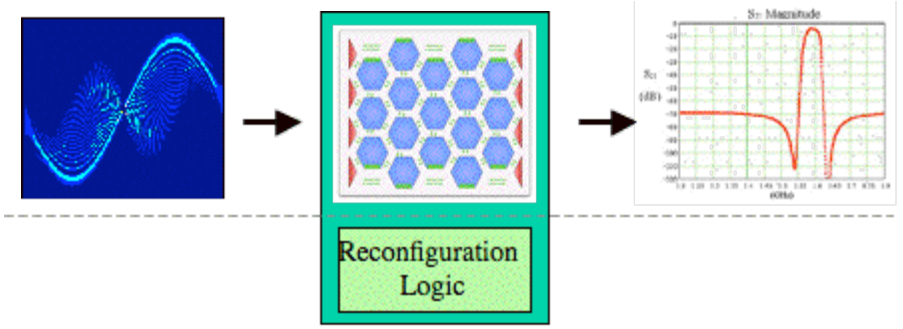
A Field Programmable Analog Array (FPAA) for reconfigurable instantiation of continuous-time filters.

Students: Joachim Becker, Fabian Henrici, Stanis Trendelenburg
Advisors: Prof. Maurits Ortmanns, Prof. Yiannos Manoli

Microelectronics Laboratory
Department of Microsystems Engineering (IMTEK)
University of Freiburg, Germany

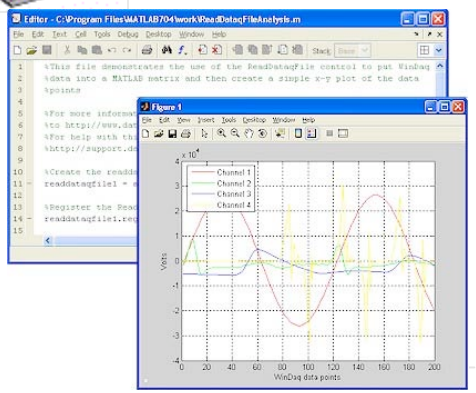
Field Programmable Analog Array

- Reconfigurable analog filter with time-continuous signal processing.



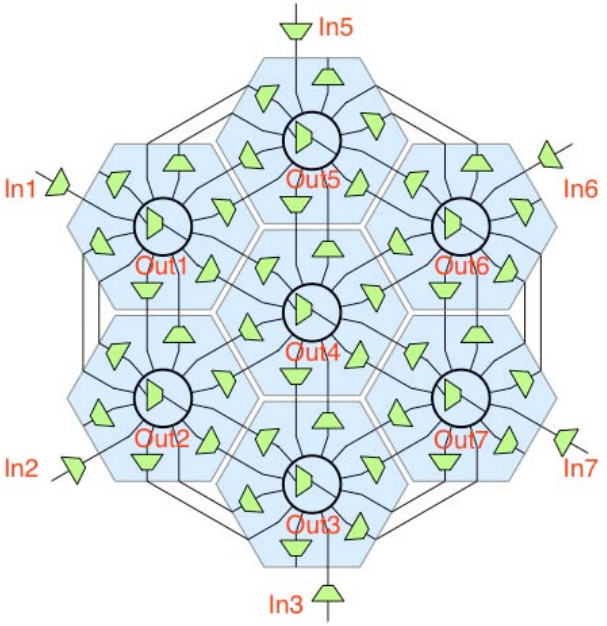
- Analog transfer-function
  - Analog signal (0.1 - 200 MHz)
  - Processing through filter on FPAA
  - Spectrum Analysis or A/D conversion

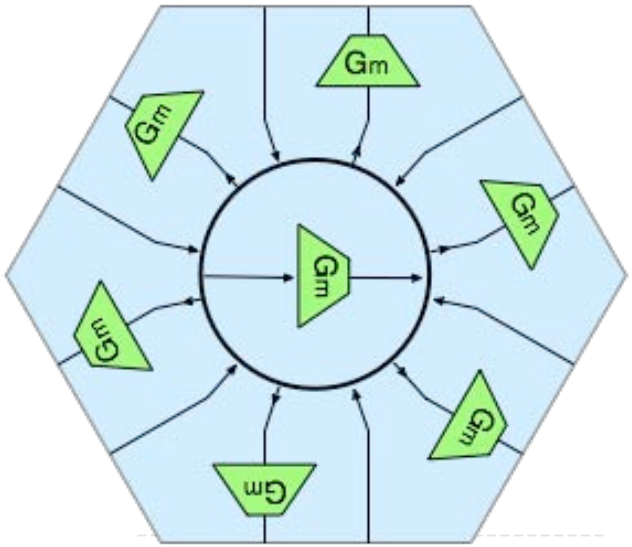
- Digital logic for filter reconfiguration
  - Graphical entry of configuration on PC through MATLAB toolbox
  - Download via USB/JTAG interface to chip
  - Setting of digital configuration memory
  - Instancing of filter on FPAA



FPAA structure

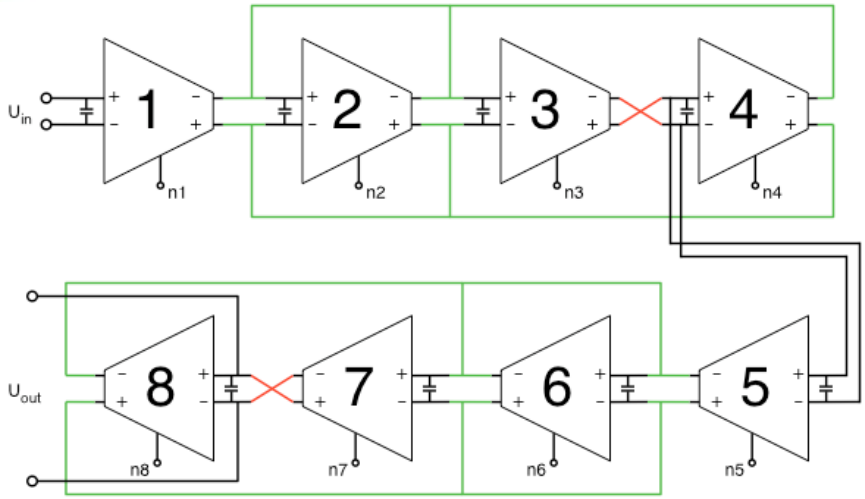
- Novel hexagonal topology
  - provides configurable signal routing between cells
  - allows all orders of feedback
  - arbitrary filter structures possible
  - no global routing network and bandwidth limiting switches
- Each cell has
  - One input node
  - Gm-cells to 6 neighbors + 1 for self-feedback
  - Each branch is a digitally tunable transconductor
  - Each branch has an inverting and non-inverting output

