Compact Model Standardization and Implementation Using Verilog-A

Outline

- Motivation
- Compact Model Standardization
  - Present, Status and Future
- Benefits Using Verilog-A
- Procedures and Tools
  - Spice-LS
  - Paragon
  - ADMS
  - Tiburon
- Summary

Motivation

Simulation & MODELING

Standardizing Compact Models for IC Simulation

By Britt Brooks

The idea of standard compact (SPICE-like) model equations has gained support recently throughout the semiconductor industry. In the past, compact models have been developed independently either by a single company or by a university or research group. These models have lacked diverse technology coverage and normally were not fully tested or productized. The concept of standardization is embraced by the semiconductor industry in several other areas, yet simulation has lagged behind due to the difficult notion of standardizing software. In this article, the idea of a standard compact model will be described as well as the industry consortium supporting the stan-

and redesign it based upon the testing results. The original program has been updated in several ways and has spawned many similar programs all with the same intent—simulating a circuit without fabricating it first.

As the semiconductor industry grew and new technologies were introduced, a need for electrical models that SPICE could use for simulation grew accordingly. Many companies addressed this need by focused model-development groups with the results being used internally as proprietary models. Companies that did not have the luxury of investing many man years of effort into model development depended on the models inherent

COMPACT MODEL

A SPICE model is considered a c
model because of the methods use
velop the equations and coefficien
for the electrical representation
physical behavior of a device. Th
compact is used because these eq
are simplified based upon several a
tions that are made when develop
model equations. Many of the equa
today's compact models would
seem brief or simplified; howe
sumptions are still needed to a
unique solutions that can be
SPICE programs.

Figure 1 shows the development cycle of a compact model. A model developer uses

etary. Once the cost of development and support of these models in external tools became too great, a new alternative was chosen: the standard model.

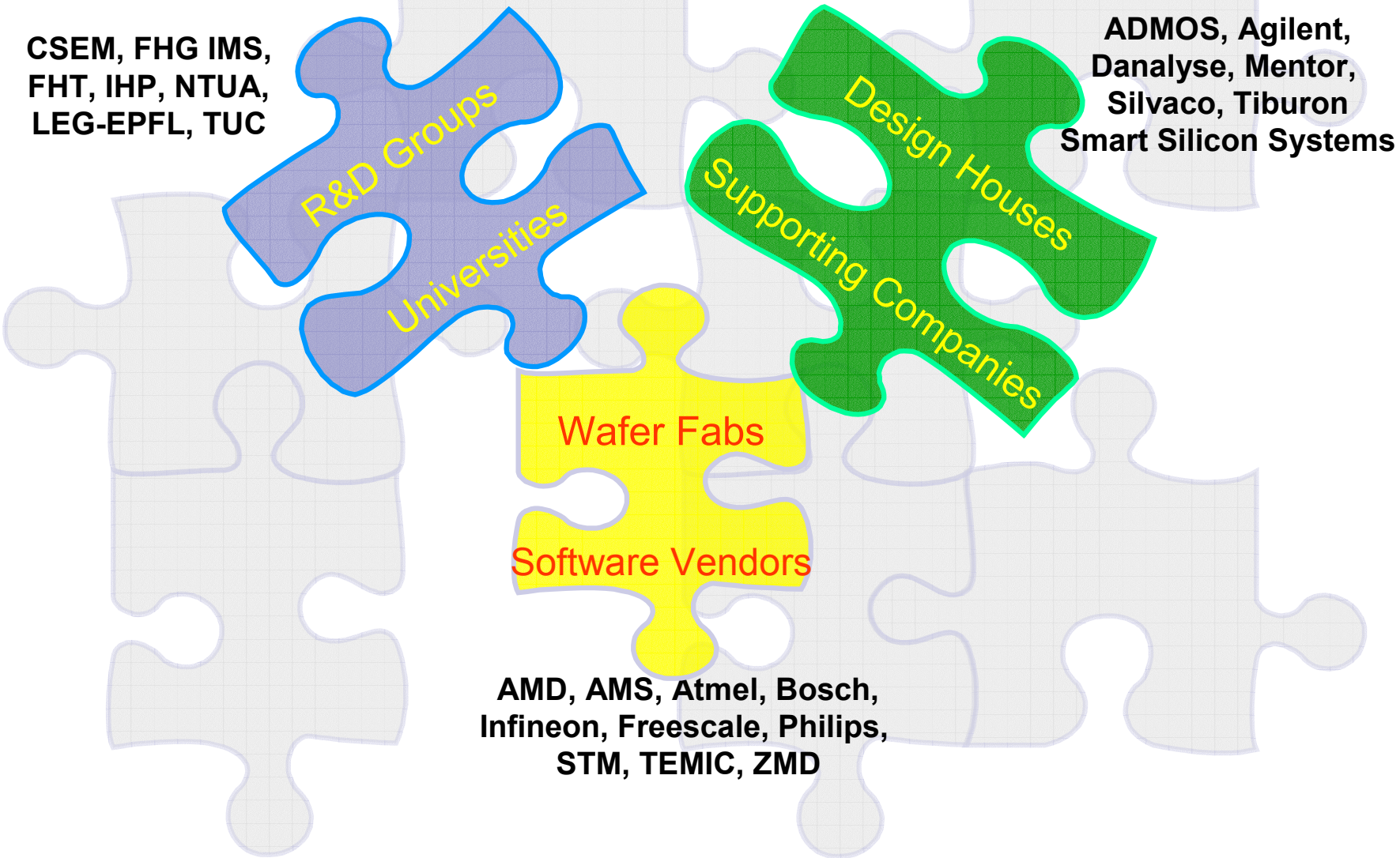
STANDARDIZATION

Standards are documented agreements containing technical specifications or other precise criteria to be used consistently as rules, guidelines, or definitions of characteristics, to ensure that materials, products, processes, and services are fit for their purpose [3]. A very simple example is a ruler. A ruler is simply a standard for length measurement. The standard ruler can be used to measure a variety of items; however, certain measurements may require more precision or a much greater capability than a simple ruler can supply, such as measuring a football field or the thickness of a human hair.

?

