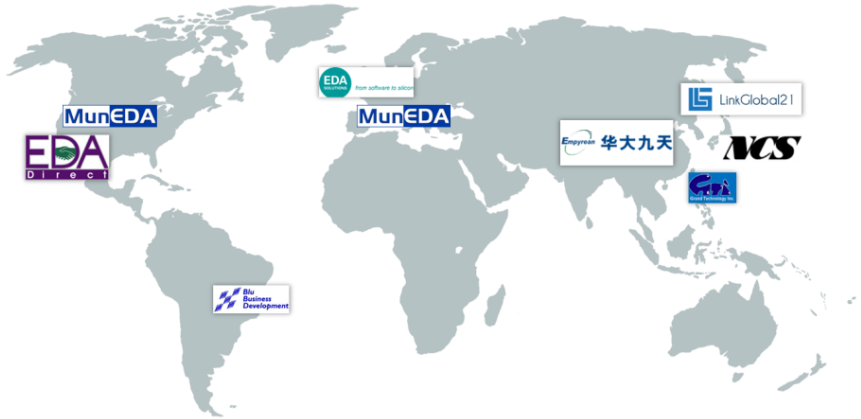
Optimization of Complex Circuits at System Level

MunEDA Corporate Overview

- EDA Software Vendor – Tools SPT & WiCkeD™ for Migration, Sizing and Verification of full custom nanometer IC designs.

- Worldwide Sales & Support Offices in USA, Korea, China, Taiwan, Japan, UK, Ireland, Scandinavia, South America



- Worldwide Customer & Partner Base with Semiconductor IDMs, Fabless Design Houses & Foundries

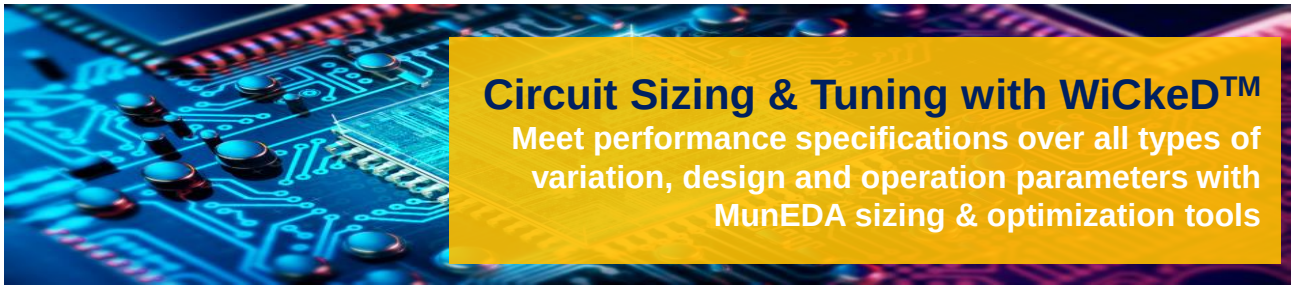
For more information visit www.muneda.com

MunEDA EDA Solutions for Full Custom IC Migration, Sizing & Verification

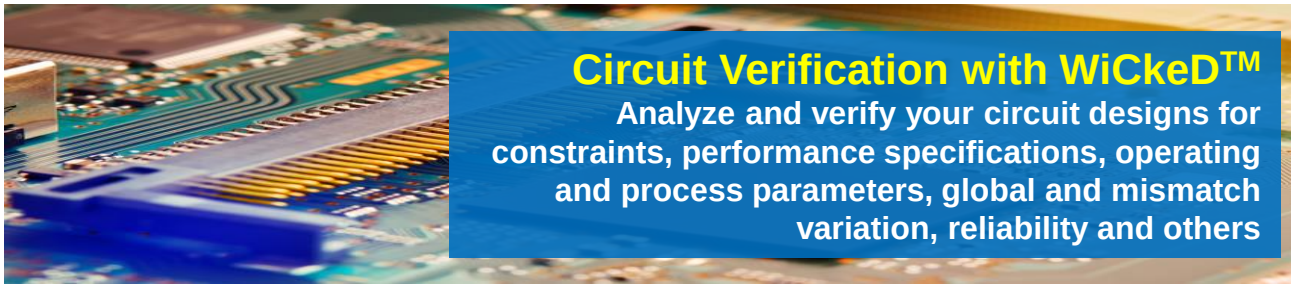
MunEDA – Comprehensive EDA Tools for Migration, Sizing and Verification of Nanometer Custom IC Designs