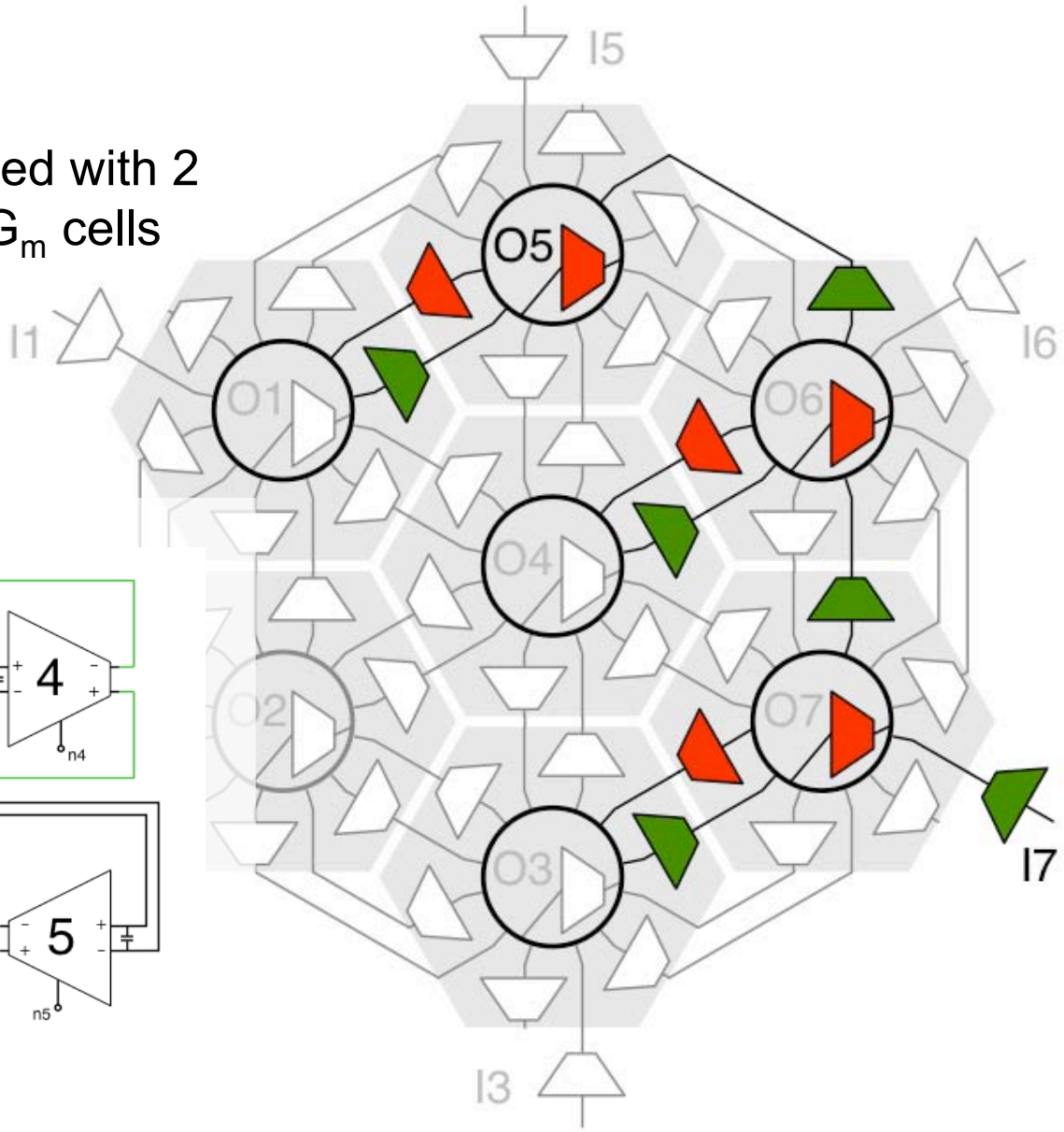




Example

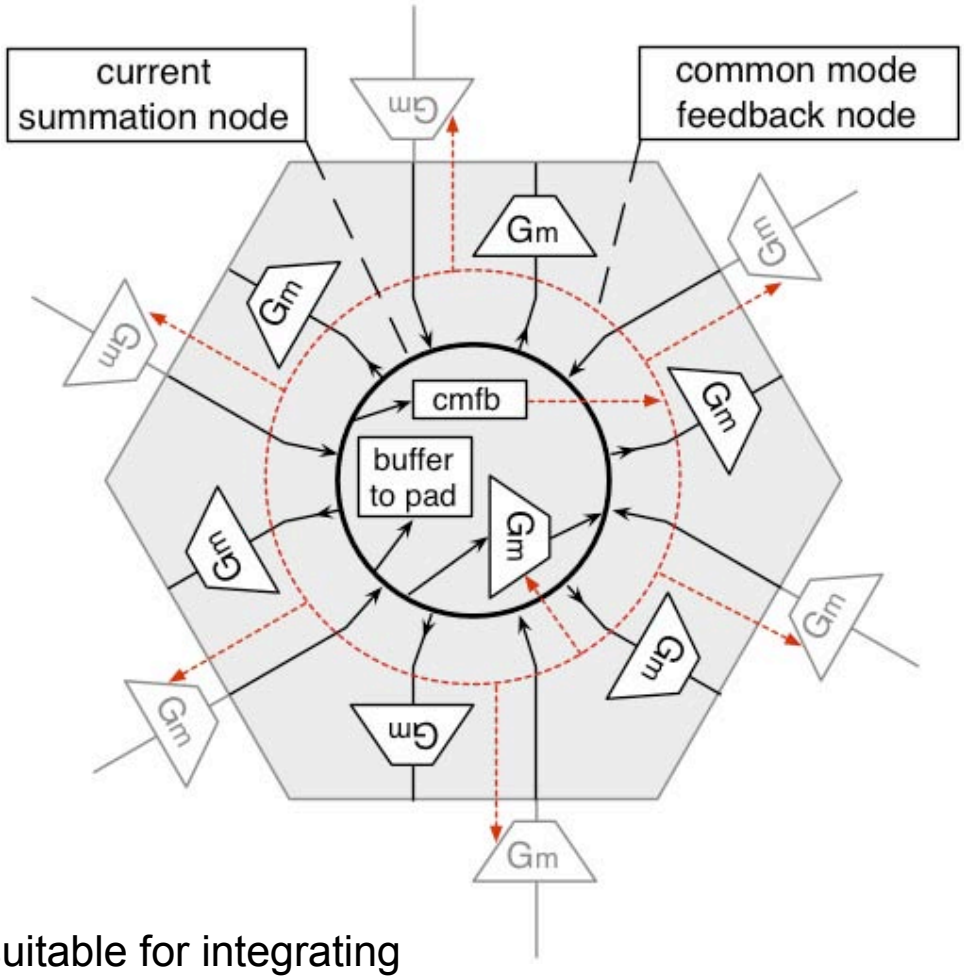
- biquad achieved with 2 nodes and 4 G_m cells
- cascade of biquads for higher order





CAB Details

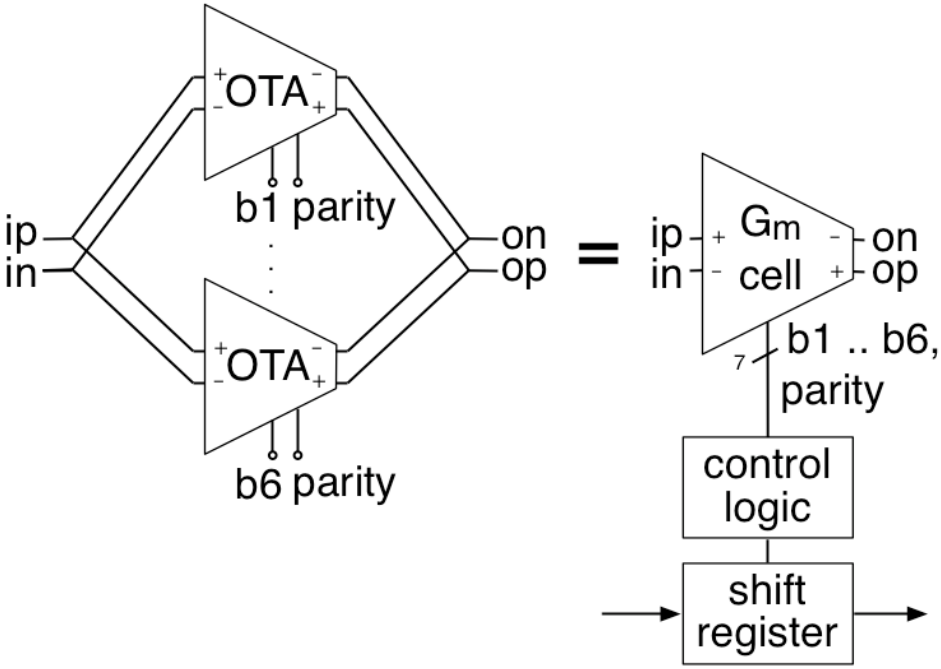
- each CAB represents one node of a filter
- each node is observable through an output buffer



- CMFB circuit regulates all external G_m cells driving that node
- filter build-up: G_m - C
  - parasitic capacitance suitable for integrating
  - control parasitics in each switching state of G_m cell

Tunable G_m-cells

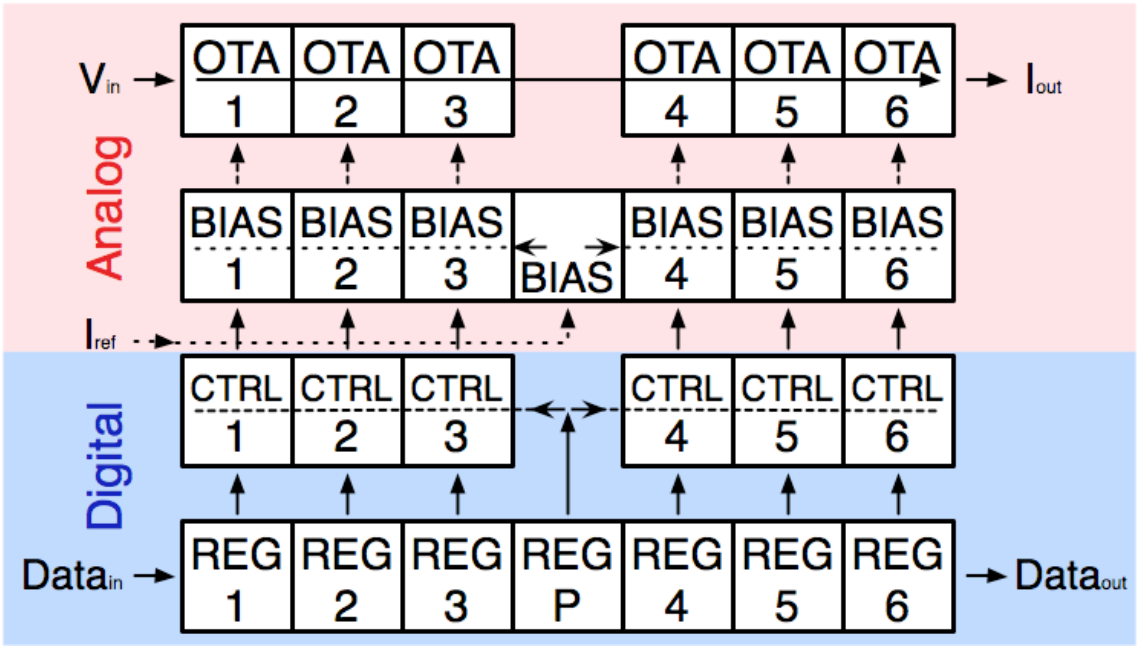
- 6 switchable OTAs
- connected in parallel
- constant input capacitance
- negligible change of output capacitance



$$f_t = \frac{G_m}{C_{load}} = \frac{\sum g_m}{\sum c_{par}} = \frac{n \cdot g_m}{6 \cdot c_{par}} \quad \text{with} \quad n \text{ in } [1..6]$$

- digital shift register for setting

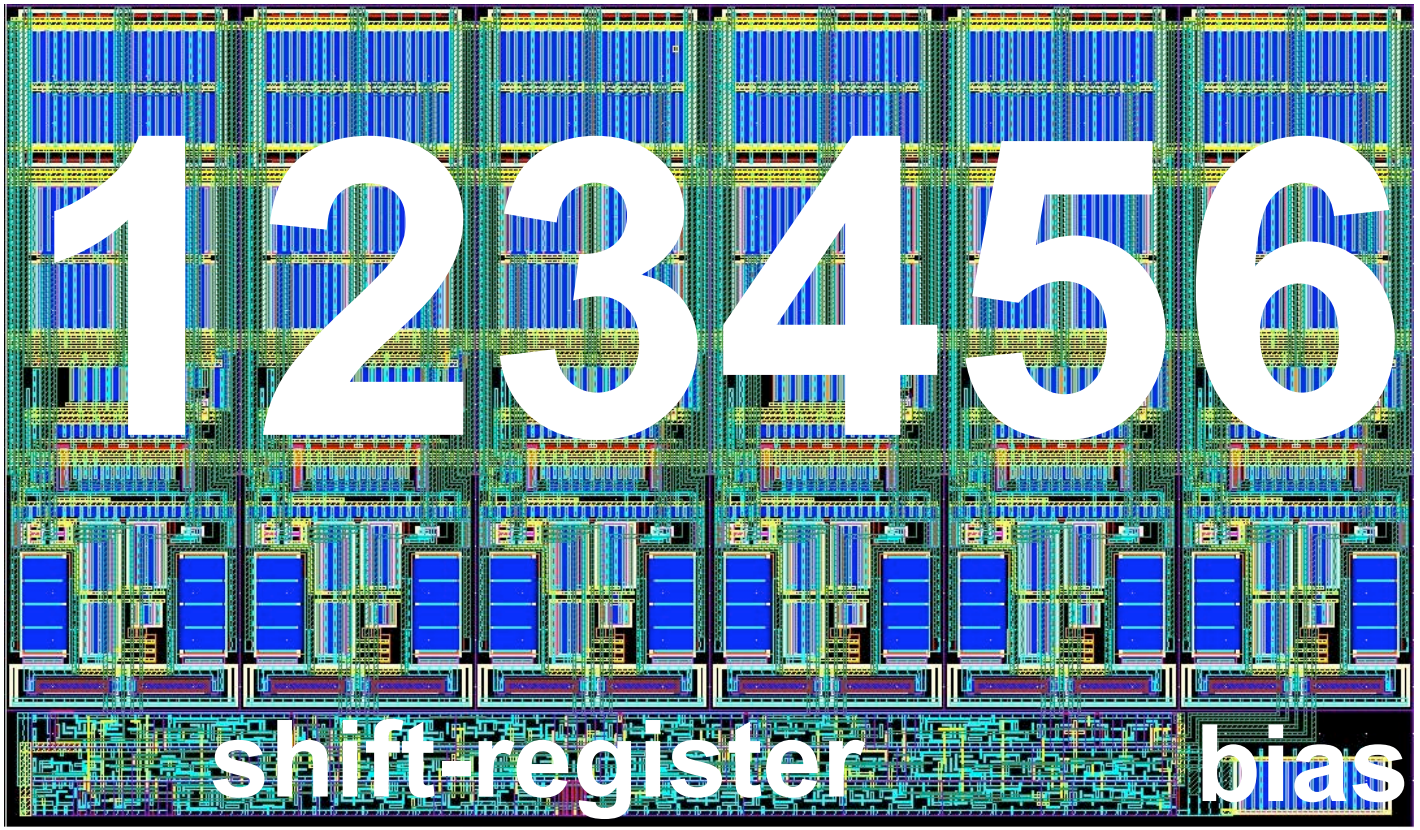
Configuration setting through shift-register



- 6 parallel connected OTAs are controlled by bias voltages
- Bias voltages are switched according to digital control signals
- Control signals are generated from information in shift-register