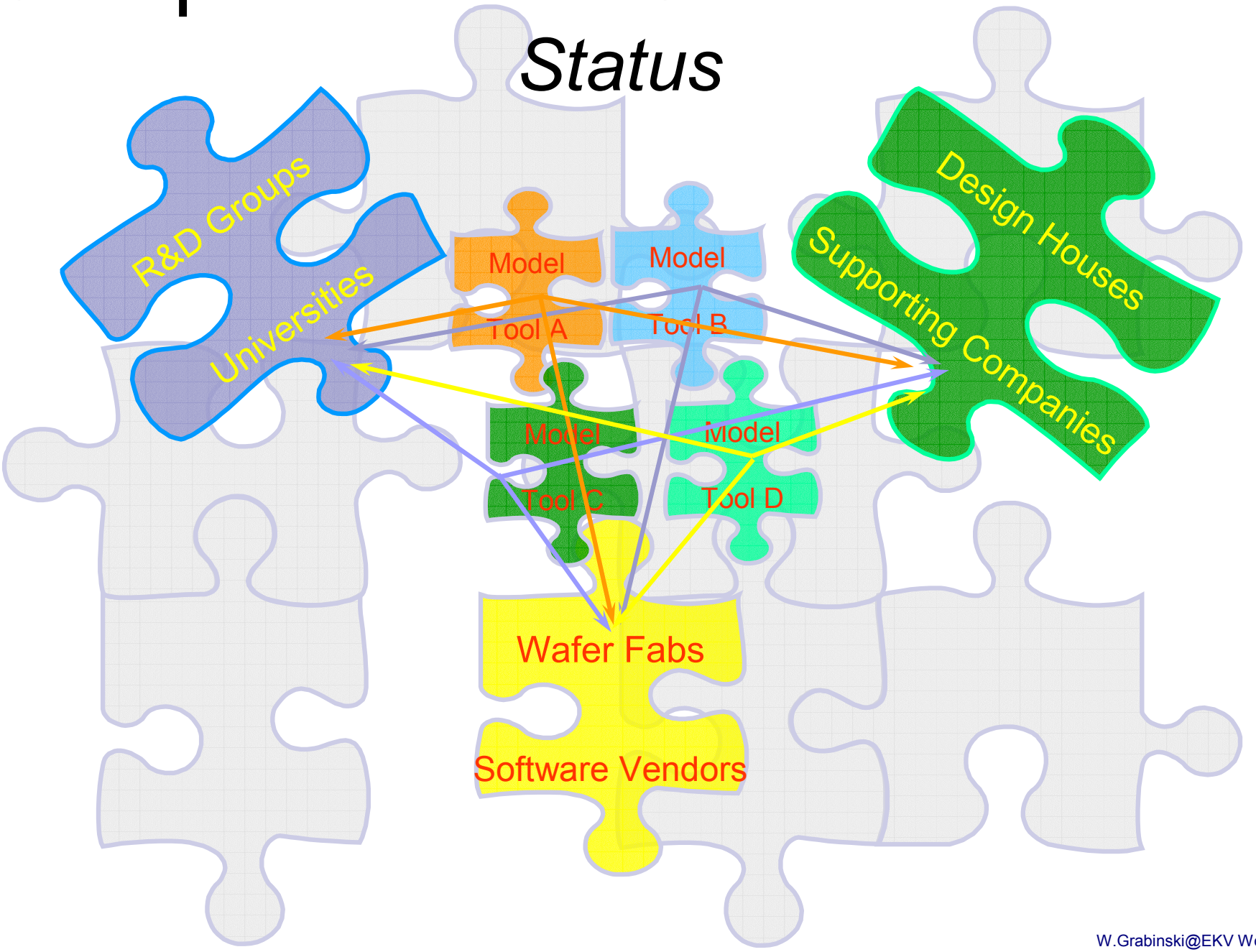
Compact Model Standardization

Present



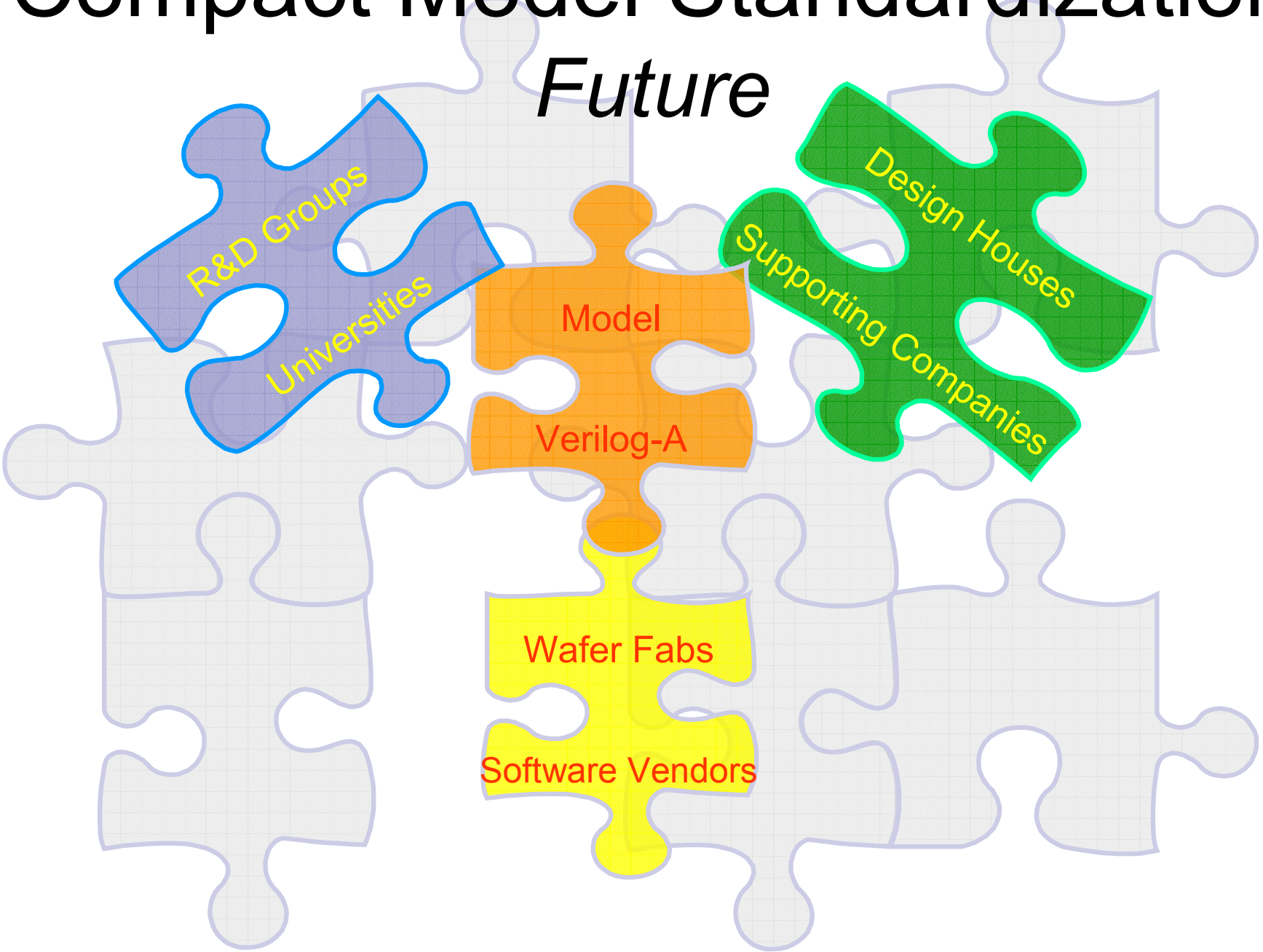
Compact Model Standardization

Status

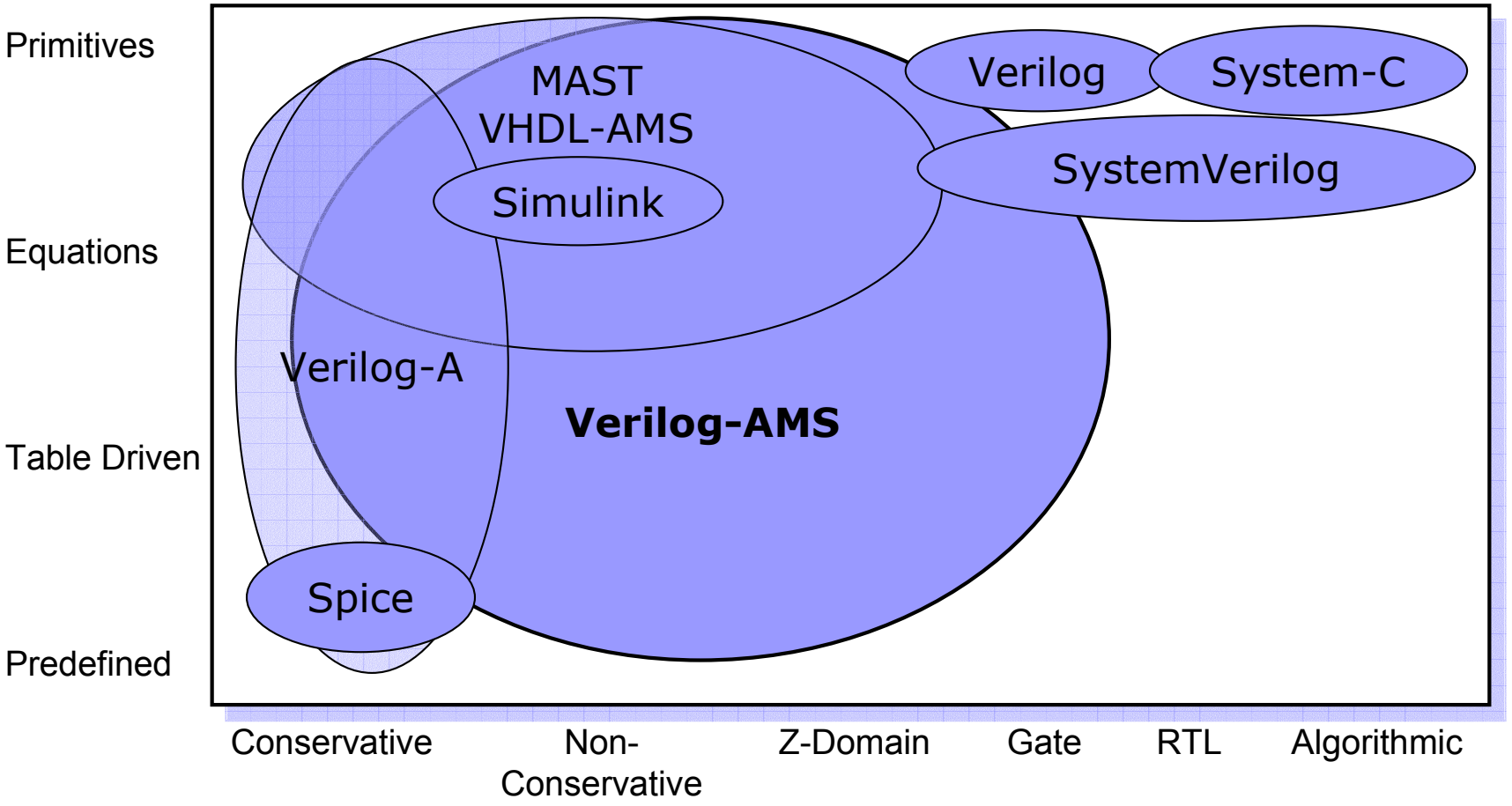


Compact Model Standardization

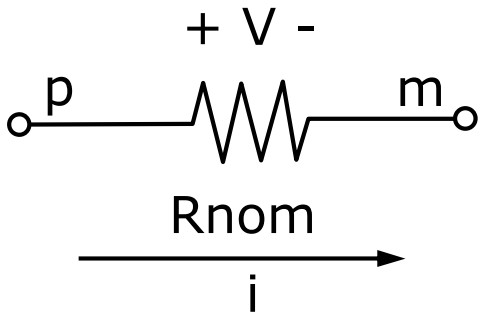
Future



Ranges of AMS Modeling



Behavioral Languages



The voltage across this device is determined by the simulator. The current through it is defined by the device characteristic equation:

$$i = \frac{V}{Rnom}$$

MAST:

```
template resistor p m = Rnom
electrical p, m

number Rnom




{
branch v=v(p, m), i=i(p->m)
equations {
    i = v / Rnom
}
}
```

VHDL-AMS:

```
use work.electrical_systems.all;
entity resistor is
generic (
Rnom : real); -- resistance value
port (
terminal p, -- positive pin
          m : electrical); -- minus pin
end entity resistor;
architecture simple of resistor is
quantity v across i through p to m;
begin
   i == v / Rnom;
end architecture simple;
```

Verilog-A:

```
`include "std.va"
module resistor (p, n);

parameter real Rnom=0 // in Ohms

inout p,n;
electrical p, m;



analog
   I(p,n) <+ V(p,n) / Rnom;
endmodule
```

