Lecture 19. Analog Models

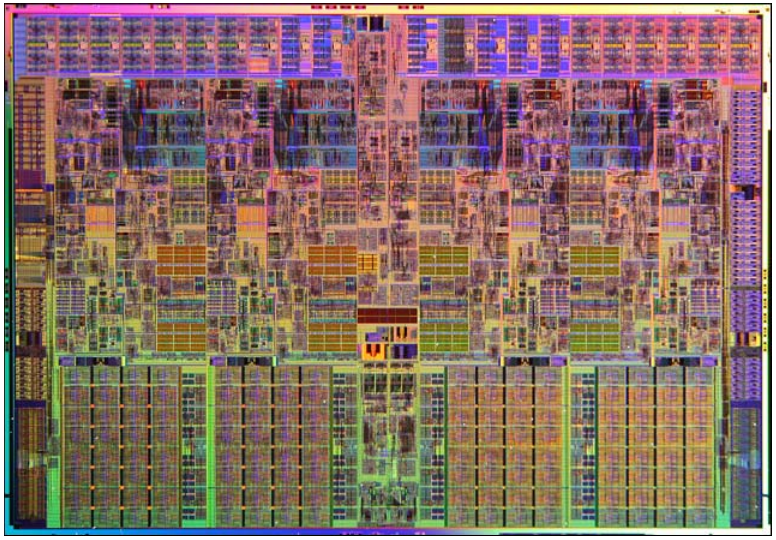
Jaeha Kim
Mixed-Signal IC and System Group (MICS)
Seoul National University
jaeha@ieee.org

Overview

- Readings
  - Three papers in the DAC'10 special session titled "Analog Model Crisis – How Do We Solve it?"
    - Luca Daniel (MIT) – Bottom-up approach (MOR)
    - Ken Kundert – Top-down approach
    - Mark Horowitz (Stanford) – Analog model equivalence check
- Introduction
  - While digital system design lies its solid foundation on models, analog design lacks a way of utilizing models effectively. This lecture highlights the current views on the analog model issues.

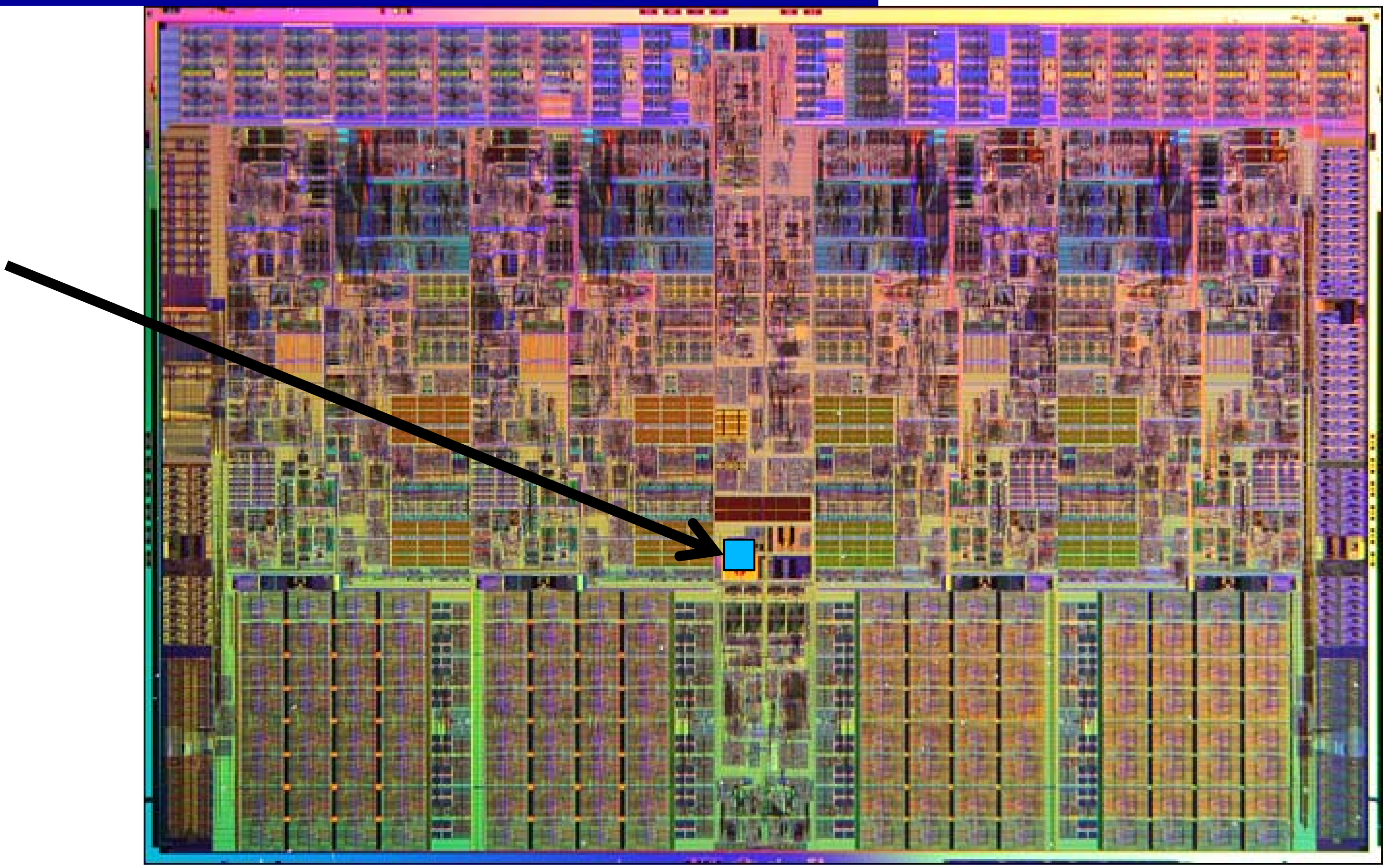
The Analog Bottleneck

- The main driving force behind the complexity scaling of CMOS systems has been digital, not analog
  - Near a billion of transistors in modern processors
- Analog circuit hasn't scaled its complexity much
  - 10~1000s of transistors
  - Technology scaling forces even simpler analog
  - Trend is to replace analog with digital for easy process migration and design management (Big D, Little A)



Intel 4-core Nehalem processor (820M)

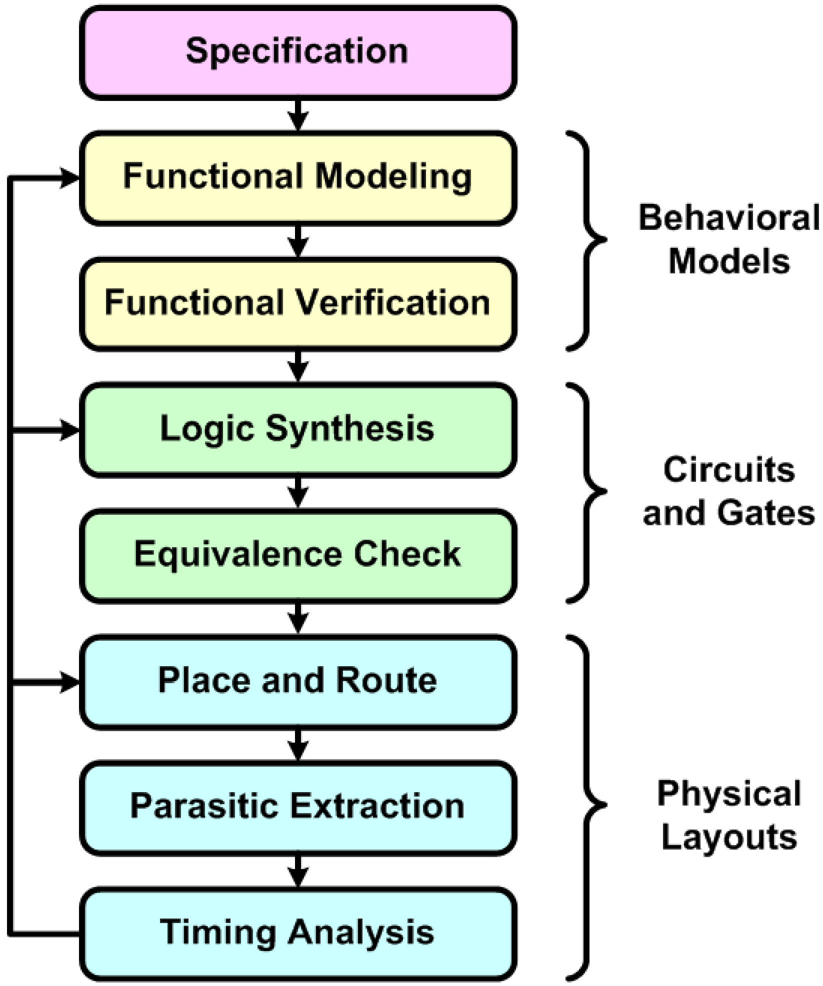
The Result



- Really big D and very little A

Digital Flow Starts with Models

- Digital flow is about turning models into reality
  - Functional models describe "what designers want"
  - Do models have correct functionalities?
  - Do circuits have same functionalities with models?
  - Are delay and power within acceptable bounds?
- Each question can be answered with help of automation tools

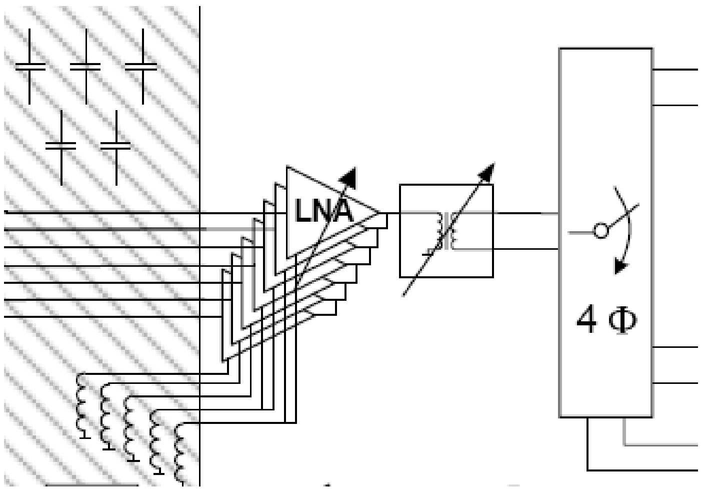


Models in Analog Design Flows

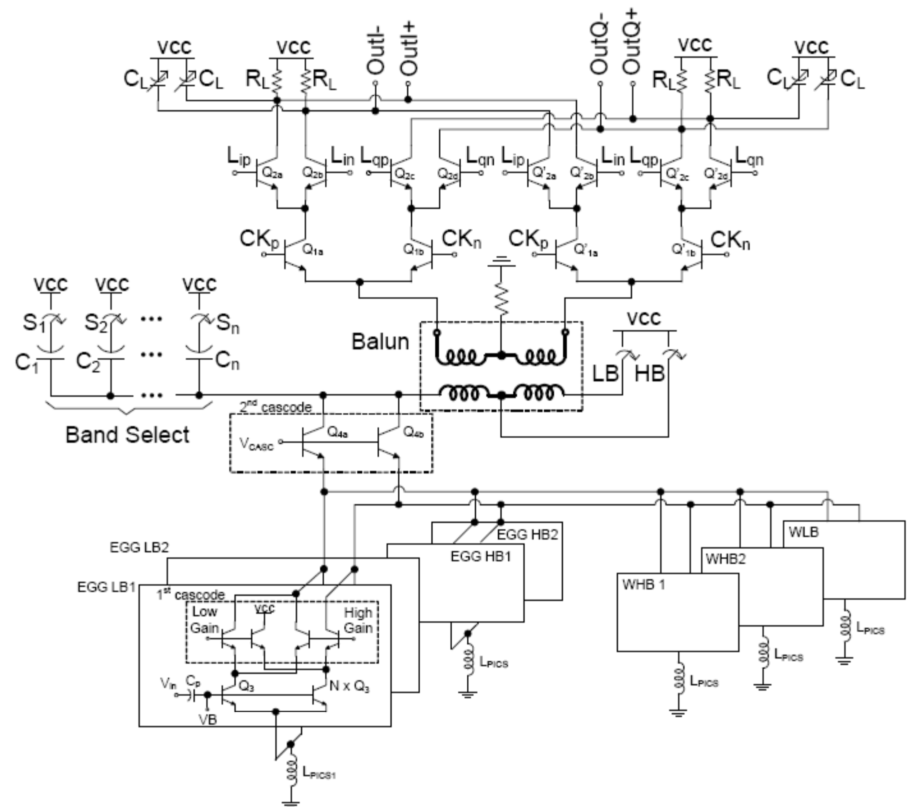
- Analog circuit designers want to use models mainly because they can speed up simulations
  - Aim is to abstract away details while preserving key behaviors of the circuits
  - Faster simulation at the system level
  - Hierarchical design flow
- Problem: lack of established flows with analog models
  - Q: how to create these models?
  - Q: how to verify these models?
  - The word “model” is perceived so differently between analog and digital designers

The Model Problem

- Which really matters



Model



Implementation

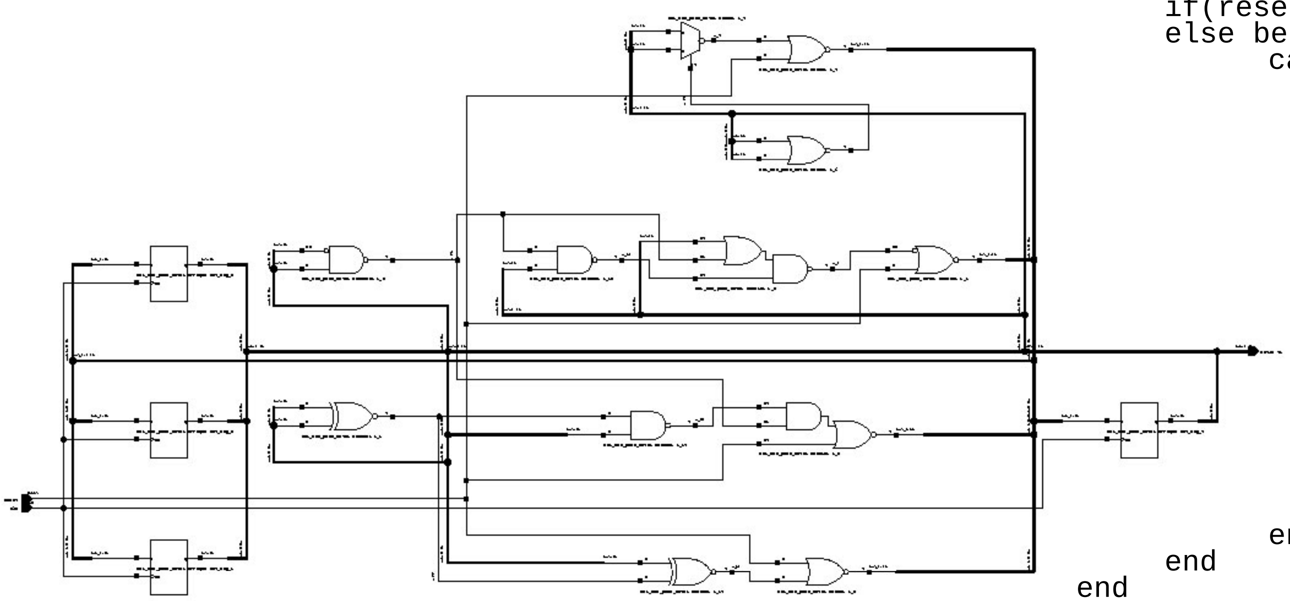
The Model Problem, cont'd

- Which really matters here?

