FASoC: Fully-Autonomous SoC Synthesis using Customizable Cell-Based Synthesizable Analog Circuits

Integration Exercise, July 2019

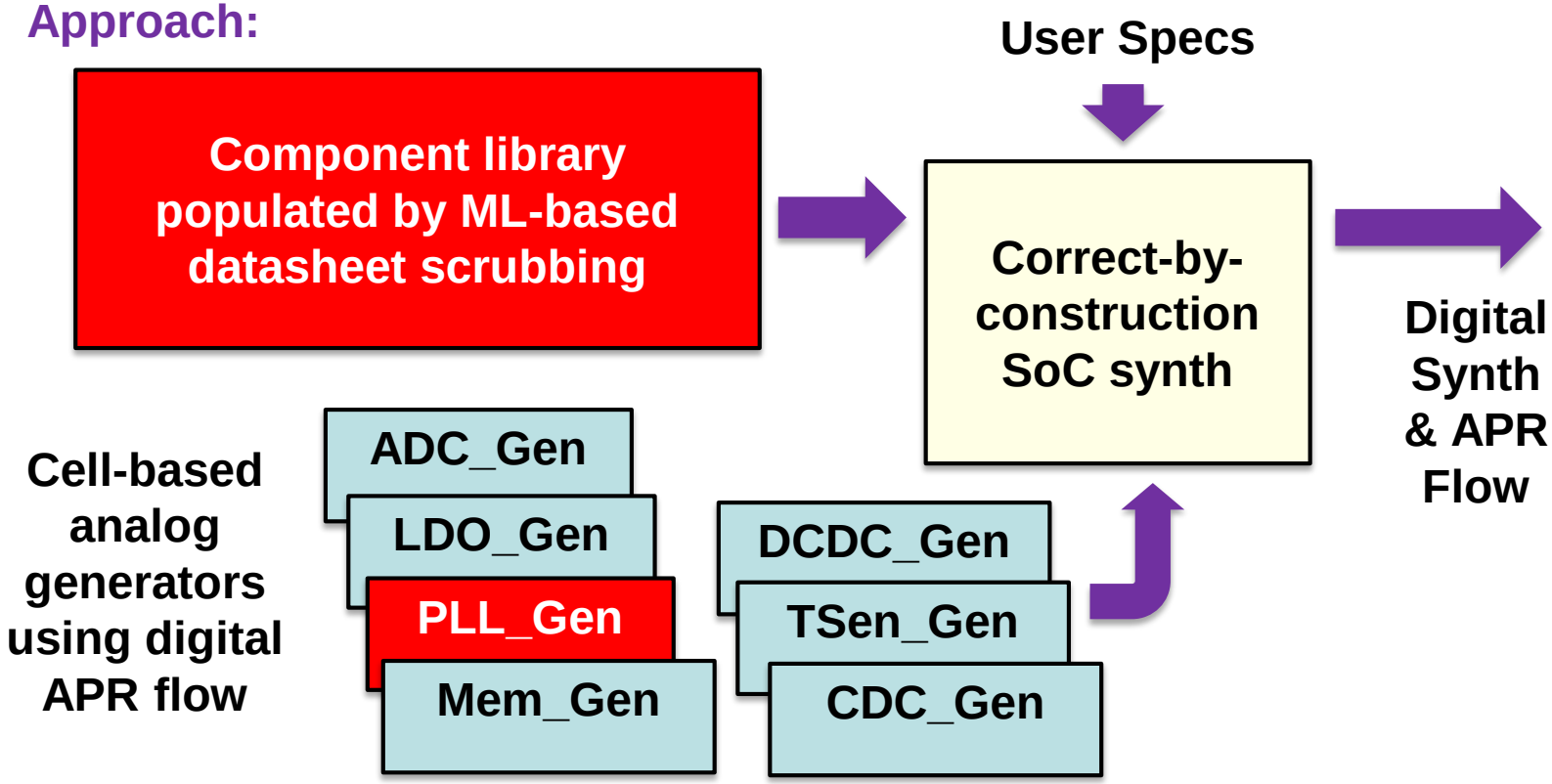
PI: David Wentzloff, UM
Co-PIs: David Blaauw, Ron Dreslinski, Dennis Sylvester (UM), Benton Calhoun (UVA), Matteo Coltella (Arm)

