CAD Techniques to Automate Analog Cell Design

Rob A. Rutenbar

Carnegie Mellon University Pittsburgh, PA, USA rutenbar@ece.cmu.edu http://www.ece.cmu.edu/~rutenbar



Talk's Emphasis

- Analog cells
- CAD & methodology issues
- Synthesis, reuse, IP options

Example: one cell on analog-side of a mixed-signal ASIC

Mixed-Signal System-on-Chip



Outline

Critical design tasks

- Circuit design: topology, sizing, centering
- Circuit layout: devices, placement, routing

About analog cells

- Why analog cells != digital cells
- Different design and reuse scenarios
- Different intellectual property (IP) issues

CAD & methodology

- Current methodologies: today's industrial coping strategies
- Evolving techniques: leading-edge strategies, universities, startups

Conclusions

Historically—Why is this so Hard?

Mediocre analog point tools Aaargh Tools!

Ad hoc, incomplete capture of design intent

...what the heck is that?

Too much art,

not enough science

Outline

Critical design tasks

- Circuit design: topology, sizing, centering
- Circuit layout: devices, placement, routing
- About analog cells
- CAD & methodology
- Conclusions

Just What Is An "Analog Building Block?"

Typical analog cell

- ~5-75 devices (if bigger, usually use some hierarchy)
- Active devices (FET, BJT, etc) and passives (R, L, C)
- Often requires precision devices/passives for performance
- Often requires sensitive device placement, wiring



Analog Cells: Common Examples

Common cells



Common subsystems composed from basic cells



Analog Cell Design: Critical Tasks

No matter how you do it, you have to do these tasks

Basic device-level circuit design



Analog Cell Design: Critical Tasks

No matter *how* you do it, you have to do these tasks

Basic device-level layout design



Outline

Critical design tasks

About analog cells

- Why analog cells != digital cells
- Different design and reuse scenarios
- Different IP issues
- CAD & methodology

Conclusions

Why Is This Actually Difficult...?

Common misperceptions here

 Based mostly on familiarity with digital cells, digital libraries, and with digital design scenarios

Myth of "limited size"

- "Hey--only 50 transistors? How hard can that be to design?"
- "I don't see people obsessing over NAND gate design!"

Myth of "limited libraries"

"There's not much analog on chip, and it's mostly understood functions like A/D and D/A, so why not just design all the required cells once, put them in a library, reuse them?"

Reminder: Cell-Based Digital Design

Digital ASIC design

- Often starts from assumed library of cells (maybe some cores too)
- Supports changes in cell-library; assumed part of methodology
- Cell libraries heavily reused across different designs



Cell-Based Design Strategies: Digital

Where do digital cells come from?



Migration Tools: Old cells -> new cells



3rd Party IP: Emphasize portability, quick use



Manual, Custom Design: Proprietary or custom library



Cell-Based Design Strategies: Analog



Where do analog cells come from?

- Mainly manual design
- Often, manual redesign
- Not much device-level reuse
- Significant design effort here
- ◄ (Some IP is emerging...)

Why is this?

Analog Cells: Strong Fab Dependence

No digital abstraction to "hide" process

◄ No logic levels, noise margins, etc, on analog cells



Exploits physics of fab process, instead of avoiding it

- Individual devices designed to achieve precise behaviors
- Especially true with precision passive devices, which might require separate process steps (eg, double poly for capacitors)
- Circuits sensitive to all aspects of device/interconnect behavior, even modest changes due to simple dimensional shrinks

Analog Cells in Digital Processes

For SoC designs, want analog in standard digital process

Common problems

Low supply voltages preclude some circuit topologies



- Precision structures may be hard/impossible to build if special layers are unavailable (eg, poly-poly capacitor)
- Digital processes do not characterize devices for analog uses, eg, models do not capture subthreshold ops, matching, etc

Analog Cell Myths Revisited

Cell design difficulty, libraries

- ◄ OK, so, maybe it's hard to design an analog cell.
- So, why not just **design it once**, add to lib, reuse it?

Problem: leverage not same for analog libraries

 How big is a digital library? Big enough to get all necessary logic functions, IO variants, timing variants, drive strengths, to first order



Analog Cell Libraries: Dimensionality

Problem: many continuous specs for analog cells



Can't just build a practical-size, universal analog library

Analog Cell Libraries: Dimensionality

Dimensionality: Reality check

- OK, do you really need all 1000 of those variants?
- Can't we make do with just a few--like we do for digital gates?