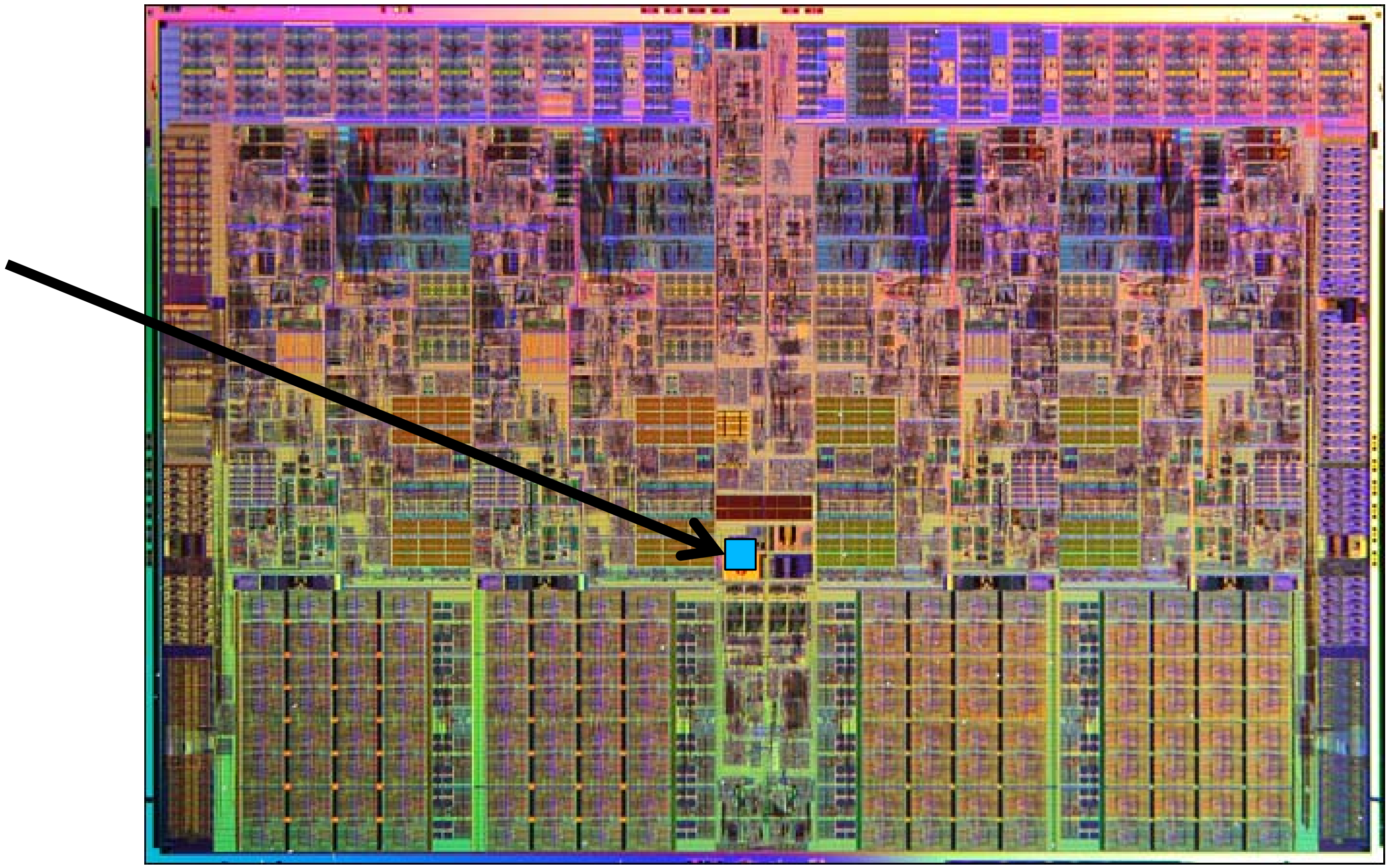
Model

```
module gray(clk, reset,out);

input clk, reset;
output [3:0] out;

wire clk,reset;

reg [3:0] out;

always @(posedge clk)
begin
     if(reset == 1) out = 4'b0000;
     else begin
           case(out)
                      4'b0000: out = 4'b0001;
                      4'b0001: out = 4'b0011;
                      4'b0010: out = 4'b0110;
                      4'b0011: out = 4'b0010;
                      4'b0100: out = 4'b1100;
                      4'b0101: out = 4'b0100;
                      4'b0110: out = 4'b0111;
                      4'b0111: out = 4'b0101;
                      4'b1000: out = 4'b0000;
                      4'b1001: out = 4'b1000;
                      4'b1010: out = 4'b1011;
                      4'b1011: out = 4'b1001;
                      4'b1100: out = 4'b1101;
                      4'b1101: out = 4'b1111;
                      4'b1110: out = 4'b1010;
                      4'b1111: out = 4'b1110;

           endcase
     end
end
endmodule
```

Implementation



But Who Controls Validation?

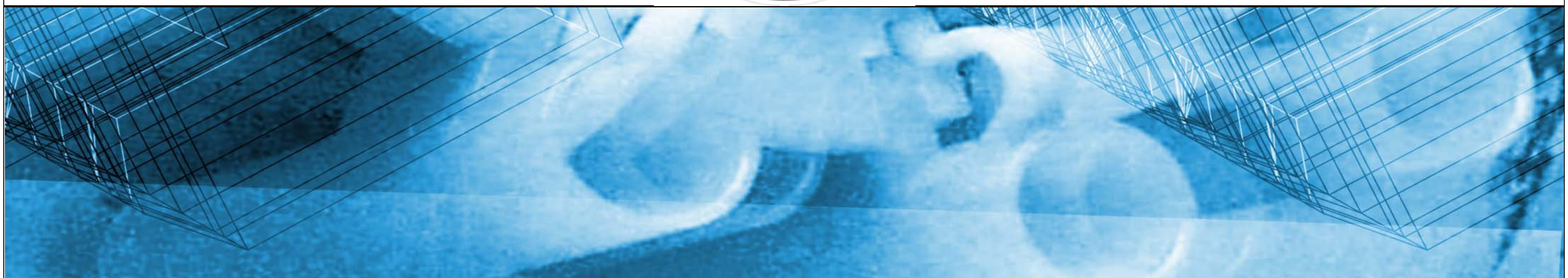


The Problem:

- Digital designers control validation
  - Model is a *golden reference*
  - They believe their “model” of the chip
- But for analog designers
  - Model is an *approximation*
  - They validate the circuits but not necessarily the models
- Leads to errors in mixed signal design
  - Bugs slip when digital designers *trust* analog models
  - Many bugs are trivial:
    - Mislabeled pins, inverted polarity, wrong bus ordering/encoding, missing connections, etc.
  - Even worse, bugs are repeated

Viewpoints on Analog Models

- At present, there is no consensus on how analog models should be created and validated
- Bottom-up approach ("MOR")
  - Models are extracted from the circuits with reduced order
  - Validity is guaranteed by the automatic MOR tool
  - But model is still an approximation (i.e. behavioral model); it may not reflect the design intent
- Top-down approach (digital-like)
  - Models are "functional models" – they describe the intent
  - Then the job is to see if circuits realize the models correctly
  - But, no established way to verify model-circuit equivalence



Automated Compact Dynamical Modeling: An Enabling Tool for Analog Designers

Luca Daniel, Massachusetts Institute of Technology

Bradley N. Bond, Coventor

The Typical Development Flow for Analog Systems

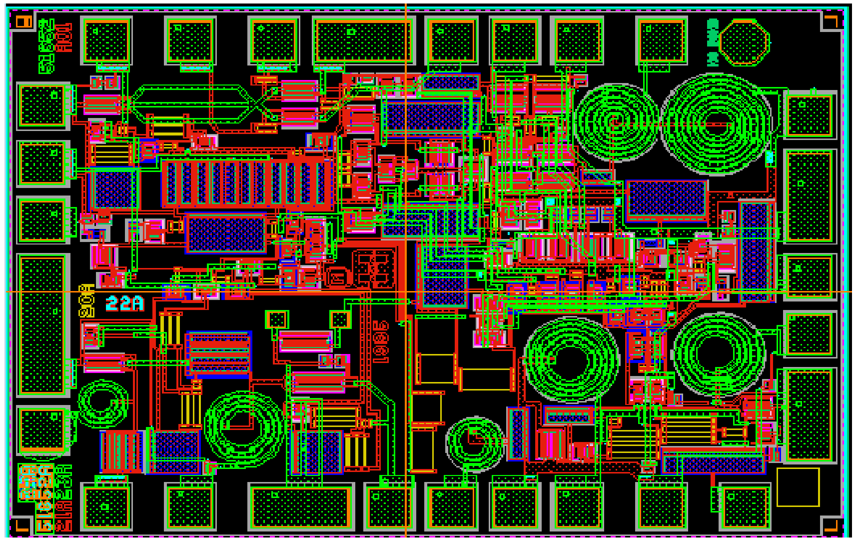
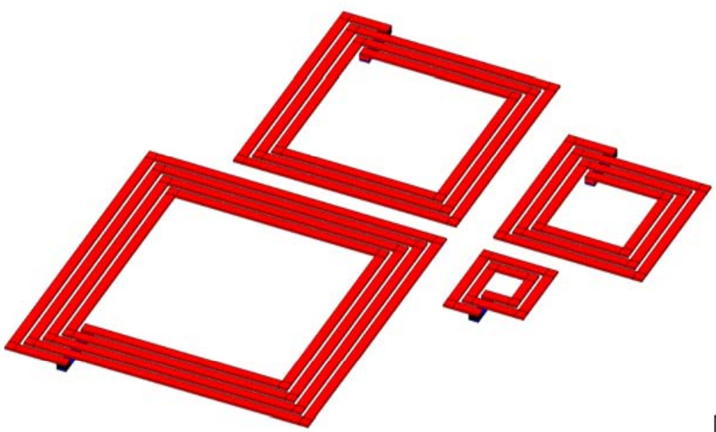
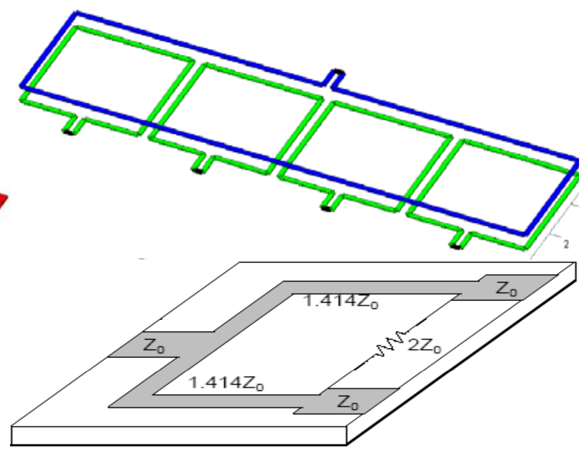


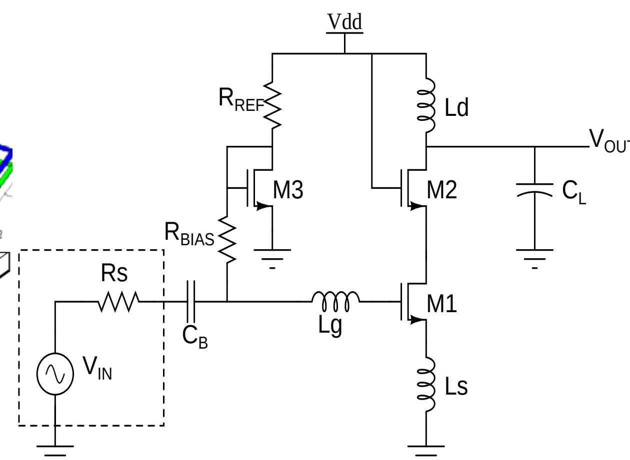
Image Harris semiconductor



RF Inductors

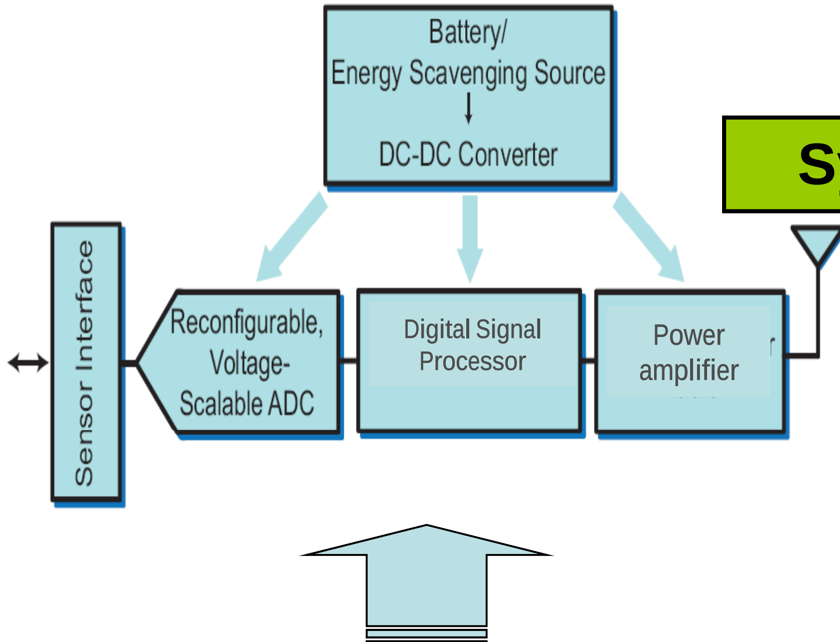


Power Combiners



Low Noise Amplifier

The Typical Development Flow for Analog Systems



System Architect

Will it work?
Is it optimal?