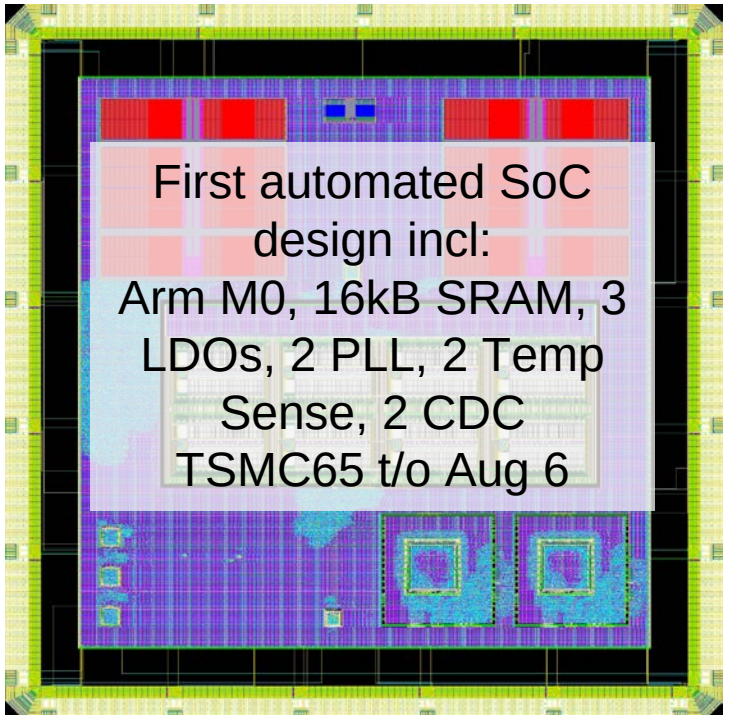
FASoC: Fully-Autonomous SoC Synthesis

Correct-by-construction SoC design leveraging IP-XACT and Arm Socrates. Analog generation tools for xDC, PLL, SRAM, DCDC, temp sense

https://fasoc.engin.umich.edu/

Approach:



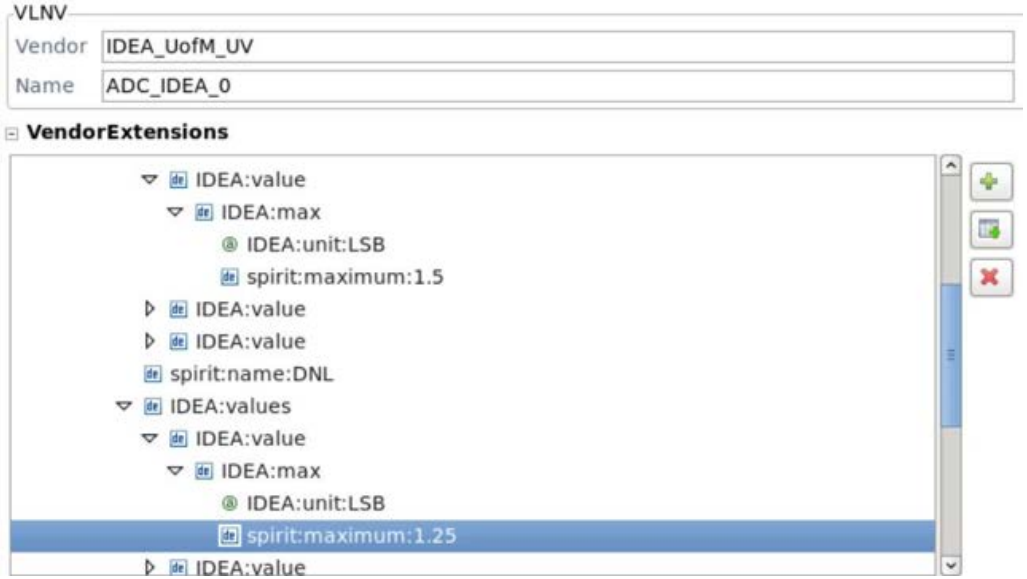


NO custom layout!

Datasheet Scrubber Demo

- **Motivation:**
  - Growing available IP and enabling reusability
  - Tracking available IP options, their revisions, and functionality is difficult for SoC designers
  - Manual approach is time consuming especially if the database grows towards millions of parts

- **Store extracted information in IP-XACT++ format using VendorExtensions**



- **Outputs:**
  - Categorization (e.g. ADC, LDO, etc.)
  - Data extraction from…
    - Text
    - Tables

Datasheet Scrubbing - Approaches

- **January integration:**
  - Simple ML based techniques
  - Analyzing 500 datasheets and academic papers, 7 categories
  - Average accuracy of 84% for category recognition and 86% for data extraction
- **July integration:**
  - More robust CNN based techniques
  - Analyzing 3000 datasheets and academic papers, 13 categories
  - Average accuracy of 96% for category recognition and 96% for data extraction
- **Phase 2 goal:**
  - 700,000 datasheets in 100+ categories from Digikey IC Parts
- **Beyond:**
  - All 6M Digikey Parts