Maybe: depends on your application



Analog Cells: Design & Reuse Strategies

2 major issues

- How do I make it easier to design this cell in the first place?
- How do I avoid designing it again? Can I reuse it, wrap/buy it as IP?
- Actually, *interdependent* set of technical responses here

Design: focuses at 3 levels

- Device-level design
- Cell-level design
- Core-level design (this is mostly ongoing research)

IP/reuse: focuses on 3 strategies

- Hard
- Firm
- Soft

Analog Cells: Design & Reuse Strategies

Simple taxonomy

		IP/REUSE					
		hard	firm	soft			
	device	Libraries of difficult, exotic device layouts	Parametric device layout generators				
Ν	cell	Libs of generic cell layouts for specific fab	Parametric templates for schematic, layout	Analog ckt synthesis and layout synthesis			
DESIG	core	Libs of useful block layouts for specific fab	Parametric templates for useful cores	Mixed-signal system synthesis			
		Focus is on <i>layout</i> reuse	Focus is on reusable circuit & layout <i>templates</i>	Focus is on <i>synthesis</i> , from spec to ckt to layout			

Outline

- Critical design tasks
- About analog cells

CAD & methodology

- Current methodologies: today's industrial coping strategies
- Evolving techniques: leading-edge strategies, universities, startups

Conclusions

Analog Cell Design & Reuse

What are people *most commonly* doing right now?

(Actually, they're mostly designing by hand, one device at a time...)

		IP/REUSE		
		hard	firm	soft
	device	Libraries of difficult, exotic device layouts	Parametric device layout generators	
Ν	cell	Libs of generic cell layouts for specific fab	Parametric templates for schematic, layout	Analog ckt synthesis and layout synthesis
DESIG	core	Libs of useful block layouts for specific fab	Parametric templates for useful cores	Mixed-signal system synthesis

First, Look at Device-Level Issues

Question: why the emphasis on *individual* devices...?

		IP/REUSE		
		hard	firm	soft
	device	Libraries of difficult, exotic device layouts	Parametric device layout generators	
Ν	cell	Libs of generic cell layouts for specific fab	Parametric templates for schematic, layout	Analog ckt synthesis and layout synthesis
DESIG	core	Libs of useful block layouts for specific fab	Parametric templates for useful cores	Mixed-signal system synthesis

Analog Device IP

Basic idea

- Analog cells require "difficult" device structures
- May need large devices, aggressive matching, unusual precision
- Can save device layouts in a library, or more commonly...
- ... write layout generators; may be provided by your foundry
- Implementations vary: can use commercial frameworks (Mentor GDT, Cadence PCELL), or write your own (C++, JAVA, etc)



Device-Level Design Issues

Focus is often on precision

 May want precise electrical characteristics, or matching among several devices, or precise ratios among devices

Central issues

- Analog devices are often large; e.g., a 40000/4 FET is not unusual
- Analog devices are often designed and laid out as a careful connection of many small, well-matched unit-size devices
- Guard-ring(s) common for electrical isolation

Result

Even one device may end up with a complex, large geometric layout

Example of Digital vs Analog Size Disparity

Digital FET



Analog FET

								 			 	202			(1998) (1998)		
11116668							1.57						E	and the second se		Siend II	ŝ
	H																
														IH II			
		, NIA		<u> </u>				: N 4.						<u>. N. I.</u>			

Common Device-Level Design Example

Consider a resistor which uses a resistive poly layer



High-precision R, add dummy bars at ends, well and guard ring





Interdigitated pair of precise-ratio 2:1 resistors

Industrial Example: Large Resistor Array



Courtesy Neolinear

Analog Device IP: Analysis

PRO

- Easier to get complex devices, device groups, laid out correctly
- Easier to get careful precision structures laid out correctly
- Insulates users from some of the nastier low-level foundry rules

CON

- Easy as a concept, hard in practice to build good generators
- Like any library (hard or generator), maintenance is an issue
- Does not help in sizing the circuit in the first place
- Does not remove requirement to place/route these devices into a functioning cell, with its own precision/performance subtleties

Next, Look at Hard Analog IP

Question: how much can you reuse complete layouts?