Need to run system level simulations

$$\nabla \times \mathbf{E} = -\mu \frac{d\mathbf{H}}{dt} \qquad \nabla \times \mathbf{H} = \varepsilon \frac{d\mathbf{E}}{dt} + \mathbf{J} \qquad C(v)\frac{dv}{dt} = -G(v) + Bv_{in}$$

Step 1: Run EM Field Solvers and Circuit Simulations

Will it work?

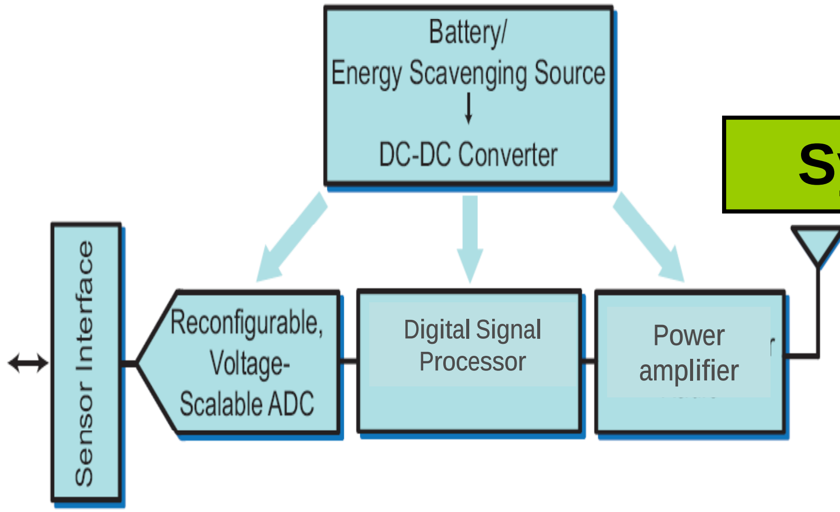
Will it work?

Will it work?

Passives + Circuit Designers

RF Inductors **Power Combiners** **Low Noise Amplifier**

The Typical Development Flow for Analog Systems



System Architect

Will it work?
Is it optimal?

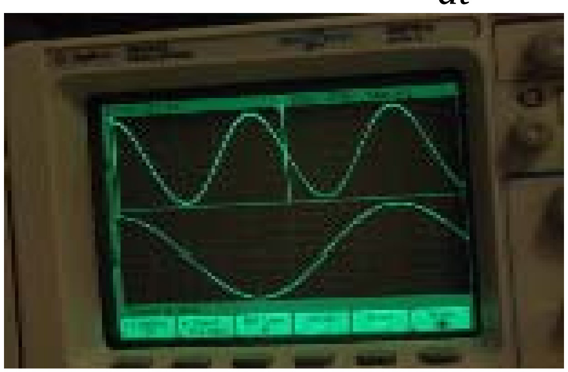
Need to run system level simulations

Step 2: Need techniques that generate compact dynamical models automatically from field solvers <u>AND</u> from device measurements

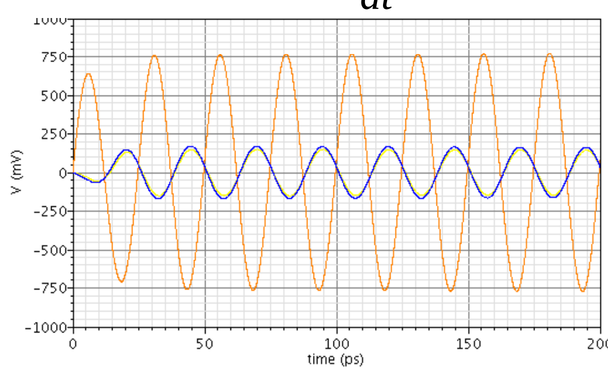
$$\nabla \times \mathbf{E} = -\mu \frac{d\mathbf{H}}{dt}$$

$$\nabla \times \mathbf{H} = \varepsilon \frac{d\mathbf{E}}{dt} + \mathbf{J}$$

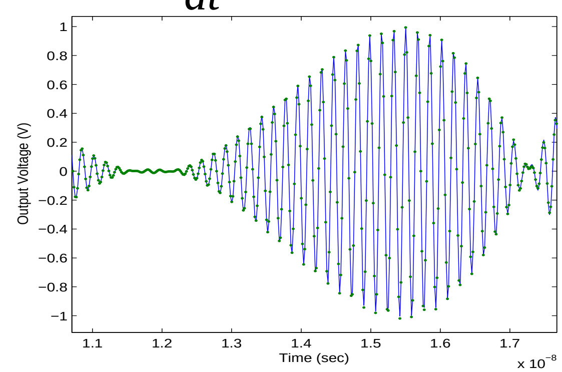
$$C(v)\frac{dv}{dt} = -G(v) + Bv_{in}$$



RF Inductors



Power Combiners



Low Noise Amplifier

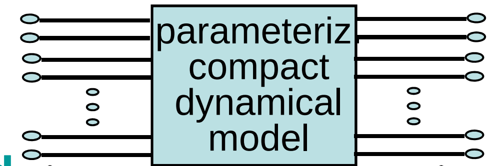
Modeling Requirements for Analog Systems

Models must be:

• dynamical	⇨	allow time domain/periodic simulation (e.g. distortion, spectral overgrowth).
• compact	⇨	run fast in system simulators.
• stable	⇨	allow stable component simulation
• passive	⇨	allow stable system level simulation
• parameterized and handle variations	⇨	allow robust design optimization
• able to handle linear components	⇨	e.g. for couple RF inductors, power combiners etc...
• able to handle non-linear components	⇨	e.g. entire power amplifier, or entire low noise amplifier
• able to account for non idealities:	⇨	allow realistic system performance eval.
• from field solvers	⇨	helps component prototyping stage and enables early system design
• from measurements	⇨	after component prototype available

Step 2: Automated Parameterized Compact Dynamical Modeling for LINEAR Systems

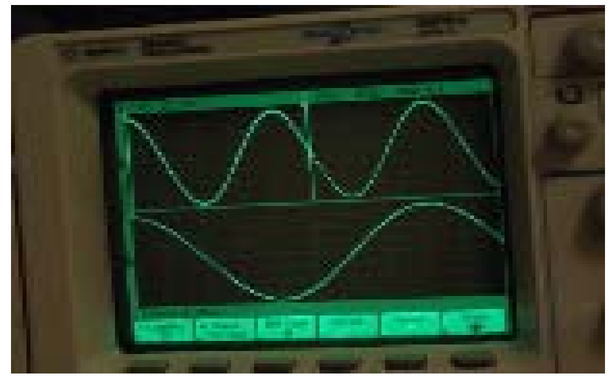
- automatically
- with field solver accuracy
- small (only 10-15 Eqns.)
- geometrically parameterized



Step 2. Model Order Reduction or System Identification

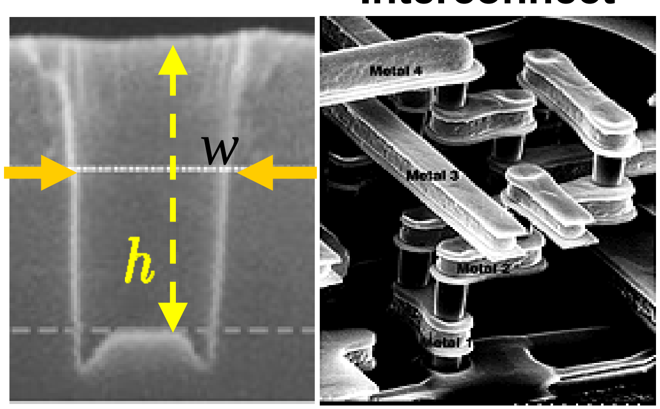
$$\frac{dx}{dt} = \left[A(w,h,d) \right] \left[x(t) \right] + \left[B \right] u(t)$$

(1 Million Eqns)

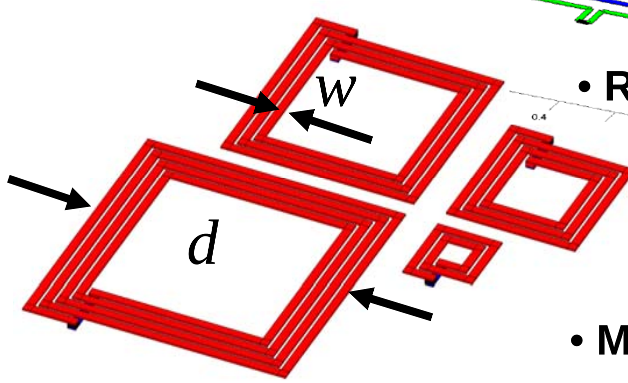


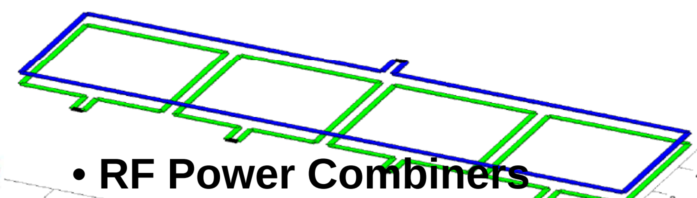
Step 1. from Field Solvers "guts" or Measurement Data

- Interconnect

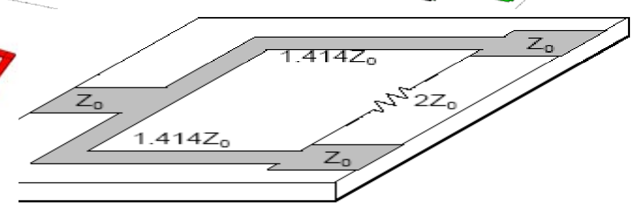


- RF inductors



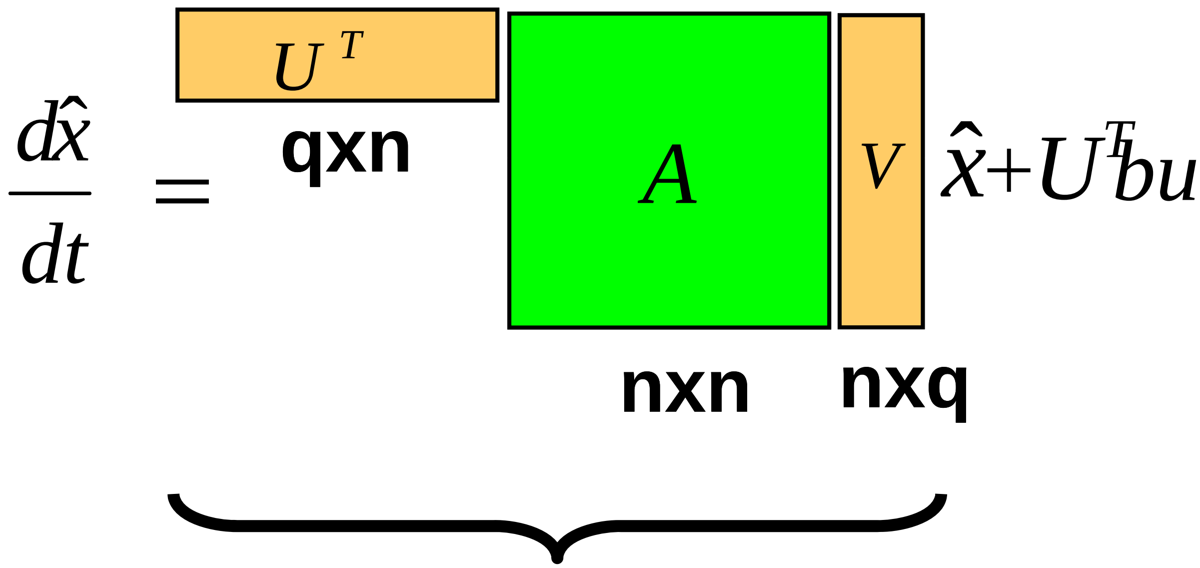


- RF Power Combiners



- Microwave Wilkinson Divider

The Standard Projection Framework (graphically)