Input Deck with Verilog-A

```
* EKV long channel MOSFET Model
* using Verilog-A
.verilog  "ekv.va"
vd  1  0 3
vg  2  0 5
vb  4  0 0
xekv  1 2 0 4 ekv L=20E-6 W=20E-6
.dc vd 0 5 0.1
.end
```

- EKV Verilog-A example:
  - http://legwww.epfl.ch/ekv/verilog-a/

EKV Verilog-A Model

```
`include "std.va"
`include "const.va"
// ***********************************
// * EKV MOS model (long channel)
// * http://legwww.epfl.ch/ekv
// ***********************************
module ekv(d,g,s,b);
//
// Node definitions
inout           d,g,s,b ;   // external nodes
electrical      d,g,s,b ;   // external nodes
//
//*** Local variables
real x, VG, VS, VD, VGprime, VP;
real beta, n, iff, ir, Ispec, Id;
//
//*** model parameter definitions
parameter real L      =  10E-6  from[0.0:inf];
parameter real W      =  10E-6  from[0.0:inf];
//*** Threshold voltage
//    substrate effect parameters (long-channel)
parameter real VTO    =  0.5    from[0.0:inf];
parameter real GAMMA  =  0.7    from[0.0:inf];
parameter real PHI    =  0.5    from[0.2:inf];
//***  Mobility parameters (long-channel)
parameter real KP     =  20E-6   from[0.0:inf];
parameter real THETA  =  50.0E-3 from[0.0:inf];
```

```
analog begin // EKV v2.6 long-channel
VG = V(g); VS = V(s); VD = V(d);
// Effective gate voltage (33)
VGprime = VG - VTO + PHI + GAMMA * sqrt(PHI);
// Pinch-off voltage (34)
VP = VGprime - PHI - GAMMA
   * (sqrt(VGprime+(GAMMA/2.0)*(GAMMA/2.0))-(GAMMA/2.0));
// Slope factor (39)
n = 1.0 + GAMMA / (2.0*sqrt(PHI + VP + 4.0*$vt));
// Mobility equation (58), (64)
beta = KP * (W/L) * (1.0/(1.0 + THETA * VP));
// forward (44) and reverse (56) currents
x=(VP-VS)/$vt; iff = (ln(1.0+exp( x /2.0)))*(ln(1.0+exp(
x /2.0)));
x=(VP-VD)/$vt; ir  = (ln(1.0+exp( x /2.0)))*(ln(1.0+exp(
x /2.0)));
// Specific current (65)
Ispec = 2 * n * beta * $vt * $vt;
// Drain current (66)
Id = Ispec * (iff - ir);
//
// Branch contributions to EKV v2.6 model (long-channel)
//
I(d,s) <+ Id;
end // analog
endmodule
```

EKV Verilog-A Model

```
`include "std.va"
`include "const.va"
// ***********************************
// * EKV MOS model (long channel)
// * http://legwww.epfl.ch/ekv
// ***********************************
module ekv(d,g,s,b);
//
// Node definitions
inout           d,g,s,b ;   // external nodes
electrical      d,g,s,b ;   // external nodes
//
//*** Local vari
real x, VG, VS,
real beta, n, if
//
//*** model para
parameter real L
parameter real W
//*** Threshold
//    substrate
parameter real V
parameter real G
parameter real P
//***  Mobility parameters (long-channel)
parameter real KP     =  20E-6   from[0.0:inf];
parameter real THETA  =  50.0E-3 from[0.0:inf];
```

```
analog begin // EKV v2.6 long-channel
VG = V(g); VS = V(s); VD = V(d);
// Effective gate voltage (33)
VGprime = VG - VTO + PHI + GAMMA * sqrt(PHI);
// Pinch-off voltage (34)
VP = VGprime - PHI - GAMMA
   * (sqrt(VGprime+(GAMMA/2.0)*(GAMMA/2.0))-(GAMMA/2.0));
// Slope factor (39)
n = 1.0 + GAMMA / (2.0*sqrt(PHI + VP + 4.0*$vt));
// Mobility equation (58), (64)
beta = KP * (W/L) * (1.0/(1.0 + THETA * VP));
// forward (44) and reverse (56) currents
x=(VP-VS)/$vt; iff = (ln(1.0+exp( x /2.0)))*(ln(1.0+exp(

                                                *(ln(1.0+exp(

                                                (long-channel)

end // analog
endmodule
```

Ports

Ports reflect the potential and flow descriptions of electrical, mechanical, thermal, and other systems.

A port has a direction: input, output, or inout,

EKV Verilog-A Model

```
`include "std.va"
`include "const.va"
// ***********************************
// * EKV MOS model (long channel)
// * http://legwww.epfl.ch/ekv
// ***********************************
module ekv(d,g,s,b);
//
// Node definitions
inout           d,g,s,b ;
electrical      d,g,s,b ;
//
//*** Local variables
real x, VG, VS, VD, VGprime
real beta, n, iff, ir, Ispe
//
//*** model parameter defin
parameter real L      =  10
parameter real W      =  10E-6  from[0.0:inf];
//*** Threshold voltage
//    substrate effect parameters (long-channel)
parameter real VTO    =  0.5    from[0.0:inf];
parameter real GAMMA  =  0.7    from[0.0:inf];
parameter real PHI    =  0.5    from[0.2:inf];
//***  Mobility parameters (long-channel)
parameter real KP     =  20E-6   from[0.0:inf];
parameter real THETA  =  50.0E-3 from[0.0:inf];
```

Data Types

integer
real
parameter
discipline data types

```
analog begin // EKV v2.6 long-channel
VG = V(g); VS = V(s); VD = V(d);
// Effective gate voltage (33)
VGprime = VG - VTO + PHI + GAMMA * sqrt(PHI);
// Pinch-off voltage (34)
VP = VGprime - PHI - GAMMA
   * (sqrt(VGprime+(GAMMA/2.0)*(GAMMA/2.0))-(GAMMA/2.0));
// Slope factor (39)
                                                VP + 4.0*$vt));

                                                TA * VP));
                                                rrents
                                                /2.0)))*(ln(1.0+exp(

                                                /2.0)))*(ln(1.0+exp(
// Drain current (66)
Id = Ispec * (iff - ir);
//
// Branch contributions to EKV v2.6 model (long-channel)
//
I(d,s) <+ Id;
end // analog
endmodule
```