		IP/REUSE		
		hard	firm	soft
	device	Libraries of difficult, exotic device layouts	Parametric device layout generators	
DESIGN	cell	Libs of generic cell layouts for specific fab	Parametric templates for schematic, layout	Analog ckt synthesis and layout synthesis
	core	Libs of useful block layouts for specific fab	Parametric templates for useful cores	Mixed-signal system synthesis

Hard Analog Cell IP

Basic idea

- ◄ Hard IP (layouts) for common, generic cell functions
- Performance ranges estimated to target common application areas (eg, audio, video, LAN, IO driver, etc)
- Available from some foundries; also some 3rd party IP shops who design for standard digital fabs



Hard Analog Cell IP: Analysis

PRO

- Again, makes it easy to do some simple functions
- CON
 - Unlike digital libraries, unlikely that 100% of needed cells available
 - And, cell portfolio may differ significantly from vendor to vendor



Hard Analog Core IP (= Mixed-Signal IP)

Recent commercial idea

- Don't focus on basic cells, focus on bigger mixed-signal cores
- ▼ Industry standards **fix** many specs; target big ASIC foundries
- Interesting technical (& business) issues here



Analog Cores: Design Issues

Not necessarily all hard (fixed layout) approaches here

- Can do modest parameterization on cells--if they don't vary much
- Can relax foundry rules to create "subset" rules that work across several similar processes (e.g., foundry 0.25µm); lose some density and performance, gain some reuse
- Can design some of the circuits themselves to be programmable, eg, a programmable bandgap voltage reference, programmable gain stage etc. Again, trade some density/performance for reuse.

Of course...

- The people who actually design these cells still have all the problems of anybody who has to design custom analog
- **You** get lucky if you can buy it from them...

Hard Analog Core IP: Analysis

PRO

- Good idea--when it works technically, and as a business
- Scene evolving quite rapidly here
- Lots of common IO interfaces require analog; productivity benefit to be able to buy this functionality

CON

- Functionality, versatility still limited
- Obtaining an analog core != integrating an analog core; noise, coupling issues still difficult for big mixed signal ICs
- No guarantees to be able to find function, speed, power, etc. you need, in the fab process you use today...or tomorrow
- If you can't buy it...you still have to design it yourself
Focus Now on Design & Synthesis

OK, suppose you can't just buy the analog cells you need; what can you do to help *design* them faster, better?

		IP/REUSE		
		hard	firm	soft
DESIGN	device	Libraries of difficult, exotic device layouts	Parametric device layout generators	
	cell	Libs of generic cell layouts for specific fab	Parametric templates for schematic, layout	Analog ckt synthesis and layout synthesis
	core	Libs of useful block layouts for specific fab	Parametric templates for useful cores	Mixed-signal system synthesis

Cell-Level Strategies

Aside from doing everything manually, are there options?

Template-based design

- If you keep designing the same cells, for similar ranges of performance, try to capture central characteristics as a template
- Parameters fill in the template, change resulting design

Analog synthesis

- For more general case, specify critical performance constraints (electrical, geometric, etc)
- Synthesis tool uses numerical/geometric search to create circuit to match your design goals

Actually, these are variants on same technical theme...

Analog Cell Synthesis

Basic idea

Circuit synthesis:Circuit layout:

transform cell spec into sized/biased schematic transform device-level netlist into laid-out cell



About Synthesis Strategies

		IP/REUSE		
		hard	firm	soft
	device	Libraries of difficult, exotic device layouts	Parametric device layout generators	-
DESIGN	cell	Libs of generic cell layouts for specific fab	Parametric templates for schematic, layout	Analog ckt synthesis and layout synthesis
	core	Libs of useful block layouts for specific fab	Parametric templates for useful cores	Mixed-signal system synthesis

- Central idea is *not* to start from scratch on each new design.
- Difference here is who does most of the work...

Parametric templates:

Designer has initiative, makes effort Identifies commonalities among designs Extracts & encodes in reusable way More designer effort, less CPU time

Circuit/layout synthesis:

Designer specifies specs, constraints New discipline: need complete specs Tools do numerical, geometric search *More CPU time, less designer effort*

Cell-Level Analog Circuit Synthesis

Basic task



Design topology Design sizing/biasing Center (*maybe*)

Major strategies

- Procedural scripting
- Equation-based search-- flat and hierarchical
- Symbolic analysis
- Simulation-based optimization

