how big can you dream?™

cadence

Future Directions in Mixed-Signal Behavioral Modeling

Ken Kundert

October 2002

2002 IEEE INTERNATIONAL WORKSHOP ON
BEHAVIORAL MODELING AND SIMULATION

On The Cusp

Mixed-signal behavioral modeling is starting to take off …

- Standard languages are here
  - Verilog-A, Verilog-AMS, VHDL-AMS
- Simulators are ready or soon will be …
  - Spectre and AMS Designer from Cadence
  - Advance-MS from Mentor
  - More than ten others
- Mixed-signal IP is becoming available with behavioral models
  - Barcelona, Neolinear, AMI Semi, Tality, etc.
- Slowly building up a supply of trained modeling engineers

Emerging Trends

The following are expected to be focus areas for the next few years

- Top-down design
- RF
- MEMS
- Compact modeling

Top-Down Design

Design Challenge: Size and Complexity

cadence

- Increasing complexity as circuits become larger
  - Increasing integration
    - To reduce cost, size, weight, and power dissipation
  - Digitalization
    - Both digital information and digital implementation
- Increasing complexity of signal processing
  - Implementation of algorithms in silicon
    - Adaptive circuits, error correction, PLL's, etc.
- Designers must improve their productivity to keep up

Slides from EPD 2001, AACD 2000

Productivity: Improving CAD is not Enough

cadence

"Fundamental improvements in design methodology and CAD tools will be required to manage the overwhelming design and verification complexity"

Dr. H. Samueli, co-chairman and CTO, Broadcom Corp. Invited Keynote Address, "Broadband communication ICs: enabling high-bandwidth connectivity in the home and office", *Slide supplement 1999 to the Digest of Technical Papers*, pp. 29-35, International Solid State Circuits Conference, Feb 15-17, 1999, San Francisco, CA

Design Productivity

- Huge productivity ratio between design groups
  - As much as 14x (Collett International, 1998)
- In a fast moving market
  - Cannot overcome this disparity in productivity by working harder
  - Must change the way design is done
- Cause of poor productivity: Using a bottom-up design style
  - Problems are found late in design cycle, causing substantial redesign
  - Simulation is expensive, and so usually inadequate
  - Inadequate verification requires silicon prototypes
  - Today's designs are too complex for bottom-up design style
  - Too many serial dependencies

What is Needed

- To handle larger and more complex circuits
  - Need better productivity
  - Need divide and conquer strategy
- To address time-to-market
  - Must effectively utilize more designers
  - Must reorganize design process
    - More independent tasks
    - Reduce number of serial steps

The Solution

- A formal top-down design process …
  - That methodically proceeds from architecture to transistor level
  - Where each level is fully designed before proceeding to next level
  - Where each level is fully leveraged in design of next level
  - Where each move is verified before proceeding
- Careful verification planning involving ...
  - System verification through simulation
  - Mixed-level verification through simulation
  - A modeling plan that maximizes efficacy and speed of simulation
  - Full chip simulation only when no alternatives exist
- Test development that proceeds in parallel with design

Architectural Exploration & Verification

cadence

- Rapidly explore and verify architecture via simulation
  - Using Verilog-AMS provides a smooth transition to circuit level
  - VHDL-AMS or Simulink could also be used, but more cumbersome
- Provides greater understanding of system early in design process
  - Rapid optimization of architecture
  - Discard unworkable architectures early
- Moves simulation to front of design process
  - Simulation is much faster
  - Block specs driven by system simulation

Partitioning

- Find appropriate interfaces and partition
  - Clever partitioning can be source of innovation
    - Joining normally distinct blocks can payoff in better performance
      - LO and mixer, S&H and ADC, etc.
  - Budget specifications for blocks
    - System simulation and experience used to set block specifications
  - Document interfaces
- Formal partitioning supports concurrent design
  - Better communication
  - Design of blocks proceeds in parallel
  - Allows more engineers to work on the same project

Pin-Accurate Top-Level Schematic

- Develop pin-accurate top-level schematic
  - Behavioral models represent the blocks
  - Faithfully represents block interfaces
    - Levels, polarities, offsets, drive strengths, loading, timing, etc.
- Distribute to every member of the team
  - Acts as executable specification and test bench
  - Acts as DUT for test program development
- Owned by chip architect
  - Cannot be changed without agreement from affected team members
  - Changes to interfaces not official until TLS updated and redistributed

Mixed-Level Simulation (MLS)

- Verify circuit blocks in context of system
  - Individual blocks simulated at transistor level
  - Rest of system at behavioral level
- Simulate with pin-accurate block models
  - Verifies block interface specifications
  - Eases integration of completed blocks
- Only viable approach to verify complex systems
  - Can improve simulation speed by order of magnitude over full transistor level simulation

Simulation and Modeling Plans

- Identify areas of concern, develop verification plans
  - Maximize use and efficacy of system-and mixed-level simulation
  - Minimize need for full-chip transistor-level simulation
- Modeling plan developed from simulation plan
  - There may be several models for each block
    - Several simple models often better than one complex one
    - Consider loading, bias levels and headroom, etc.
- Developed and enforced by the chip architect
- Up front planning results in ...
  - More complete and efficient verification
  - Fewer design iterations

SPICE Simulation

cadence

- Use selectively as needed
  - Mixed-level simulation
    - Verify blocks in context of system
  - Hot spots
  - Critical paths
  - Start-up behavior
- The idea is not to eliminate SPICE simulation, but to ...
  - Reduce the time spent in SPICE simulation while ...
  - Increasing the effectiveness of simulation in general

Top-Down Design Is ...

cadence

- A way of trading ...
  - An up-front investment in planning and modeling
- For ...
  - A well controlled design process
    - More predictable
    - Fewer unpleasant surprises
    - Fewer design iterations
  - More parallelism

Top-Down Design ...

- Is not going to happen on its own
- It is a formal top-down design process that requires a serious commitment through out the entire design process
- It requires a substantial investment in education and infrastructure
- Any design group that attempts it without adequate training, management support, and planning is likely to fail
- It is much easier the second time around

Top-Down Design Impediments

- Lack of acceptance
  - Designers use bottom-up design or lazy top-down design
  - They do not follow formal top-down design principles
    - Partition design using well specified and verified interfaces
    - Develop verification and modeling plans in advance
    - Avoid unverified translations
      - Mixed-level simulation
- Lack of qualified engineers
  - Need strong modeling, simulation, and application background

What is Needed

- A long term perspective
- Extensive model libraries
  - Reduced barrier to entry
  - Learn by example
- Improved education and training
  - Must train modeling
  - Must train top-down design
  - Must train both inside and outside universities
    - University classes, workshops, books, articles, etc.
- Model extraction and fitting tools

It Can Happen, Stranger Things Have

Bears muzzle No. 12 Huskies

CAL DEFEATS WASHINGTON FOR 1ST TIME IN 26 YEARS

San Jose Mercury News, 6 October 2002

cadence

RF Design

Modeling Challenges in RF Design

cadence

- High-level modeling
- Transition from high-level model to implementation
- Distributed component modeling

High-Level Modeling of RF Systems

cadence

- Today
  - Simulators (Matlab & Simulink, Ptolemy, SPW)
  - Spreadsheets (Excel)
- Missing in action
  - VHDL-AMS
  - Verilog-AMS
  - Verilog-A

What is Working Today, What is Not

cadence

+ RF model libraries (generic & standards)

+ RF analysis (test benches and measurement)

– Smooth transition to implementation

- AMS languages can help solve implementation issue

  But must catch up in libraries & analysis

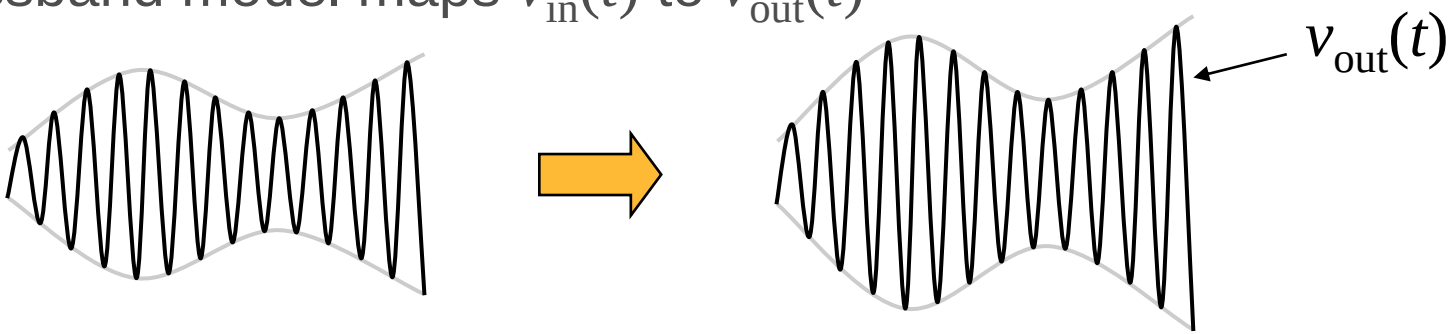
Why Use Behavioral Modeling?