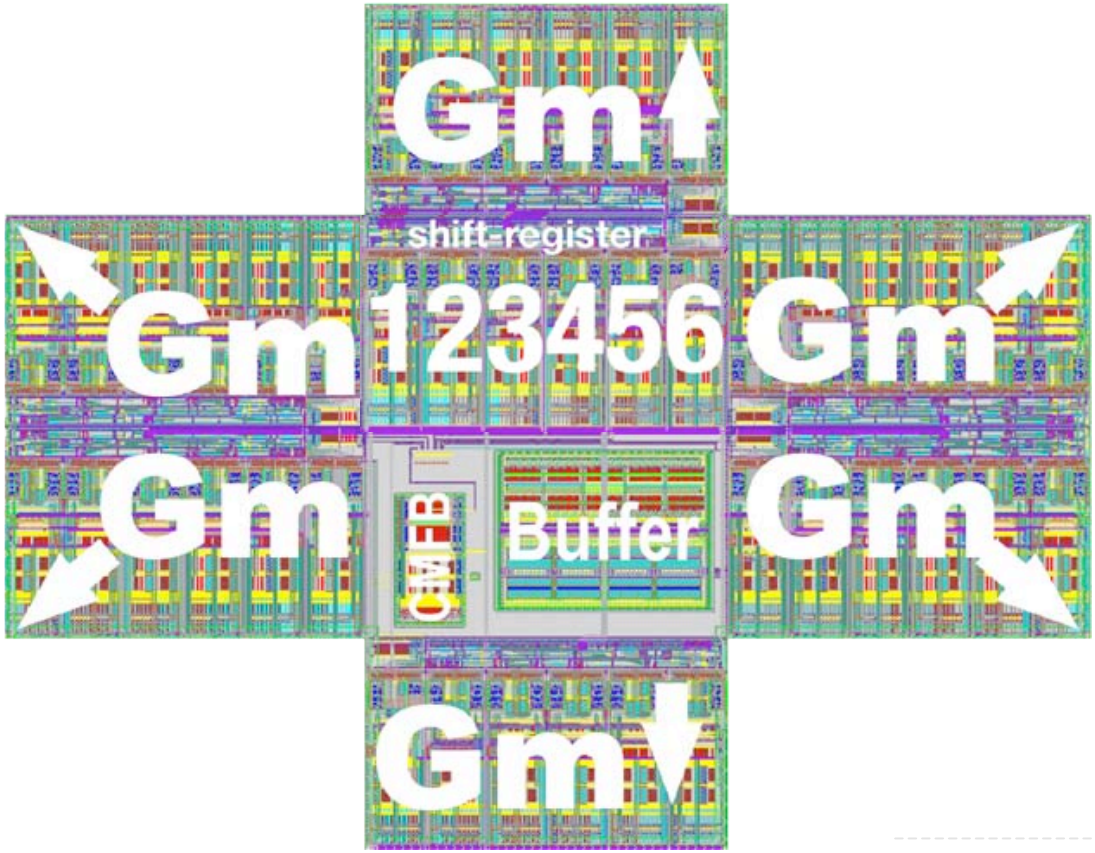
FPAA Layout (bottom level: tunable Gm cell)

- 6 switchable transconductance amplifiers
- reference-voltage current-mirrors
- digital configuration logic

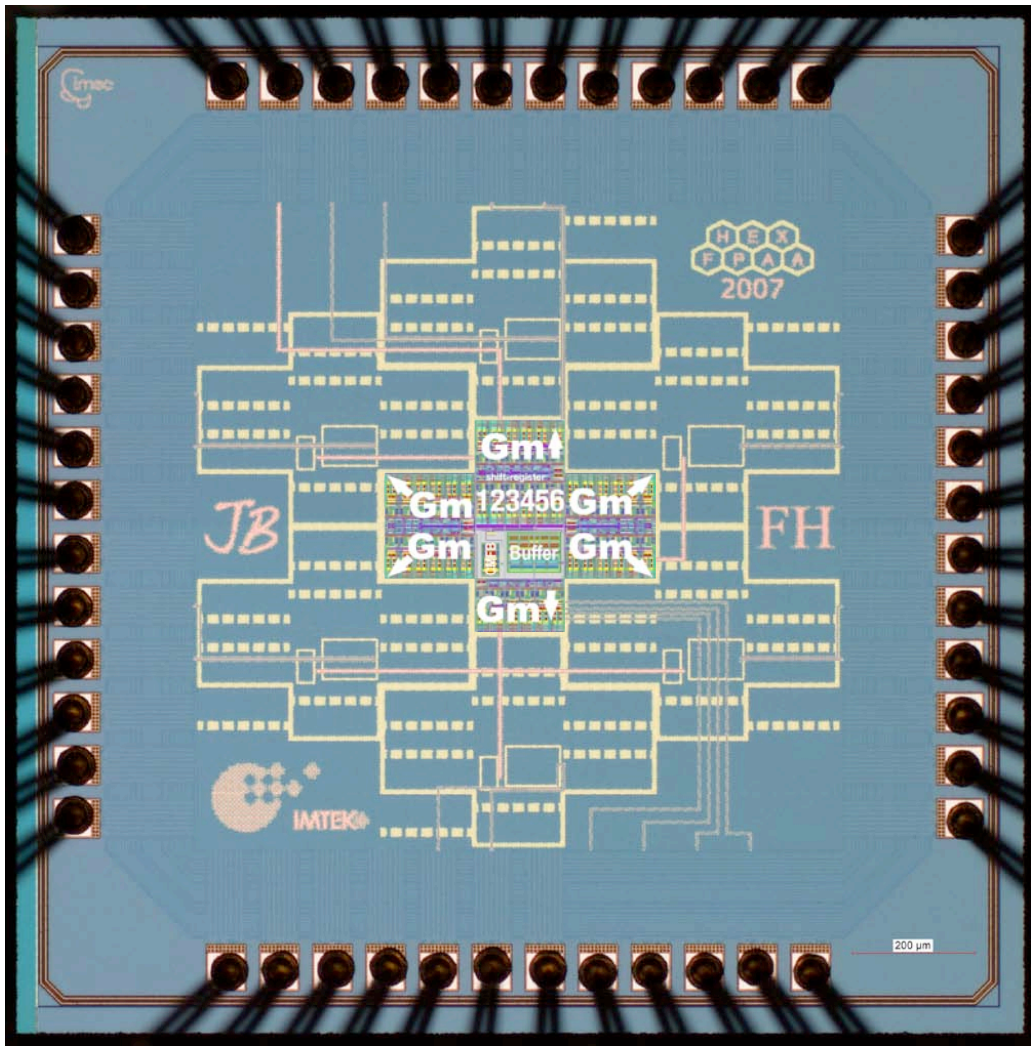


FPAA Layout (intermediate level)

- 7 tunable transconductance amplifiers (branches)
- common-mode feedback circuit, analog output buffer



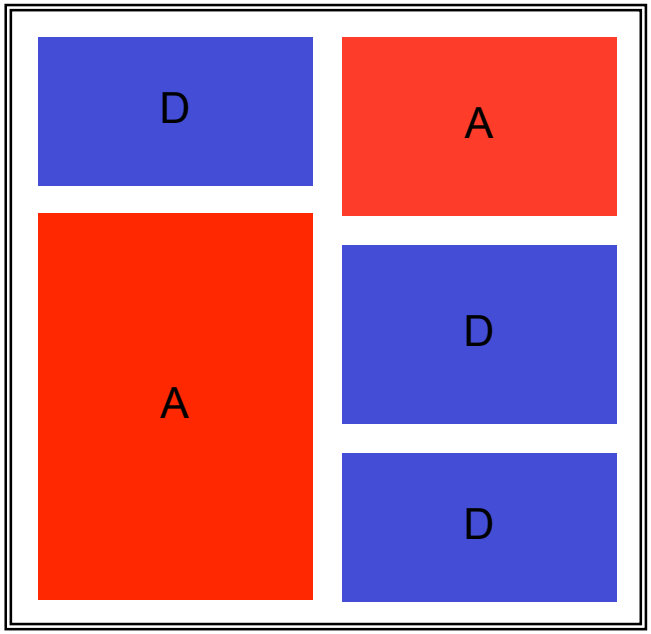
Chip Photo of FPAA with 7 CABs

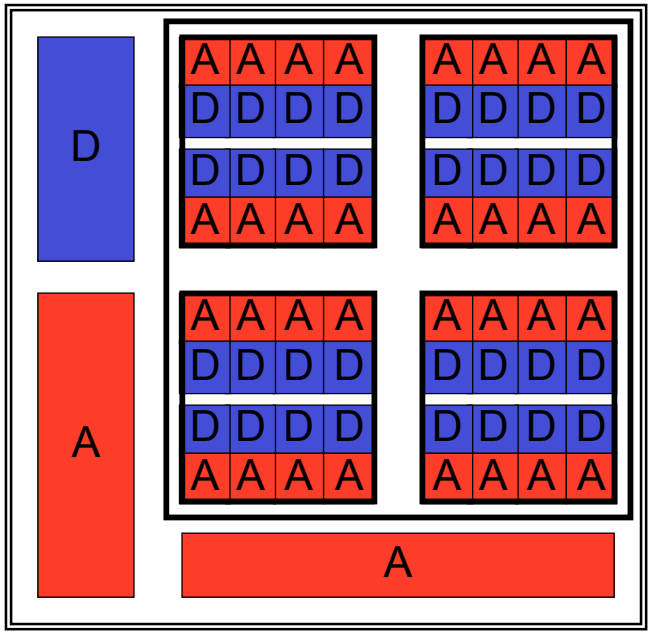


- 7 configurable analog blocks
- padring IOs
- 0.13 µm CMOS
- 1.5 x 1.5 mm^2
- 6 diff. inputs
- 7 diff. outputs
- 7 nodes
- 55 G_m cells
- 385 config bits

- Achieved using Cadence Tools
- presented at ISSCC 2008

Methodology aspects: mixed-signal hierarchy

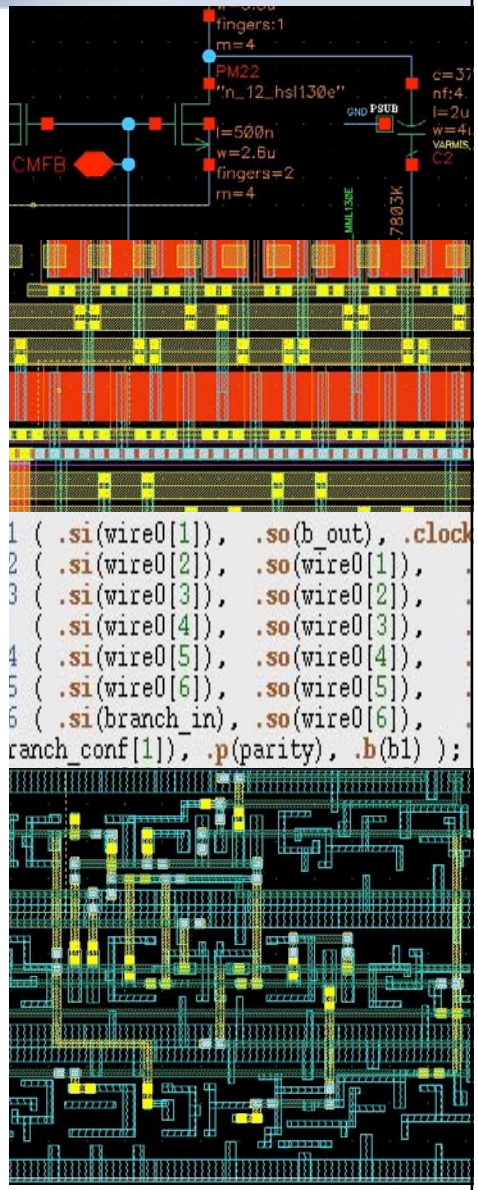




- Hierarchical design with mixed-signal regular structures
  - Leaf-cell: Analog OTA and digital shift-register
  - Bottom-up assembly of CABs and CAB-Array
  - Neither *Analog-on-top* nor *Digital-on-top,* rather ***mixed-anywhere***...