Cell-Level Synthesis: Framework

Most approaches have this overall structure



Uses heuristic or numerical search

- Optimization engine:
- **Evaluation engine:**
- Cost-based search:
- proposes candidate circuit solutions
- evaluates quality of each candidate
- cost metric represents "goodness" of design

Synthesis: Procedural Scripting



Examples:

[DeGrauwe, JSSC'87] [Harvey, TCAD'92]

Basic idea

- Capture equations, models, calculations you keep re-solving in sensible, *solvable* order
- Write a program--a script--that does it
- Analogy: a spreadsheet

Issues

- OK for simple circuits, if you have good models, require modest parameter changes
- ◄ Hard (impossible) to write for complex ckts
- Can't get good analytical model for all specs
- Often problems with accuracy (vs. simulation models), robustness

Procedural Scripting: Mirror Example



Synthesis: Equation-Based Optimization



Examples:

[Koh, TCAD'90] [Hershenson, ICCAD'98]

Basic idea

- Capture equations, models, etc.
- Can't script everything analytically; use numerical search
- Styles vary: gradient search, annealing, geometric (convex) programming, ILP, ...

Issues

- Supports wider set of design, goals
- Writing correct equations still very hard, laborious; eqns often fragile, short lifespan
- Can't get good analytical model for all specs
- Accuracy problems (vs. simululation), numerical starting-point dependency

Eqn-Based Optimization: Example

Example: posynomial-formulation [Hershenson ICCAD98]

- If you can render all equations as posynomials (like polynomials, but real-valued exponents and only positive terms, eg 3x²y^{2.3}z⁻²), can show resulting problem is convex, has one unique minimum
- Geometric programming can solve these to optimality



Courtesy Mar Hershenson, Stanford

Optimal trade-off curves



Synthesis: Hierarchical Search



Examples:

[Harjani DAC'87] [Gielen, JCTh'95]

Basic idea

- Equation-based search, but use hierarchical representation of circuit
- Even small circuits have components: mirrors, references, gain stages, etc
- Build eqns for pieces, assemble into circuit

Issues

- More easily supports search over circuit topology and circuit sizing at same time
- Eases some of the burden of writing eqns--but still have to get eqns for components
- Some "deep" optimizations more difficult when circuit partitioned into pieces
- ▼ Same accuracy/robustness problems of eqns

Hierarchical Circuit Synthesis

- **Selection** = pick an abstract design style (sub-block topology)
- **Refinement** = decompose parent performance specs for child



[Harjani DAC'87]

Aside: Gets More Interesting at System Level

Use these ideas to explore system spec/architecture tradeoffs



Synthesis: Symbolic Analysis



Examples:

[Gielen, JSSC'90] [Wambacq, JSSC'95] [Sechen, TCAD'97]

Basic idea

- Automatically derive eqns--when you can
- Support powerful symbolic manipulation
- Add designer-derived eqns for remainder
- Use numerical optimization on these eqns

Issues

- Works well, but restricted to linear, weaklynonlinear specifications, behaviors
- Can work for continuous/discrete time (t/z)
- Can support useful interactive modes
- "Transient waveform" specs not well captured
- Same accuracy/robustness problems as eqns

Symbolic Analysis: Simple Example

Basic idea: prune symbolic form

< Symbolically manipulate determinant of admittance matrix



Symbolic Analysis: Realistic Example

Katholieke Univ. Leuven, ISAAC/SYMBA tool [Gielen JCTh'95]



Courtesy Georges Gielen, KUL

Bigger Circuit Example

KU Leuven, AMGIE tool,
[Gielen JCTh'95]

	Spec.	unit	Manual	Optimization
Detector capacitance	80	pF	80	80
Peaking Time	1.5	μs	1.1	1.1
Counting rate	200	kHz	200	294
Noise	< 1000	e ⁻ RMS	1000	905
Gain	20	mV/fC	20	21
Output Voltage range	2	V	2	2
Power consumption	< 40	mW	40	7



Synthesis: Custom Simulator + Optimizer



Examples:

[Medeiro, ICCAD'94] [Ochotta, TCAD'96]

Basic idea

- Build fast, custom simulator just for synthesis
- Simulate inside numerical search loop
- Better accuracy (avoid eqns), more CPU time

lssues

- Better accuracy, robustness
- Usually used with stochastic search, like annealing, to avoid many local minima
- Building a simulator is very hard
- Usually lacks features regarded as critical in commercial simulators; may still need eqns
- Requires yet more, different input deck info

