Bringing Topology and Technology Variations from Circuit Models into COSIDE® System Models

Sören Kwasigroch, TU Kaiserslautern

Frank Schenkel, MunEDA GmbH

Reimund Wittmann, IMST GmbH

Coseda User Group Meeting 24th November 2022

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

AnastASICA

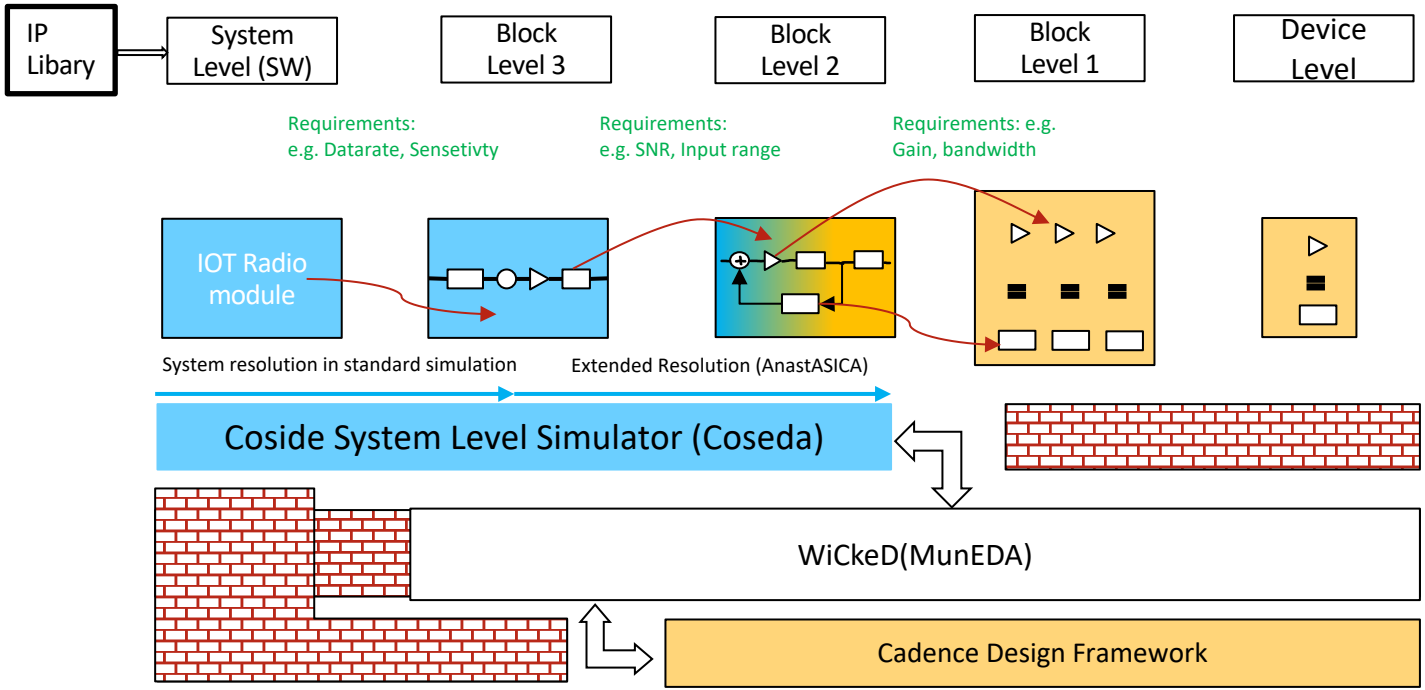
GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

Structure of the Presentation

- **Problem**
- Approach
- Example: Tire Pressure Metering System
- Results
- Conclusion

Design Flow with Coside, WiCkeD, IIP and the Cadence Design Framework



State of the Art: How can we combine Circuit & System Level?