- Behavioral models translate RF metrics into system metrics
  - Through simulation
- Standard RF metrics
  - Gain, iIP_3, noise figure
- Standard system metrics
  - BER, EVM, ACPR
  - Determining these metrics is very compute intensive
  - Simulations must be very fast

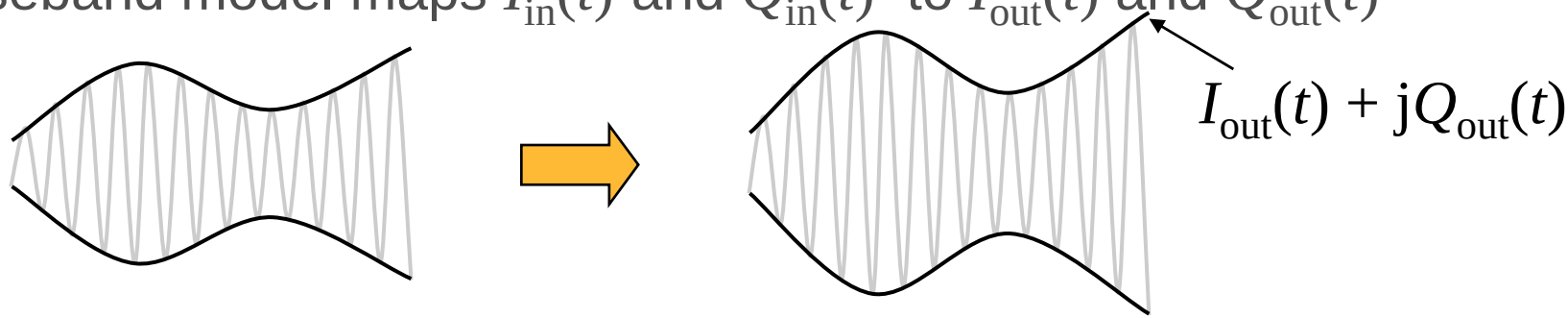
Slides derived from those of Jess Chen (BMAS 2001)

Suppressed Carrier Modeling

- Assume signals of the form $v(t) = \mathrm{Re}\{(I(t) + \mathrm{j}Q(t))e^{\mathrm{j}\omega t}\}$
- Passband model maps $v_{in}(t)$ to $v_{out}(t)$



- Baseband model maps $I_{in}(t)$ and $Q_{in}(t)$ to $I_{out}(t)$ and $Q_{out}(t)$



- More efficient because time point density is much lower

Two Types of RF Models

- Passband Models
  - Does not suppress the carrier
  - Good for implementation and validation
  - Too slow for architectural exploration
- Baseband Equivalent Models
  - Highly effective at exploring the architectural space.
  - Specify components in terms of RF metrics.
  - Measure performance in terms of system metrics.

Transition to Implementation

- Use baseband-equivalent models for architectural exploration
- Use passband models in transition to implementation
  - Allows more detail in the model
  - Compatible with transistor-level simulation
- Co-simulate baseband and passband models
  - Use modulators and de-modulators as interface elements
- Issue
  - Baseband models pass 2 numbers (I & Q) per wire
    - Perhaps more, harmonics, impedance, etc.
  - Verilog-A/MS does not support this

What is Needed

- Extension of Verilog-A/MS to support composite signals
- Library of self-consistent passband and baseband models
- Application support
  - Standard-based test benches
  - RF analyses and measurements

Distributed Components

- Generally described in the frequency-domain
  - Naturally compatible with harmonic balance
  - More work needed to support in transient-based simulators
- VHDL-FD or VHDL-RF/MW
  - Proposes to add frequency-domain modeling to VHDL-AMS
- Trend is away from describing distributed models with equations
  - Measurements use tables of *S*-parameters
  - Electromagnetic solvers either use S-parameters or ROMs

Frequency-Domain Modeling

- Few situations where users are comfortable writing a distributed model using expressions involving frequency
  - Skin effect: $R = R_0\sqrt{f}$
  - System level models: $v_{out}(f) = v_{in}(f)$ for $f < f_0$ and 0 otherwise
- Often models are non-physical
  - Non-causal, non-passive, etc. (ex: above models are noncausal)
  - Results in large errors and various numerical problems
  - Transient-base simulators struggle with such models
  - Harmonic balance simulators increasingly struggle
    - Envelope is time-domain based; pre-convergence transient
- Very easy and common for users to write non-physical models

Extending AMS into Frequency-Domain

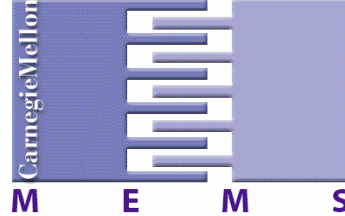
cadence

- Provide hooks to test equipment and EM solvers
  - Table models (S-parameter files) or ROMs
- Wait on providing support in language for expressions involving frequency
  - Improve simulator implementations
  - Improve frequency-domain expression language
    - Develop language that naturally avoids causality issues

cadence

MEMS

Hierarchical Levels of Abstraction in Suspended MEMS

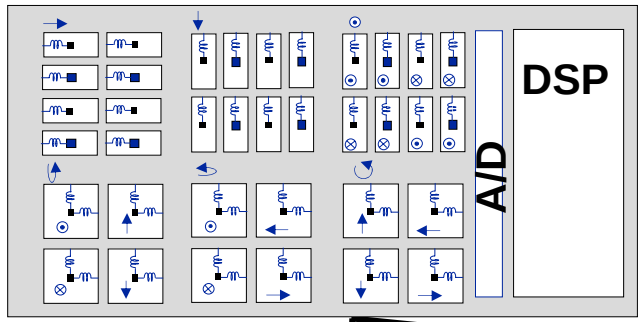


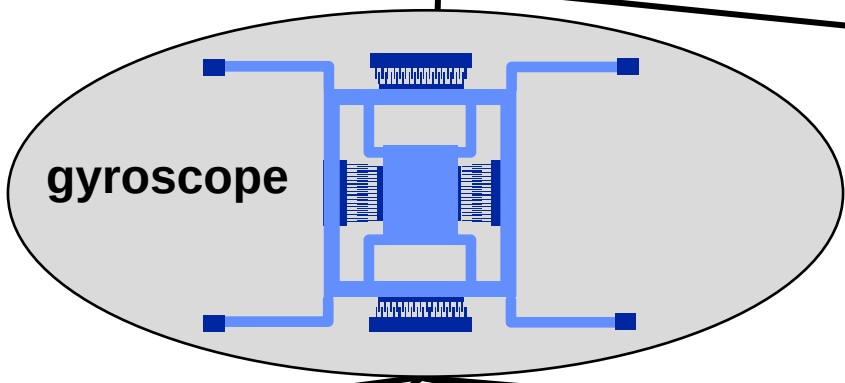
System

Component (Subsystem)

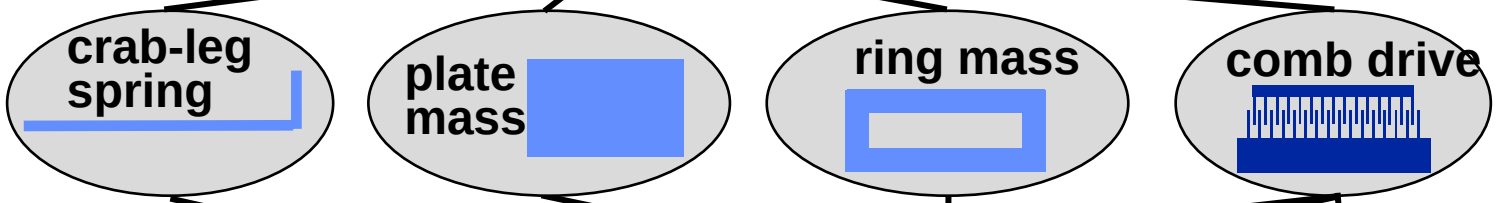
Functional element

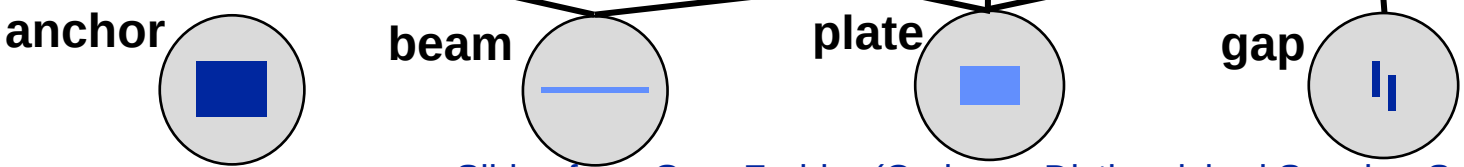
Atomic element





other components

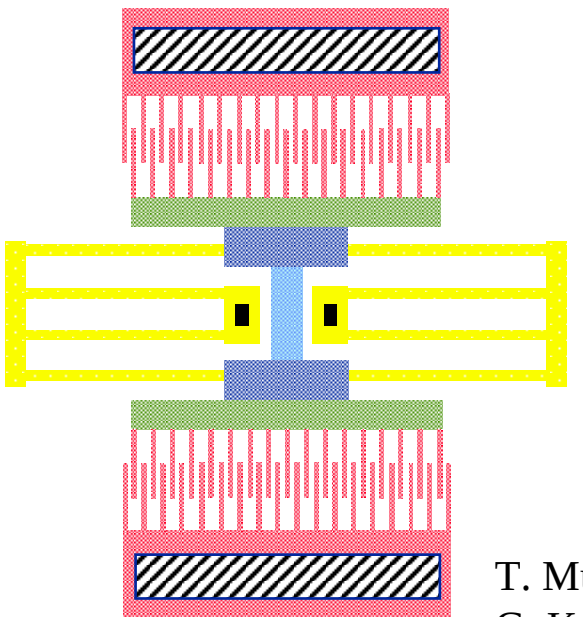




Slides from Gary Fedder (Cadence Distinguished Speaker Series)