Custom Simulator Example





Synthesis: Commercial Sim + Optimizer



Examples:

[Phelps, CICC'99] [Krasnicki, DAC'99] [Phelps, DAC'00] [Phelps TCAD'00]

Basic idea

- Designers are busy people--don't ask them to do extra work to do synthesis
- Treat the circuit + SPICE deck as the real IP
- Use exact same simulation/verification environment inside numerical optimization
- Use distributed workstations for CPU cycles

Issues

- Best accuracy, robustness
- Relies on clever, vigorous global search: annealing, genetic, pattern search
- ▼ No equations. None. Zero.
- ▼ CPU resource intensive

Example: Industrial Cell from TI

CMU ANACONDA tool [Phelps CICC99]



- Folded cascode opamp, high-drive output stage
 - **3**3 devs, 2 Rs, 2 Cs; 0.8um CMOS
- Difficult goals
 - ◄ High drive amplifier, 5Ωload
 - Nominal THD, 0.1%
 - 1kHz, 2.6V p-p input voltage



Larger Synthesis Example: TI ADSL CODEC

- [R. Hester, et al.. IEEE Int'l Solid-State Circuits Conf., 1999]
- [R. Phelps, et al., ACM/IEEE Design Automation Conf, 2000]



EQF Block: What It Looks Like

- 5 low-noise amps, ~100 passives, 36 program switches, 6 op-modes,
- ~400 devices, flat; ~2-3hrs to SPICE

Vin



© R.A. Rutenbar 2001

59 of 91

Synthesis Results: Noise vs Area

Full sizing/biasing ~10hours on 20 CPUs; all TI specs met



Synthesis Results: Spectral Mask

Eq0 Passband



One More Issue: Design Centering

Cannot ignore this entirely in analog synthesis flow

Optimization-based attacks can find "bad" corners of design space



2 broad, overall strategies

- Use first-order heuristics in numerical synthesis, then run centering
- Combine full statistical optimization in with numerical synthesis
- Examples: [Mukherjee TCAD'00], [Debyser, ICCAD'98]

Example: Centering Heuristics in Synthesis

Simple designer-derived constraints in ANACONDA synthesis

 Require matched devices to be "big"; sensitive devices to be "far enough" into desired region of operation (eg, 250mV above V_T)



Cell-Level Analog Layout Synthesis

Basic task



From schematic + geometric constraints to physical layout

Major strategies

- Enhanced polygon-editing
- Analog compaction & templates
- Physical synthesis: full device-level custom place/route

Layout: Enhanced Polygon Editing



Basic idea

- Pushing polygons is *painful*
- Add nicer editing features to your editor
- Examples: connectivity-maintenance, device-level layout generators, interactive routing, interactive DRC, etc.
- Real example: Cadence VirtuosoXL

lssues

- ◄ Good, useful stuff (ie, even beyond analog)
- Editability enhancements *always* popular in a tool you have to live with for *long* hours
- Still, not a *radical* productivity win...still really manual layout here, just nicer

Analog Layout: Compaction

Basic idea

Draw the layout loose, use compaction to tighten up

Issues

- Analog is not just about density--also about precision
- Symmetry, align, device internals, etc, critical; can't treat as digital



Analog Layout: Templates

Manually capture regularities as procedures for high-use cells

- Can mix device generators, cell generators, compaction ideas, etc.
- **The Still requires significant manual setup & maintenance investment**







Courtesy Koen Lampaert, Conexant



Another Template Example: CYCLONE

Optimizes LC-oscillators from specs to layout [Deranter DAC'00]

- Simulated annealing in combination with circuit simulations and some equations
- FEM simulations to characterize inductor coils
- Automatic template-based generation of VCO layout

Parameter	Low resistive sub CMOS 0.35?m	High resistive sub BiCMOS 0.65?m
Ls	1.26 nH	2.3 nH
Rs	6.5 ?	5.2 ?
Rad, W, Turns	109 ?m, 40??m, 2	141 ?m, 5??m, 2
Power	32 mW	8.2 mW



Analog Layout: Physical Synthesis

Basic tasks



Analog-Specific Optimizations: Place/Route

Placement symmetric and diffusion merging



Routing: differential symmetric and coupling avoidance

Wiring task with **Obstacle**

Symmetry No crosstalk





No symmetry No crosstalk



© R.A. Rutenbar 2001 70 of 91

Analog-Specific Optimizations: Merging

Optimal construction of diff-merged FET groups

TExample: merging with analog symmetry [Basaran DAC96]