Assembly of Mixed-Signal Cells

- things to be considered
  (decisions made for this particular design)
  - analog designkit is reference for tapeout
  - digital designkit faulty, uses different layer table
  - most of the chip-area is analog devices, take advantage of Virtuoso features
  - full-custom place and route, hand-drawing ability, “feeling” for symmetry and hexagonal layout
  - hierarchy of schematics rather then HDL-netlists
  - analog or AMS simulation integrated in IC-package
- Analog-on-top seems to be the way
- issues to address
  - export digital layouts from Encounter to Virtuoso
  - connection of analog and digital netlists
  - digital global routing, clock tree synthesis
  - global timing analysis

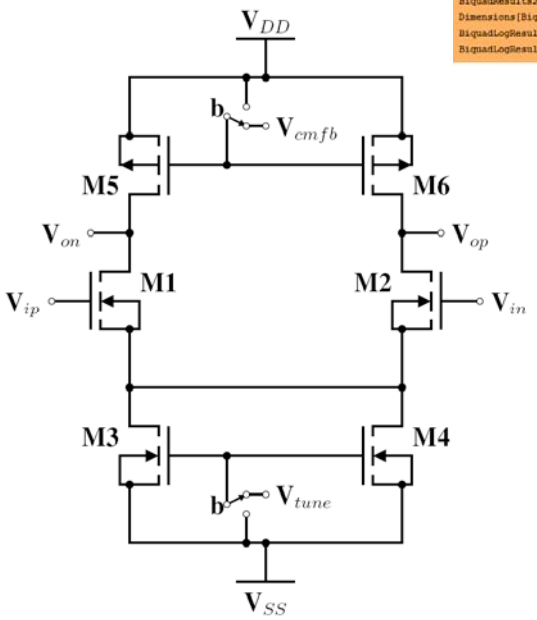


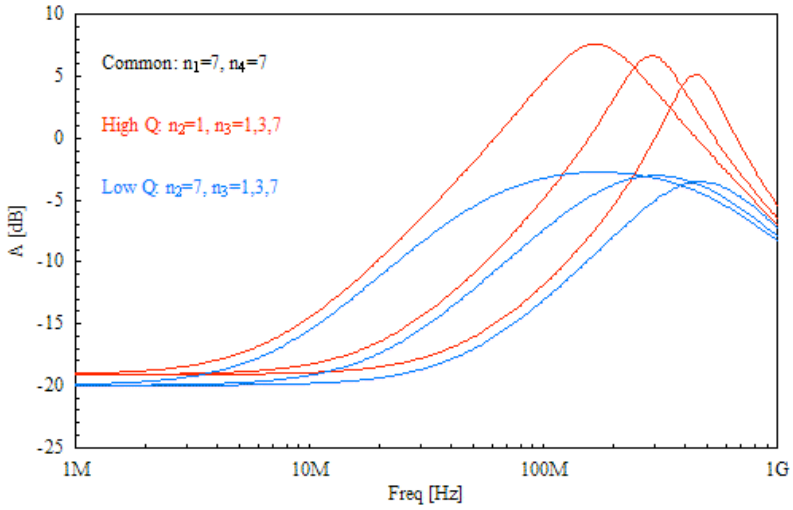
Simulation of analog part at transistor-level

- Transfer function is dependent on configuration setting.
- During analog design, many sweeps of simulations had to be run.
- Changes in the transistor-level design had to be iterated many times.

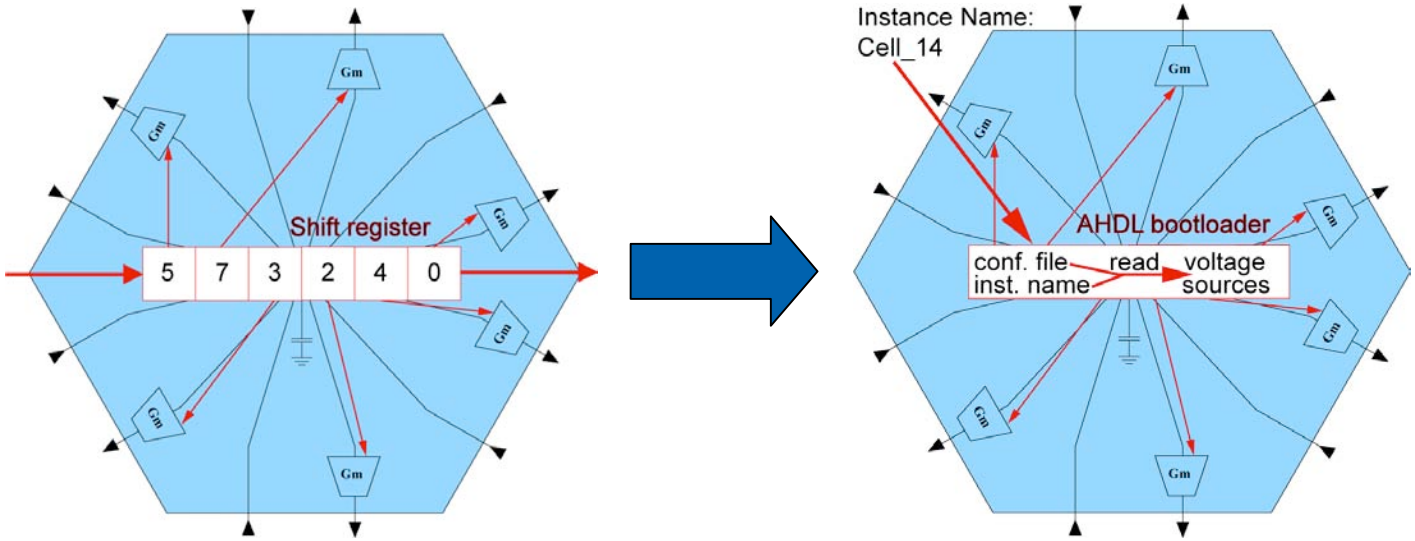
```
SweepParameter = {{1, 7}, {1, 3, 7}, {1, 3, 7}};
SubDir = "biquad";
Results = ParaSweepLoader[SweepParameter, SubDir];
```

```
BiquadResults = (Results[[1, 3, 3, 9, All, All]], Results[[1, 2, 3, 9, All, All]], Results[[1, 1, 3, 9, All, All]]);
Dimensions[BiquadResults]
BiquadLogResults = BiquadResults /. (x_, y_) → (x, 20 Log[10, y]);
BiquadLogResultsLogPlots = (LogLinearListPlot[#, PlotStyle → {Hue[0]}]) & /@ BiquadLogResults;
BiquadLogResultsLogPlotsb = (LogLinearListPlot[#]) & /@ BiquadLogResults;
BiquadResults2 = (Results[[2, 3, 3, 9, All, All]], Results[[2, 2, 3, 9, All, All]], Results[[2, 1, 3, 9, All, All]]);
Dimensions[BiquadResults2]
BiquadLogResults2 = BiquadResults2 /. (x_, y_) → (x, 20 Log[10, y]);
BiquadLogResultsLogPlots2 = (LogLinearListPlot[#, PlotStyle → {Hue[0.6]}]) & /@ BiquadLogResults2;
```





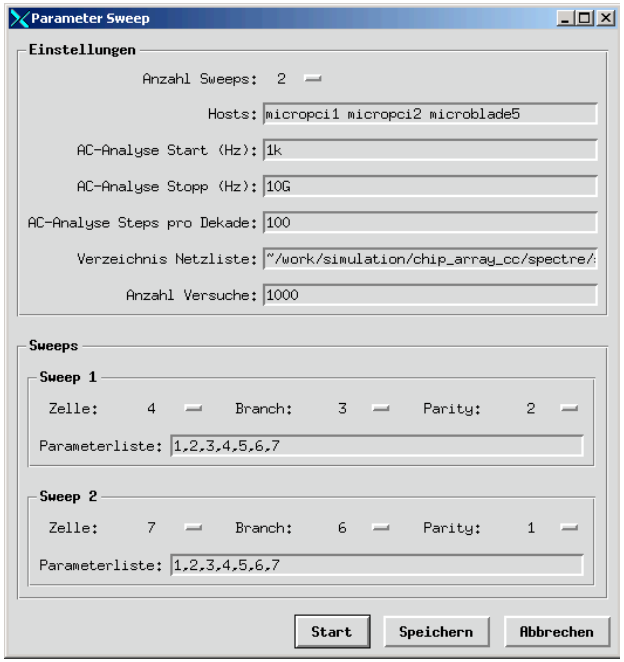
AHDL implementation



- Read configuration bit stream from shared file and load into matrix
- Decide by own instance name which cell is to be configured
  - instname = $str("%l");
  - pos = $strstr(instname,"Cell_")
  - cell_number = $strtoint($substr(instname,pos+5,pos+7))
- Set voltage outputs to corresponding levels
- Start analog simulation

Distributed simulation

- Job manager
  is started by GUI and runs in the background



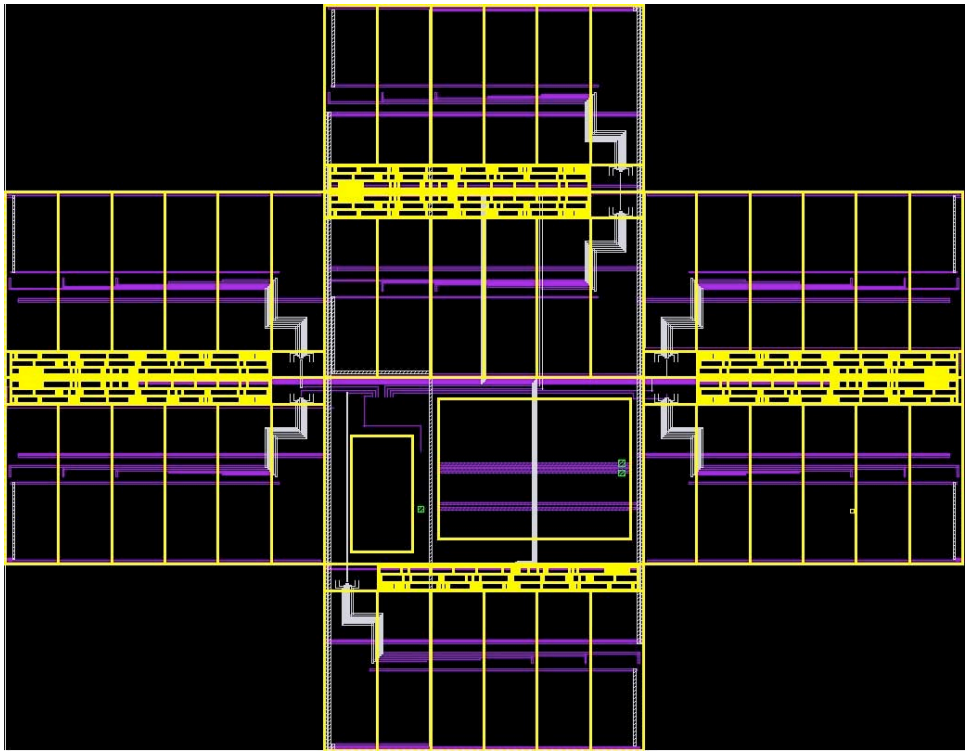
- Configuration file and OCEAN script for each setting of parameter sweep are written to UNIX home directory