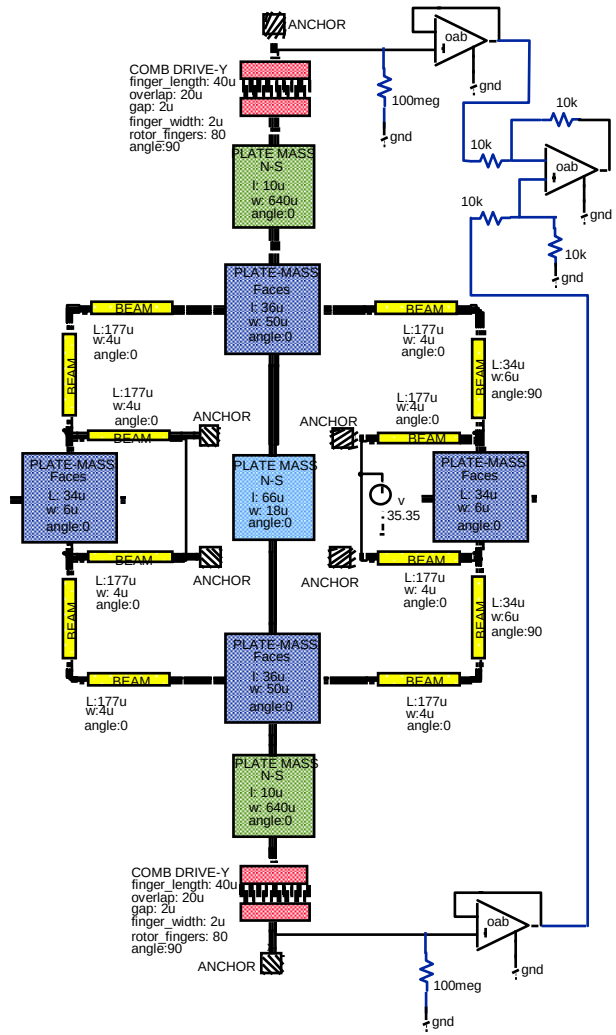
NODAS MEMS Cell Library

- NODAS is the MEMS schematic design library developed at Carnegie Mellon.
- Goal: Develop and validate reusable parameterized cell library and tools



layout generation

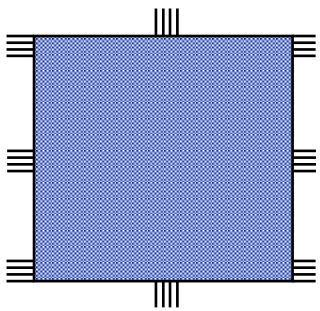
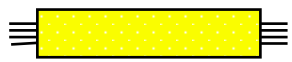
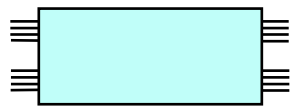
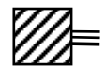
circuit extraction



T. Mukherjee, et al., *IEEE TCAD of Int. Circuits and Systems*, Dec. 2000.
G. K. Fedder and Q. Jing, *IEEE TCAS-II*, Oct. 1999.

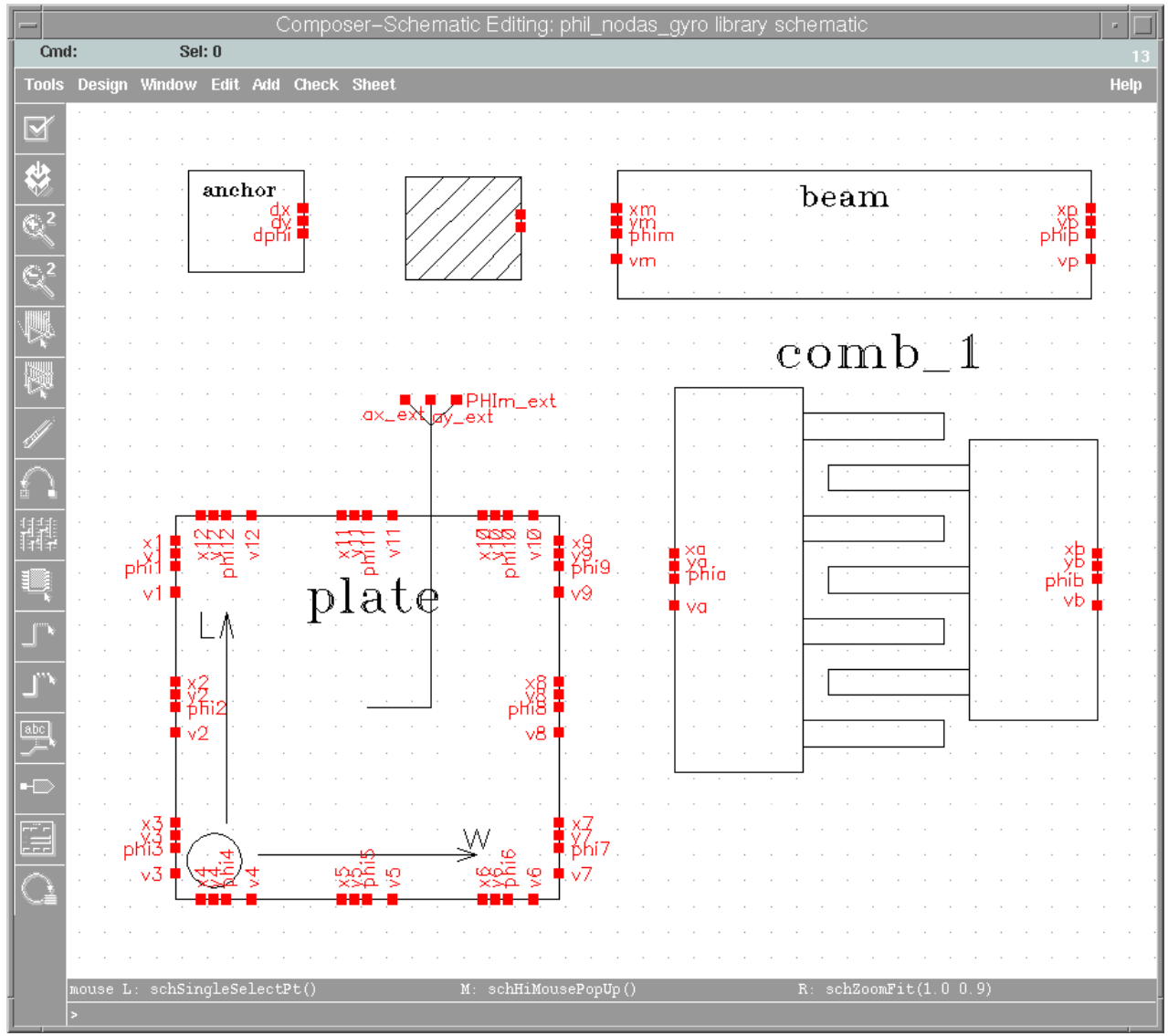
Basic Elements

- Circuit representations of suspended MEMS can be partitioned into four basic lumped-parameter elements: plates, beams, gaps, and anchors

	PLATE	BEAM	GAP	ANCHOR
symbol				
parameters	L: 50u w: 100u angle: 0	L: 100u w: 4u angle: 0	g: 4u L_o: 25u angle: 0	
layout				