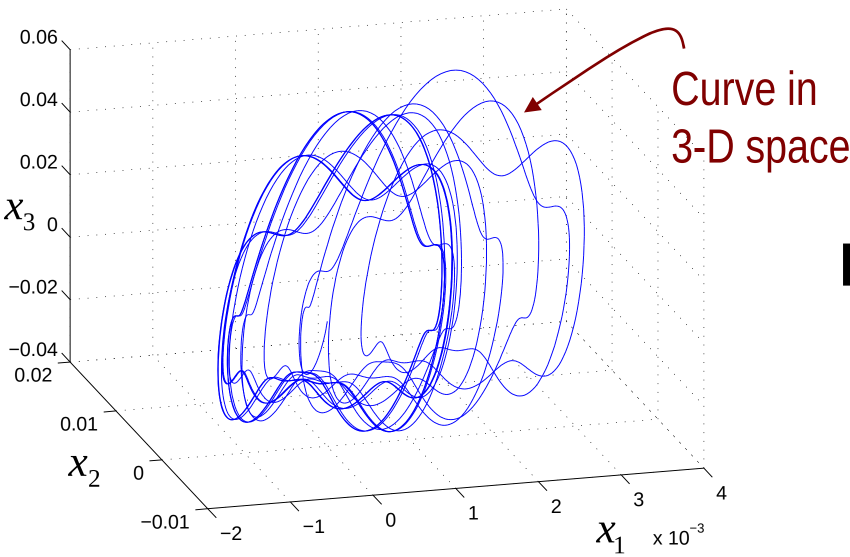
$$\frac{d\hat{x}}{dt} = \hat{A}\,\hat{x}(t) + \hat{b}u(t)$$

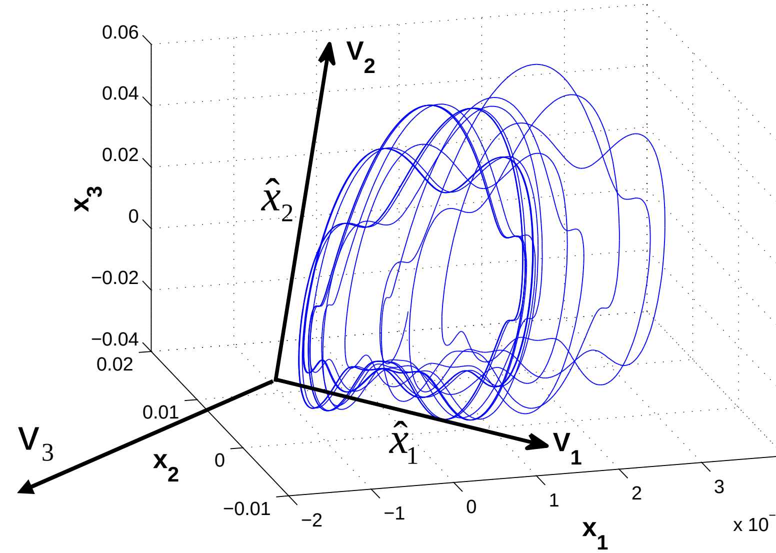
qxq

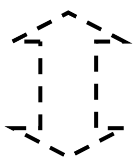
Key Question: how do you choose V and U?

How to choose V and U?

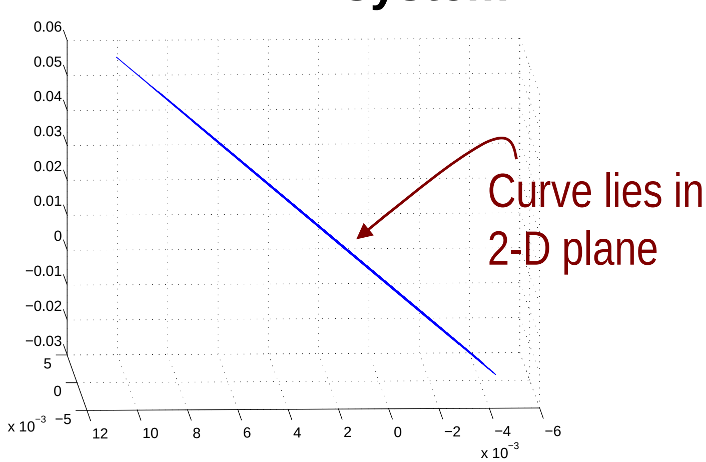
$$\dot{x} = Ax + Bu$$



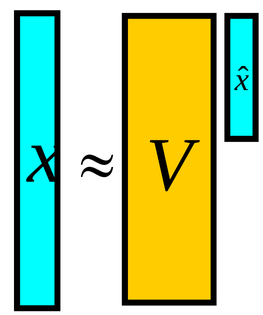




Rotate coordinate system



Projection: Rotation + Truncation



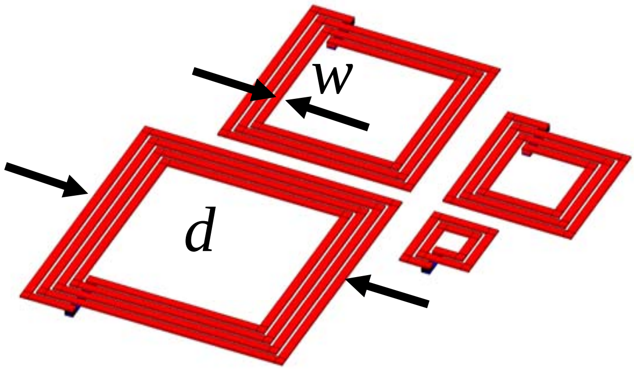
Step 2: Automated Compact Dynamical Modeling for LINEAR Systems

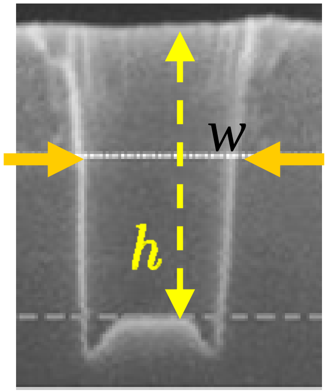
	Basic Technique	
Optimal but small systems	TBR 81 Hankel 84	**Control/Systems**
POD, KL, PCA, SVD	Wilcox Peraire91, PMTBR04	**Mechanical** **Aero/Astro** **Statistics, E.D.A.**
Moment, Matching	AWE90, PVL94	**E.D.A.** **Num Linear Algebra**

All Projection Frameworks with different ways of finding V and U

Step 2: Automated Compact Dynamical Modeling for LINEAR Systems

	Basic Technique	Parameters/ Variations
Optimal but small systems	TBR 81 Hankel 84	Heydari01
POD, KL, PCA, SVD	Wilcox Peraire91, PMTBR04	Phillips04
Moment, Matching	AWE90, PVL94	Weile99 one-param, Daniel04, Multi-param Moselhy10, Statistical

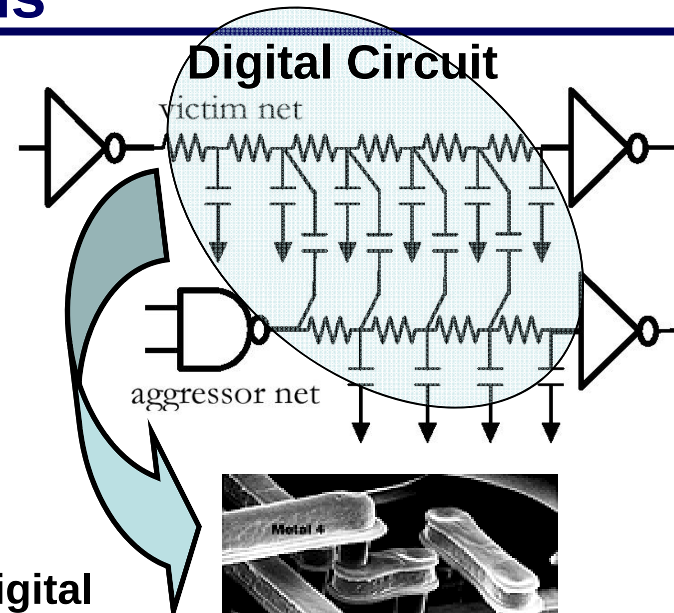




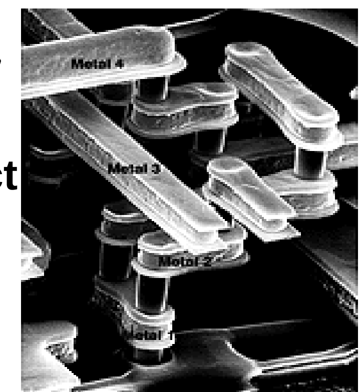
Step 2: Automated Compact Dynamical Modeling for LINEAR Systems

	Basic Technique	Parameters/ Variations	Stability/ Passivity
Optimal but small systems	TBR 81 Hankel 84	Heydari01	Phillips02
POD, KL, PCA, SVD	Wilcox Peraire91, PMTBR04	Phillips04	
Moment, Matching	AWE90, PVL94	Weile99 one-param, Daniel04, Multi-param Moselhy10, Statistical	PRIMA97 (only if A>0)

Great!



Digital Interconnect



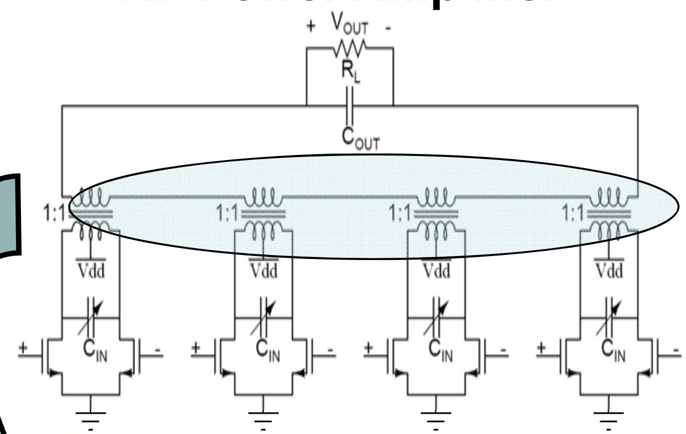
Field Solver/ Parasitic Extractor

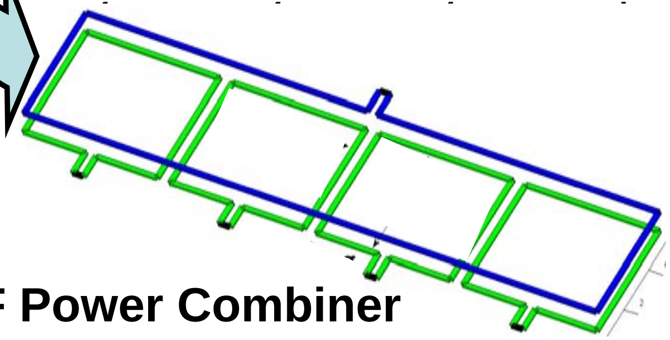
E>0, A>0 positive definite easily!

Step 2: Automated Compact Dynamical Modeling for LINEAR Systems

	Basic Technique	Parameters/ Variations	Stability/ Passivity
Optimal but small systems	TBR 81 Hankel 84	Heydari01	Phillips02
POD, KL, PCA, SVD	Wilcox Peraire91, PMTBR04	Phillips04	
Moment, Matching	AWE90, PVL94	Weile99 one-param, Daniel04, Multi-param Moselhy10, Statistical	PRIMA97 (only if A>0)

RF Power Amplifier





RF Power Combiner

Field Solver

- non-symmetric EM formulations
- discretization errors
- substrate green functions
- fullwave effects

A: NOT positive definite

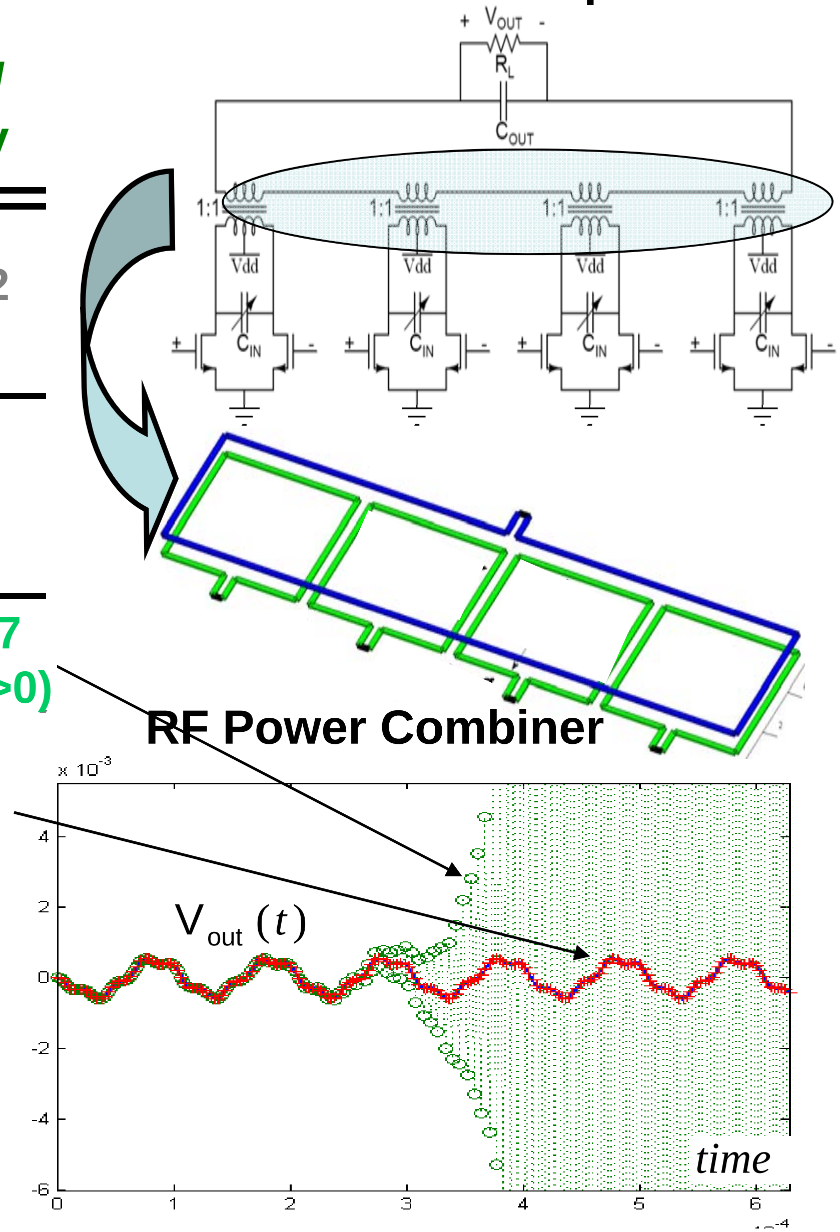
System simulation becomes unstable!