- Use of Mixed-Level simulator coupling
  - Critical/ unknown components as circuit-level simulations
  - Too slow for software; requires licenses; **inappropriate for embedded SW**
- Automated model generation
  - Not enough desired automation; needs additional model verification
- Use of pre-existing SystemC AMS models & automated characterization of circuit
  - **choosen Tool MunEDA WiCkeD**
- ***Novelty:*** ***We model & maintain dependencies/sensitivities of circuit-level variations***

Structure of the Presentation

- Problem
- **Approach**
- Example: Tire Pressure Metering System
- Results
- Conclusion

Overview for the Workflow of the proposed solution

Library of parameterizable
Block models

Performance verification
System optimization

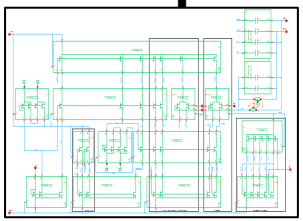
CADENCE
Circuit Designer

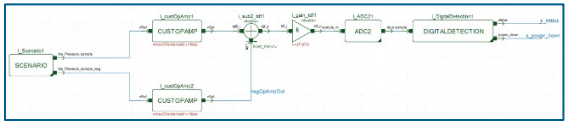
Performance values

COSIDE
System
Model
Designer

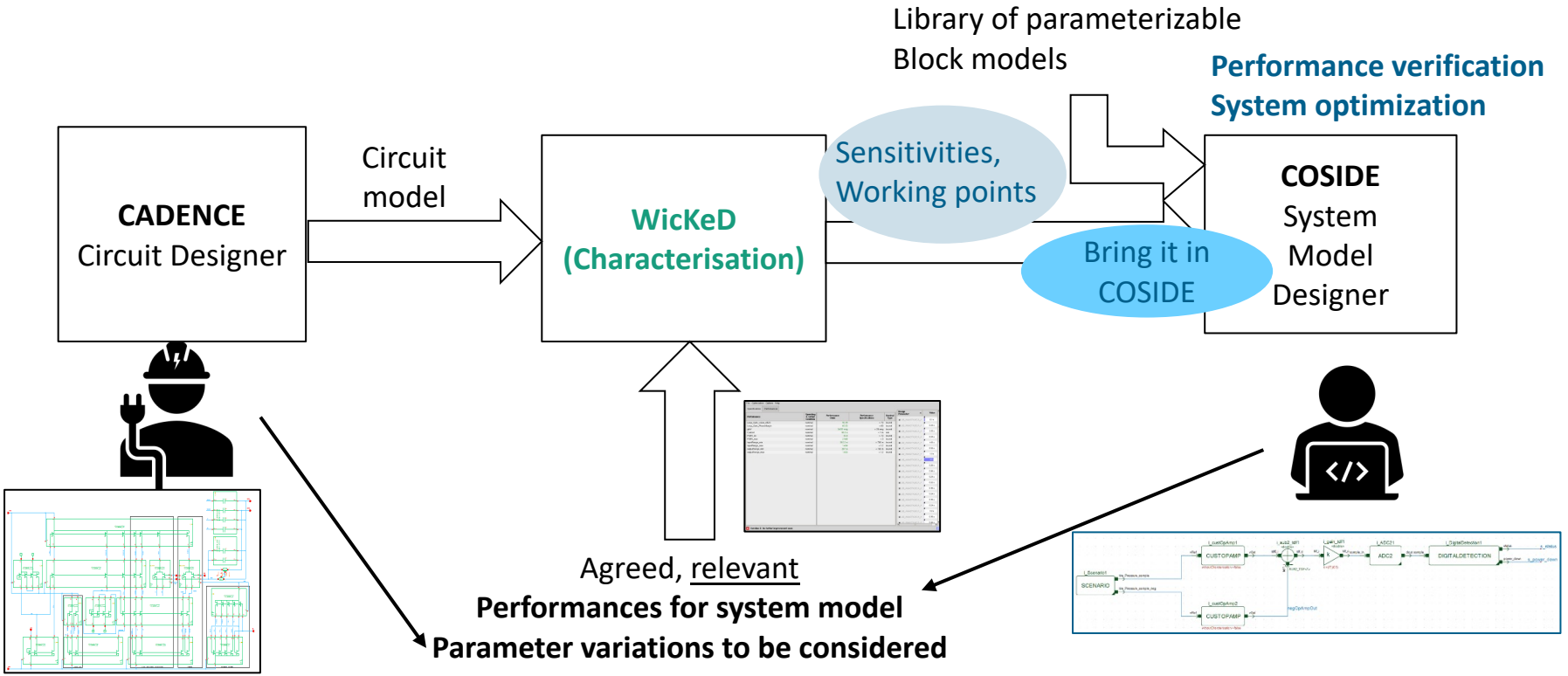
Only one single circuit instance modeled