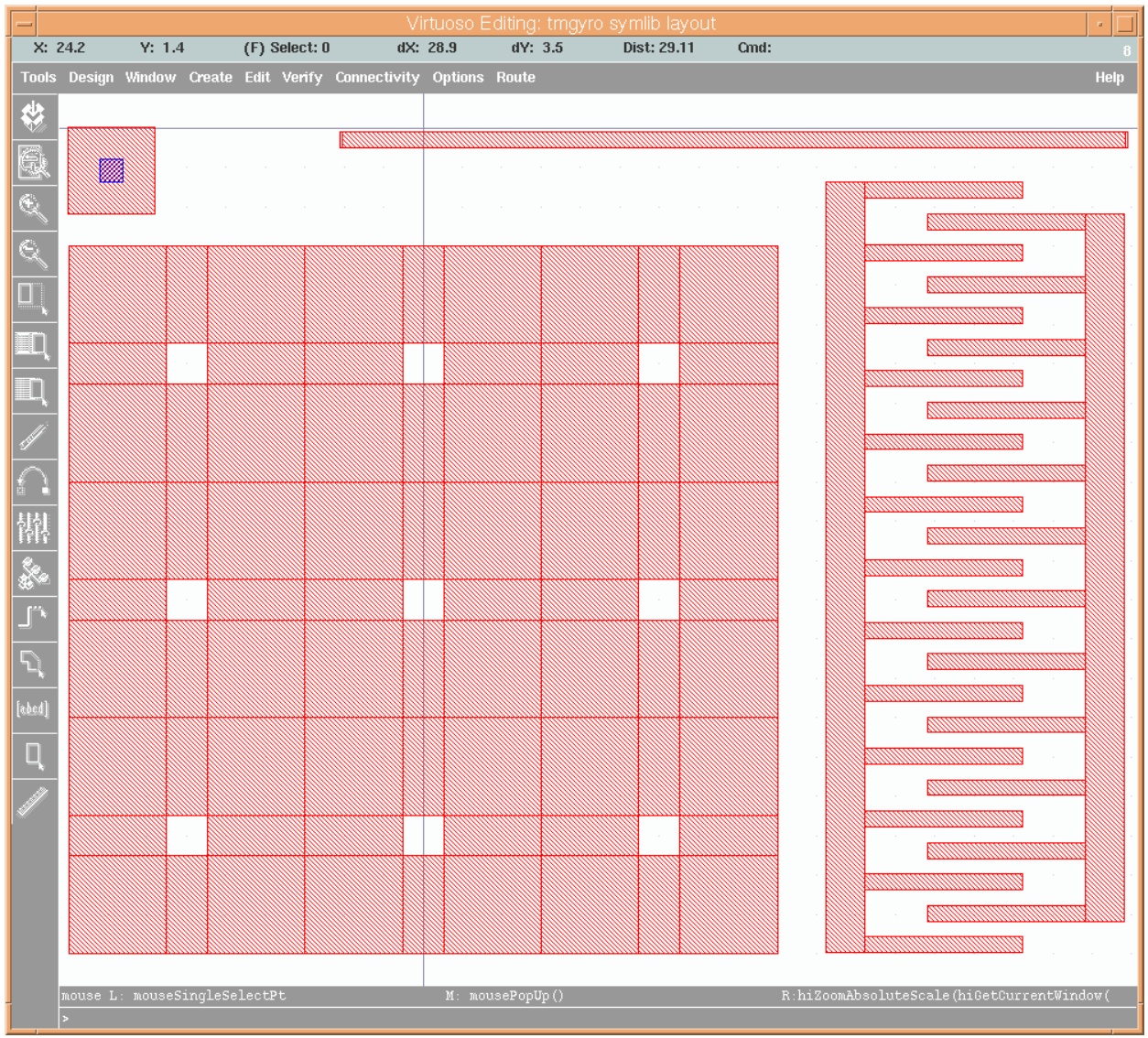
Multi-Level Design Reuse

- Elements (symbols and models) can be reused in new designs
- Low-level elements are:
  - Anchor
  - Beam
  - Plate
  - Gap
  - Comb



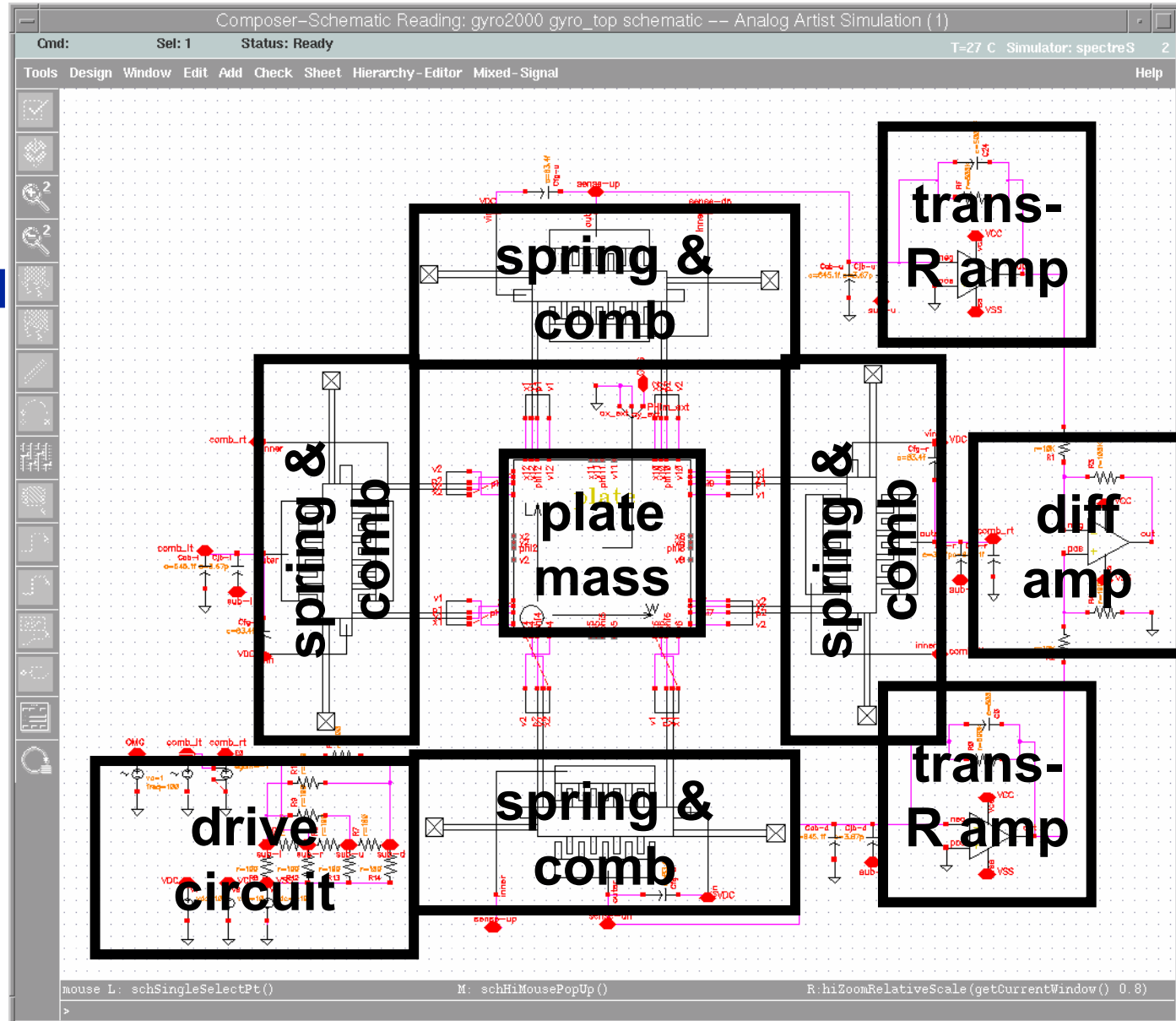
Layout Generation

- Automated layout is hierarchically p-cell (parameterized cell) driven directly from elements

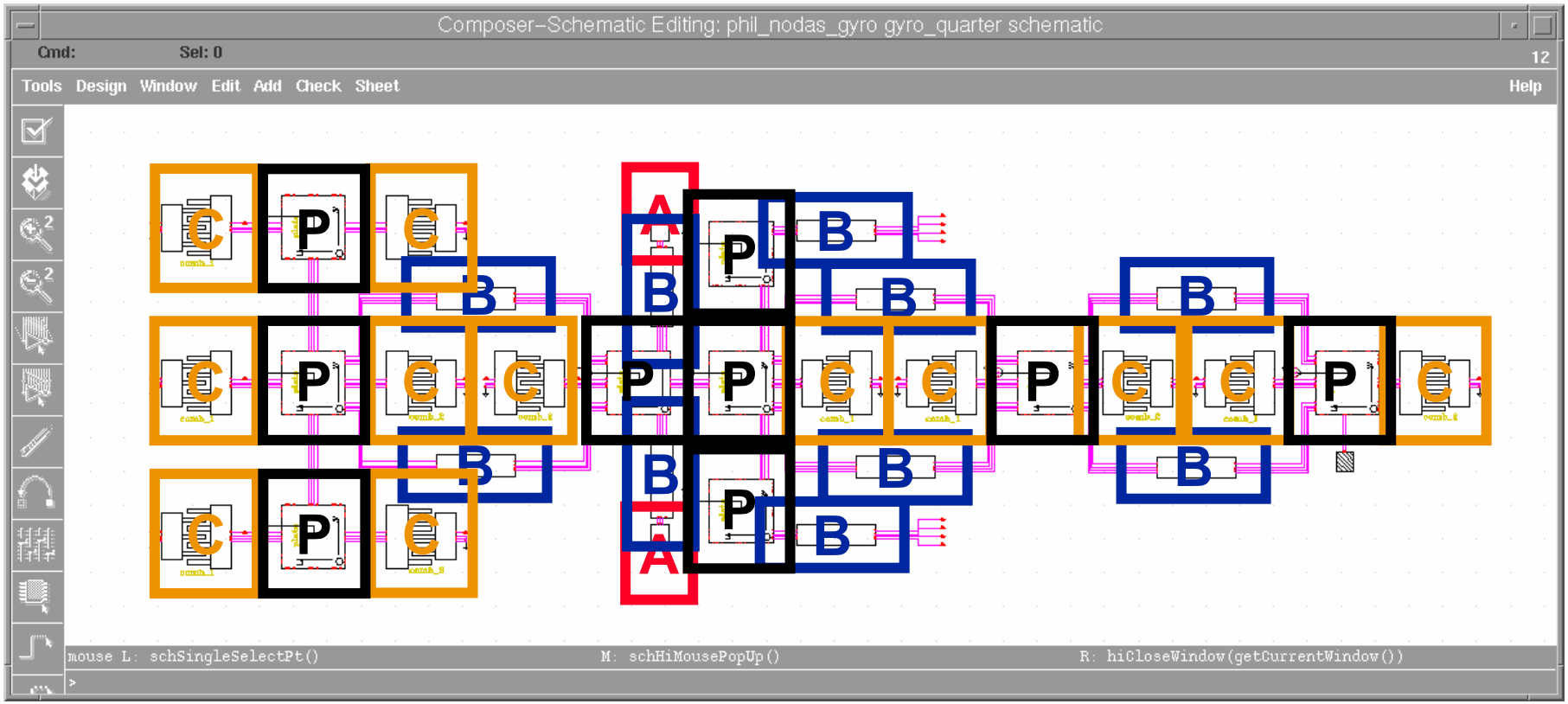


Gyroscope Schematic Design Entry

- MEMS hierarchical cells provide ability to start simple and add detail later
- Interoperable components at several levels of abstraction
- Only two kinds of MEMS components in this view