Step 2: Automated Compact Dynamical Modeling for LINEAR Systems

	Basic Technique	Parameters/ Variations	Stability/ Passivity
Optimal but small systems	TBR 81 Hankel 84	Heydari01	Phillips02
POD, KL, PCA, SVD	Wilcox Peraire91, PMTBR04	Phillips04	Bond08 any A
Moment, Matching	AWE90, PVL94	Weile99 one-param, Daniel04, Multi-param Moselhy10, Statistical	PRIMA97 (only if A>0) Bond08 any A

In analog, need stability/passivity for any A

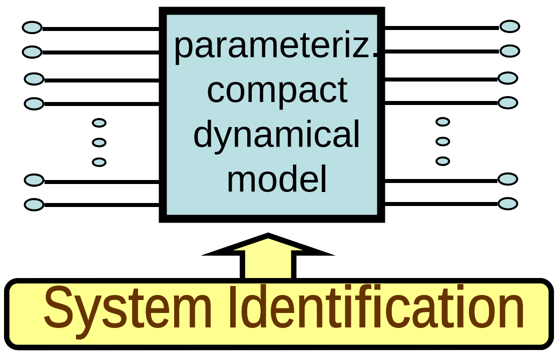
RF Power Amplifier

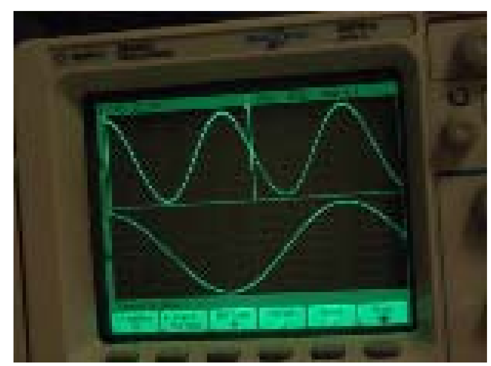


Step 2: Automated Compact Dynamical Modeling for LINEAR Systems

	Basic Technique	Parameters/ Variations	Stability/ Passivity
Optimal but small systems	TBR 81 Hankel 84	Heydari01	Phillips02
POD, KL, PCA, SVD	Wilcox Peraire91, PMTBR04	Phillips04	Bond08 any A
Moment, Matching	AWE90, PVL94	Weile99 one-param, Daniel04, Multi-param Moselhy10, Statistical	PRIMA97 (only if A>0) Bond08 any A
Fitting, Optimiz Based, System ID	Gustavs99	Dhaene01	

In analog, need to construct models also from measurements





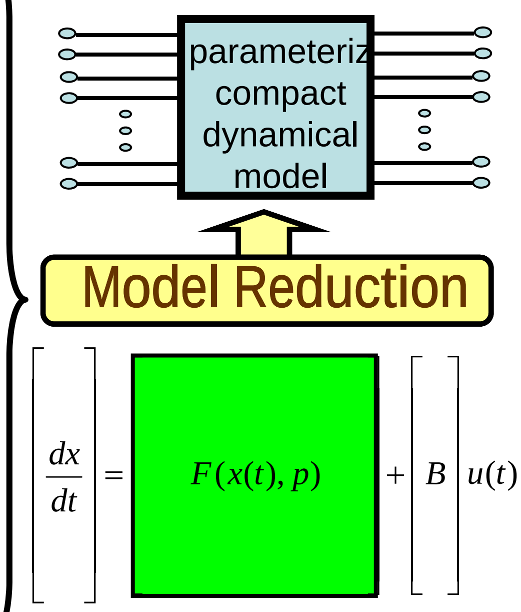
Step 2: Automated Compact Dynamical Modeling for LINEAR Systems

	Basic Technique	Parameters/ Variations	Stability/ Passivity
Optimal but small systems	TBR 81 Hankel 84	Heydari01	Phillips02
POD, KL, PCA, SVD	Wilcox Peraire91, PMTBR04	Phillips04	Bond08 any A
Moment, Matching	AWE90, PVL94	Weile99 one-param, Daniel04, Multi-param Moselhy10, Statistical	PRIMA97 (only if A>0) Bond08 any A
Fitting, Optimiz Based, System ID	Gustavs99	Dhaene01	Coelho01, Talocia03, Mahmood10 (multiport)
	Sou08 Quasi-Convex Optimiz.		

Warning: basic vector fitting gives no stability/passivity

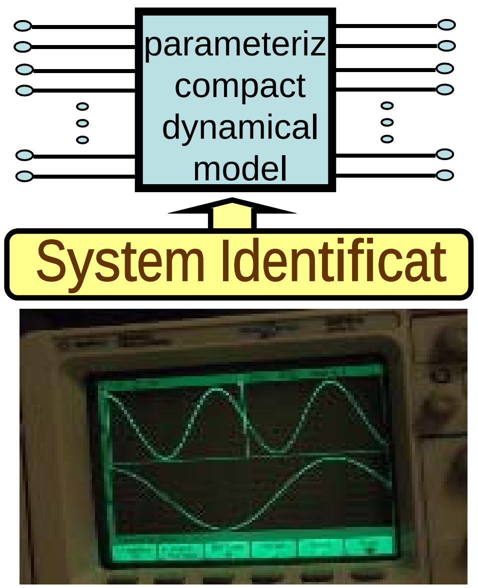
Step 2: Automated Parameterized Compact Dynamical Modeling for NON-LINEAR Systems

	Linear Systems			Non-Linear Systems
	Basic Technique	Parameter/ Variations	Stability/ Passivity	Basic Technique
Optimal (small system)	TBR 81 Hankel 84	Heydari01	Phillips02	TBR-TPWL Vasilyev03
POD, KVL, PCA, SVD	Wilcox Peraire91, PMTBR04	Phillips04	Bond08 any A	Wilcox Peraire99
Moment, Matching	AWE90, PVL94	Weile99 one-param, Daniel04, Multi-param Moselhy10, Statistical	PRIMA97 (only if A>0) Bond08 any A	Quadratic Chen00, TPWL01, PWP03, NORM03
Fitting, Optimiz Based, System ID	Gustavs99	Dhaene01	Coelho01, Talocia03, Mahmood10 (multiport)	
	Sou08 Quasi-Convex Optimiz.			



Step 2: Automated Parameterized Compact Dynamical Modeling for NON-LINEAR Systems

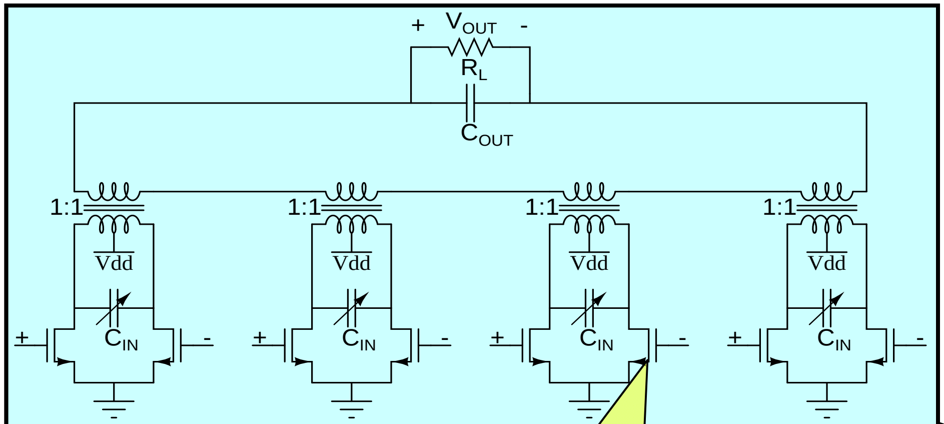
	Linear Systems			Non-Linear Systems
	Basic Technique	**Parameter/ Variations**	**Stability/ Passivity**	**Basic Technique**
Optimal (small system)	TBR 81 Hankel 84	Heydari01	Phillips02	TBR-TPWL Vasilyev03
POD, KVL, PCA, SVD	Wilcox Peraire91, PMTBR04	Phillips04	Bond08 any A	Wilcox Peraire99
Moment, Matching	AWE90, PVL94	Weile99 one-param, Daniel04, Multi-param Moselhy10, Statistical	PRIMA97 (only if A>0) Bond08 any A	Quadratic Chen00, TPWL01, PWP03, NORM03
Fitting, Optimiz Based, System ID	Gustavs99	Dhaene01	Coelho01, Talocia03, Mahmood10 (multiport)	Volterra91, Haber99, Wiener- Hammerst99
	Sou08 Quasi-Convex Optimiz.			

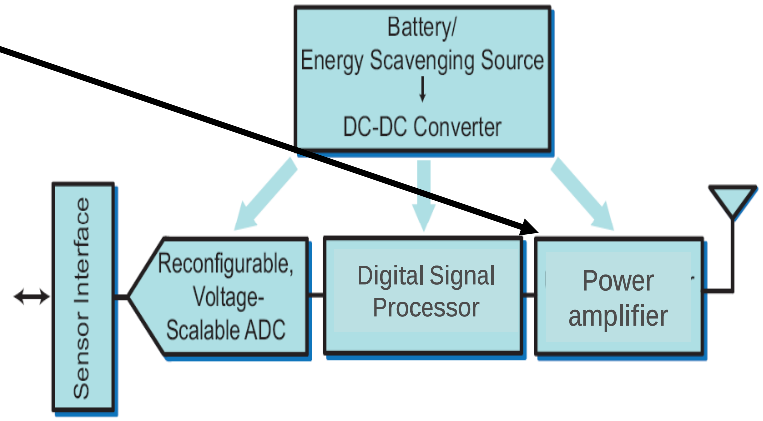


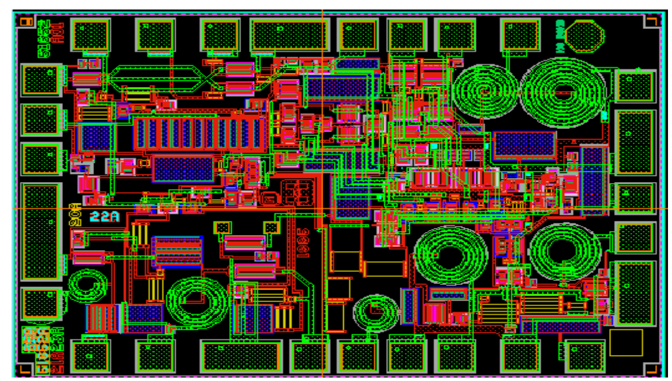
Step 2: Automated Parameterized Compact Dynamical Modeling for NON-LINEAR Systems

	Linear Systems			Non-Linear Systems		
	Basic Technique	Parameter/ Variations	Stability/ Passivity	Basic Technique	Parameter/ Variations	Stability/ Passivity
Optimal (small system)	TBR 81 Hankel 84	Heydari01	Phillips02	TBR-TPWL Vasilyev03		
POD, KVL, PCA, SVD	Wilcox Peraire91, PMTBR04	Phillips04	Bond08 any A	Wilcox Peraire99	Parameter-TPWL Bond07	Stable-TPWL Bond07
Moment, Matching	AWE90, PVL94	Weile99 one-param, Daniel04, Multi-param Moselhy10, Statistical	PRIMA97 (only if A>0) Bond08 any A	Quadratic Chen00, TPWL01, PWP03, NORM03		
Fitting, Optimiz Based, System ID	Gustavs99 Sou08 Quasi-Convex Optimiz.	Dhaene01	Coelho01, Talocia03, Mahmood10 (multiport)	Volterra91, Haber99, Wiener-Hammerst99	Wiener-Hamm[Suo08], Incremental stability [Bond10]	

Why isn't Compact Dynamical Modeling already a Fully Working Solution for Analog Designers?







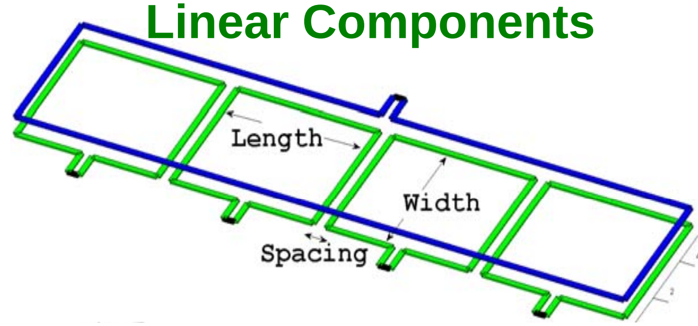
Non-Linear Systems: Need PASSIVITY otherwise cannot connect them at system level

Why isn't Compact Dynamical Modeling already a Fully Working Solution for Analog Designers?