- Spectre is started on each host by calling OCEAN interpreter via ssh

```
IN1
0000000
CAB1
0000000
0000000
0000000
0000000
0000000
0000000
0000000
0000000
CAB2
0000000
0000000
0000000
0000000
0000000
0000000
0000000
0000000
IN2
0000000
IN3
0000000
CAB3
0000000
  ...
```

Digital top level

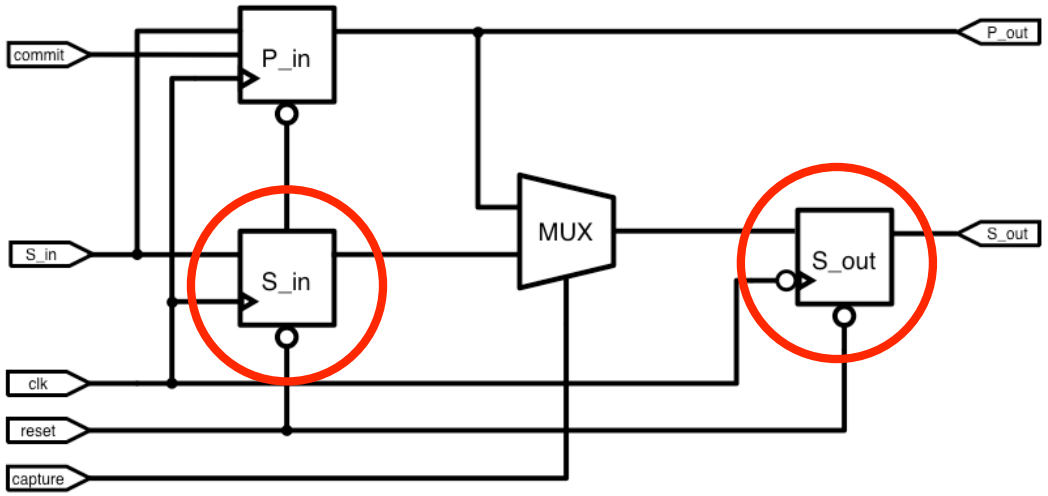
- Distributed digital parts
- Clock tree synthesis
- Buffers
- Antenna-Diodes



- Digital hierarchy hidden in Virtuoso schematics
- Automated CTS and timing analysis
  - back to Encounter?
  - netlists to Verilog
  - GDS export
  - return to Virtuoso?
- For simple digital parts, it could be done by hand
  - balanced, symmetrical control signal routing
  - precautionary placement of buffers and antenna-diodes

Digital timing verification

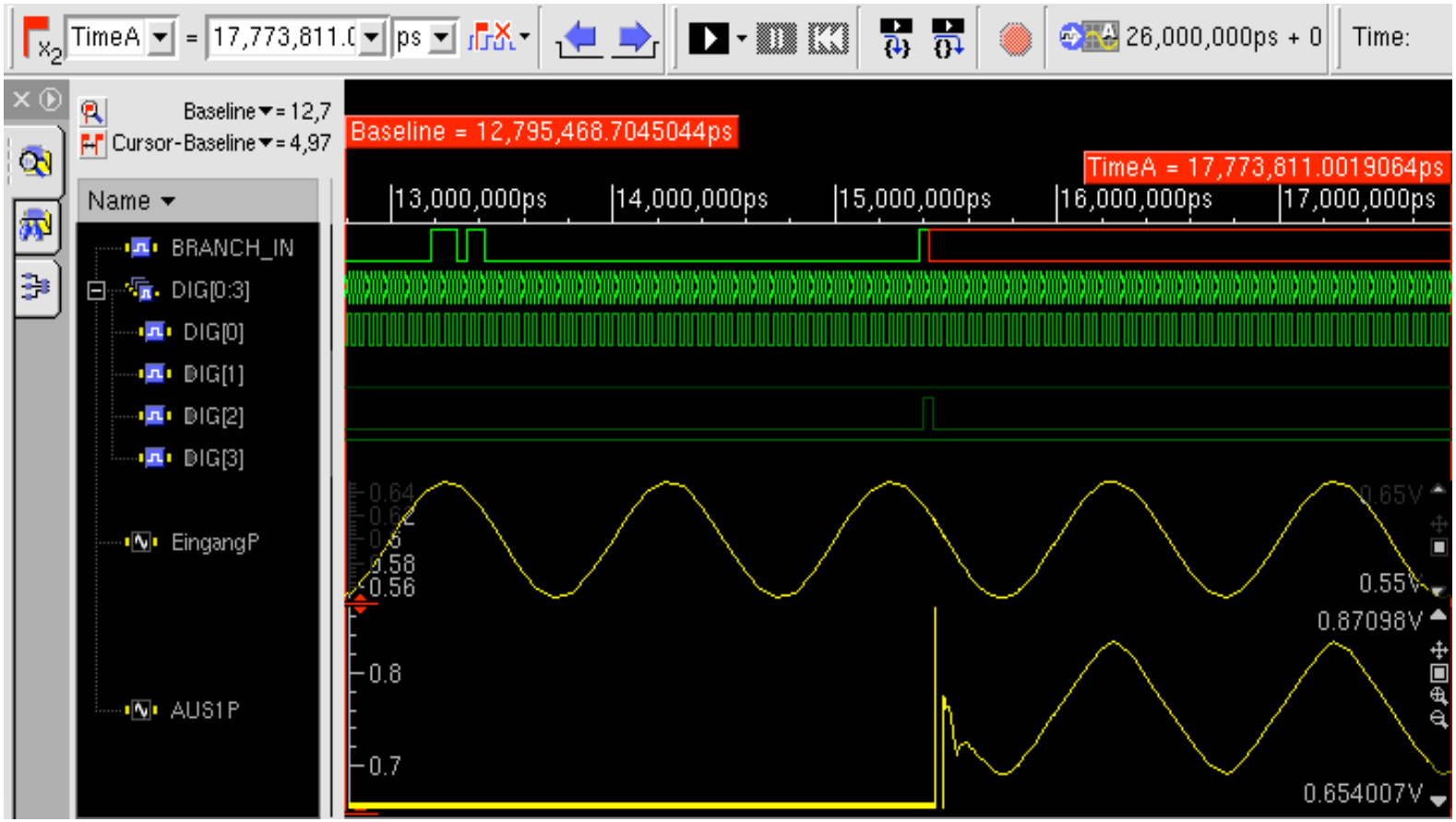
- No global digital netlist
- No global digital timing analysis
- Make bottom-level insensitive to clock skew
  - similar to JTAG, introduce extra flip-flops to avoid hold-violations
  - data signal inputs are latched at rising clock edge
  - data signal outputs change at falling clock edge



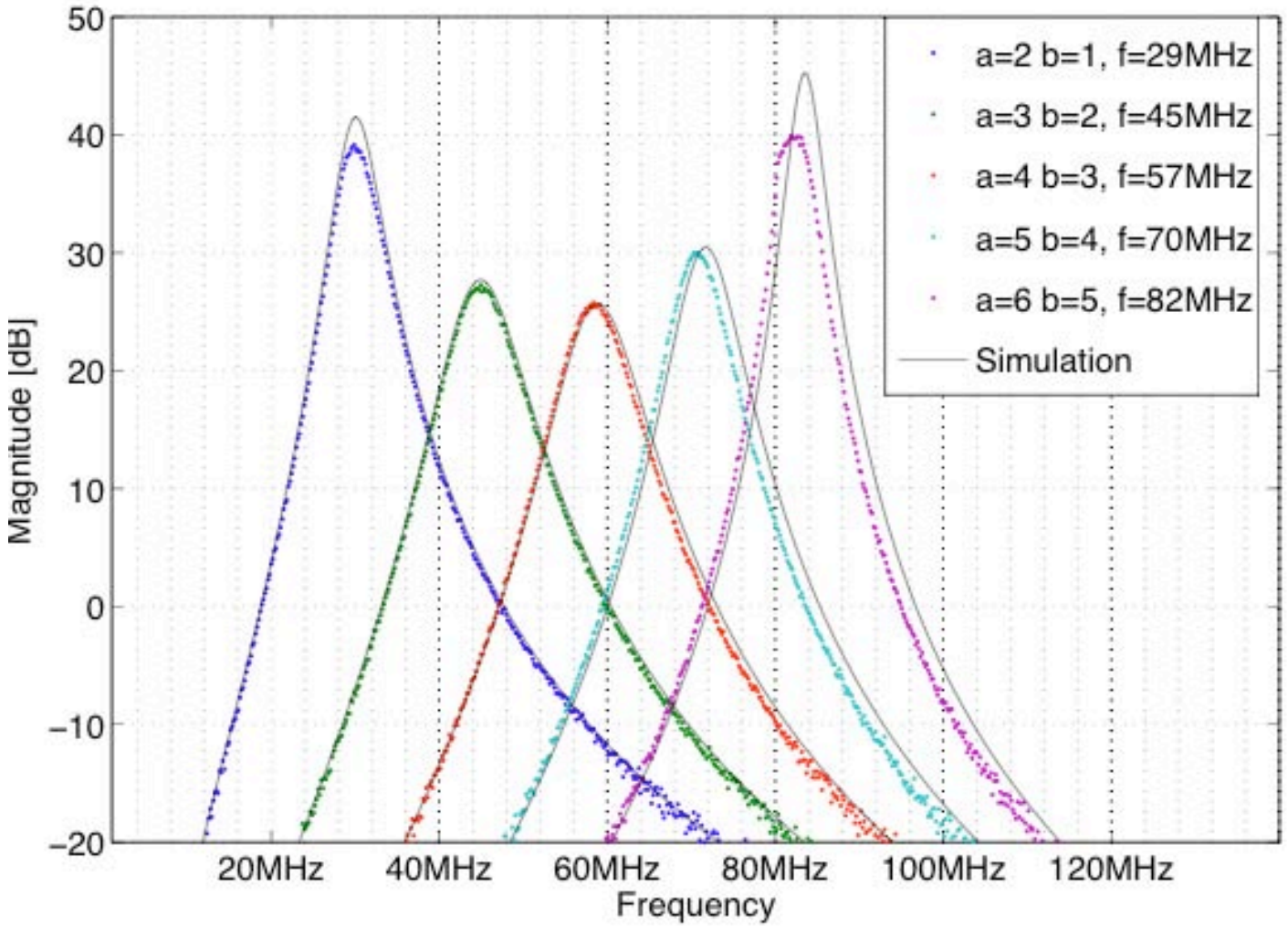
- Timing within bottom-level blocks verified
- Data has half clock-cycle to move to next stage.
  - hold-violations between distant registers avoided by lower clock frequency