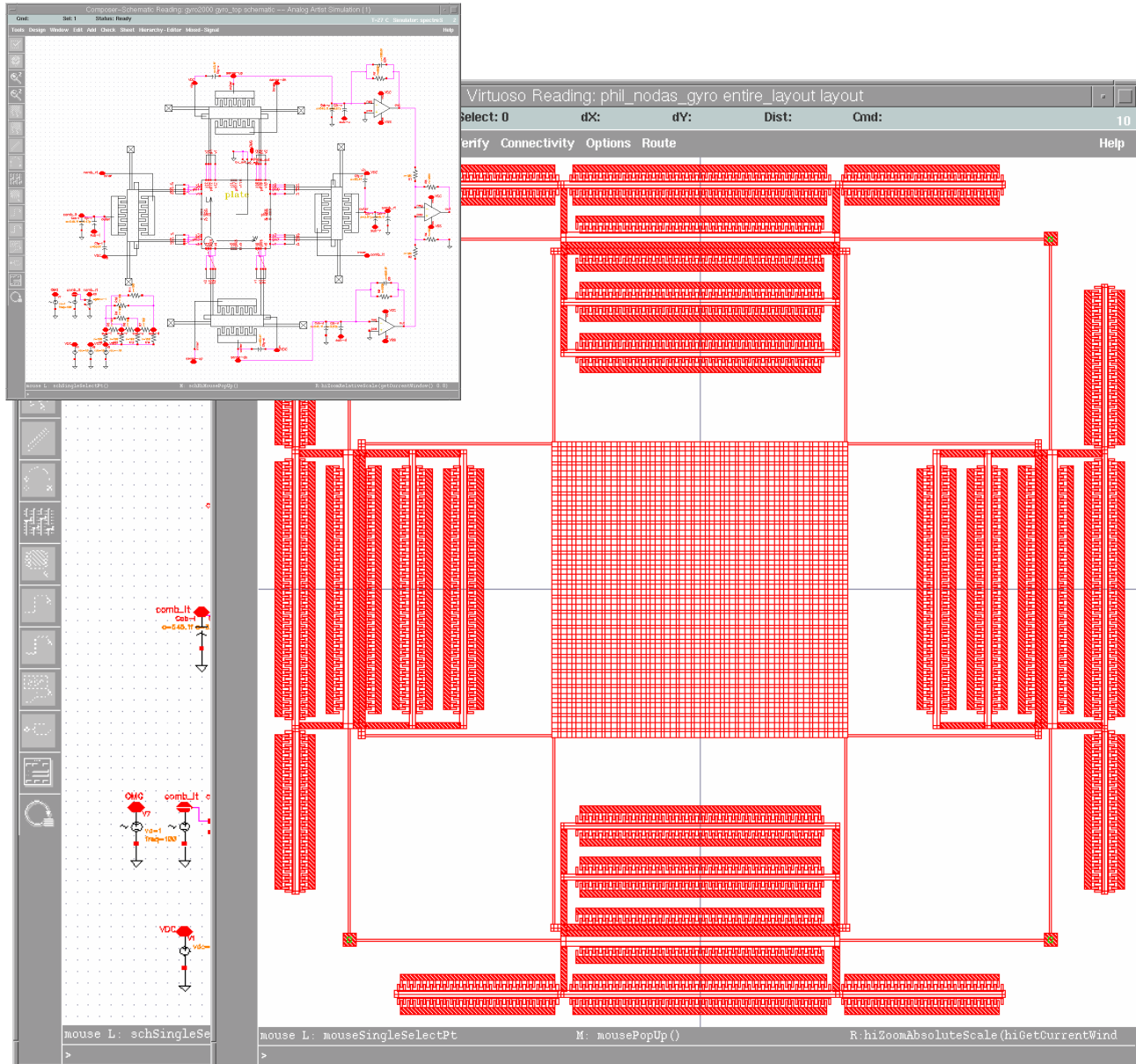
MEMS at Next Level: Spring & Comb Cell



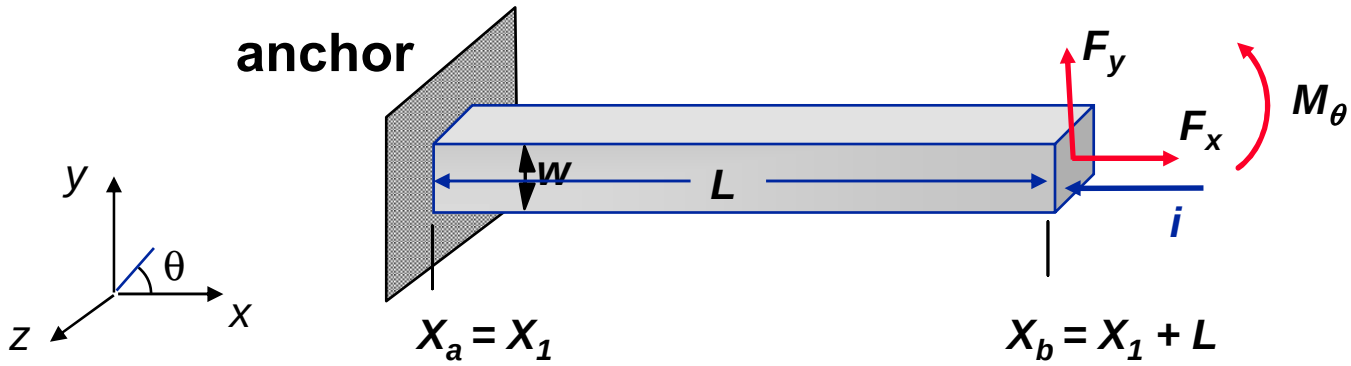
- Extreme detail with only four MEMS components
  - A – anchor
  - B – beam
  - C – comb-finger capacitor
  - P – plate mass

Gyroscope Layout Generation

- **All necessary geometric information embedded in schematic**

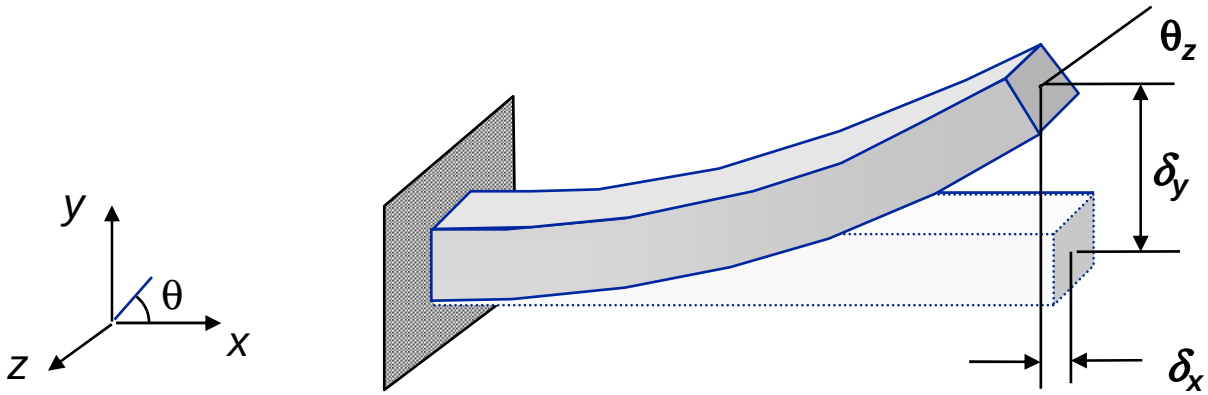


Cantilever Beam Example

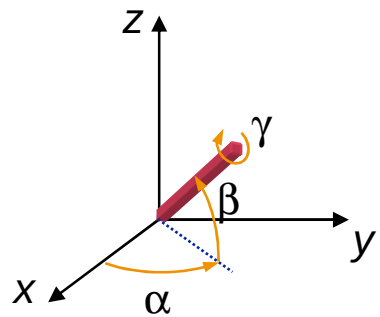


- Assume conducting beam:
  - Kirchhoff's current law (KCL)
    - $\Sigma i = 0$
- Assume 2-D operation (x-y plane):
  - Force balance:
    - $\Sigma F_x = 0; \Sigma F_y = 0$
  - Moment balance:
    - $\Sigma M = 0$