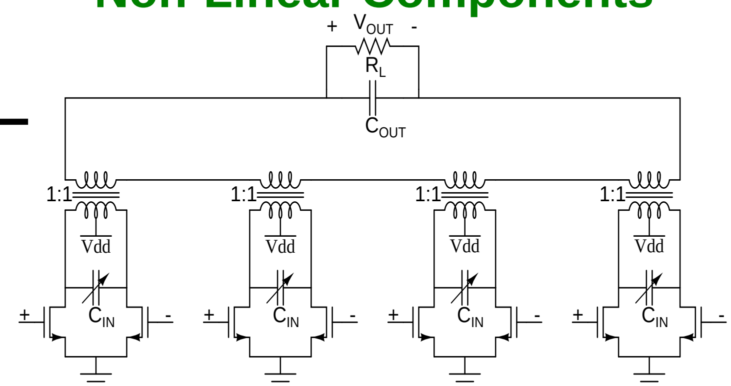
- Stability: needed to simulate single component
- Passivity: needed to connect components (system level simulation)
- Param + Stability: needed to optimize component
- Param + Passivity: needed to optimize system

		Stability	Passivity	Param + Stability	Param + Passivity
Linear Components	From Field Solvers "Guts"	2008	2008	2008	✗
	From Commercial Field Solvers or Measurem.	2003	2003	2010	✗
Non-Linear Components	From Field Solvers "Guts"	2007	✗	2007	✗
	From Commercial Field Solvers or Measurem.	2010	✗	2010	✗

Linear Components



Non-Linear Components



The Analog Model Crisis – How Can We Solve it?

The Bottom-Up approach (e.g. automatic generation of Parameterized Compact Dynamical Models):

1. **must contribute to the solution,**
   - **can propagate to the higher levels the effects of non-idealities**
   - **allows to build automatically flexible libraries to be used in many styles of design methodologies**

2. **but has not reached its maturity point yet for analog systems.**
   - **still needs a-priori passivity guarantees for any parameter**

3. **...although some significant breakthroughs are beginning to happen since the last 3-4 years**

Designer's Guide Consulting

Analog Verification

Model-Based Analog Functional Verification

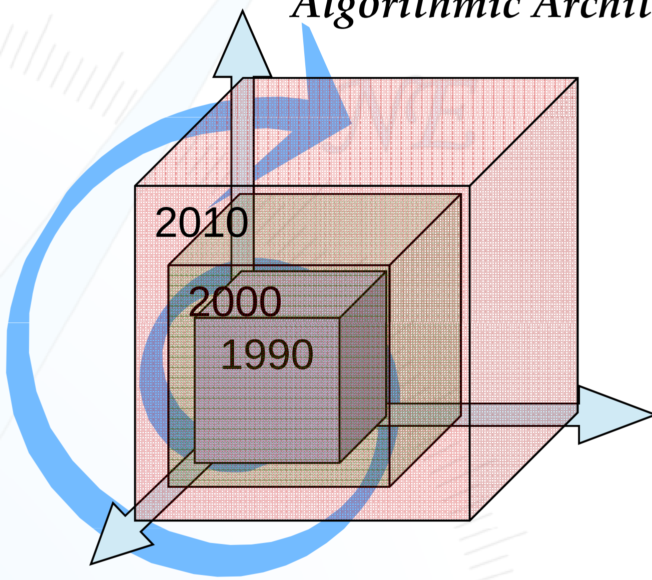
Ken Kundert

Henry Chang

Designs They Are A-Changin'

Bob Dylan, 1964

The Complexity of Design is Growing Rapidly



Algorithmic Architectures

ΔΣ architectures
Auto calibration
Adaptive filtering
Etc.

Modes & Settings

Power modes
Digital trimming
Multiple standards
Etc.

Size

> 100K transistors

In Multiple Dimensions!

Modes Aplenty, Modes Galore

Move to CMOS has resulted in …

- Many more modes
- Many more settings

This has greatly multiplied testing requirements

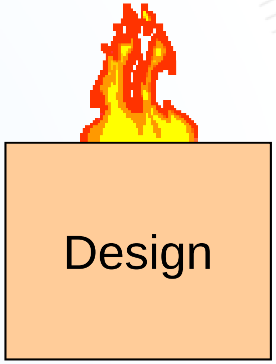
- Each represents a hiding place for errors

Makes analog verification increasingly like digital

- Must test every mode and setting
- Need rapid functional verification

Functional Errors

- Functional errors are often very simple errors
  - Inverted signals
  - Corrupt logic
  - Flipped busses
  - Unaccounted for dependencies (chicken/egg problem)
  - Communication errors
- But are generally catastrophic



The Three Basic Issues

- Detailed verification only performed at block level
  - All requires signals are assumed to be present
  - Assumptions on inter-block dependencies never verified
- Verification on most settings never performed
  - Only typical or min/max settings
  - Any control logic that supports untested mode or setting could contain hidden error
- No analog-digital co-verification

The Answer

- Functional verification with …
  - Model-based verification
    - Dramatically accelerates the simulation
    - Moves it earlier in design cycle
  - Exhaustive regression testing
    - Check every mode and every setting
    - Automated pass/fail tests (self-checking tests)

Why Functional Verification?

- Designers focus on block-level performance
  - It's largely covered
- Designers generally only verify a few modes and settings
  - Rest are assumed to work
- Functional errors are often the most devastating
- It is possible and cost effective

Exhaustive Testing

- When confronted with a large number of settings, designers will usually test a typical setting and the extreme settings.
  - But what if two LSB lines are swapped?
- Typical & worst case testing of settings is not enough
- Must systematically check functionality for every mode and setting

Regression Testing

- Today, most designers test functionality at most once, when first designed
  - Redesign can break existing functionality
- In regression testing, we test all functionality on every design change
  - Greatly reduces risks of redesign

AMS Simulation

- Verilog-AMS
  - Combines logic and circuit simulation
  - Combines Verilog, Verilog-A, SPICE, plus more
- Required when testing model against circuit
- May also be used to represent model
  - Though often models are pure Verilog

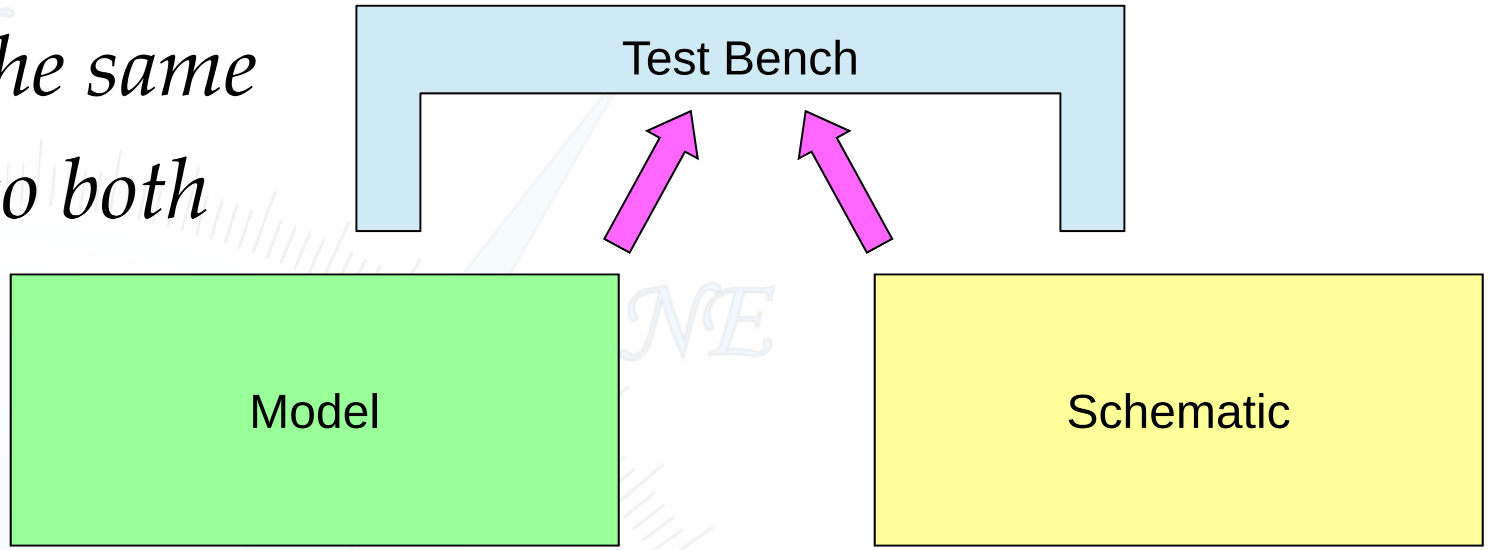
Model-Based Verification

- Replace transistor-level circuit with model
  - Dramatically accelerates simulation
  - Verification can start before schematics are available
  - Model can be used for system level verification, test development, etc.

But how does one assure model matches implementation?

Model Verification

Apply the same tests to both



- Model must be 'pin accurate'
- Testbench must be comprehensive
- Model can be developed before schematic
- Generally takes too long to simulate with full schematic at top-level

This is Analog Verification

- Exhaustive regression testing
- Traceable to transistor level
- Verifies both models and circuits
  - Test benches verify behavior of models
  - Methodology assures models are consistent with circuit

We can now imagine a future where we are surprised when an analog chip does not function the first time.

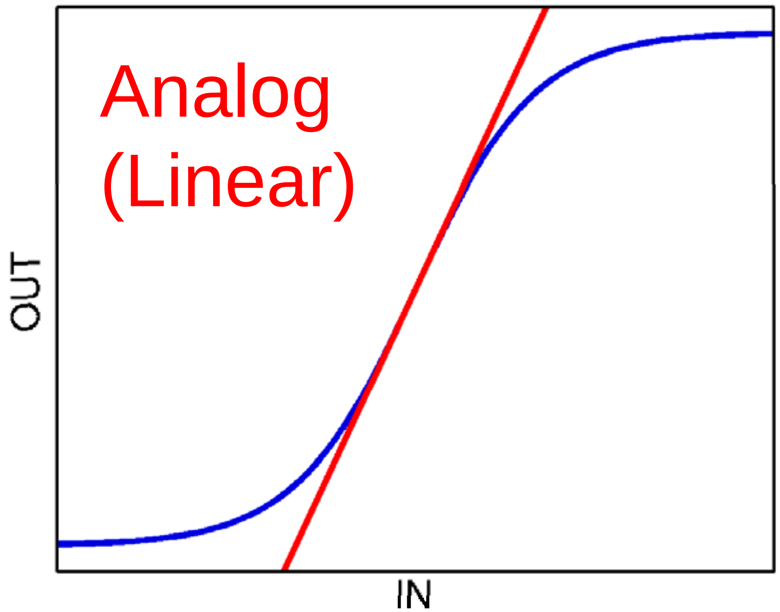
An Efficient Test Vector Generation for Checking Analog/Mixed-Signal Functional Models

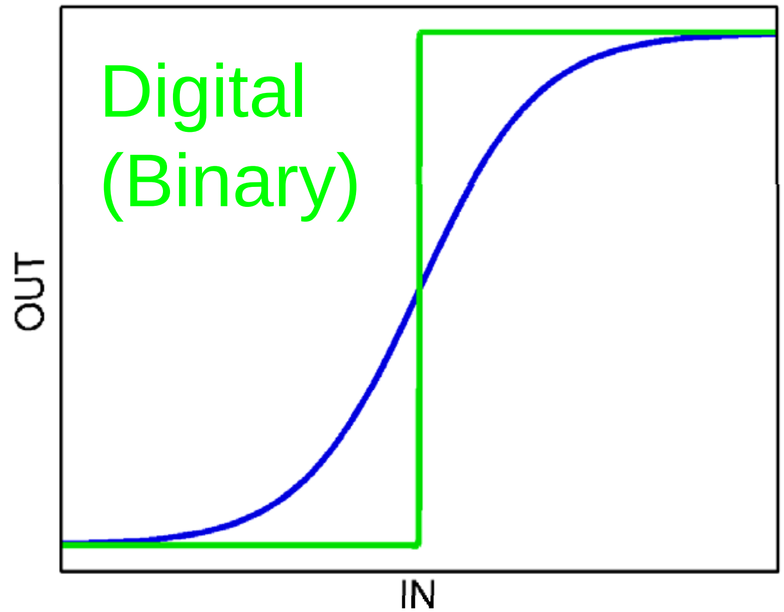
Byong Chan Lim[1], Jaeha Kim[2], Mark A. Horowitz[1]
[1]Stanford University, [2]Seoul National University
June 17, 2010

Remember This Slide?

- What was the key difference between A and D?

What do you see in this picture?



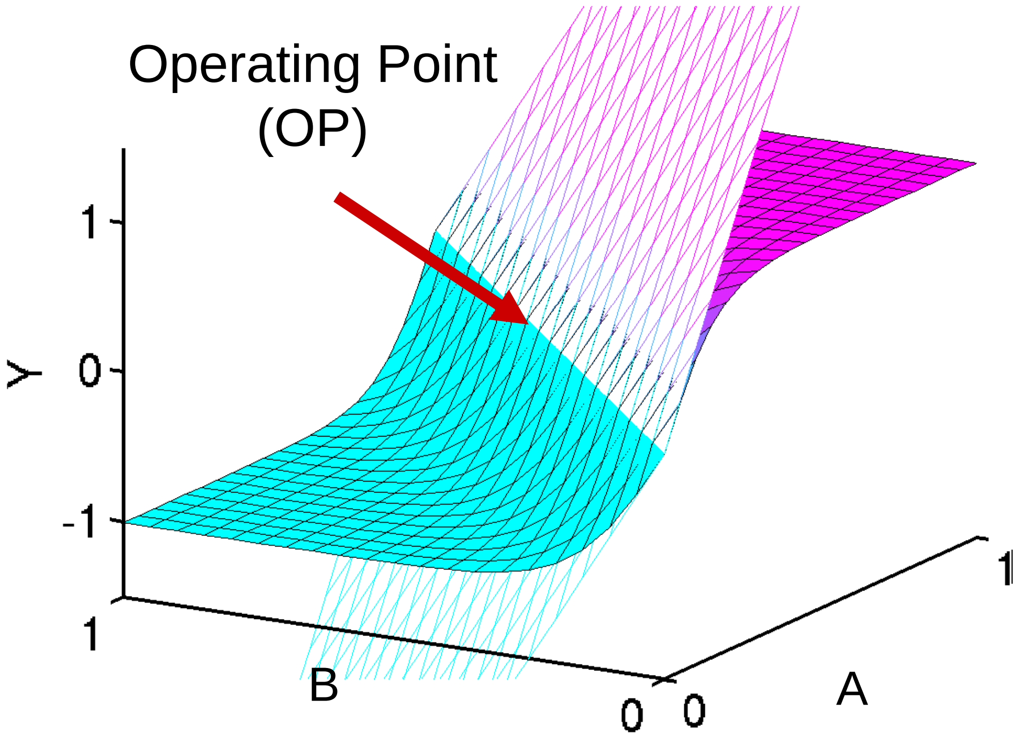


The Missing Piece in Analog

- Digital tools leverage "abstraction" effectively
  - Digital abstraction: Boolean (value), synchronous (time)
  - Leverage abstractions to:
    - Check circuits, measure coverage, check equivalence, etc.
  - Designers don't just rely on fast circuit simulators

- Analog tools do not
  - No notion of analog abstraction; focus mainly on fast simulation with accurate device models
  - Designer think *faster* SPICE is the answer; but it will never be fast enough

Analog Abstraction: Linear System



- Design intent is to use the linear region around the OP
- The ideal circuit has linear I/O relationship $\Delta Y = \alpha \cdot \Delta A + \beta \cdot \Delta B$
- In general, it's a linear dynamical system

- If all analog circuits have a *linear system* in mind, what is the proper way to leverage it in validating models?