AMS Hierarchy Editor

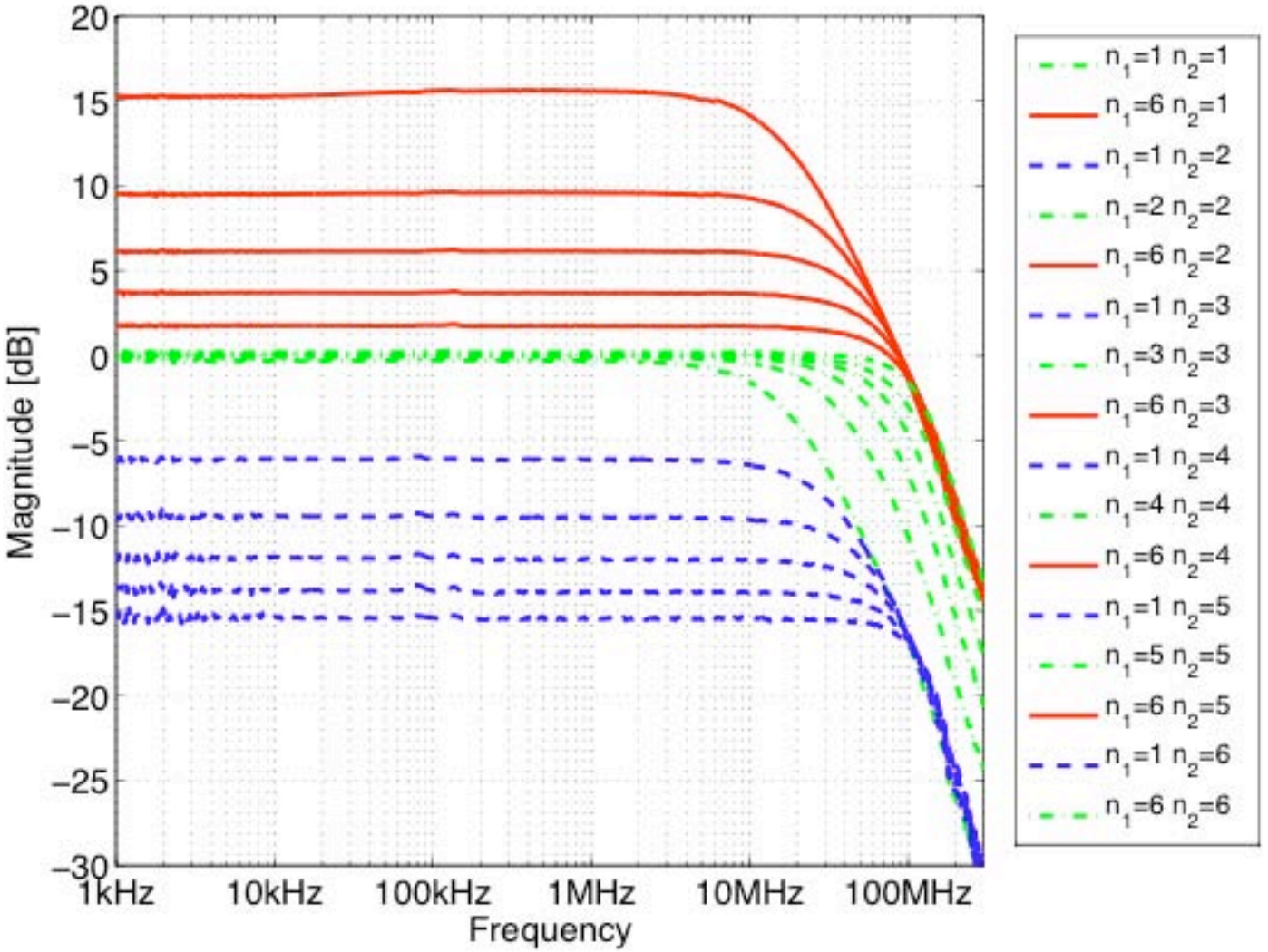
- mixed-signal simulation needed for verification
  - co-simulation of digital configuration input and analog behavior



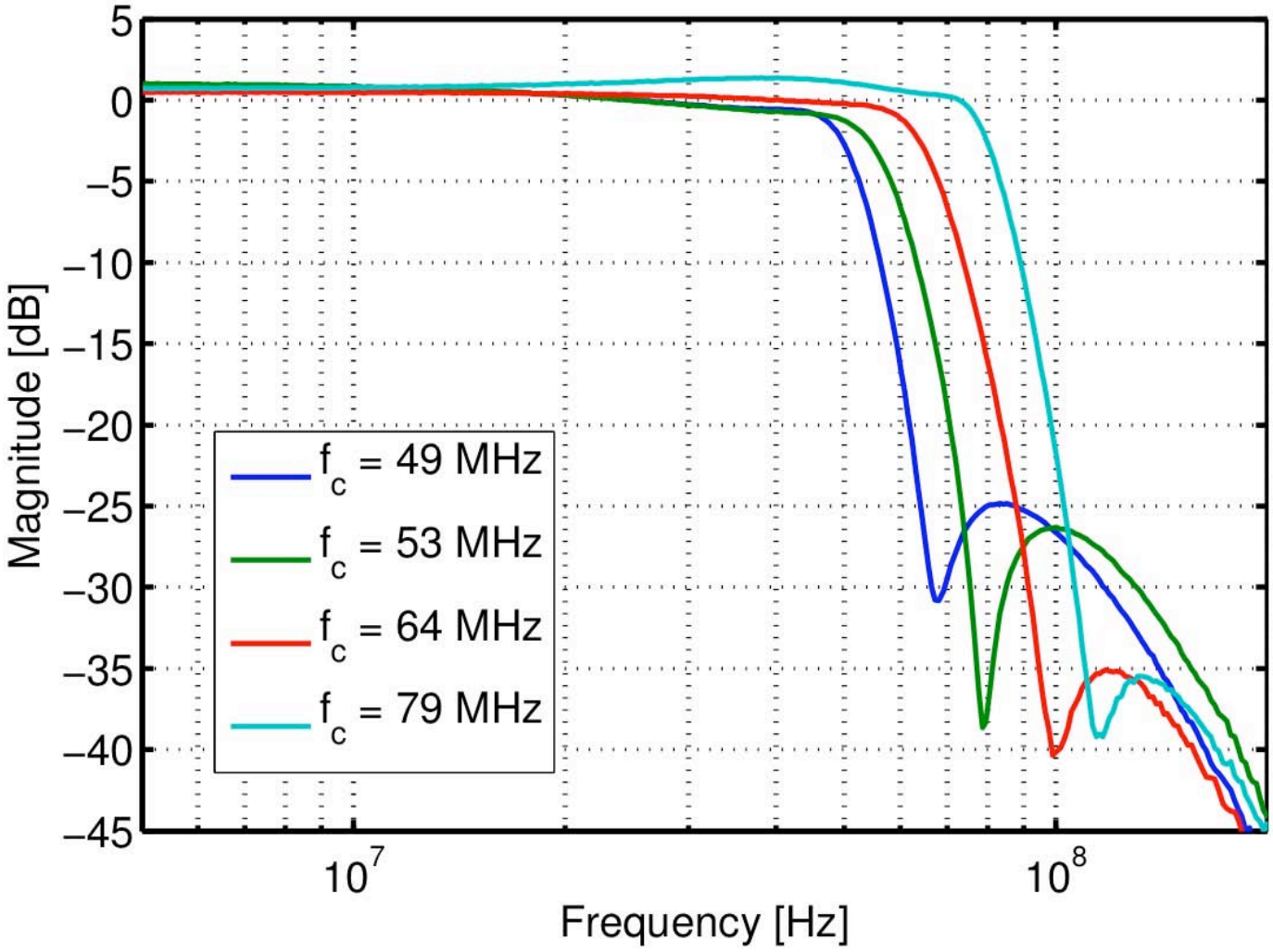
Measurements: biquad bandpass



Measurements: lossy integrator

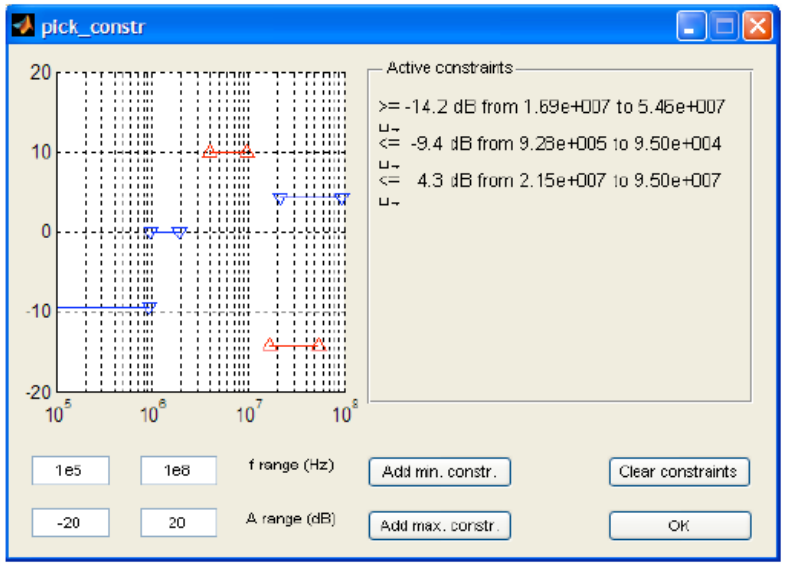


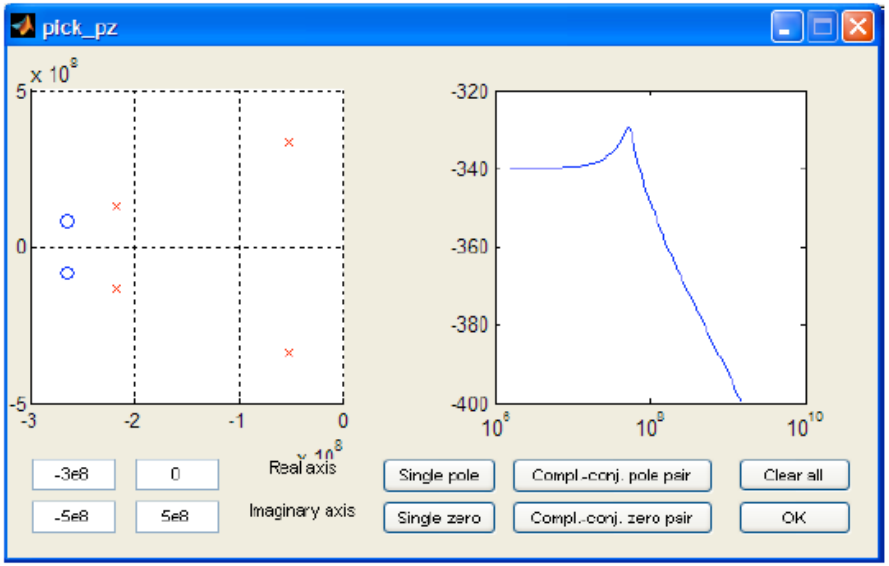
Measurements: elliptic filters



Filter Synthesis through Genetic Algorithm (GA)