EKV Verilog-A Model

```
`include "std.va"
`include "const.va"
// ***********************************
// * EKV MOS model (long channel)
// * http://legwww.epfl.ch/ekv
// ***********************************
module ekv(d,g,s,b);
//
// Node definitions
inout           d,g,s
electrical      d,g,s
//
//*** Local variables
real x, VG, VS, VD, V
real beta, n, iff, ir
//
//*** model parameter
parameter real L  =  10E-6  from[0.0:inf];
parameter real W  =  10E-6  from[0.0:inf];
//*** Threshold voltage
//    substrate effect parameters (long-channel)
parameter real VTO    =  0.5    from[0.0:inf];
parameter real GAMMA  =  0.7    from[0.0:inf];
parameter real PHI    =  0.5    from[0.2:inf];
//***  Mobility parameters (long-channel)
parameter real KP     =  20E-6   from[0.0:inf];
parameter real THETA  =  50.0E-3 from[0.0:inf];
```

Verilog-A Extensions

```
instance real L   =  10E-6 "m" - "drawn length";

model    real VTO =  0.5;
```

```
analog begin // EKV v2.6 long-channel
VG = V(g); VS = V(s); VD = V(d);
// Effective gate voltage (33)
VGprime = VG - VTO + PHI + GAMMA * sqrt(PHI);
// Pinch-off voltage (34)
VP = VGprime - PHI - GAMMA
   * (sqrt(VGprime+(GAMMA/2.0)*(GAMMA/2.0))-(GAMMA/2.0));
// Slope factor (39)
                                  + VP + 4.0*$vt));

                                  THETA * VP));
                                  ) currents
                                  p( x /2.0)))*(ln(1.0+exp(

                                  p( x /2.0)))*(ln(1.0+exp(
// Specific current (65)
Ispec = 2 * n * beta * $vt * $vt;
// Drain current (66)
Id = Ispec * (iff - ir);
//
// Branch contributions to EKV v2.6 model (long-channel)
//
I(d,s) <+ Id;
end // analog
endmodule
```

EKV Verilog-A Model

```
`include "std.va"
`include "const.va"
// ***********************************
// * EKV MOS model (long channel)
// * http://legwww.epfl.ch/ekv
// ***********************************
module ekv(d,g,s,b);
//
// Node definitions
inout           d,g,s,b ;
electrical      d,g,s,b ;
//
//*** Local variables
real x, VG, VS, VD, VGprime
real beta, n, iff, ir, Ispe
//
//*** model parameter defin
parameter real L      =  10
parameter real W      =  10E-6  from[0.0:inf];
//*** Threshold voltage
//    substrate effect parameters (long-channel)
parameter real VTO    =  0.5    from[0.0:inf];
parameter real GAMMA  =  0.7    from[0.0:inf];
parameter real PHI    =  0.5    from[0.2:inf];
//***  Mobility parameters (long-channel)
parameter real KP     =  20E-6   from[0.0:inf];
parameter real THETA  =  50.0E-3 from[0.0:inf];
```

Analog Events

```
@(initial_step)

@(initial_step("tran","ac","dc"))

@(final_step("tran"))
```

```
analog begin // EKV v2.6 long-channel
VG = V(g); VS = V(s); VD = V(d);
// Effective gate voltage (33)
VGprime = VG - VTO + PHI + GAMMA * sqrt(PHI);
// Pinch-off voltage (34)
VP = VGprime - PHI - GAMMA
    (sqrt(VGprime+(GAMMA/2.0)*(GAMMA/2.0))-(GAMMA/2.0));
                                                  VP + 4.0*$vt));
                                                  TA * VP));
                                                  rrents
                                                  /2.0)))*(ln(1.0+exp(
                                                  /2.0)))*(ln(1.0+exp(
// Drain current (66)
Id = Ispec * (iff - ir);
//
// Branch contributions to EKV v2.6 model (long-channel)
//
I(d,s) <+ Id;
end // analog
endmodule
```

EKV Verilog-A Model

```
`include "std.va"
`include "const.va"
// ***********************************
// * EKV MOS model (long channel)
// * http://legwww.epfl.ch/ekv
// *********
module
//
// Nod
inout
electr
//
//***
real x
real b
//
//***
parame
parame
//***
//
parameter real VTO    =  0.5    from[0.0:inf];
parameter real GAMMA  =  0.7    from[0.0:inf];
parameter real PHI    =  0.5    from[0.2:inf];
//***  Mobility parameters (long-channel)
parameter real KP     =  20E-6   from[0.0:inf];
parameter real THETA  =  50.0E-3 from[0.0:inf];
```

Branch Contribution

```
V(n1, n2) <+ expression;
I(g, b)   <+ ddt(QB);

Gm  = $ddx(Ids, V(g,s));
Cgb = $ddx(Qg, V(g,b));

I(n1, n2) <+ V(n1, n2) / R + white_noise(4*TK/R,"thermal");
I(n1, n2) <+ flicker_noise(KF * pow(abs(I(n1,n2)), AF), 1.0, "flicker");
```

```
analog begin // EKV v2.6 long-channel
VG = V(g); VS = V(s); VD = V(d);
// Effective gate voltage (33)
VGprime = VG - VTO + PHI + GAMMA * sqrt(PHI);
// Pinch-off voltage (34)
VP = VGprime - PHI - GAMMA
                                  // Branch contributions to EKV v2.6 model (long-channel)
//
I(d,s) <+ Id;
end // analog
endmodule
```

