Code(Equation) Based Models, an accurate way to represent, and transfer simulation models

Tudor Secasiu – Technology and Research Labs (tudor.secasiu@intel.com)

Motivation

Product cycle:

- Technology readiness (~years)
- Product definition (marketing, development freeze)
- Product Execution (~months)

Frequencies:

- GHz (~ ps of margin) for bus cycles.

Different design needs are addressed by different CAD tools:

  - Technology readiness - Proof of concept/model validation
  - Product Execution - Manufacturability

Motivation – cont.

Implications:

- Most of the model qualification/simulation work has to be done during technology readiness.
- ACCURATE (~ps) simulation models have to be transferred to product teams, in a short time, and in a DIE-TO-DIE format that includes power delivery, driver/pre-driver, receiver, package, connectors and interconnect.
  - No time to re-qualify models
  - No luxury to allocate margins due to model differences

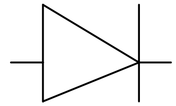
Transfers between simulators are still necessary:

- Net lists – not a major source of error
  - Flattened – easiest way
  - Hierarchical
- Layout – small structures ~ O.K.
- Models –most frequent cause of errors.

Simulators Today

How do (most) simulators work?

Model Instance



User input

Is=

Model Card - mathematical equation

Vt = boltzmann*temp_kelvin/q_electron

I(1,0)=Is*(exp(_v1/Vt)-1)

User has no control

Output to the simulation engine

I,V,derivatives, integrals, etc.

Simulators Today – cont.

Issues with existing simulators:

- Models are "pre-defined" by the tool vendor.
- Technology differences
  - ❑ Usually based on an industry standard that does not account for company specifics in silicon process.
- Time differences
  - ❑ Any model improvement requires a new "revision" from CAD vendors.
- Does not protect confidential information.

Issues with model transfers between simulators:

➢Model compatibility ⟶ errors

➢Solution:

❖Transfer of mathematical equations

❑no errors

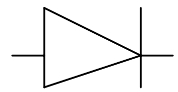
or

❑User controlled errors (ex: Taylor expansions)

What is a Code Based Model?

User definition of both Model Card and Model Instances using mathematical functions and equations.

Model Instance



Is=
Co=
Vo=

User controlled

User defined Model Card - mathematical equation

```
Qf(v)=Co*(v-alpha*Vo+(v-alpha*Vo)^2/(4*Vo*(1-alpha)))/sqrt(1-alpha)+Qr(alpha*Vo)
Qr(v) = -2*Co*sqrt(Vo*(Vo-v))
Q(v) = if (v<alpha*Vo) then Qr(v) else Qf(v) endif
My_exp(x) = if (x < max_arg) then exp(x) else (x+1-max_arg)*max_exp endif
Vt = boltzmann*temp_kelvin/q_electron
I(1,0)=Is*(My_exp(_v1/Vt)-1)
I(1,1)=differential(Q(_v1))
```

User controlled

Output to the simulation engine

I,V,derivatives, integrals, etc.

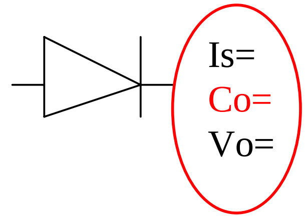
Advantages

- Very accurate and intuitive.
  - Can be made available in early development
- Portable (just mathematics) from a system to another.
- Can represent a single device, or a whole design block.
  - transfer function
- Win-Win situation:
  - CAD vendors don't have to "spin-around" their model each time a new model (standard) appears.
  - Users don't have to rely on CAD vendors and industry standards to model their own process (in time).
- Protection of confidential process information.
  - By eliminating any process references (how you write the equations).
  - Possibility to encode the model card into binary (object) files.

Protecting sensitive information

Allows flexible encoding options:

Model Instance



If desired can be "hidden" in the model card

User defined Model Card - mathematical equation

Qf(v)=Co*(v-alpha*Vo+(v-alpha*Vo)^2/(4*Vo*(1-alpha)))/sqrt(1-alpha)+Qr(alpha*Vo)
Qr(v) = -2*Co*sqrt(Vo*(Vo-v))
Q(v) = if (v<alpha*Vo) then Qr(v) else Qf(v) endif
My_exp(x) = if (x < max_arg) then exp(x) else (x+1-max_arg)*max_exp endif
Vt = boltzmann*temp_kelvin/q_electron
I(1,0)=Is*(My_exp(_v1/Vt)-1)
I(1,1)=differential(Q(_v1))

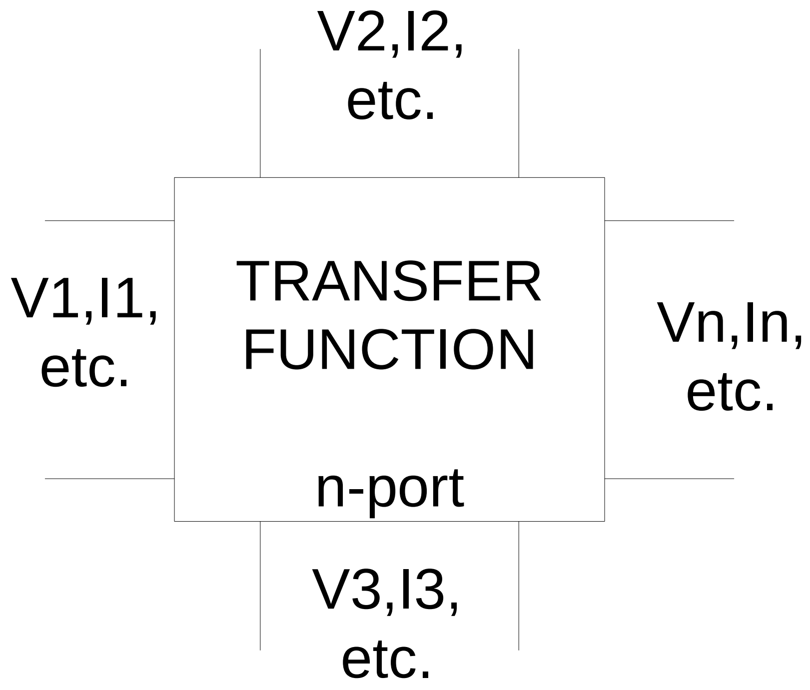
Can be encoded

Output to the simulation engine

I,V,derivatives, integrals, etc.

Implementation

1. Define a set of functions/equations and **methods of encoding** with an industry standard
2. Define a common I/O interface (I,V,C, etc) and adopt an existing programming language → encoding in binary format



N-port "black box" with input/output and transfer function

Summary

- Code Based Model - a real time model.
  - Treated in the same way as an internal simulator model.
- User has to have full control of the model
- Encoding capabilities
- Definition of a common I/O interface and a real time link (DLL) of a (encoded) file
  - Ex: object file from a C code

    or
- A common set of functions with encoding capabilities
  - Ex:IBIS extension