We want subspace to analyze!





Overview for the Workflow of the proposed solution



Linear Model of Dependencies

1. Given and defined **working point** and range (min, max) for each considered parameter
   - Use same parameter for dependent variations
   - Use combination of parameter to model dependencies (correlations)

2. WiCkeD determines a linear dependency model of performances from all parameter variations

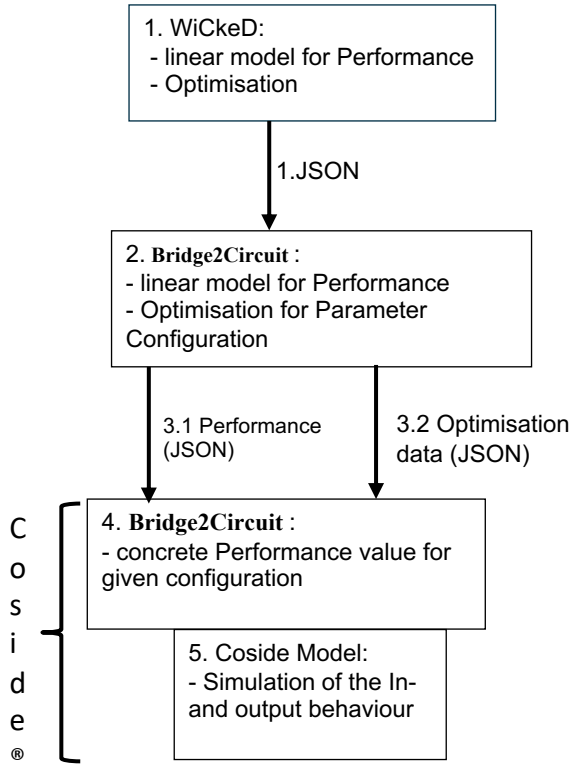
3. Minima and Maxima of performances are optimized by WiCkeD

4. Exported of model in JSON

Import into SystemC AMS / COSIDE

TU KL's Bridge2Circuit C++ library is an "add-on" for each block, but does **globally**

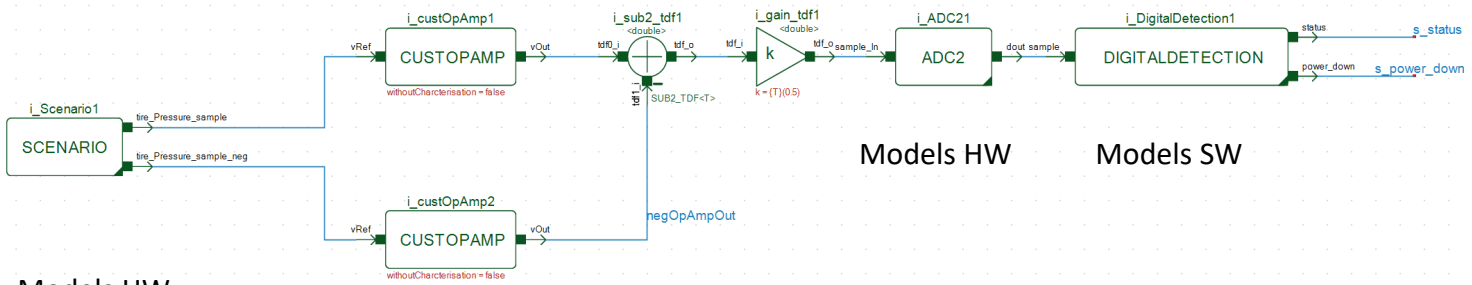
- Read JSON File with dependency models
- Set parameter to concrete values in simulation run
  - (e.g. working points, corner cases, optimization)
  - Computes performances of each block with **globally set** variations
- Run (Coside) system simulation(s) to get system performances