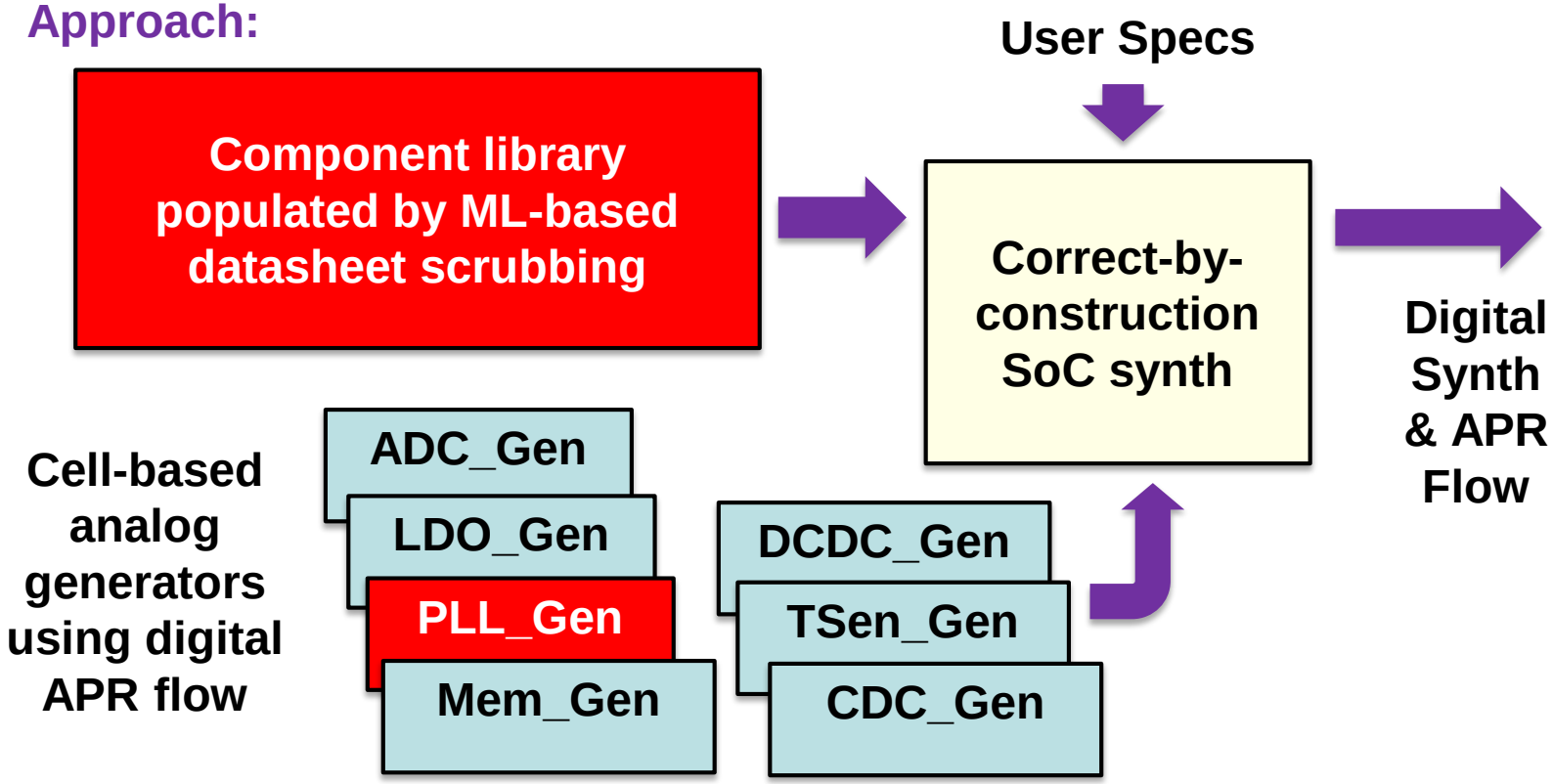
https://github.com/idea-fasoc/datasheet-scrubber