Validating Analog Models

- Ideally, we'd like to have a checker that validates the equivalence between the analog circuit and its model
  - Similar to the equivalence checkers in digital
  - A modest, starting goal is to verify the *I/O consistency* first (i.e. whether the components are hooked up correctly)

- I/O consistency check:
  - Validates if each I/O port of the model has the same *functionality* with the corresponding port in the circuit
  - For analog, the functionality of an I/O port is determined by its role for the underlying *linear (or weakly nonlinear) system*
  - Linearity in their characteristics enables efficient validation

The Power of the Linear Abstraction

- As Boolean abstraction did for digital, the linear abstraction greatly simplifies analog verification

- The key is that superposition holds

$$y = \sum_i \alpha_i \cdot x_i \qquad \text{(superposition)}$$

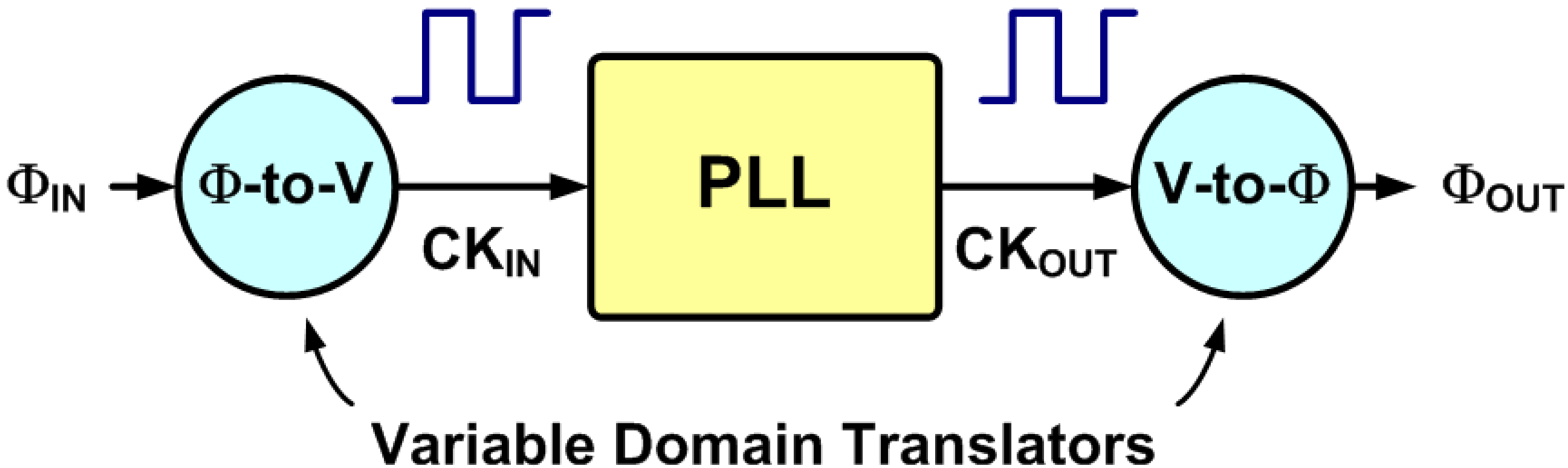
- This means generating input vectors is easy
  - Output is the sum of the change from each input
  - The output surface is smooth
    - Opposite of a digital system

Dealing with Non-Linear, Linear Circuits

- No real circuit is linear
  - But that does not mean it doesn't have a linear intent
  - Can we describe the circuit by its approximate linear function
    - And its deviation from that function?
    - Weakly non-linear function
- Two major types of non-linearity
  - Linear in a different domain than V and I
  - Controllable systems
    - Can control gain / frequency of linear system
- Both of these are easily handled in this framework

Variable Domain Transformation

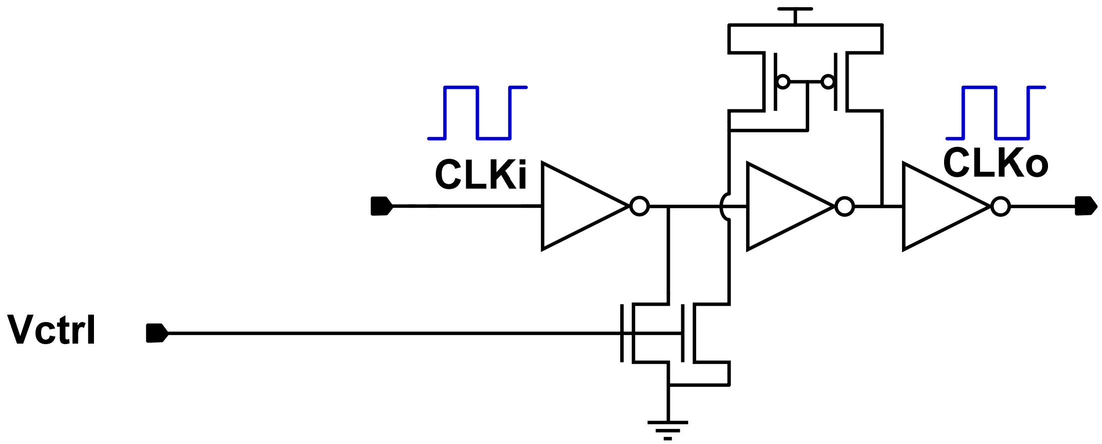
- PLL example:
  - PLL is a strongly nonlinear system in V/I but
  - A linear system in *phase* domain



Ref: J. Kim, et al., "Variable Domain Transformation for Linear PAC Analysis of Mixed-Signal Systems," ICCAD'07.

Variable Domain Translation: Example

- Duty-Cycle Adjuster

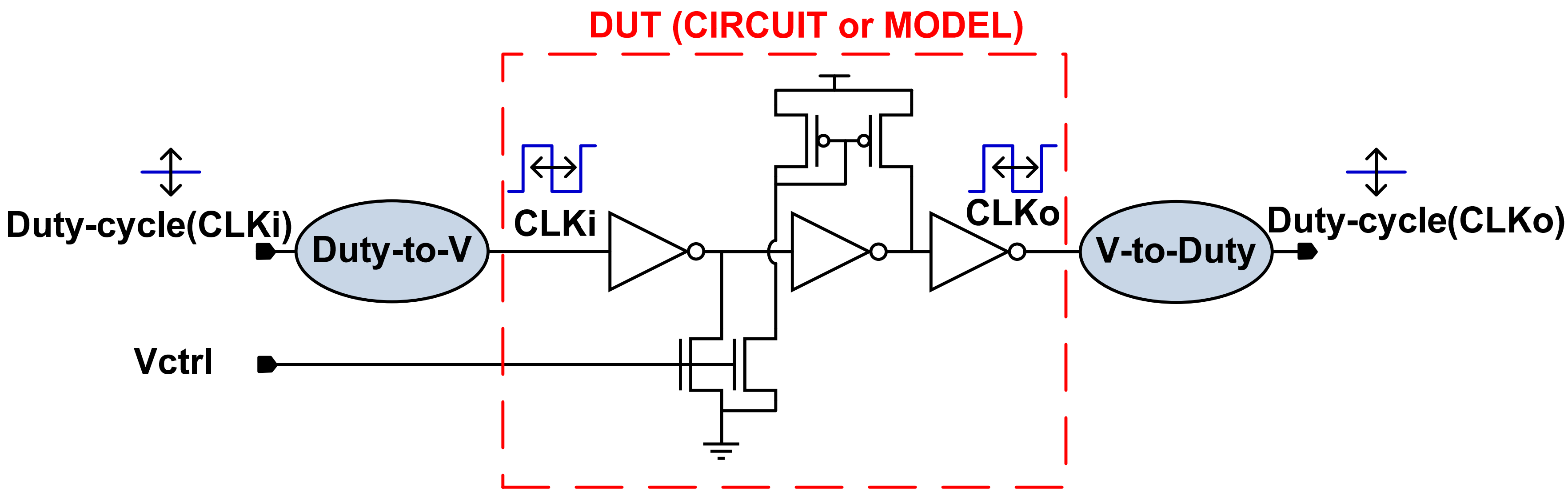


It's strongly non-linear !

CLKo = *f*(*CLKi*, *Vctrl*) = ?

Variable Domain Translation: Example

- Duty-Cycle Adjuster

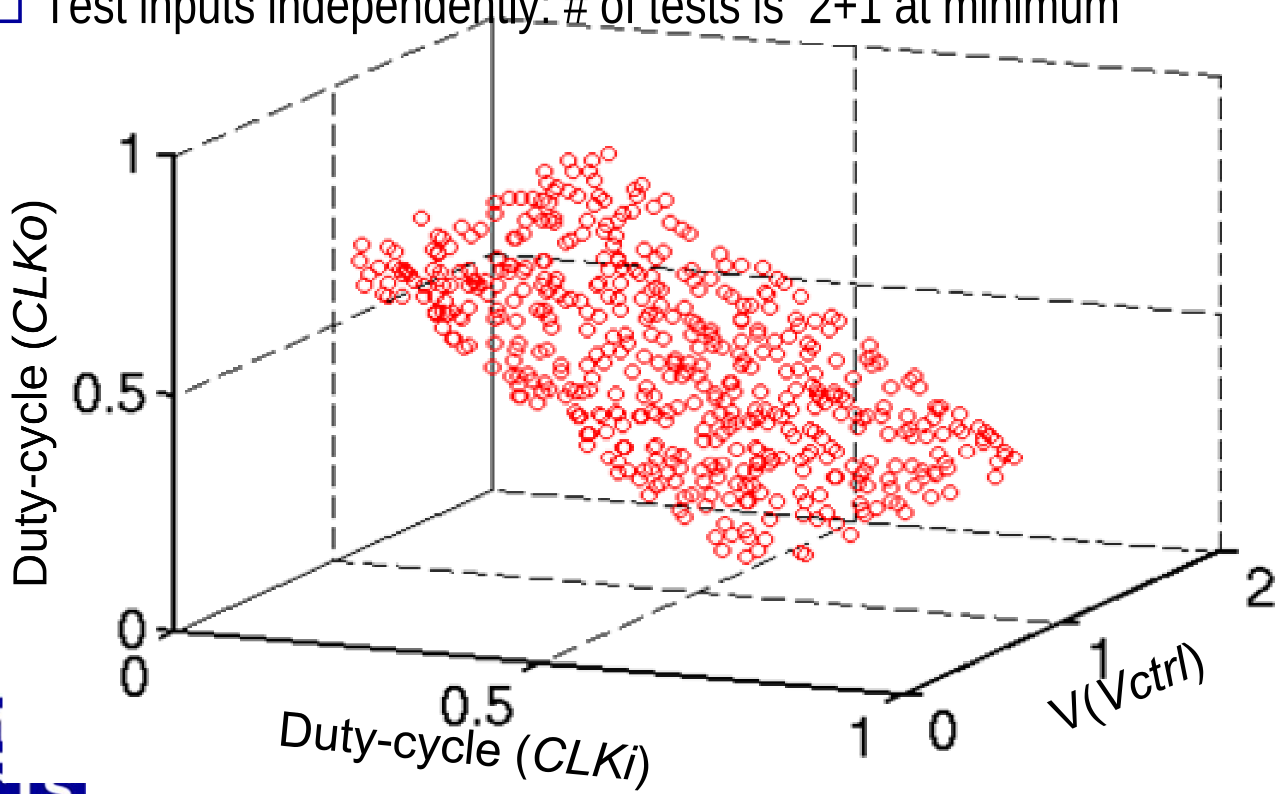


Design Intent is *Linear in Duty-cycle domain !*

$$\text{Duty}(CLKo) = \alpha \cdot \text{Duty}(CLKi) + \beta \cdot V(Vctrl)$$

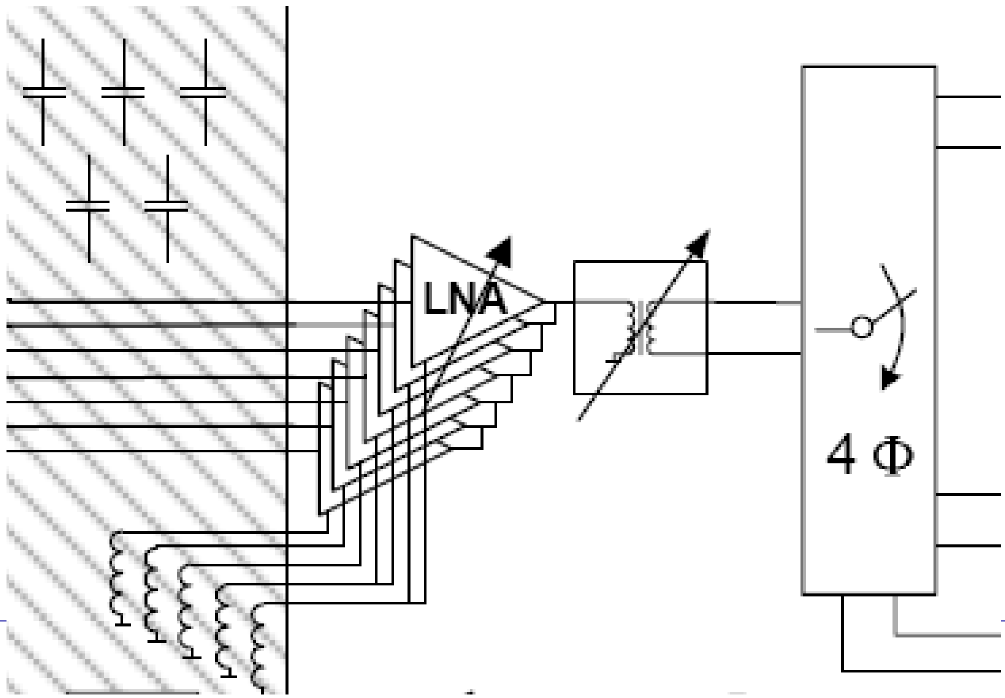
Variable Domain Translation: Example

- The response surface is hyper-plane in *duty-cycle* domain
- Linearity holds
  - Gain matrix comparison shows the equivalence
  - Test inputs independently: # of tests is 2+1 at minimum