Structure of the Presentation

- Problem
- Approach
- **Example: Tire Pressure Metering System**
- Results
- Conclusion

Use Case Example: Tire Pressure Metering System (TPMS)

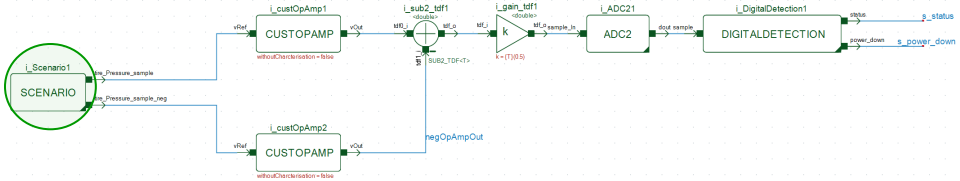


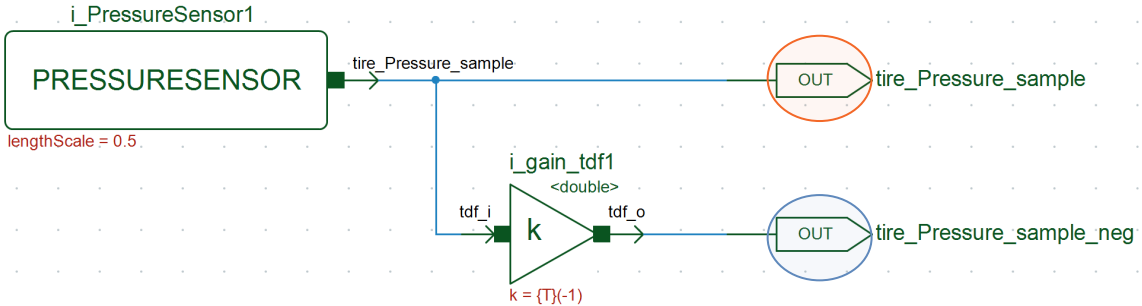
Goals:

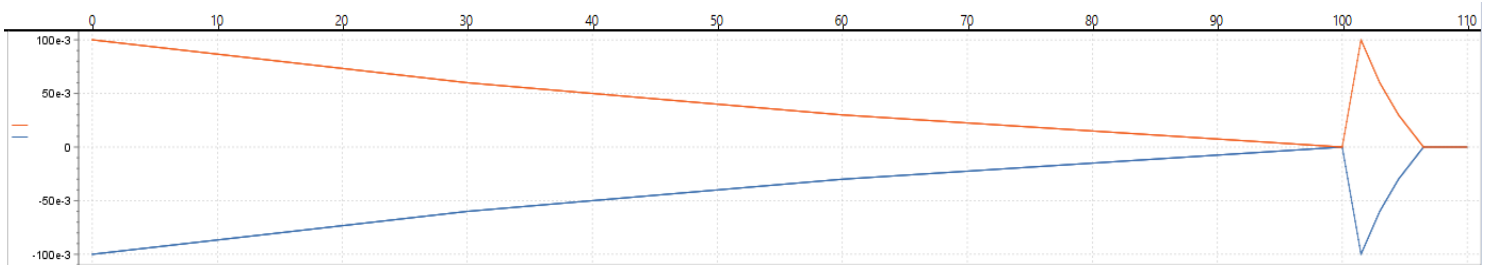
- Validating of the System performances
- Target System for System optimization