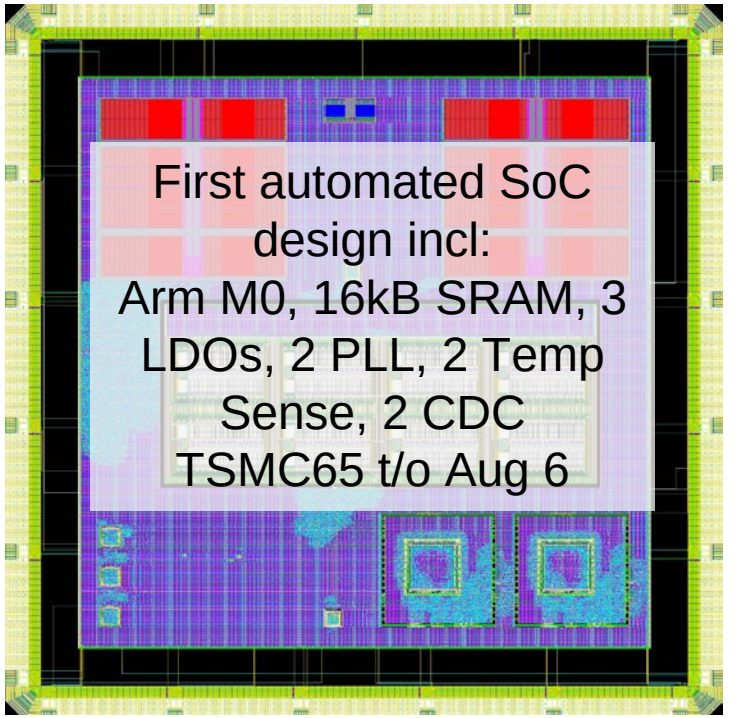
FASoC: Fully-Autonomous SoC Synthesis

Correct-by-construction SoC design leveraging IP-XACT and Arm Socrates. Analog generation tools for xDC, PLL, SRAM, DCDC, temp sense

https://fasoc.engin.umich.edu/

Approach:





NO custom layout!

PLL_Gen Demo

1. git clone

idea-fasoc / fasoc Private

Clone or download

2. check input

tool setup:
cad/cadenv/setup.tcl

environment setup:
platform.config

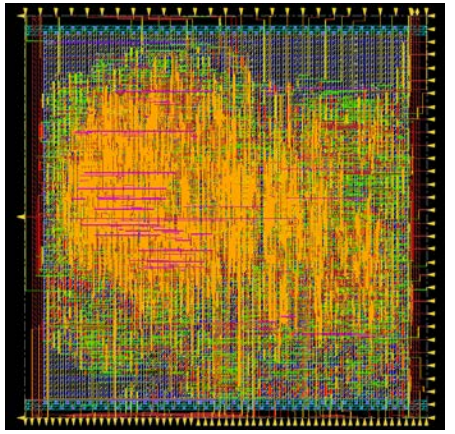
pll input spec:
spec_in.json
(ref. README)

3. $ make pll

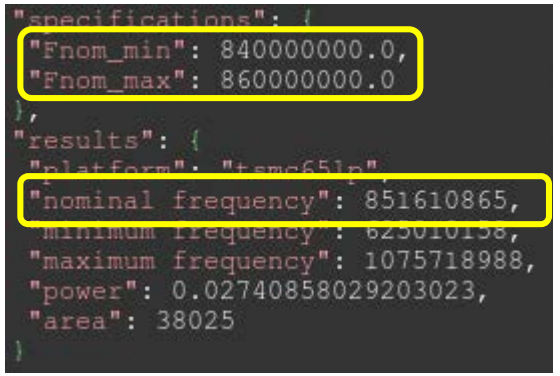
PLL_GEN

1. pick design w/ pll model
2. generate Verilog
3. synth & APR w/ CADRE
4. calibre (drc, lvs, rcx)
5. run pex simulation
6. generate deliverables