Position and Displacement

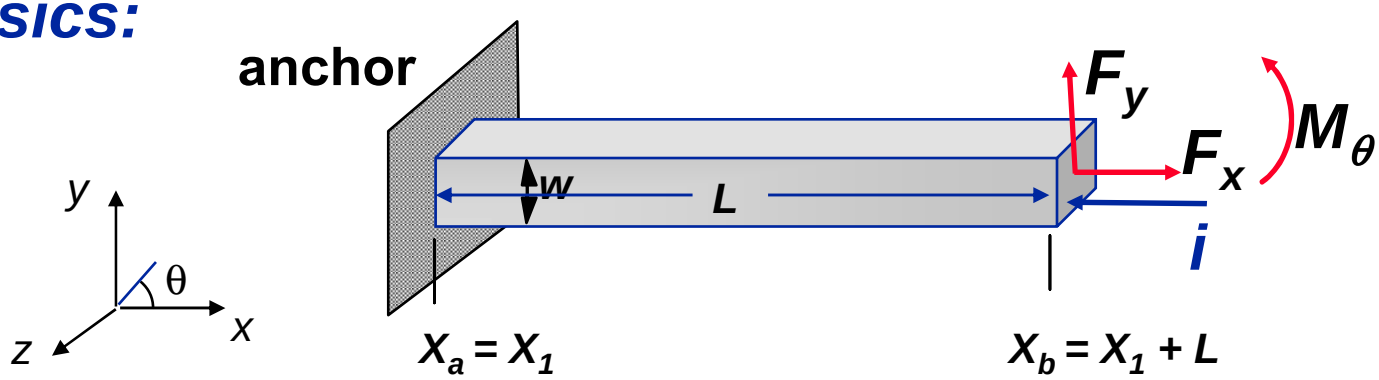


- Layout position
  - X, Y, Z
  - Orientation angles, α, β, γ
- Displacement
  - $\delta_x, \delta_y, \delta_z$
  - $\theta_z, \theta_x, \theta_y$

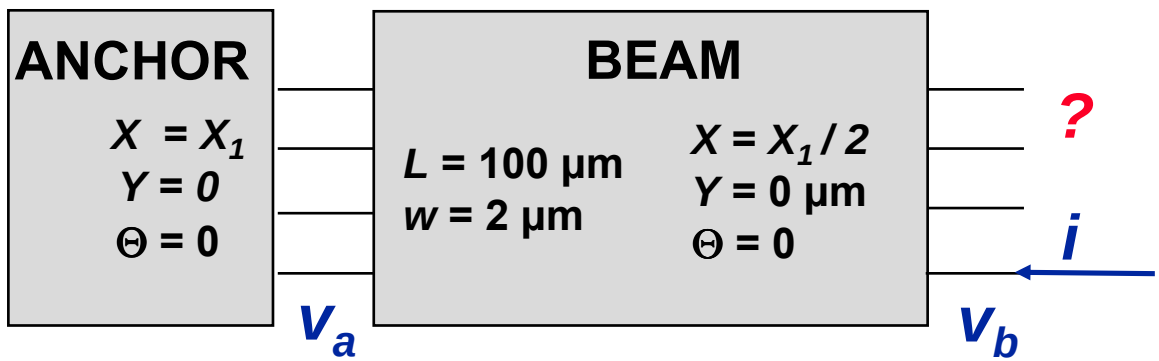


MEMS Circuit Representation: Cantilever Beam Example

Physics:



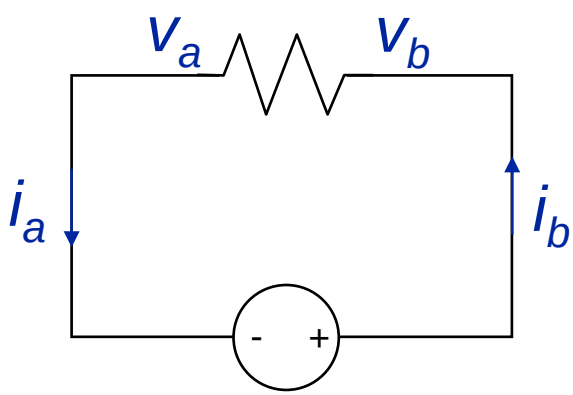
Circuit:



Schematic components have geometric parameters

‘Through’ and ‘Across’ Variables

- **Electrical ‘across’ variable is voltage**
- **Electrical ‘through’ variable is current**



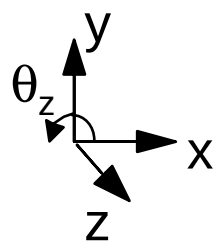
- Nodes are labeled *a* and *b*
- Across variables are v_a and v_b
  - Voltage ‘across’ resistor is v_b - v_a
- Through variables are i_a and i_b
  - Current ‘through’ resistor is i_a (or i_b)

Model determines $i_a = f(v_a, v_b)$ and $i_b = f(v_a, v_b)$

Mechanical Nodal Conventions

- **Across variables (x, y, θ_z)**
  - Positive valued displacements are in positive axial direction
  - Positive valued angles are counterclockwise around axis

Example: moving beam in x



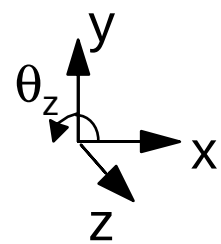
both x_a and x_b are positive

Equivalent schematic:

Mechanical Nodal Conventions

- **Across variables (x, y, θ_z)**
  - Positive valued displacements are in positive axial direction
  - Positive valued angles are counterclockwise around axis

Example: beam in tension



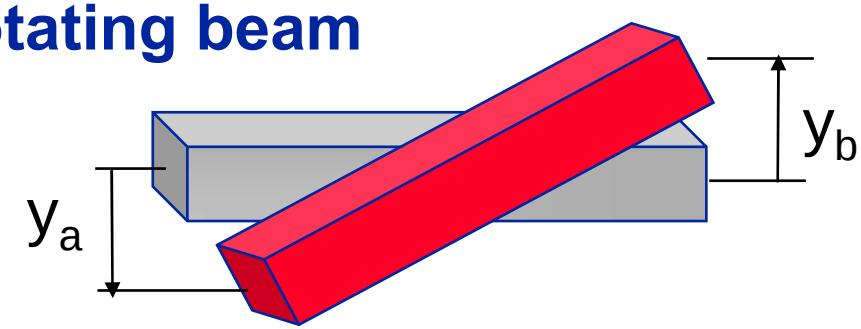
x_a is negative; x_b is positive

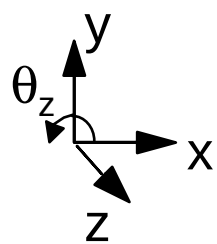
Equivalent schematic:

Mechanical Nodal Conventions

- **Across variables (x, y, θ_z)**
  - Positive valued displacements are in positive axial direction
  - Positive valued angles are counterclockwise around axis

Example: rotating beam





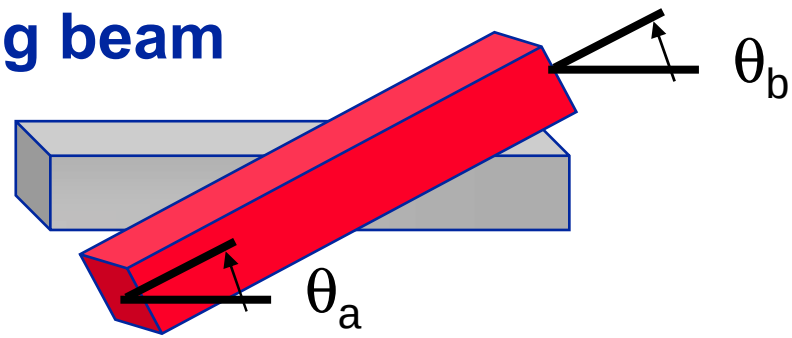
y_a is negative; y_b is positive

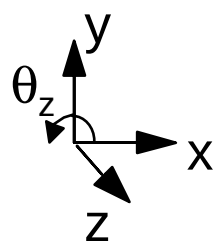
Equivalent schematic:

Mechanical Nodal Conventions

- **Across variables (x, y, θ_z)**
  - Positive valued displacements are in positive axial direction
  - Positive valued angles are counterclockwise around axis

Example: rotating beam





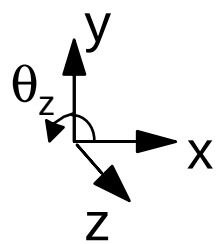
both θ_a and θ_b are positive

Equivalent schematic:

Mechanical Nodal Conventions

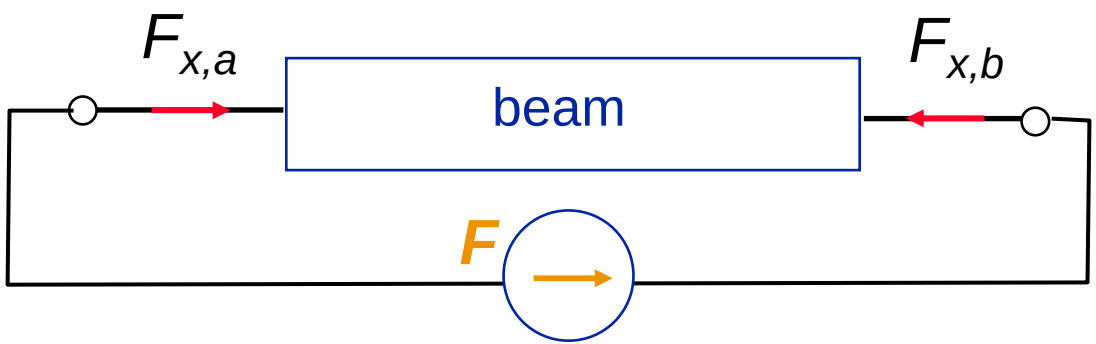
- **Through variables (F_x, F_y, M_z)**
  - Force flowing into node acts in positive axial direction
  - Moment flowing into node acts counterclockwise around axis