Analog-Specific Optimizations: Wells

Example: dynamic optimization of wells/latchup during place





Courtesy Neolinear
University Layout Synthesis Example



Courtesy Georges Gielen, K.U. Leuven

Industrial Layout Synthesis Example

Proprietary CMOS comparator auto-layout; Neolinear NeoCelltm analog layout tool



Courtesy Neolinear

IP = Capture + Front-to-Back-Synthesis

Commercial example from Neolinear NeoCircuit/NeoCell flow



IP = Capture + Front-to-Back-Synthesis

Commercial example from Neolinear NeoCircuit/NeoCell flow



Analog Cell Ckt/Layout Synthesis: Analysis

PRO

- Good idea--getting more "real" with very recent work
- Supports more dynamic libraries, handles flexibility and variability requirements of custom analog in more natural way
- Removes many problems with hard IP (layout) bound to one fab
- Trades time/quality: good designs for most common cases; same trade-offs as for ASIC-style design

CON

- Very recent, research-oriented tools and flows
- Until recently only available from universities; in the last 24months, some startup activity

Last Point: Different Design Discipline

Synthesis: requires of users more *clarity of intention*

- Digital folks have already figured this out for cell-based synthesis
- Analog folks will need to run up the same learning curve
- CAD tools still can't read designers minds (yet)



Example: constraint capture/editing

Wrong...

Just like that, but *better*...



What's Left to Do: System-Level Design

OK, you design/buy/synthesize all your cells...then what? Chip-level assembly. (...and, problems don't get easier)

		IP/REUSE		
		hard	firm	soft
DESIGN	device	Libraries of difficult, exotic device layouts	Parametric device layout generators	
	cell	Libs of generic cell layouts for specific fab	Parametric templates for schematic, layout	Analog ckt synthesis and layout synthesis
	core	Libs of useful block layouts for specific fab	Parametric templates for useful cores	Mixed-signal system synthesis

"When Bad Things Happen to Good Cells"

Noise upsets on delicate/precise analog

- From noisy digital wires nearby
- ▼ From noisy shared substrate
- From noisy power grid

Thermal issues

- Large digital blocks switching, or large analog devices: heat
- Temperature changes can affect precision analog

Solutions

- Segregate (away from digital)
- Isolate, shield (from noise)



One Assembly Example: IBM Data Channel

Digital switching is the source of (almost) all evil for analog



CAD Solution: Power Grid Synthesis

Auto power grid synthesis

- Re-synthesized IBM grid
- Power grid routed, sized
- Power IOs assigned
- Substrate contacts configured
- Decoupling caps added





Static IR Drop (mV)



Conclusions

Analog cells are not like digital cells, viz CAD & methodology

- Not as easily library-able; can't build one "complete" library
- Tightly bound to fab process, have difficult precision requirements

Design strategies

- Device-level IP: many people use libraries or generators here
- Cell-level design: templates (designer-initiative), synthesis (tool-based) are workable. Synthesis increasingly real, commercial.

IP/Reuse strategies

- Hard IP is often hard to use; even more true for analog
- Emerging cores for common interface functions, targeting major foundries, hide much of the unpleasantness here; very new business

Closing Observation: What We Really Want

Practical analog synthesis / IP / reuse



Analog IP

General Analog CAD Survey

- R. A. Rutenbar, "Analog Design Automation: Where are We? Where are we Going?", Proc. 1993 IEEE Custom Integrated Circuits Conference (CICC), May 1993.
- L. Richard Carley, Georges Gielen, Rob A. Rutenbar, Willy Sansen, "Synthesis Tools for Mixed-Signal ICs: Progress on Frontend and Backend Strategies," Proc. 1996 ACM/IEEE Design Automation Conference, June 1996.
- Computer Aided Design of Analog and Mixed-Signal ICs, B. Antao, G. Gielen, R.A. Rutenbar, eds., IEEE Press, to appear late 2000.
- G. G.E. Gielen, R.A. Rutenbar, "Computer Aided Design of Analog and Mixed-Signal Integrated Circuits", *Proceedings of the IEEE*, December 2000, to appear.