4. check output



pll.gds



spec_out.json

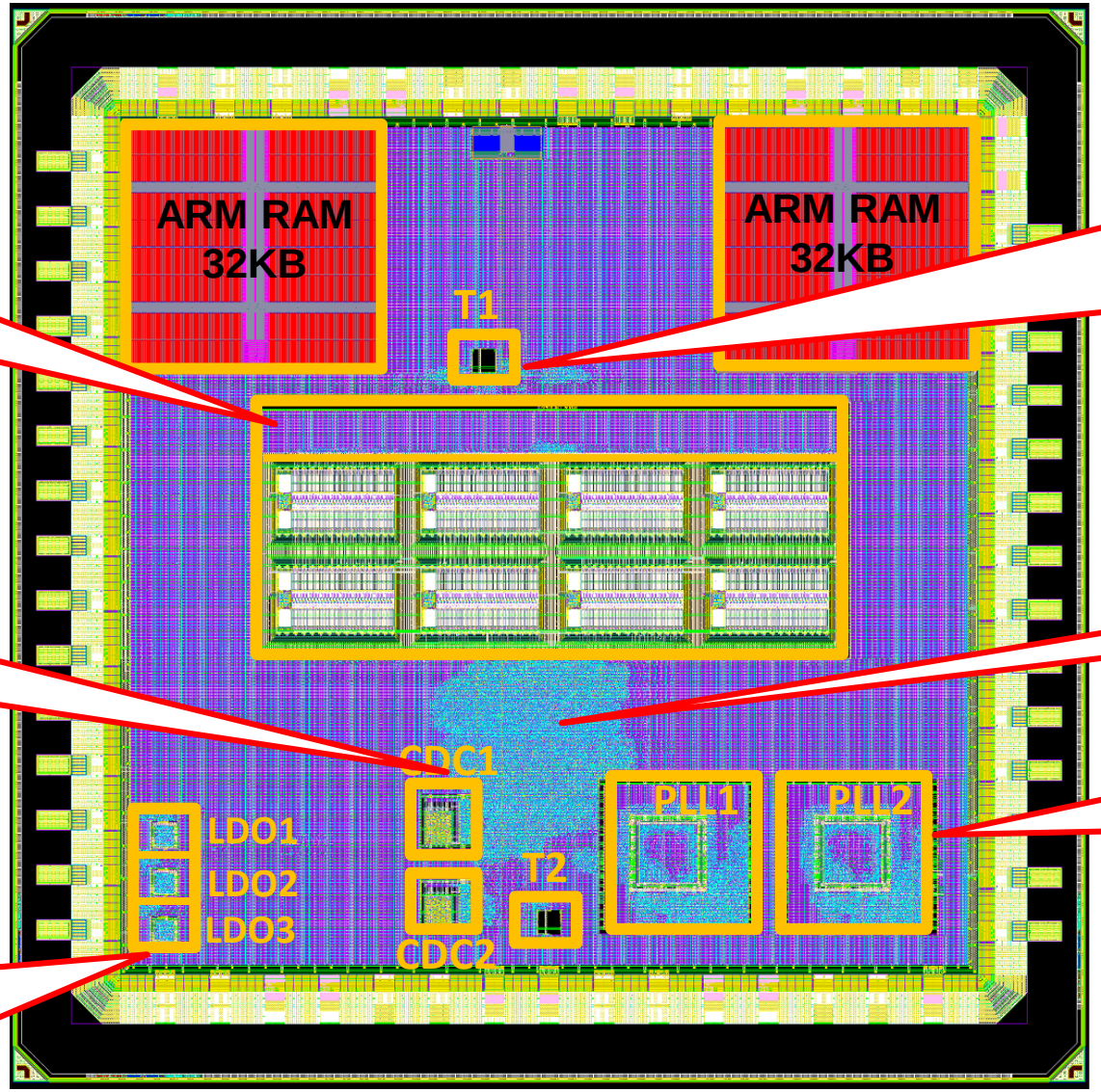
Fnom_min < nominal frequency < Fnom_max
840MHz < 852MHz <860MHz

Auto-Generated SoC – TSMC 65LP – Aug 6 T/O

16kB RAM
250MHz
0.5-1.2V

2 CDC versions
1- C_{Range} = [1pF 10nF]
2- C_{Range} = [1pF 1nF]

3 LDO versions:
1- $I_{load,max}$ = 2mA
2- $I_{load,max}$ = 5mA
3- $I_{load,max}$ = 5mA w/ ADC based controller



2 Temperature versions
T_{Range} = [-20 100] ℃:
1- Optimization: **Min. Error**
0.21 ℃ | E/conv.: 53.7 pJ
2- Optimization: **Min. Power**
0.39 ℃ | E/conv.: 21 pJ