Scenario

- Differential output
- Frequency: 1 Khz
- Ideal sampling





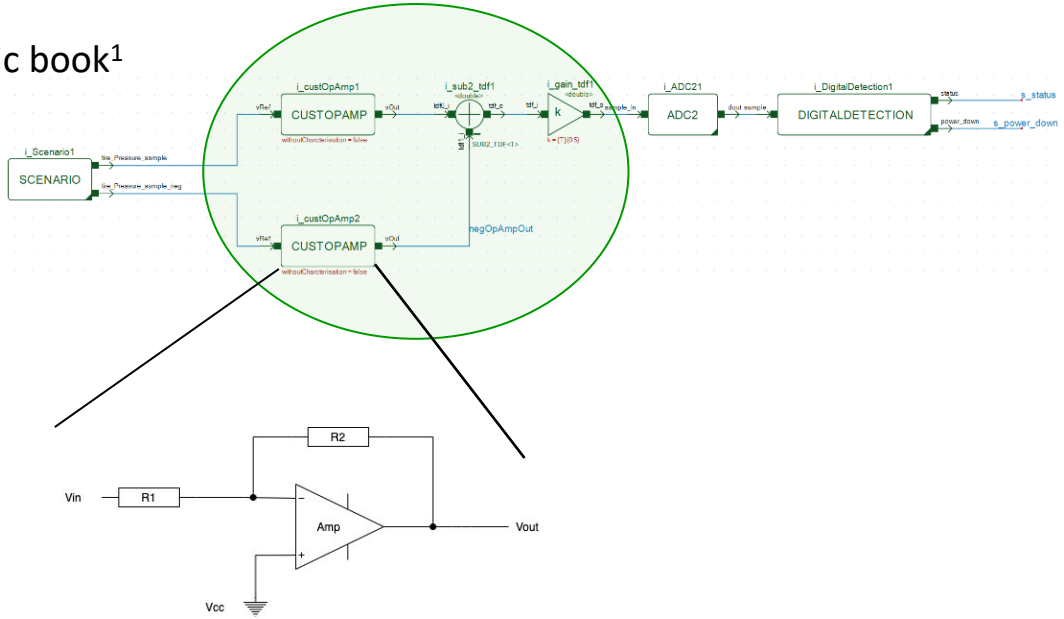


Pos_Sample

Neg_Sample

Circuit Level Part: Operational Amplifer

- Derived from a measurement electronic book[1]
- Input performances
  - Outputrange_min/max
  - Offset Error
  - loop_gain_20db
- Main characteristics
  - Out = min{outputrange,v0*Vin}