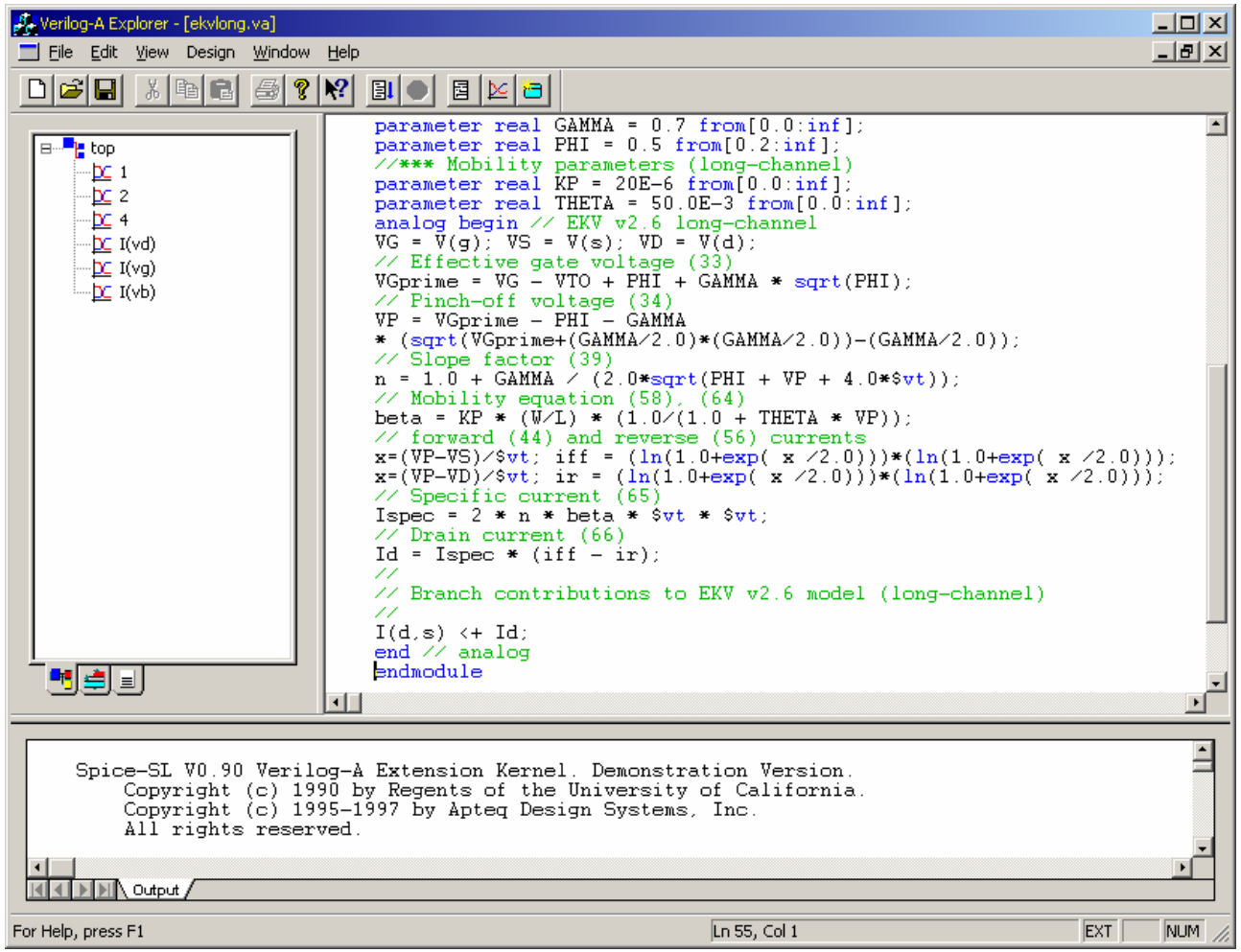
Benefits Using Verilog-A

- For the model developers
  - Develop once and run everywhere
  - Focus on model equation, not on implementation
- For the software vendors
  - Simplified implementation of the standard models
  - Proprietary Verilog-A models are also supported
- For the silicon fabs
  - Standardized model parameter set
- For the end-users (designers)
  - Standardized libraries and design kits

Tools for CM Standardization

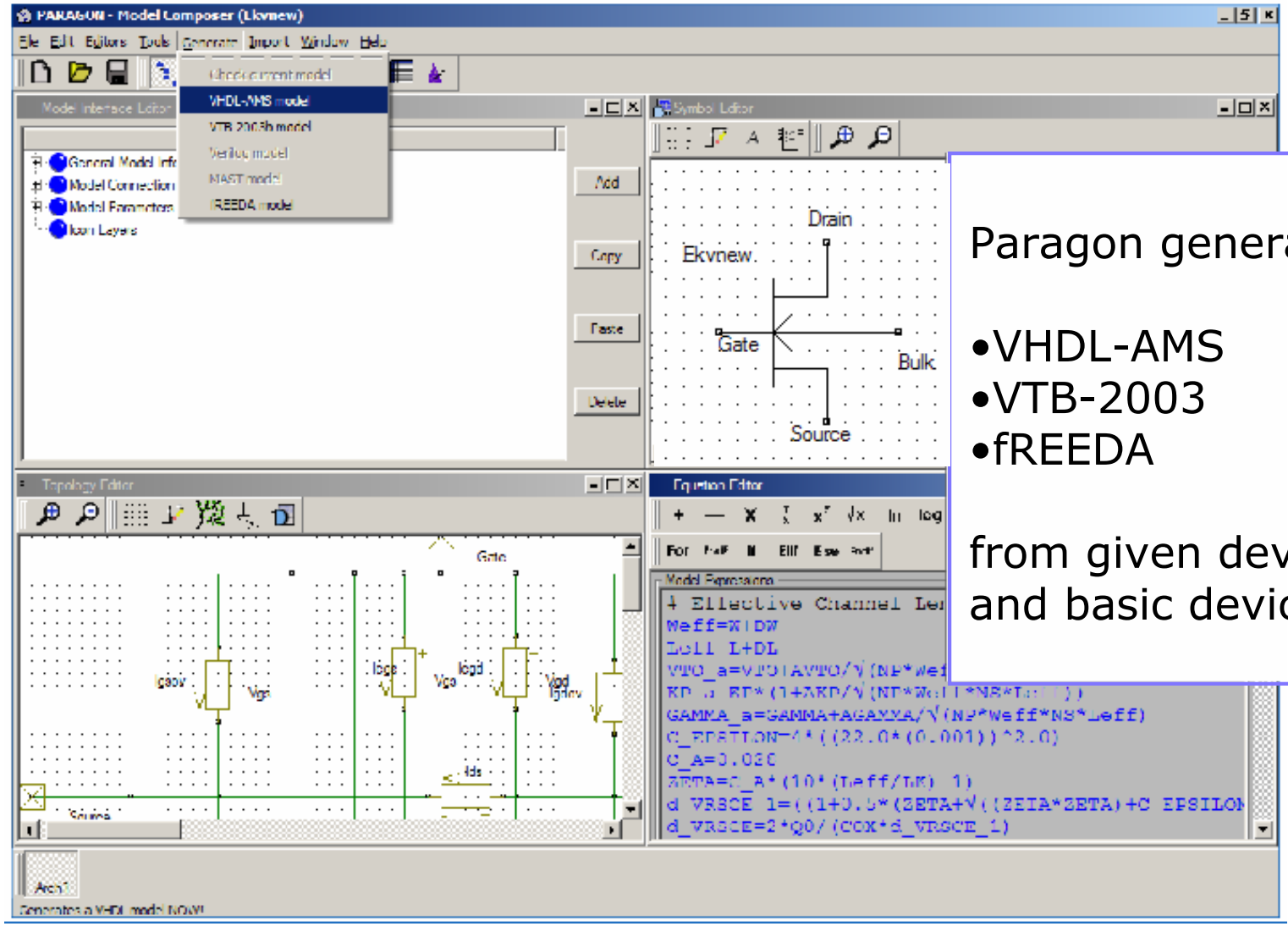
- Spice-SL/Verilog-A simulator
- PARAGON: HDLs graphical interface
  - http://mixedsignal.eleg.uark.edu/paragon.html
- ADMS for Verilog-A defined models
  - Laurent Lemaitre
  - http://sourceforge.net/projects/mot-adms
  - http://sourceforge.net/projects/mot-zspice
- RTE environment and Verilog-A compiler
  - Marek Mierzwinski, Tiburon DA Solution
  - http://www.tiburon-da.com