Arm M0
w/ debug access port

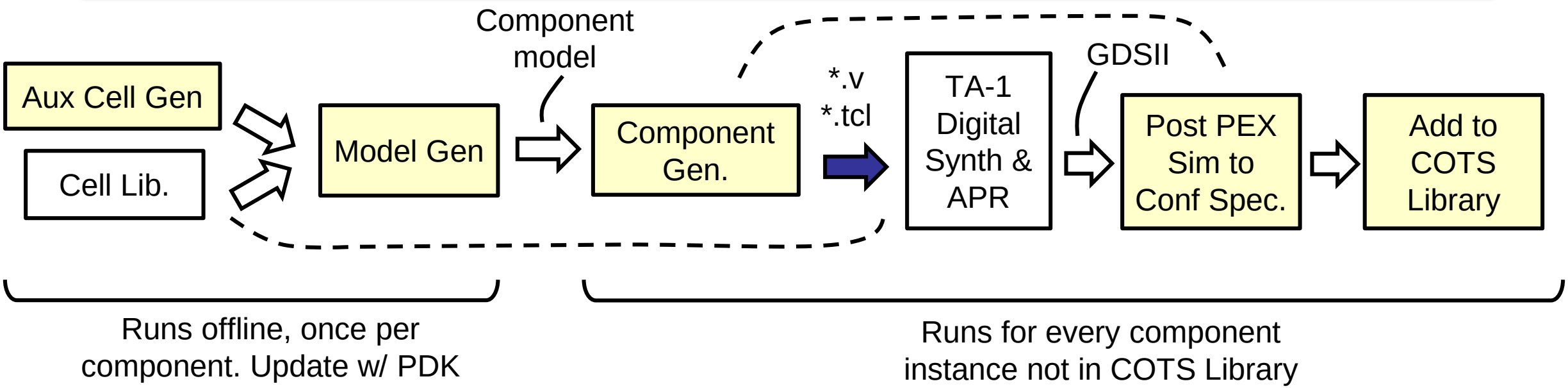
2 PLL versions:
1- 400MHz
2- 200MHz

Landing Page

https://fasoc.engin.umich.edu/

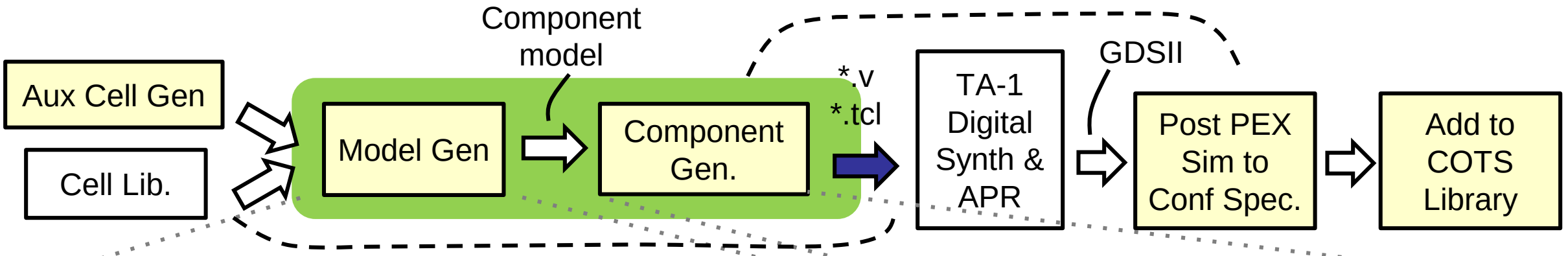
Backup

Cell-Based Analog Design Flow



- **Same flow for ADPLL, ADC, CDC, DC/DC, LDO, Temp Sensor, Memory**
- **Structural and behavioral description of components**
- **Use "digital" synthesis and APR flow for physical design**
  - **No custom analog layout. No analog layout tool required.**