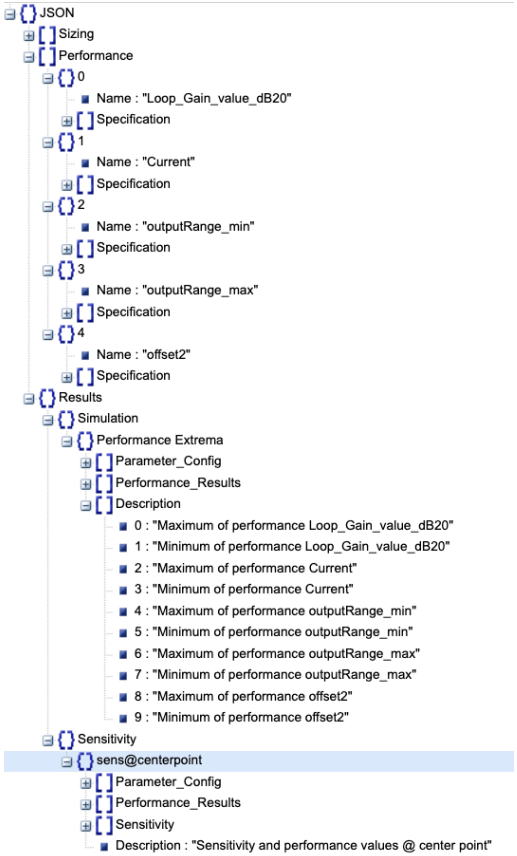


Structure of the Presentation

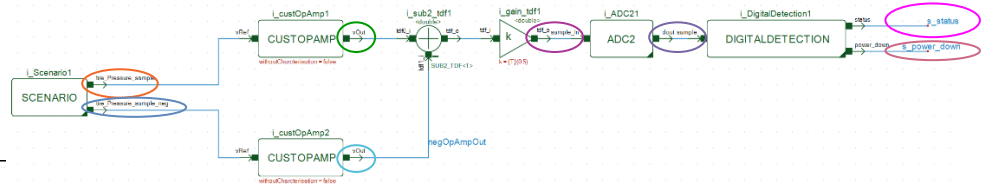
- Problem
- Approach
- Example: Tire Pressure Metering System
- **Results**
- Conclusion

Dimensions of the Import

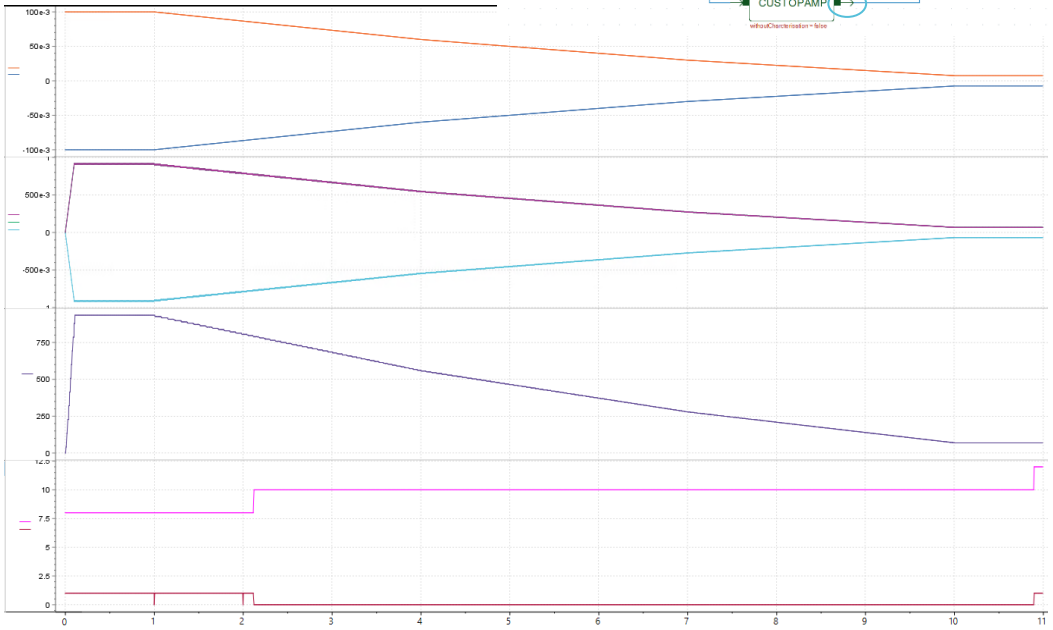
- Characterization data of MunEDAs WiCkeD Tool:
  - 69 parameter
  - 5 performances
  - 10 evaluation of simulation, for minima and maxima of the performances
  - Sensitivity of the parameter to the performances
  - Simulated 35334 simulations with 2 simulators in parallel for around 36 hours



System Model results with educated guessed performances



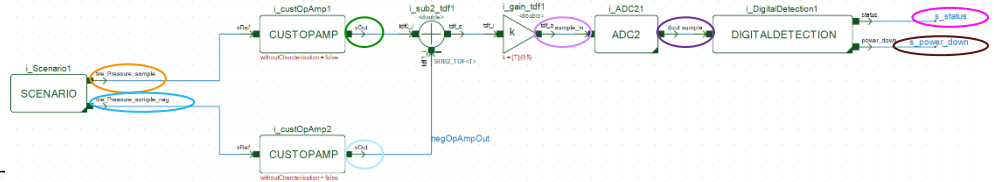
- Pos_Sample
- Neg_Sample
- result_Amplified
- Pos_Amplified
- Neg_Ampliefied
- Digital_Sample
- Status
- Power_Down