Example: beam in tension



$F_{x,a}$ is negative; $F_{x,b}$ is positive

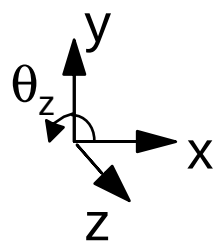
Equivalent schematic:



Mechanical Nodal Conventions

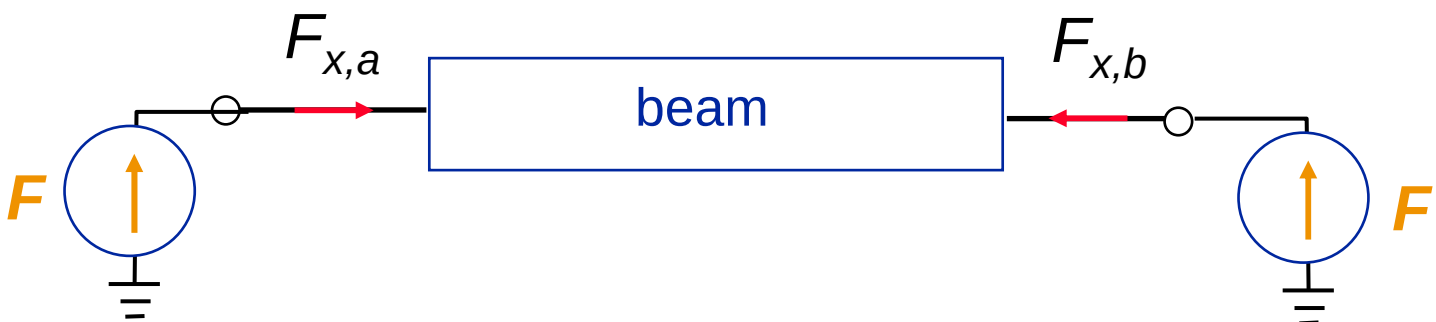
- **Through variables (F_x, F_y, M_z)**
  - Force flowing into node acts in positive axial direction
  - Moment flowing into node acts counterclockwise around axis

Example: moving beam



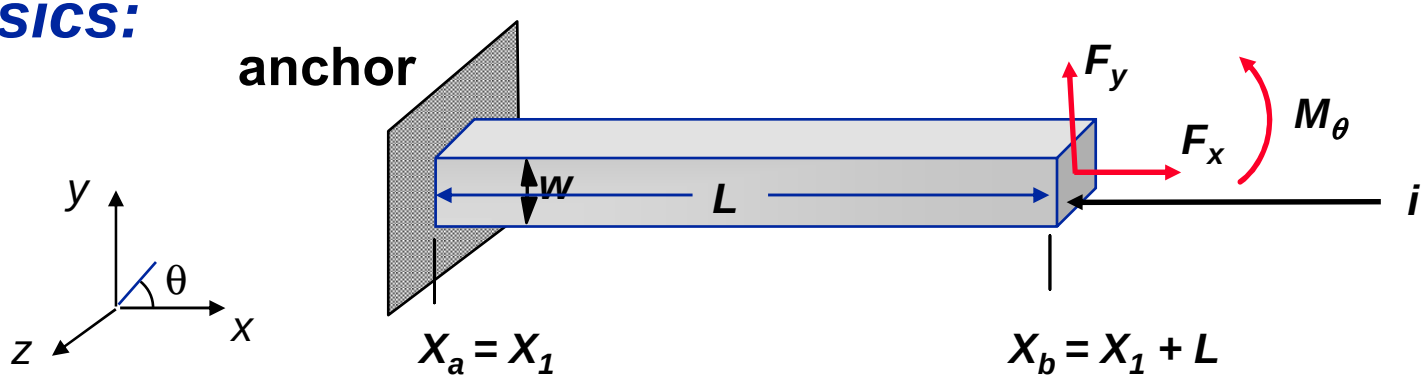
both $F_{x,a}$ and $F_{x,b}$ are positive

Equivalent schematic:

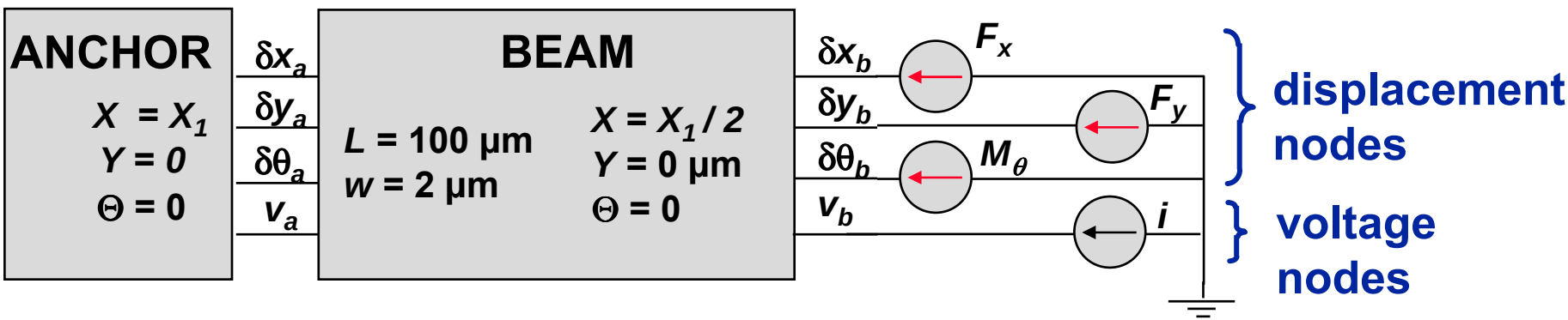


MEMS Circuit Representation: Cantilever Beam Example

Physics:



Circuit:



- Across variables: displacement, angle, voltage
- Through variables: force, moment, current
  - Branch relations: Σ i = 0; Σ F = 0; Σ M = 0

What is Needed to Support MEMS?

- Extend Verilog-A/MS to support composite, multidimensional signals (X, Y, Z, α, β, γ, V)
- Carefully resolve tolerancing issues
  - Develop natures with appropriate tolerances, pointers to derivative and integral natures
  - Develop modeling guidelines to improve use of tolerances
  - Implement tolerancing features of language in simulators
- Develop MEMS library
- Improve visualization tools

cadence

Compact Modeling

Issues in Compact Modeling

cadence

- BSIM3v3 requires 40k lines of code
  - Can take 1 year or more to schedule a model
  - Can take 6 months or more to implement a model
  - Can take 2 months or more to enhance a model
  - Including time to develop the model, and time to adopt release that contains it, it can take several years between when an engineer requests a model and when it is available
- Simulation vendors only support most popular models
  - Access to specialty models suffers
  - Many modeling groups struggle to contribute
  - Users must make do without the models they need

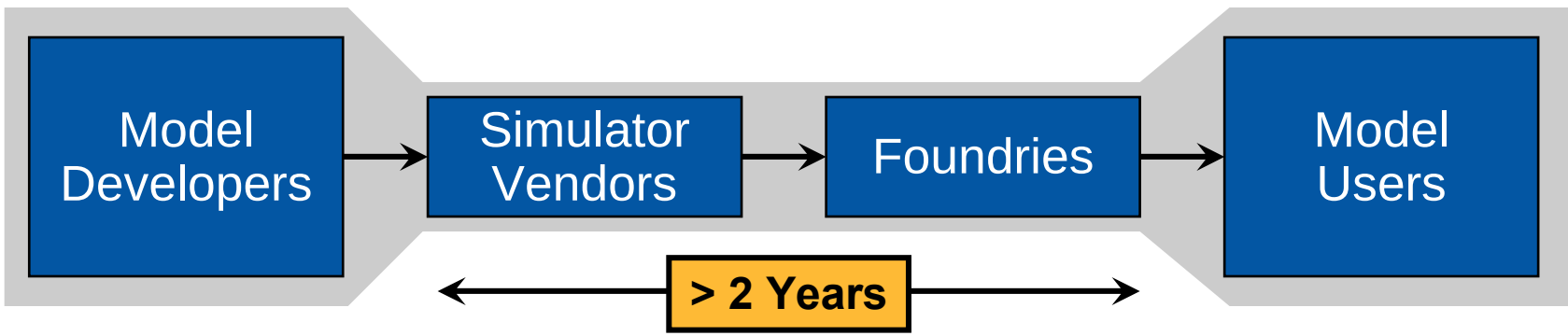
Slides from CMC 2002

Issues in Compact Modeling

- Can take a long time to include important new effect into models
  - Leakage, RF effects, etc.
- Models inefficiently implemented
  - Models too large, implementers too rushed, to effectively optimize models
    - Difficult trade-off between efficiency and time required to implement model
  - Core functions are huge (containing more than 2500 lines for BSIM3v3)
    - Too large for optimizing compilers
  - Bloated models
    - With few models, those available must do everything