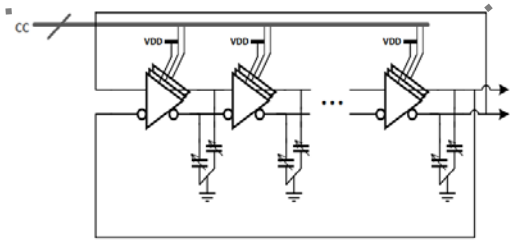
ADPLL Generator

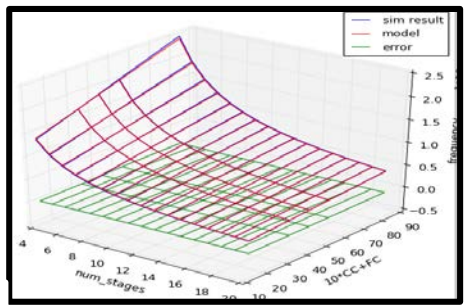


Inputs
-Technology info
-Aux cell
-DCO Architecture

DCO modelling

Outputs
-Model Parameters
-Maximum error rate





model *vs* sim-result

	Err max
	1%
	10%
	6%

Inputs
-DCO model
-Freq.range, PN, Area

ADPLL gen

Outputs
-DCO.v
-PLL.v

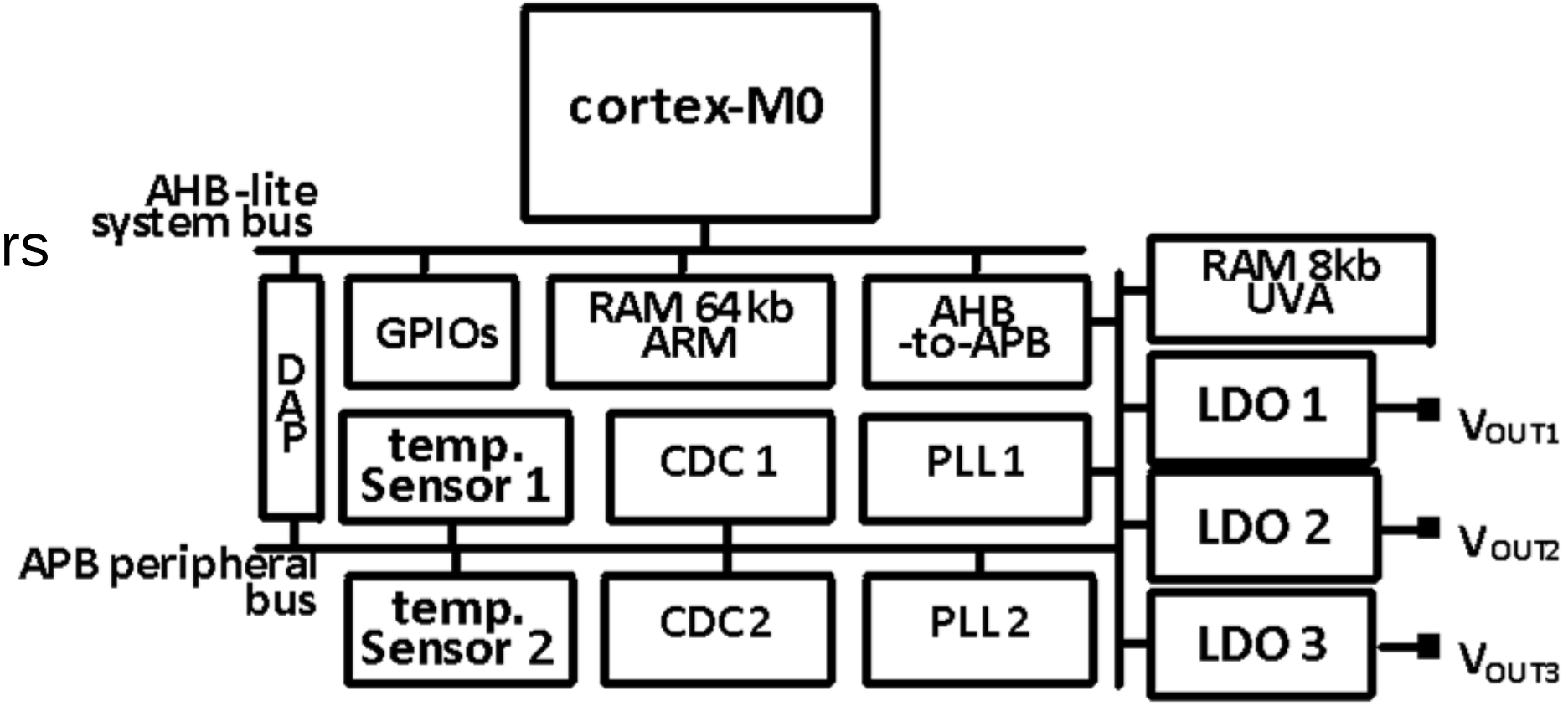
	spec
	3.5~4.5
	0.6
	-95
	20000

Spec comparison	**Auto generated DCO**	ISSCC'16[1]
Frequency range(GHz)	3.4 ~ 4.6	1.7 ~ 3.47
PN@1MHz (dBc/Hz)	-97	-98.7
	155 @(1M/4.0GHz)	165.6 @(1M/3.5GHz)
Technology	40nm	65nm

(Mehdi) SoC Diagram – Tape out

Multiple versions of each generator are included in our SoC to address different optimizations/strategies such as: operation ranges, accuracy, power, speed, etc..