IP Issues

- Steve Ohr, "Analog IP Slow to Start Trading", EETimes, Issue 1053, March 22 1999. http://www.eet.com (Steve Ohr covers analog design/EDA for EETimes)
- K.C. Murphy, "A Time for Analog Design", Electronic News Online, August 2 1999. http://www.electronicnews.com/enews/BackIssues/BackIssues.asp
- http://www.vsia.com -- Virtual Socket Interface Alliance working on specs for interchange of analog IP

Analog Synthesis

- M.G.R. DeGrauwe et. al, "IDAC: An Interactive Design Tool for Analog CMOS Circuits", IEEE Journal of Solid-State Circuits, December 1987.
- H.Y. Koh, C.H. Sequin, P.R. Gray, "OPASYN: A Compiler for CMOS Operational Amplifiers", *IEEE Transactions on CAD*, Feb. 1990.
- R. Harjani, R.A. Rutenbar and L. Richard Carley, "OASYS: A Framework for Analog Circuit Synthesis", *IEEE Transaction on CAD*, Dec. 1989.
- G. Gielen, Walscharts, W. Sansen, "Analog circuit design optimization based on symbolic analysis and simulated annealing", *IEEE JSSC*, June 1990.
- J. P. Harvey, M.I. Elmasry and B. Leung, "STAIC: An Interactive Framework for Synthesizing CMOS and BiCMOS Analog Circuits", *IEEE Transaction on CAD*, Nov. 1992.
- P.C. Maulik, L.R. Carley and R.A. Rutenbar, "Integer Programming Based Topology Selection of Cell-Level Analog Circuits", *IEEE Transactions on CAD*, April 1995.
- B. Antao and A. Brodersen, "ARCHGEN: Automated Synthesis of Analog Systems", IEEE Transaction on VLSI Systems, June 1995.
- W. Kruiskamp and D. Leenaerts, "DARWIN: CMOS Opamp Synthesis by Means of a Genetic Algorithm", *Proc. 32nd ACM/IEEE DAC*, pp. 433-438, 1995.

Analog Synthesis, cont.

- F. Medeiro, F. Fernandez, R. Dominguez-Castro, A. Rodriguez-Vazquez, "A Statistical Optimization Based Approach for Automated Sizing of Analog Cells", *Proc. ACM/IEEE ICCAD*, 1994.
- E.S. Ochotta, R. A.Rutenbar and L.R. Carley, "Synthesis of High-Performance Analog Circuits in ASTRX/ OBLX", *IEEE Transactions on CAD*, March 1996.
- M. Hershenson, S. Boyd, T. Lee, "GPCAD: a Tool for CMOS Op-Amp Synthesis", *Proc. ACM/IEEE ICCAD*, pp. 296-303, 1998
- M. Krasnicki, R. Phelps, R. Rutenbar, L. R. Carley, "MAELSTROM: Efficient Simulation-Based Synthesis for Custom Analog Cells", *Proc ACM/IEEE DAC*, June 1999.
- R. Phelps, M. Krasnicki, R. Rutenbar, L. R. Carley, J. Hellums, "ANACONDA: Robust Synthesis of Analog Circuits Via Stochastic Pattern Search", *Proc. IEEE CICC.*, May 1999.
- R. Phelps, M. Krasnicki, R. Rutenbar, L. R. Carley, J. Hellums, "A Case Study of Synthesis for Industrial-Scale Analog IP: Redesign of the Equalizer/Filter Frontend for an ADSL CODEC", Proc. ACM/IEEE Design Auto Conference, June 2000.
- R. Phelps, M. Krasnicki, R. Rutenbar, L. R. Carley, J. Hellums, ANACONDA: Simulation-Based Synthesis of Analog Circuits Via Stochastic Pattern Search, IEEE Trans CAD, June 2000.
- T. Mukherjee, L.R. Carley and R.A. Rutenbar, "Efficient Handling of Operating Rangeand Manufacturing Line Variations in Analog Cell Synthesis," *IEEE Trans CAD*, August 2000.
- G. Debyser, G. Gielen, "Efficient analog circuit synthesis with simultaneous yield and robustness optimization", *Proceedings IEEE/ACM ICCAD*, pp. 308-311, November 1998.