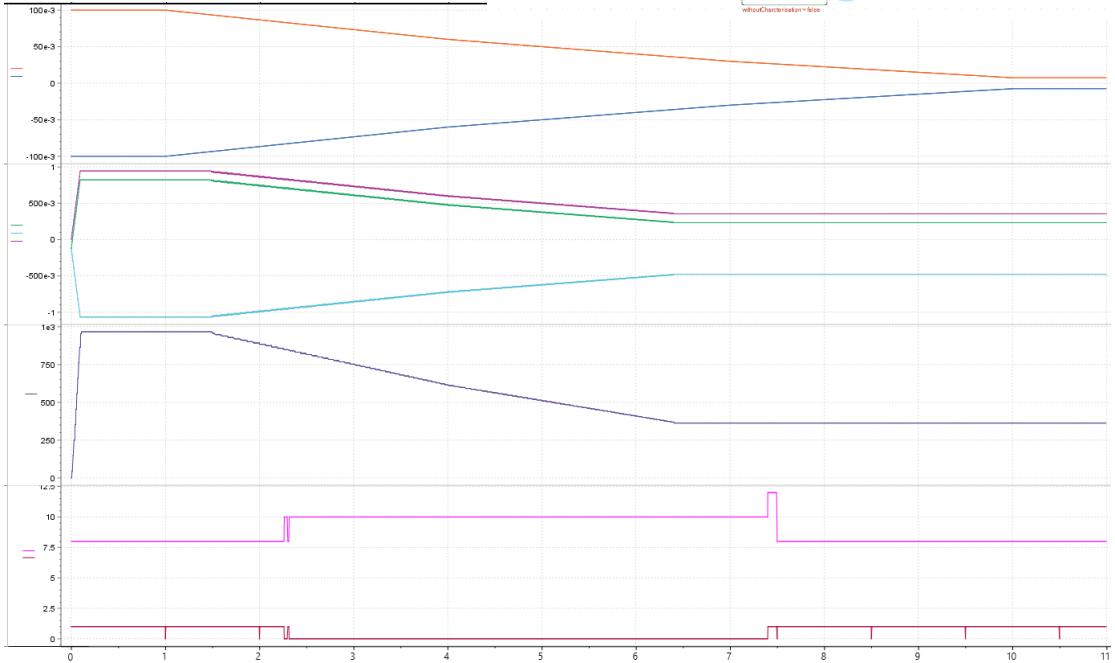
bounds_min =0
Bounds_max=5
offset=0
Open loop_gain =1000

Time in sec

System Model results with sensitivity-aware performances



- Pos_Sample
- Neg_Sample
- result_Amplified
- Pos_Amplified
- Neg_Ampliefied
- Digital_Sample
- Status
- Power_Down



bounds_min =0.359841
Bounds_max= 0.93543
offset= -0.1223326587696
Open loop_gain= 529296

