENNA cumulative_area {  
  SIZE s1 { CALCULATION = incremental;  
    HEADER {  
      AREA a1 { LAYER = metal1; }  
      AREA a0 { LAYER = poly; }  
      CONNECTIVITY { BETWEEN { metal1 diffusion } }  
    } EQUATION { CONNECTIVITY? 0.5*a1/a0 : a1/a0 } }  
  // put calculation for other layers here  
  LIMIT { SIZE { MAX = 1000; } } }
```

# ANTENNA rule evaluation

- Antenna rule checker must account for manufacturing order
- Count only top-down connections
- Combine poly areas connected top-down

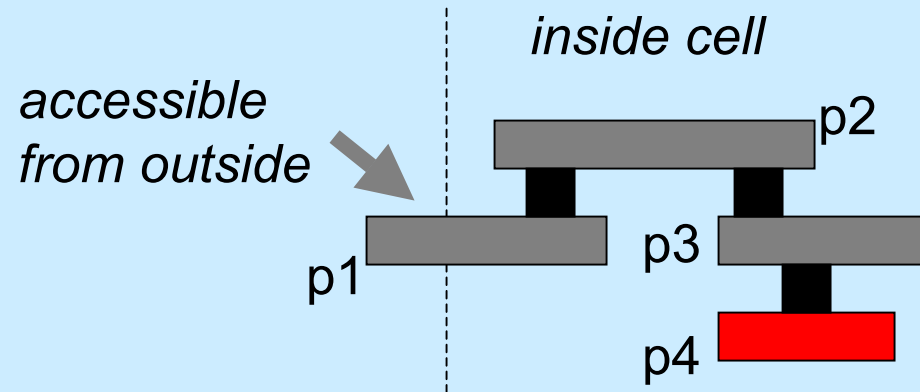


When metal1 is fabricated: check  $a1'/a0' = 50/20$  and  $a1''/a0'' = 70/30$

When metal2 is fabricated: check  $a2/(a0' + a0'') = 120/(20 + 30)$

# ANTENNA model for cell pin

- Antenna checker must look inside cell
- Abstraction of artwork required



```
PIN pin1 {  
  PATTERN p1 { LAYER=metal1; AREA=30; }  
  PATTERN p2 { LAYER=metal2; AREA=40; }  
  PATTERN p3 { LAYER=metal1; AREA=25; }  
  PATTERN p4 { LAYER=poly; AREA=20; }  
  CONNECTIVITY=1 { CONNECT_TYPE=physical;  
    BETWEEN { p2 p3 p4 } }  
  CONNECTIVITY=1 { CONNECT_TYPE=physical;  
    BETWEEN { p1 p2 } }  
  PORT port1 { PATTERN = p1; } }
```

# ANTENNA summary

- ALF supports ANTENNA technology rules
  - Layer-specific and cumulative rules
  - Partially cumulative rules (air gap layer)
  - Diffusion layer involved in rule
- ALF supports hierarchical ANTENNA model
  - Abstract artwork model for cell pin
  - Sufficient detail for accurate antenna check
  - Does not reveal artwork geometry
  - Suitable for IP modeling



# Conclusion and outlook

- ALF provides comprehensive modeling support
  - Timing with sign-off accuracy
  - Power from RTL to layout level
  - Signal integrity, reliability, manufacturability
- ALF is already deployed in the industry
  - Production-proven EDA tools
  - ASIC vendor libraries
  - Commercial library and IP providers
- ALF is a truly open standard
  - Vendor neutral
  - Forward looking
  - Recognized by IEEE and IEC

# ALF resources available to the Industry

- ALF tutorial at CICC 2001
- ALF paper at DATE 2002 Designers' Forum
- ALF specification documents

*Available for download at*

<http://www.eda.org/alf>

- Free ALF parser from Alternative Systems Concepts

*Available for download at*

<http://www.ascinc.com>



# ALF deployment in the industry

Sponsoring organization



Donation of free ALF parser



I/F to Milkyway DB

I/F to Volcano DB



Native lib for timing, power, SI tools



Native lib for RTL prototyping tool

Lib support from major ASIC vendors



# ALF covers superset of any other library combination