Controlled Linear System

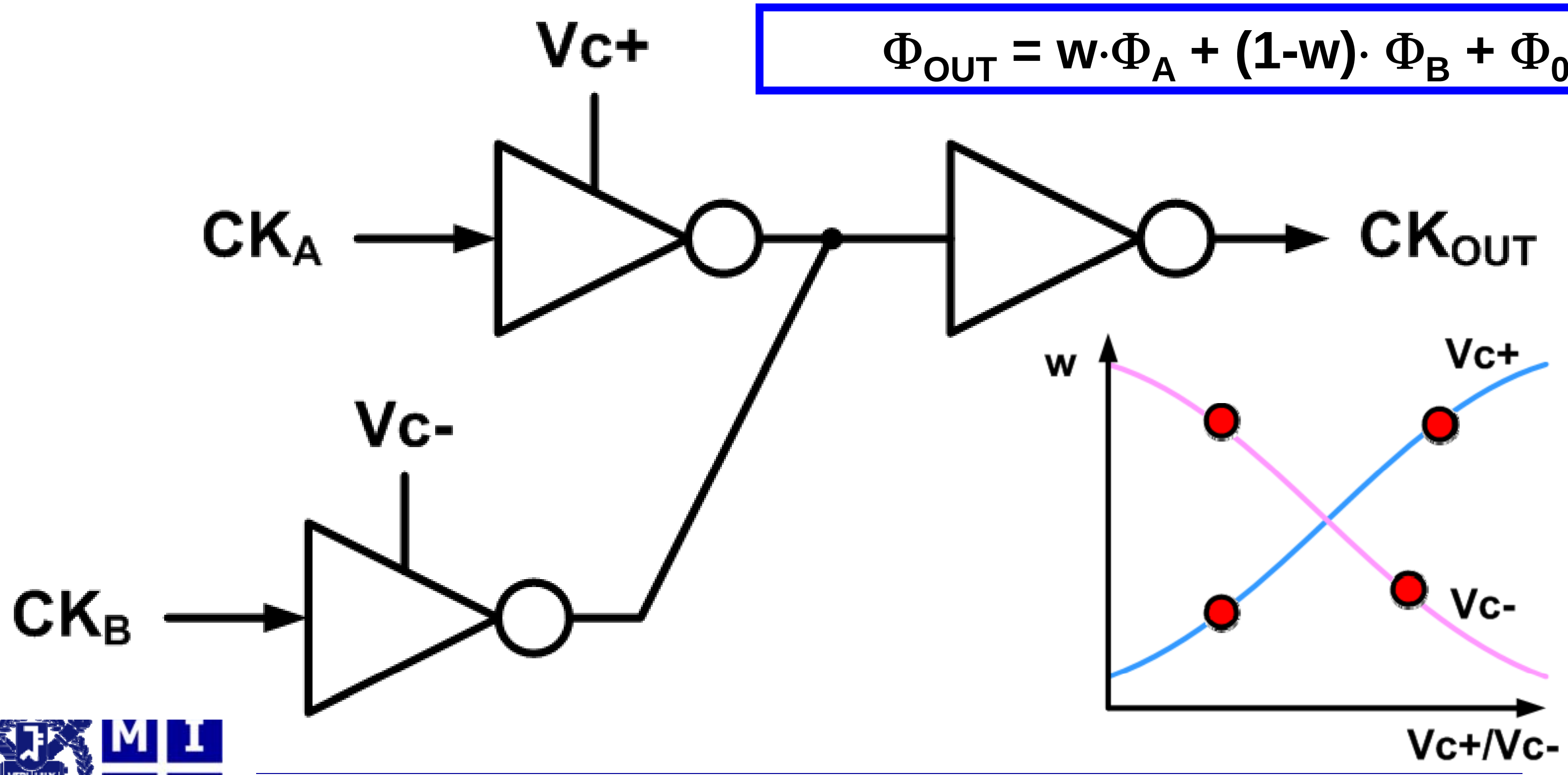
- Many systems have control inputs
  - Inputs that change the system response
- We reason about these systems
  - As two coupled systems
  - So we model them that way



Phase Interpolator with Analog Control

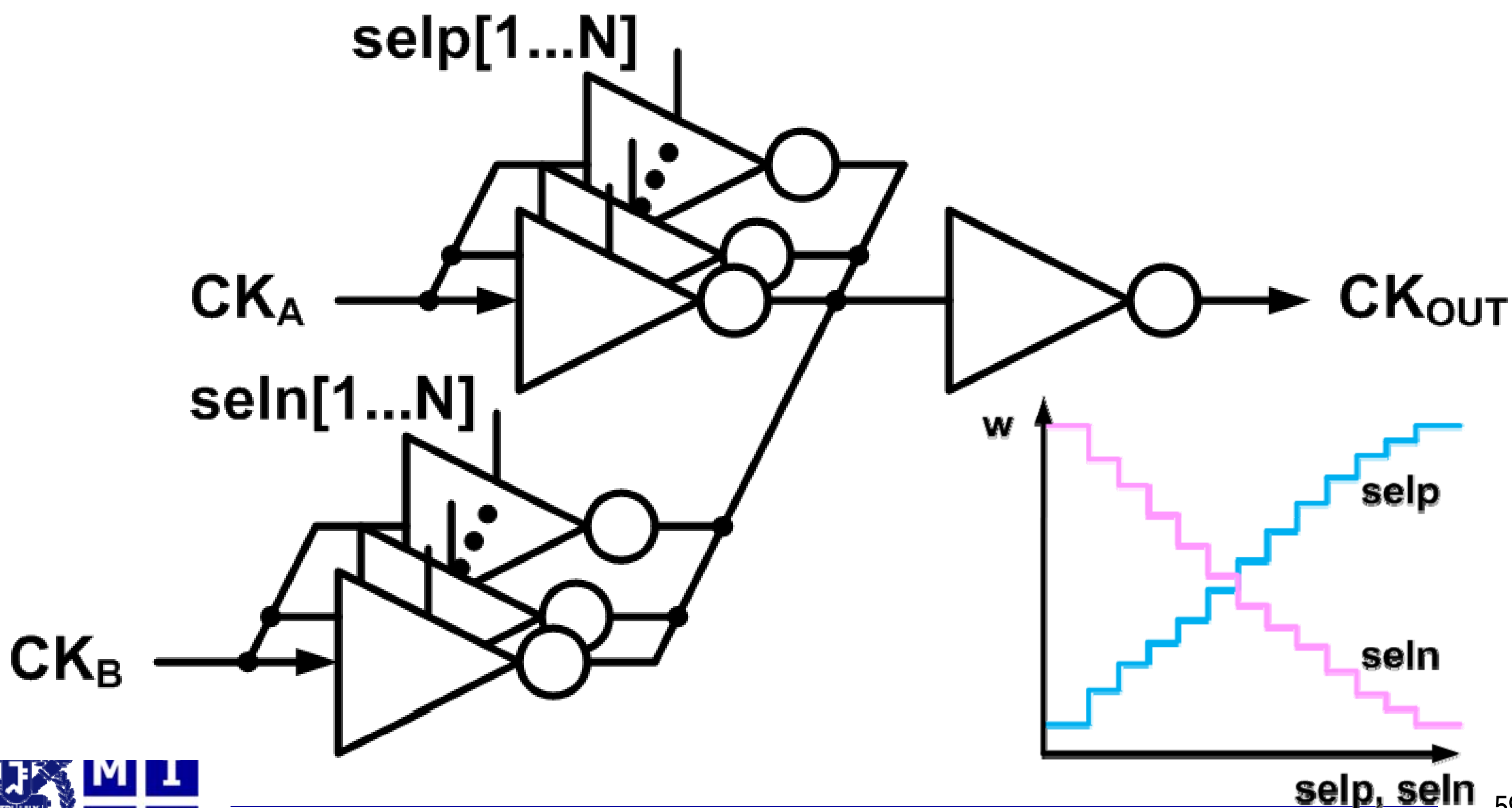
- Control inputs change the properties of the underlying system (e.g. interpolation weight; *w*)
  - Testing polarities of the Vc+/- vs. *w* requires only 4 points

$$\Phi_{OUT} = w \cdot \Phi_A + (1-w) \cdot \Phi_B + \Phi_0$$



Phase Interpolator with Digital Control

- **selp/seln** inputs represent *quantized analog* values
  - Sufficient just to verify each bit's weight; requires $N+1 < 2^N$



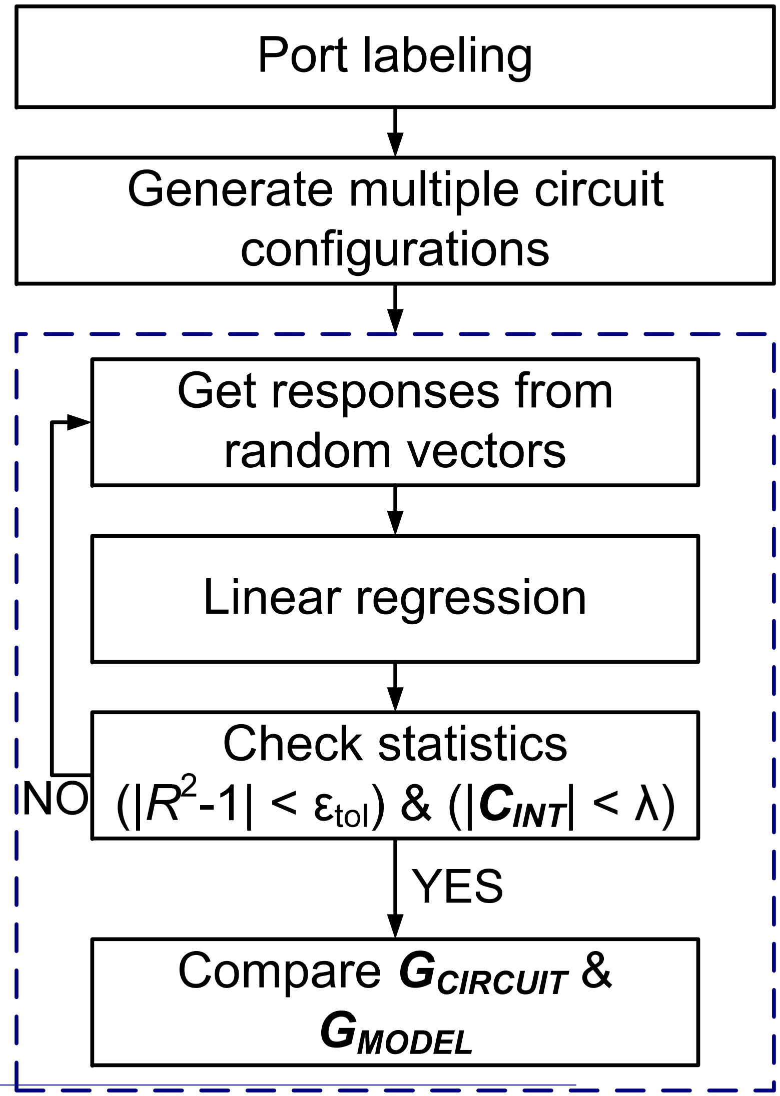
Intent of I/O Ports

- The classification of I/O ports guides the test vector generation

- Types of I/O Ports in the intended linear system
  - Analog port
    - Analog I/O port
    - Analog control port
    - Pseudo-output port
  - Digital port
    - Quantized analog port
    - True digital port
  - Function port

Checking Procedure

- Generate circuits to check
  - True digital inputs cause the linear circuit to change, and each needs to be checked
- Generate input stimulus
  - Using domain converter if needed
- Check to ensure circuit is linear
  - If not complain to user
- Check equivalence
  - Comparing gain matrices



Analog Fault Coverage

- If a circuit is defined by transfer matrix
  - One can find all faults by measuring that matrix
- Measuring that matrix is not hard
  - Since the number of required inputs is small
  - Even when the matrix is a function of control inputs
- Problem is determining what is a fault
  - Since no two matrices will ever be exactly the same
  - Need to set a tolerance
    - Is it absolute error? Relative error?
  - Unlike digital, generating the stimulus is the easy part.