Time in sec

System Optimization Main Idea

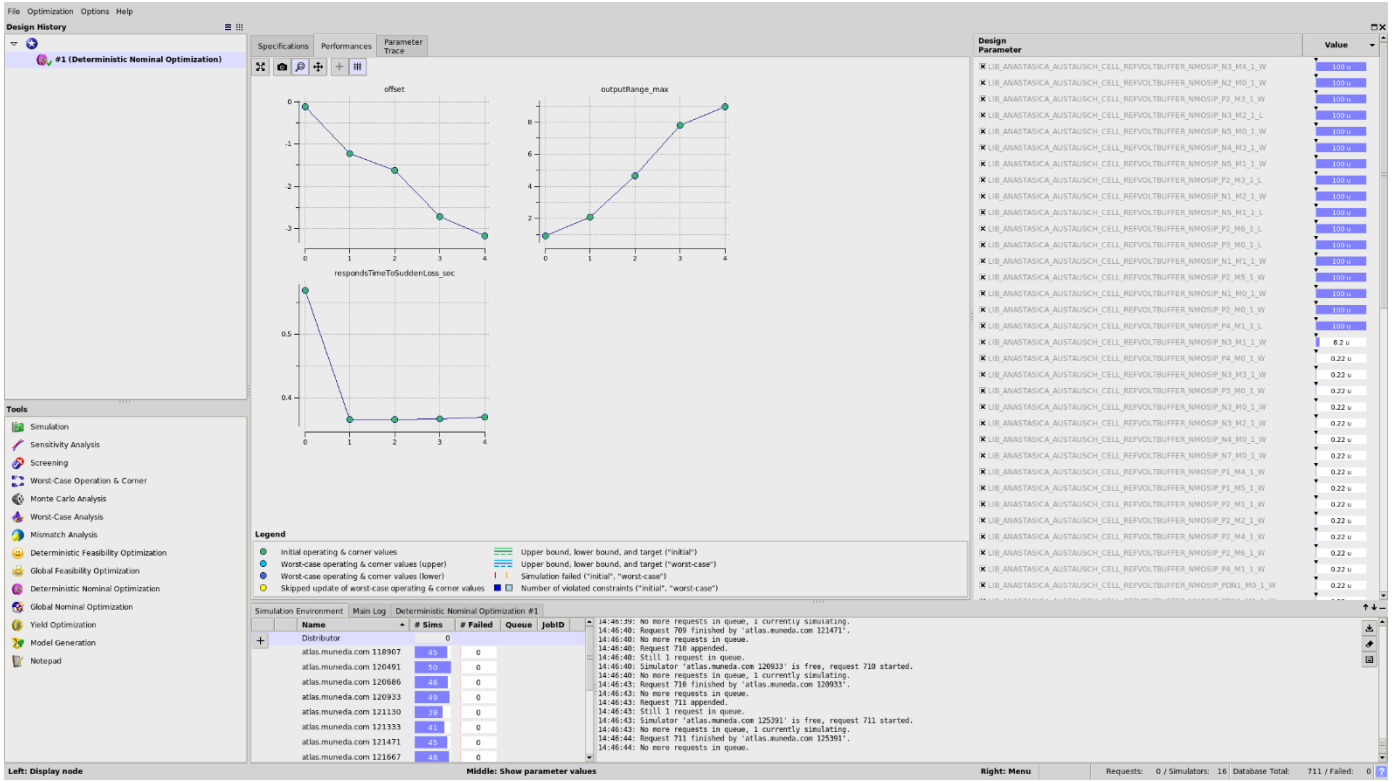
- Optimization:
  - Change design parameters (e.g. length and/or width of transistors)
  - Make performances fulfill specifications $f \leq f_b$ (e.g. power ≤ 1 µW)
  - Additional considerations
    - Operating conditions (e.g. temperature, supply voltage)
    - Process variations & mismatch
    - Ageing
- Optimization can be applied at all levels of abstraction (block, IP, system, …)

System Optimization Results

Performance	Optimization Goal	Initial	Final
outputRange_max	maximize	935.3 mV	8.947 V
responseTimeToSuddenLoss_sec	minimize	569 ms	370 ms

- Duration: 20 min
- # system evaluations (simulations): 711

System Optimization Results



Structure of the Presentation

- Problem
- Approach
- Example: Tire Pressure Metering System
- Results
- **Conclusion**

Core Advantages of this Workflows

- Approach is semi-automatic
- Based on re-use of models
- Brings selected parameter-dependencies to system level simulation
- Allows verification and system optimization within given limits by selected bounds, performances, …

Time for your Questions!

GEFÖRDERT VOM

AnastASICA

Bundesministerium
für Bildung
und Forschung