Symbolic Analysis

- G. Gielen, H. Walscharts, W. Sansen, "ISAAC: A Symbolic Simulator for Analog Integrated Circuits," IEEE Journal of Solid-State Circuits, Vol 24, No. 6, pp. 1587-1597, Dec 1989
- F. Fernandez, A Rodriguez-Vazquez, J. Huertas, "Interactive AC Modeling and Characterization of Analog Circuits Via Symbolic Analysis," *Kluwer Journal on Analog Integrated Circuits and Signal Processing*, Vol. 1, pp. 183-208, November 1991.
- J. Starzyk,, A. Konczykowska, "Flowgraph Analysis of Large Electronic Networks," IEEE Transactions on Circuits and Systems, Vol. 33, No. 3, pp 302-315, March 1986.
- B. Li, D. Gu, "SSCNAP: A Program for Symbolic Analysis of Switched Capacitor Circuits," IEEE Transactions on CAD, Vol. 11, No. 3, pp. 334-340, March 1992.
- P. Wambacq, F. Fernandez, G. Gielen, W. Sansen, A. Rodriguez-Vazquez, "Efficient symbolic generation of approximated small-signal characteristics of analog integrated circuits," *IEEE JSSC*, pp327-330, March 1995.
- Q. Yu and C. Sechen, "A Unified Approach to the Approximate Symbolic Analysis of Large Analog Integrated Circuits", IEEE Transactions on Circuits and Systems-I, vol.43, pp 656-669, August 1996
- Q. Yu and C. Sechen, "Efficient Approximation of Symbolic Network Functions Using Matroid Intersection Algorithms," *IEEE Transaction on CAD*, vol. 16, no. 10. pp. 1073-1081, October 1997.
- C. Shi, X. Tan, "Symbolic Analysis of Large Analog Circuits with Determinant Decision Diagrams," *Proc.* ACM/IEEE ICCAD, pp. 366-373, 1997.

Analog Layout

- J. Rijmenants, J.B. Litsios, T.R. Schwarz, M.G.R. Degrauwe, "ILAC: An Automated Layout Tools for Analog CMOS Circuits," *IEEE JSSC*, Vol. 24, No. 4, pp. 417-425, April 1989.
- J.M. Cohn, D.J. Garrod, R.A. Rutenbar, L.R. Carley, "KOAN/ANAGRAM II: New Tools for Device-Level Analog Placement and Routing," IEEE JSSC, Vol. 26, No. 3, March 1991.
- U. Choudhury, A Sangiovanni-Vincentelli, "Automatic Generation of Parasitic Constraints for Performance-Constrained Physical Design of Analog Circuits," *IEEE Trans. CAD*, Vol. 12, No. 2, pp. 208-224, February 1993.
- E. Malavasi, E. Felt, E. Charbon and A. Sangiovanni-Vincentelli, "Automation of IC Layout with Analog Constraints," *IEEE Transactions on CAD*, vol. 15, no. 8, August 1996.
- K. Lampaert, G. Gielen, W. Sansen, "A Performance-Driven Placement Tool for Analog Integrated Circuits," IEEE JSSC, Vol. 30, No. 7, pp. 773-780, July 1995.
- E. Malavasi, E. Felt, E. Charbon, A. Sangiovanni-Vincentelli, "Symbolic Compaction with Analog Constraints, " Int. J. Circuit Theory and Applic., Vol.23, No.4, pp. 433-452, July/Aug. 1995
- E. Malavasi, D. Pandini, "Optimum CMOS Stack Generation with Analog Constraints," IEEE Transactions on CAD, Vol. 14, No. 1, pp. 107-12, Jan. 1995.
- B. Basaran, R.A. Rutenbar, "An O(n) Algorithm for Optimum CMOS Device Stacking with Analog Constraints," Proc. ACM/IEEE Design Automation Conference, June 1996

Analog Layout

- Mitra, R.A. Rutenbar, L.R. Carley, D.J. Allstot, "Substrate-Aware Mixed-Signal Macrocell Placement in WRIGHT," *IEEE JSSC*, Vol. 30, No. 3, pp. 269-278, March 1995.
- S. Mitra, S. Nag, R.A. Rutenbar, and L.R. Carley, "System-Level Routing of Mixed-Signal ASICs in WREN," Proc. ACM/IEEE ICCAD, Noc. 1992.
- B.R. Stanisic, N.K. Verghese, R.A. Rutenbar, L.R. Carley, D. J. Allstot, "AddressingSubstrate Coupling in Mixed-Mode ICUs: Simulation and Power Distribution Synthesis," *IEEE JSSC*, Vol. 29, No. 3, Mar. 1994.