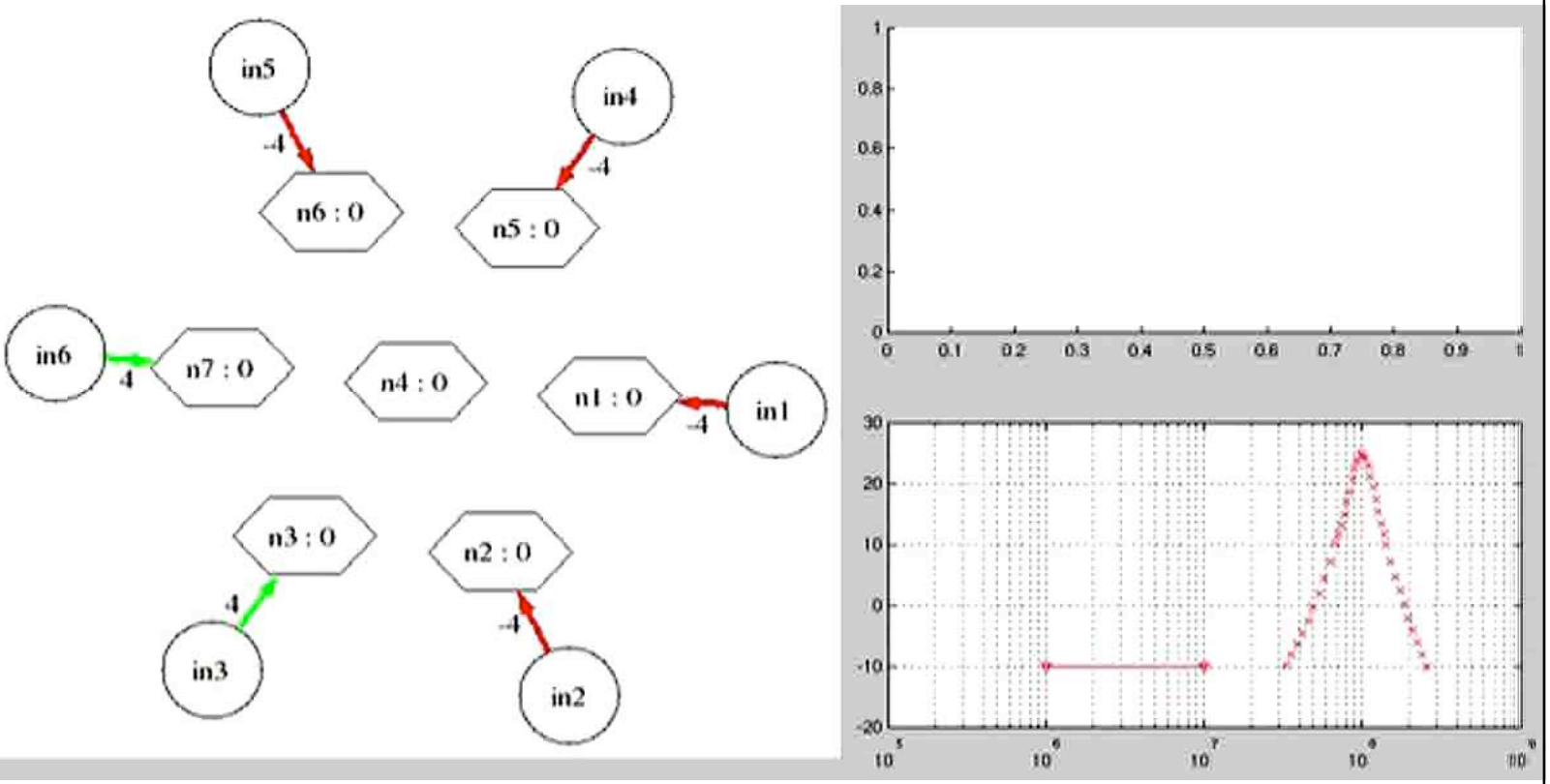
- Designer needs help for filter mapping on chip
  - Chip may have properties which are hard to handle (quantization, parasitics)
- Gradient-based algorithms like hill-climbing unsuitable
  - Parameter space is to ragged, there is no continuous way to a goal
  - There are $2^{385} = 10^{115}$ filters possible on the array
- Genetic Algorithm uses biology inspired evolution
  - Set of individuals, exchange of genes (Mutation), survival of the fittest (Selection)





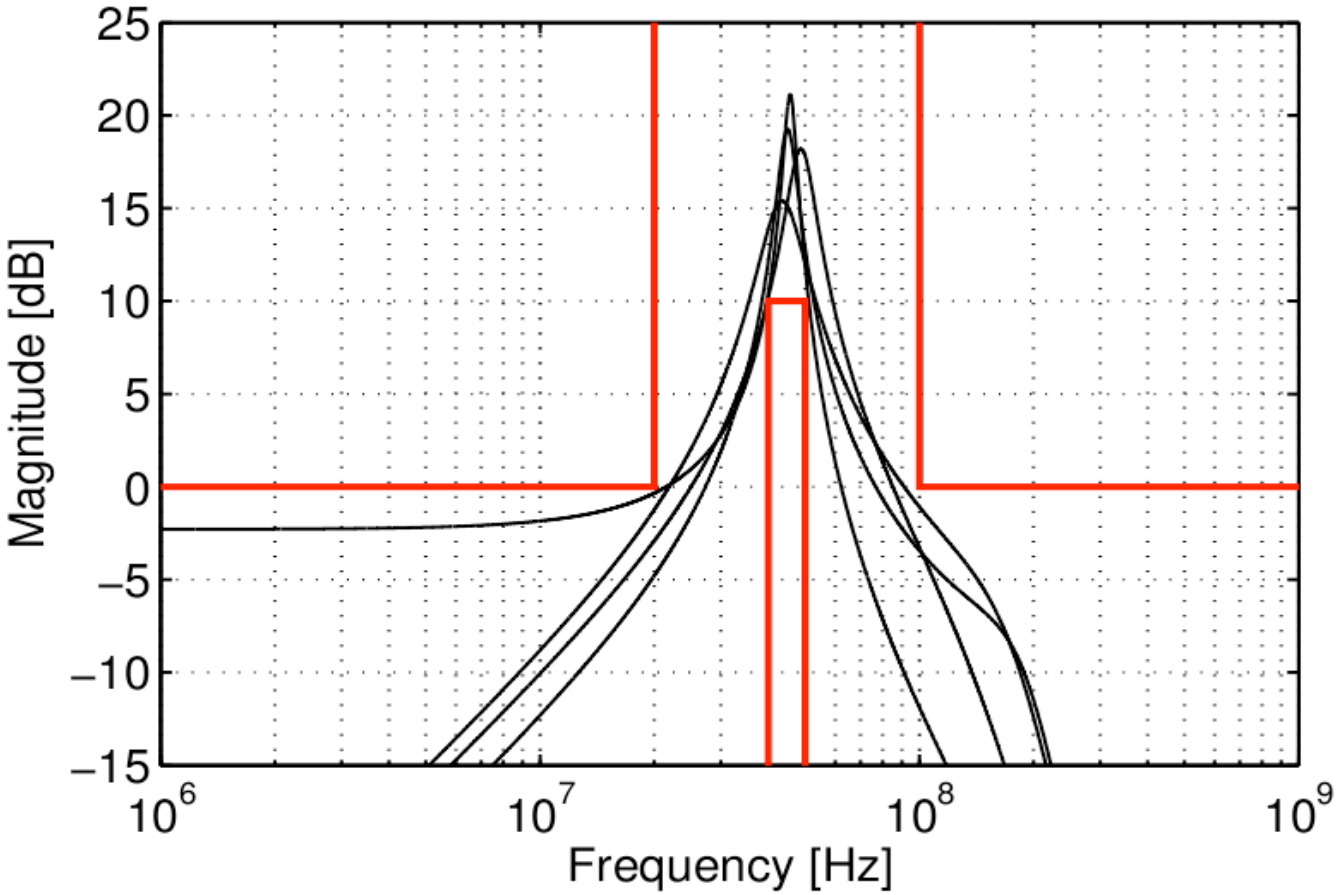
Example of GA in execution

- Visualization of best individual of evolving generations
  - Layout, magnitude, and pole/zero plot
  - Evolution of a given bandpass filter function