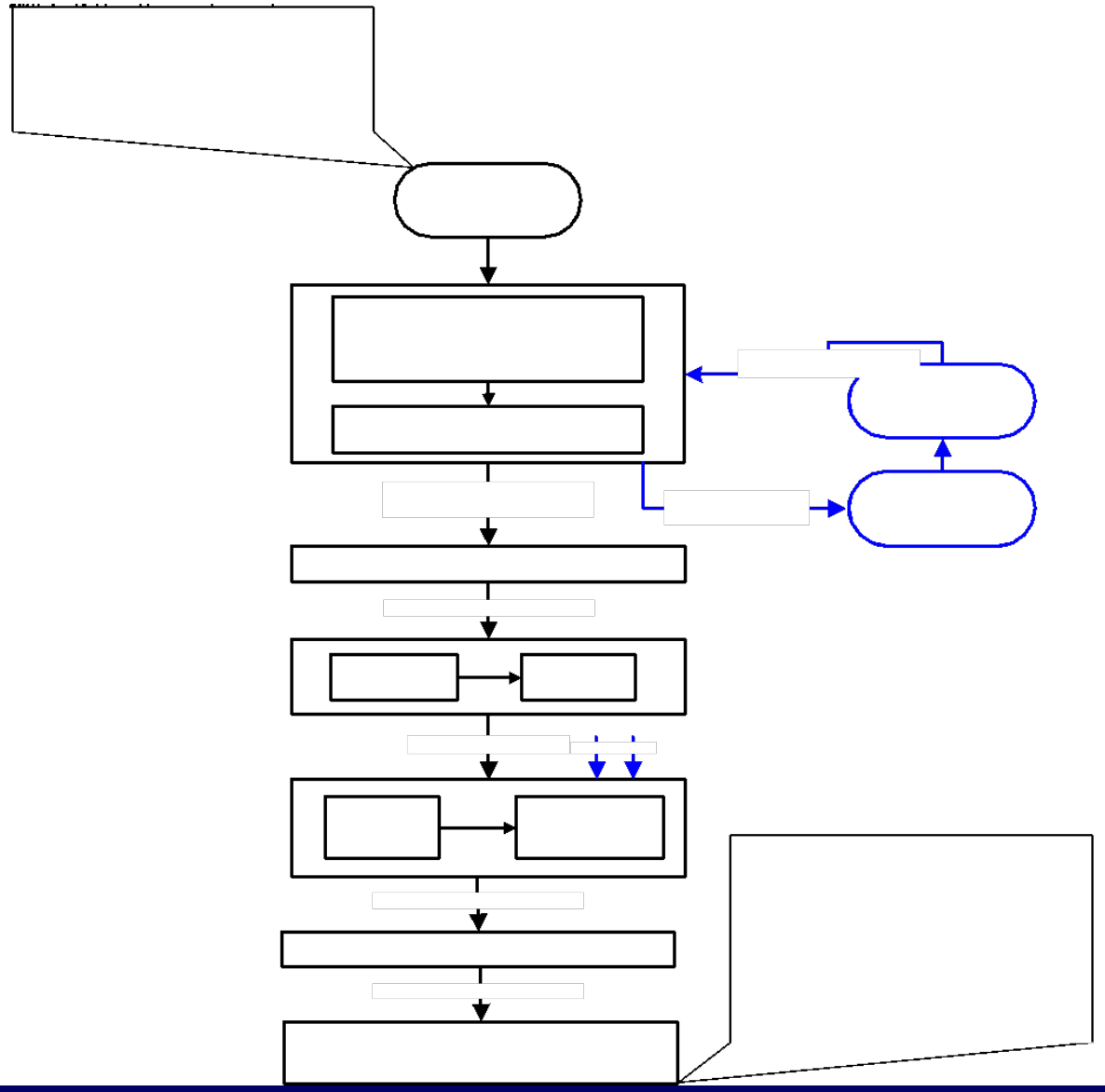
- 3 LDO
- 2 PLLs
- 2 temp. sensors
- 2 CDCs
- Memory



✓ Testing circuitry wrappers are included for each of the generators

Generator Flow : Generic flow

We can use this diagram to highlight each of the steps that the generator goes through while the demo is running in the background. I can make changes if needed



Post PEX: Summary Results – Temperature Sensor

Spec Inputs

T_{Range} = [-20 100] ℃

Optimization: **Min. Power**

Output Perf. (CDL sim)

Max. Error: 0.16 ℃

E/conversion: 5.6 E-3 pJ @25 ℃

Output Perf. (PEX sim)

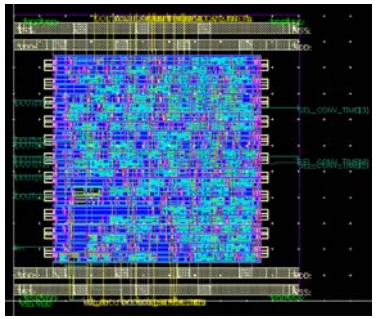
Max. Error:

0.39 ℃ [-20 100]

0.083 ℃ [0 100]

E/conversion: 21 pJ

Area: 2300 µm^2



Spec Inputs

T_{Range} = [-20 100] ℃

Optimization: **Min. Error**

Output Perf. (CDL sim)

Max. Error: 0.032 ℃

E/conversion: 6.18 E-3 pJ @25 ℃

Output Perf. (PEX sim)

Max. Error:

0.21 ℃ [-20 100]

0.08 ℃ [0 100]

E/conversion: 53.7 pJ @25℃

Area: 2621 µm^2