Issues in Compact Modeling

- Slight difference in models between simulators, extractors
  - Causes extra work to extract and support multiple versions of models
  - Causes confusion, finger pointing
- There is too much distance between model developers and users
  - Takes too much time
  - Middle men are often reluctant partners with competing objectives
  - Frustrating for users (need help) and developers (want to help)



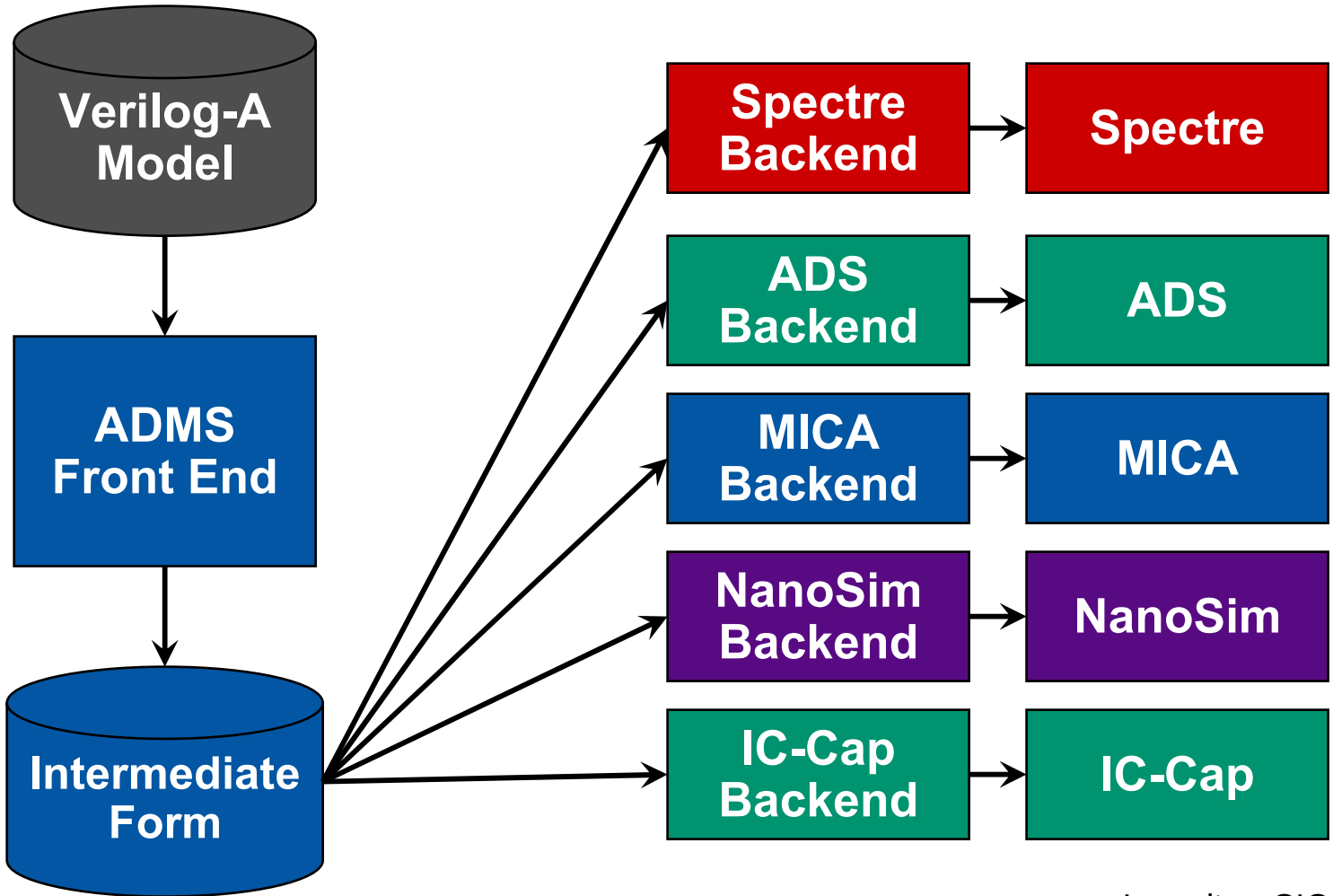
Effect of Issues

- Few models are available
- They run slowly
- They take a long time to get
- Users have little control of what they get
- Model developers and users are disempowered, frustrated
- Models are hard to develop and establish
  - Takes the fun out of modeling
  - Makes it hard to recruit new talent to the field

Model Compiler

- Develop models in Verilog-A
  - It's easy to use (a language that is designed for modeling)
  - Make and try changes with quick turn around time
  - Works in all analyses (DC, AC, noise, transient, RF, etc.)
  - Test models on real circuits (ring oscillators, etc.)
- Compile in to multiple simulators
  - Exactly the same model for all simulators & extractors
  - Expect better than hand-coded performance (eventually)
  - Avoids errors that result during conversion to C
- Eliminates middle men, empowers model developers and users
  - Encourages open-source model development

Motorola's ADMS

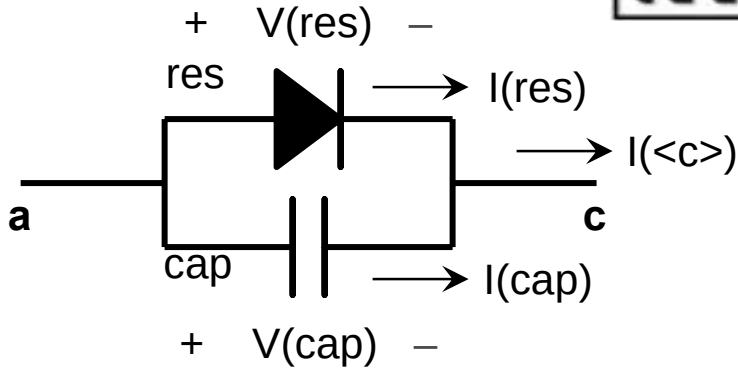


Lemaitre, CICC 2002

Why Verilog-A

- It's a good fit
  - It's a language designed for modeling
  - Compatible with SPICE-class simulators, does not require full MS simulator
- It's a standard
  - It is not proprietary, can be implemented & supported by anybody
  - Behavioral model benefit from association with compact models
    - Compilers, optimizers, documentation, etc.
  - Compact model benefit from association with behavioral models
    - Increased attention and investment
  - More people will know language
- Rapid testing of models
  - Interpreted versions exist
  - Modify and test model without compiling

Example: Diode




```
module diode (a, c);
  electrical a, c;
  branch (a, c) res, cap;
  parameter real is = 1e–14 from (0:inf);          // resistive parameters
  parameter real tf = 0, cjo = 0, phi = 0.7;       // capacitive parameters
  parameter real kf = 0, af = 1, ef=1;             // noise parameters
  analog begin
        I(res) <+ is∗(limexp(V(res)/$vt) – 1);
        I(cap) <+ ddt(tf∗I(res) – 2∗cjo∗sqrt(phi∗(phi∗V(cap))));
        I(res) <+ white_noise(2∗‘P_Q∗I(res));
        I(res) <+ flicker_noise(kf∗pow(I(res), af), ef);
  end
endmodule
```

One Simple Model Works in All Analyses — DC, AC, Noise, Transient, RF —

Extending Verilog-A

cadence

- Improved documentation
  - Model, parameters, terminals, etc.
- Modular model support
  - Declare variables where used
  - Define multiple versions with same parameters but different speed/accuracy tradeoffs
  - Allow user to easily specify version as configuration

- Optional terminals
- Required parameters
- Initialized variables
- Output, op-point parameters
- Multiplicity factor
- Gmin support
- Frequency-domain descriptions.
- Simulator specificity

Critical Success Factors

cadence

- Multiple simulator & extractor support
  - Need a compelling set of simulators & extractors supported
  - Model writers must believe that their models will see substantial use
  - Foundries, users must see compelling advantage to switch
    - Reduced model support costs
    - Improved model quality, performance, coverage, accuracy, timeliness, ...
- Available Models
  - Need a compelling set of models available in Verilog-A
  - Would be best if they were exclusively available in Verilog-A

If We Are Successful

- More nimble model development and support process
  - Companies can ask local universities to develop specialized models
  - Model fixes can be turned around in hours
  - Will be important if models begin to change at < 90nm
- A healthier, more open modeling community
  - Enables an open-source approach to development and support
  - Allows more people to contribute
  - Allows compact modeling to expand beyond MOS and BJT models

cadence

Onward …

Looking Back

cadence

- As a community, we have accomplished a great deal
  - Standard languages
  - Simulators
  - Model libraries
  - IP libraries
- While you are here …
  - Take time to celebrate all that we have accomplished

Looking Forward

cadence

- We still have much to do
  - Improve the languages, tools and libraries
  - Educate the masses
  - Branch out into new areas
  - Automate the model generation process ???
- If we are successful, we will have accomplished something great
  - We will have changed the way design and simulation are done
  - Remember to occasionally step back and look at the big picture
  - Admire it, then do something to make it better

how big can you dream?™