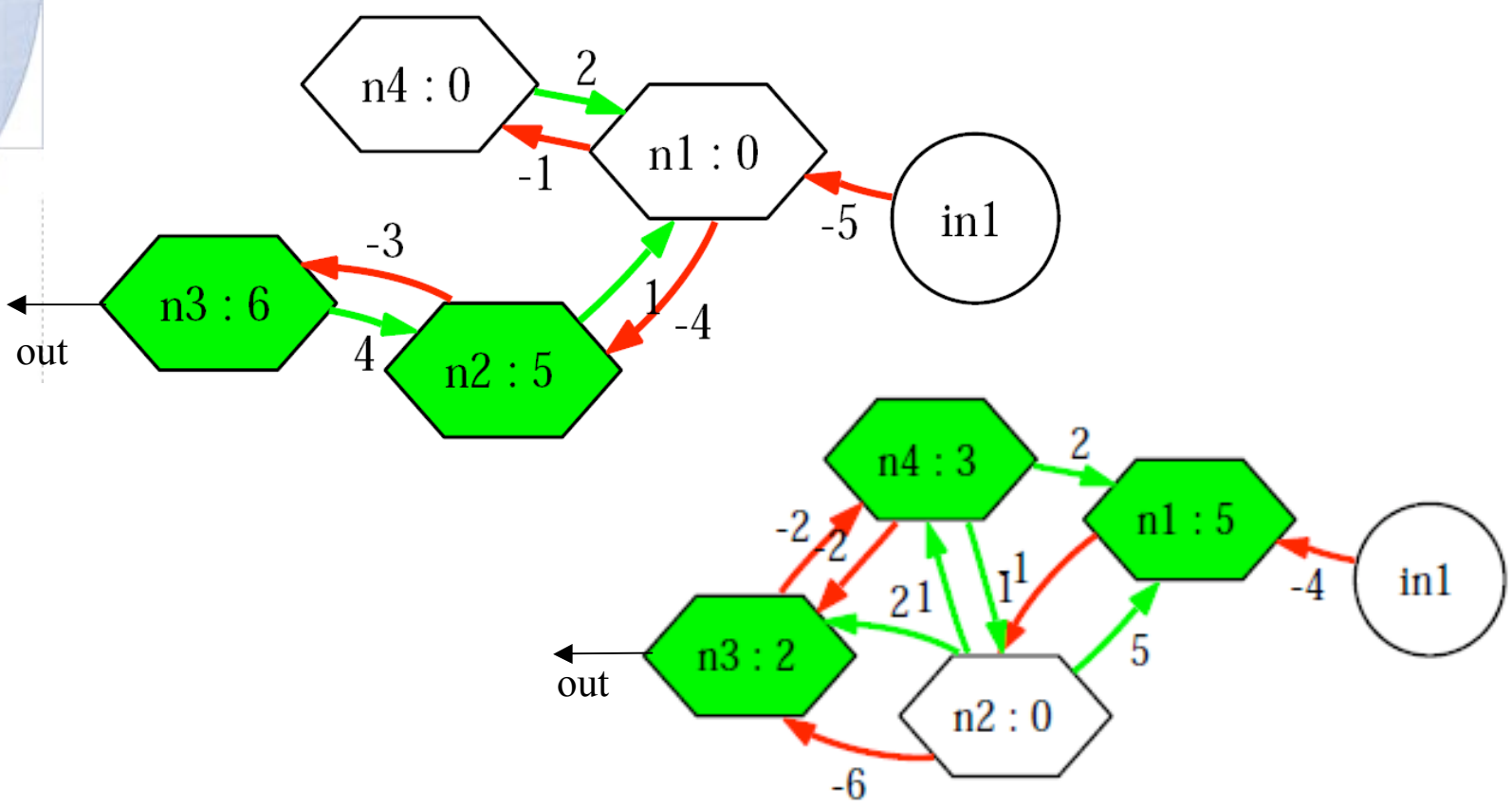
Results

Filters found for an exemplary bandpass specification



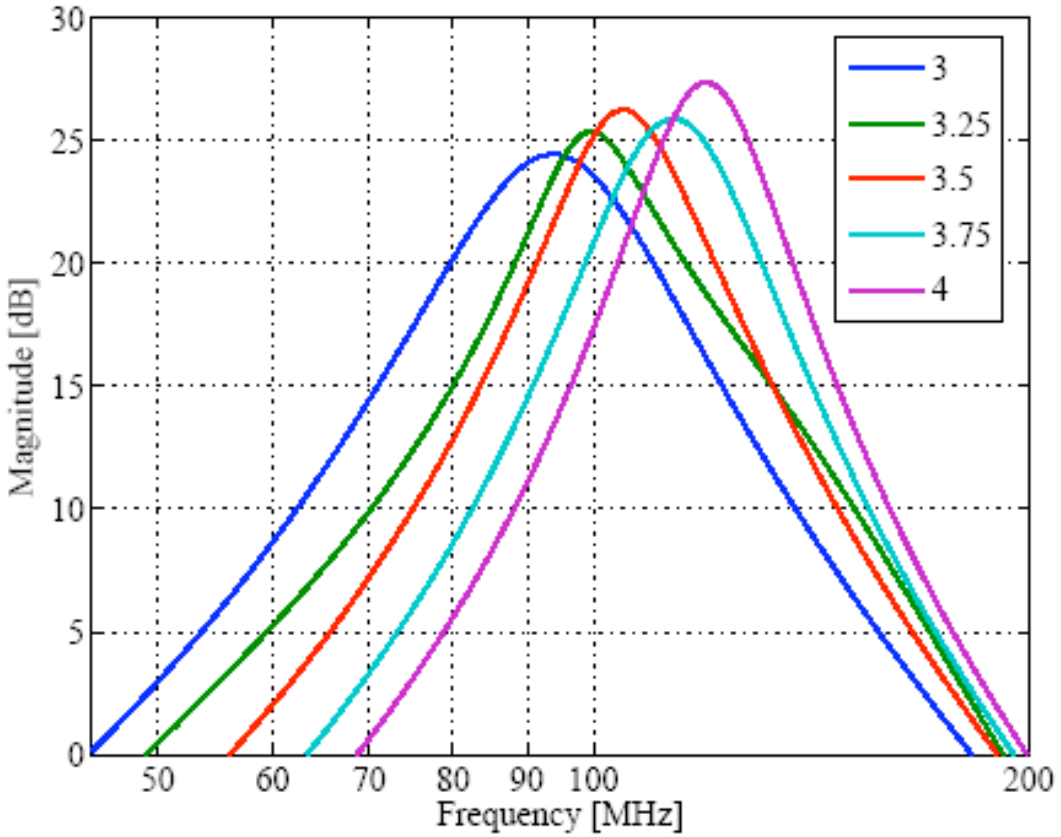
Topology of Bandpass solutions

- For more complex functions, solutions are sometimes entirely new but often variations of known structures.



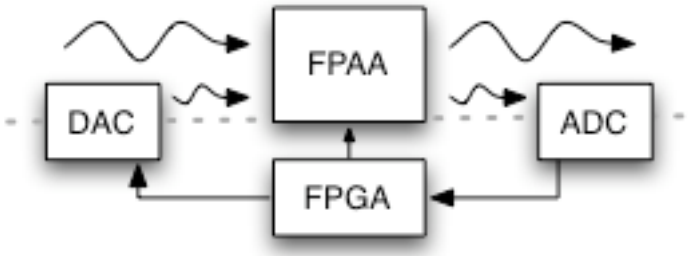
Application of GA: parameter quantization

- Digitally programmable gm-cells have quantized parameters n = 1 .. 6
- Parameter quantization limits can be overcome



Genetic Algorithm with hardware in the loop

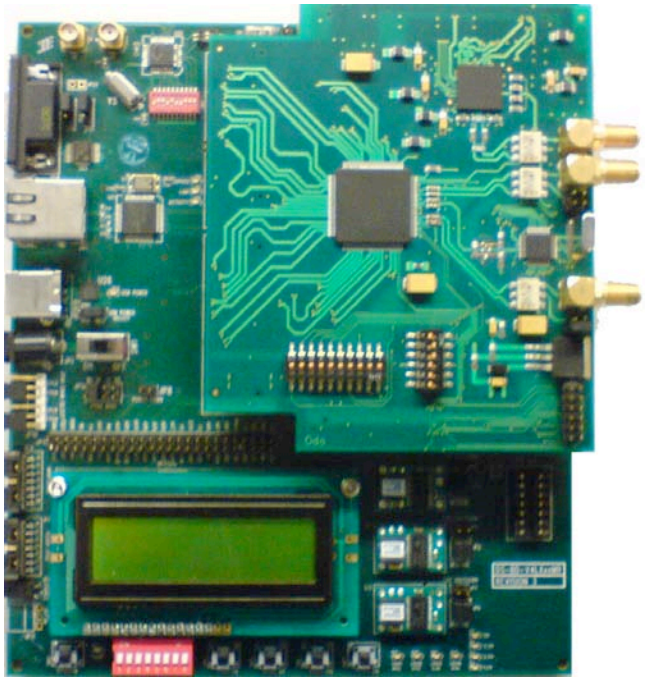
- GA with hardware in the loop uses all imperfections of the chip
  - could even take benefit of real hardware properties
- Delivers working prototype of filter
- Setup with laboratory equipment works but is very slow
  - development of an embedded system

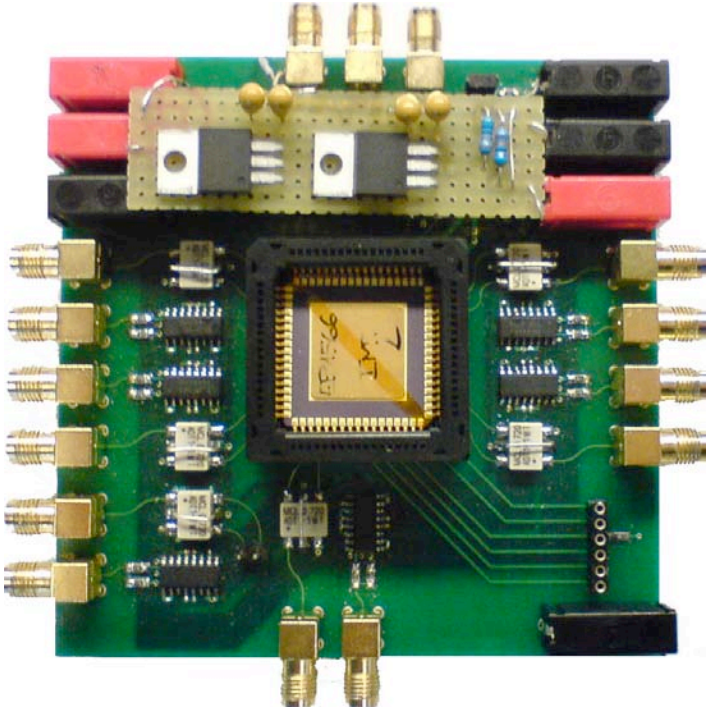


- Transfer function of a filter needs to be evaluated
  - white noise created as input to the FPAA
  - transient response measured
  - Fourier Transformation gives information on magnitude of signal-frequency
- One evaluation can be done in 200ms, whole GA takes 5 minutes.

Implementation of GA in an embedded system

- Commercial FPGA board with Virtex4 for control
  - VHDL Code for noise generation, FFT-processing, fitness-calculation, genetic algorithm
- Custom PCB with 1.4 Gsps DAC and ADC done in Allegro
  - LVDS bus with matched impedance and phase
  - RF analog signal processing
- Custom FPAA interfacing board



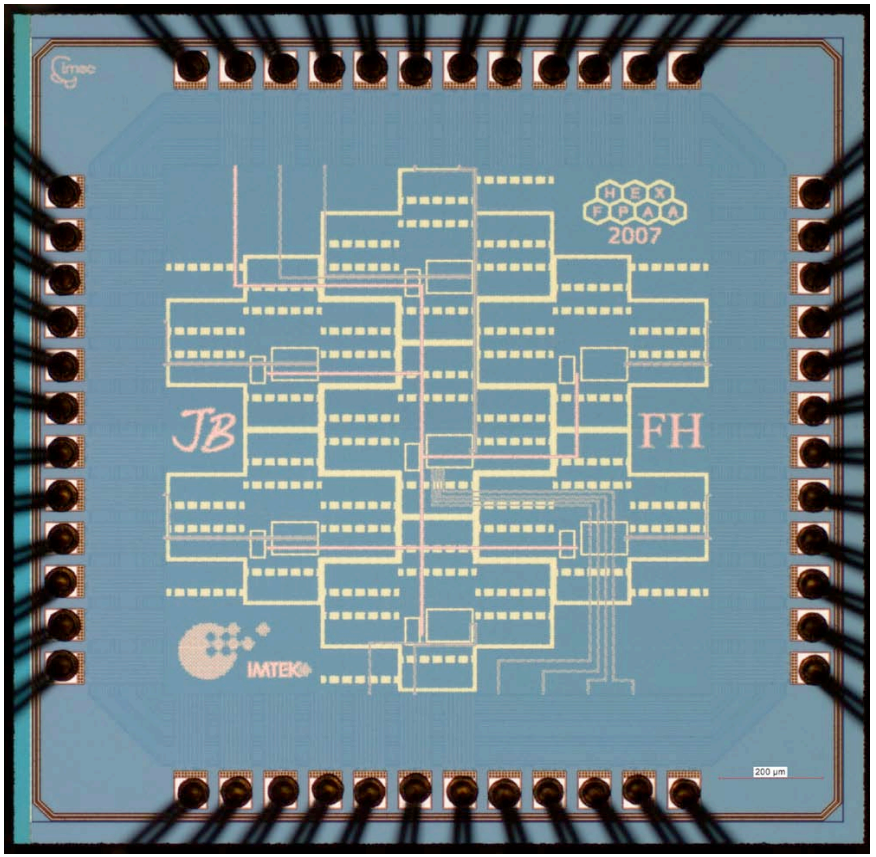


Conclusion (1)

- Innovative Idea and new concept
  - Field Programmable Analog Array for continuous-time filters
  - Unique hexagonal topology enables novel routing possibilities
  - Improves state-of-the-art by one order of magnitude in bandwidth
- Sophisticated design methodology
  - Setup of reconfigurable circuit during simulation and verification
  - GUI for configuration setting, distributed sweep of simulations
- Efficient use of Cadence tools
  - Hierarchical mixed-signal design is challanging with standart tool-chain
  - Mixed leaf-cells need to be designed for use with one single design-flow
  - Assembly of chip in *Analog-on-top* flow without digital flow support
    - ... and still got it working!

Conclusion (2)

- No rest for the wicked…
  - Complexity of project makes it a PhD-topic from hell
  - Complexity of project makes it a thrill and keeps us running...
  - IC 6.1 and SOC 7.2 have terrific new features for AMS-integration
    - unified OpenAccess database shared between IC and SOC
    - all tools of both flows available at all levels: mixed-signal hierarchy
    - Hex_FPAA already ported, tape-out in preparation



Joachim Becker, Fabian Henrici, Stanis Trendelenburg
Microelectronics Laboratory, Department of Microsystems Engineering (IMTEK), University of Freiburg, Germany
http://www.imtek.de/mikroelektronik
jmbeck@imtek.de