Spice/Verilog-A Simulator



D. Fitzpatrick and I. Miller
Analog Behavioral Modeling with the Verilog-A Language
ISBN 0-7923-8044-4

Paragon (U.Arkansas)



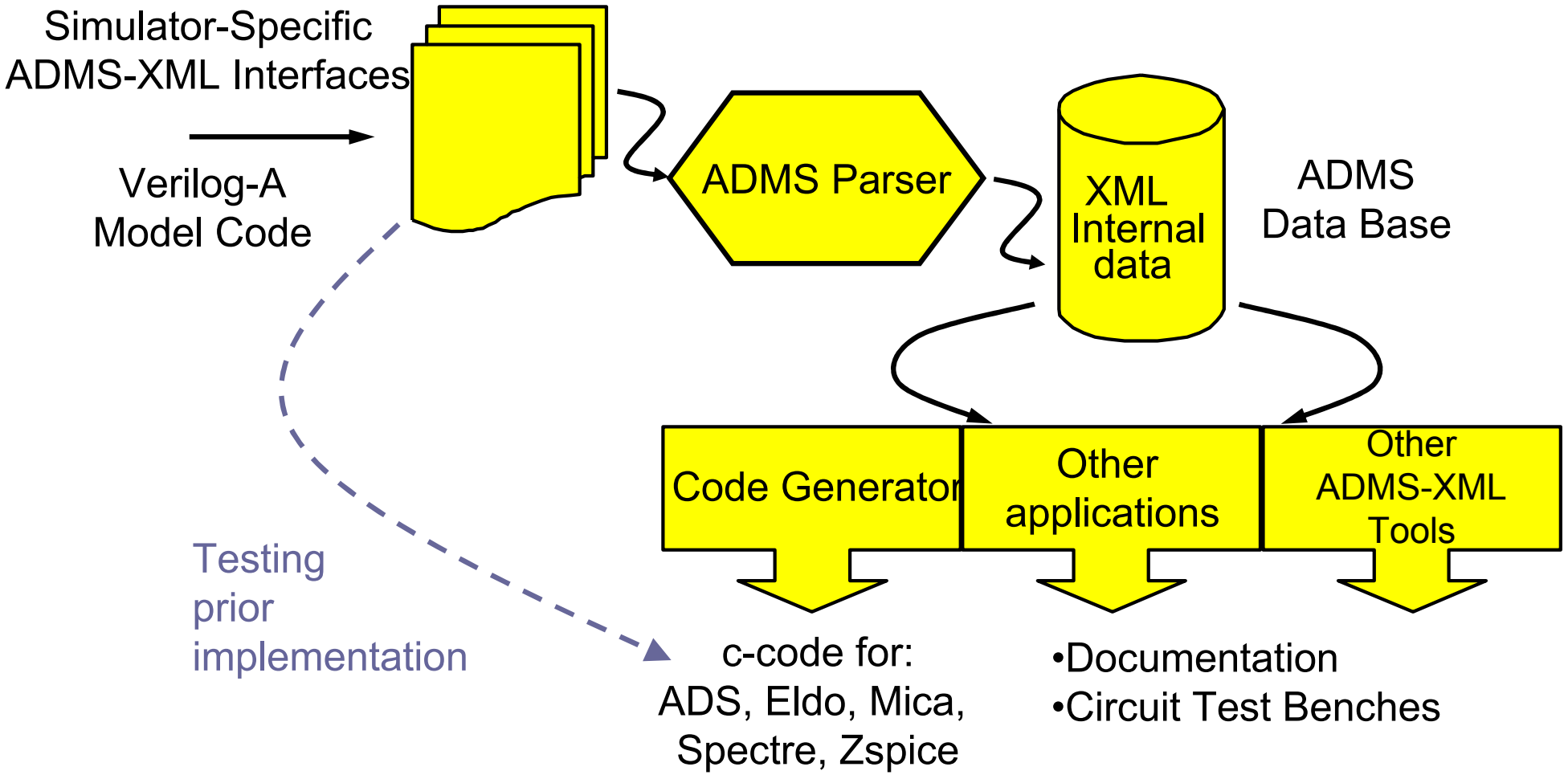
Paragon generates:

- VHDL-AMS
- VTB-2003
- fREEDA

from given device topology and basic device equations

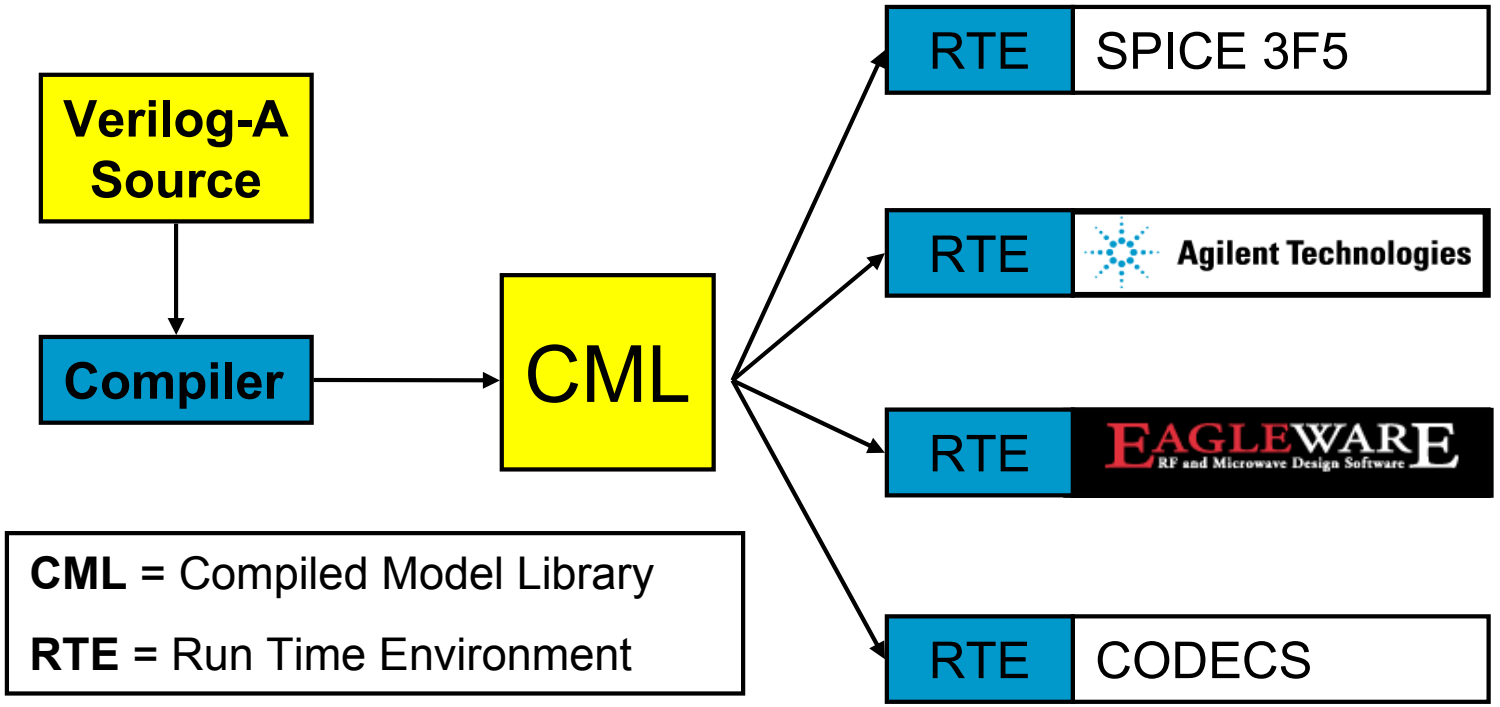
EKV Demo ➔ http://mixedsignal.eleg.uark.edu/paragon/ekv_tut.html

ADMS - Overview



http://sourceforge.net/projects/mot-adms

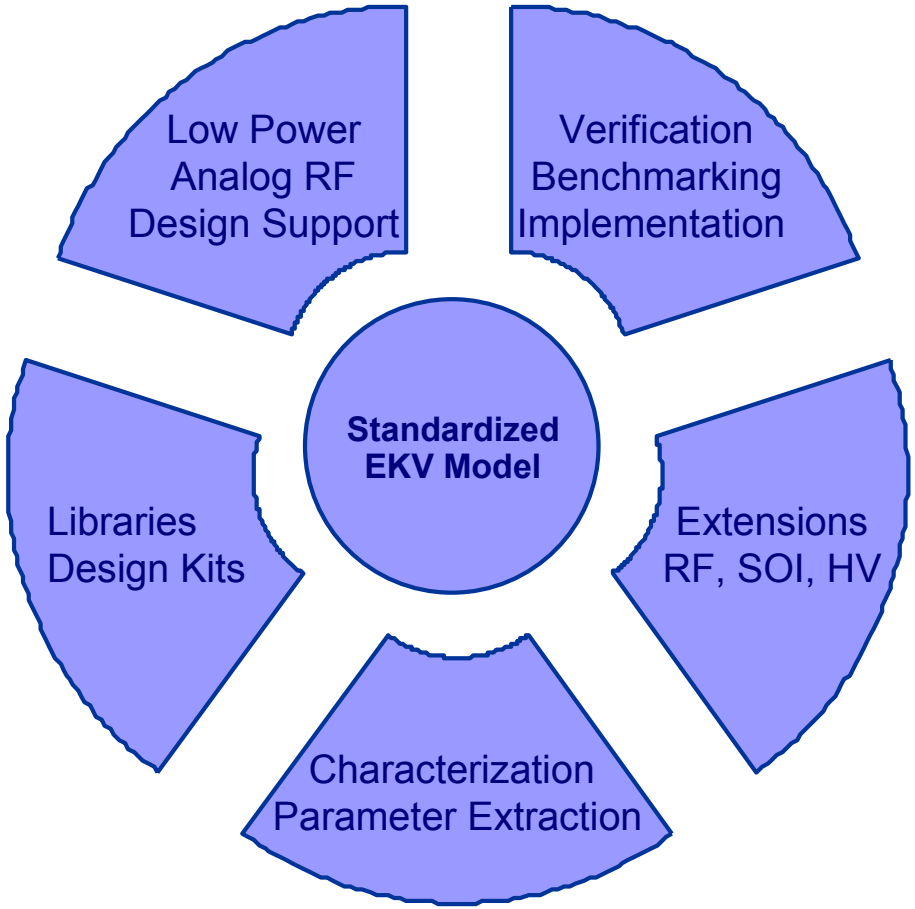
Tiburon RTE Architecture



http://www.tiburon-da.com

150+ Men/Yeas of Modeling Expertise

- Christian Enz
- Francois Krummenacher
- Eric Vittoz
- Matthias Bucher
- Christophe Lallement
- Jean-Michel Sallese
- Wladek Grabinski



+12 students associated with EKV professors

Acknowledgements

C. Enz, F. Krummenacher, E.Vittoz
Authors of the initial EKV paper

M. Bucher, J-M. Sallese, C. Lallement
EKV Development Team

F. Krummenacher, M. Bucher,
Smart Silicon Systems (S3)

C.McAndrew, L.Lemaitre, J. Victory, O. Pilloud
GMC, Freescale

M.Mierzwinski, F. Sischka
Tiburon and Agilent

All developers implementing the EKV model into public domain and commercial simulation tools

"a good model can advance fashion by ten years"
Yves Saint Laurent

standardized

"a ~~good~~ model can advance fashion by ten years"

Yves Saint Laurent