MunEDA
SPT
Schematic Porting Tool for IC & IP Migration

MunEDA
WiCkeD™
Circuit Sizing & Tuning

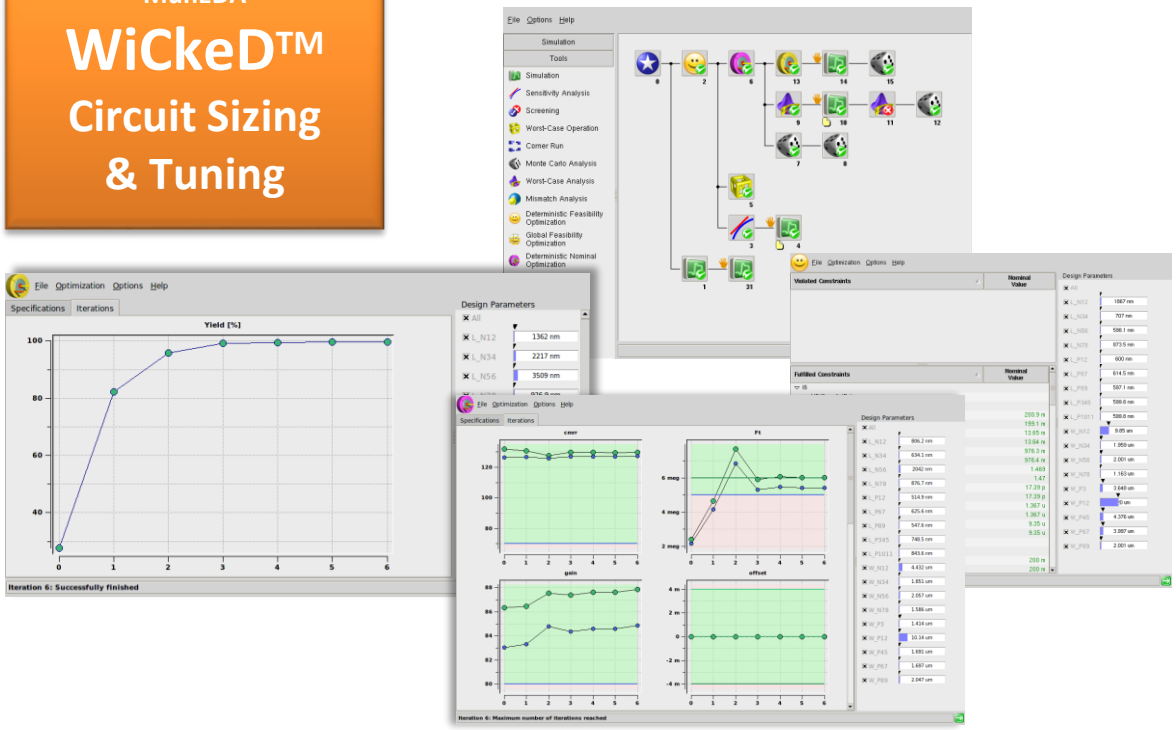


MunEDA
WiCkeD™
Circuit Analysis & Verification



MunEDA WiCkeD™ Tool Suite for Circuit Sizing & Tuning

MunEDA
WiCkeD™
Circuit Sizing & Tuning



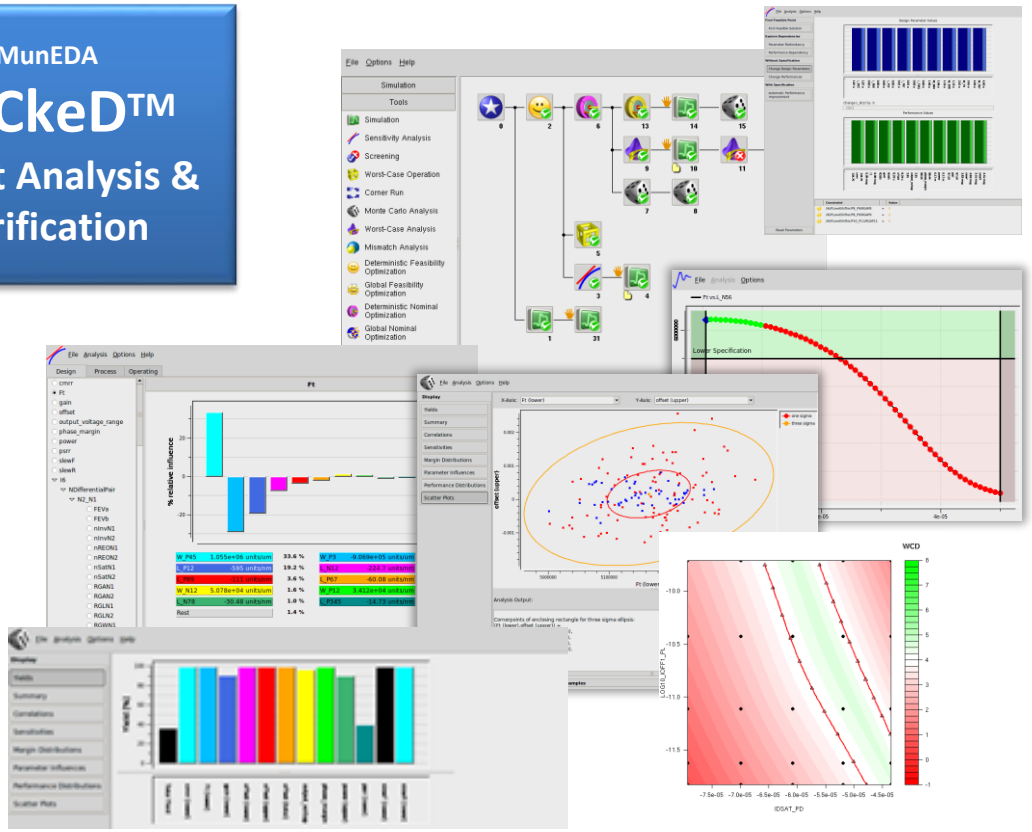
Selected Features:

- Circuit Constraint & Feasibility Optimization
- Deterministic Nominal Circuit Optimization
- Global Nominal Circuit Optimization
- Discrete Circuit Optimization
- Yield Optimization
- Circuit sizing under consideration of degradation and reliability effects
- Circuit Design History & Database
- Parallel simulation environment (supporting industry-standard SPICE simulators)
- Multitestbench-Support

- **MunEDA WiCkeD™ Circuit Sizing & Tuning: simulation-based tools to improve and optimize custom circuits**
- **Targets constraint fulfilment, performance specifications, robustness, targets for yield, area, stability, timing, power, reliability, aging, degradation, stress, self heating and more !**

MunEDA WiCkeD™ Tool Suite for Circuit Analysis & Verification

MunEDA
WiCkeD™
Circuit Analysis & Verification



Selected Features

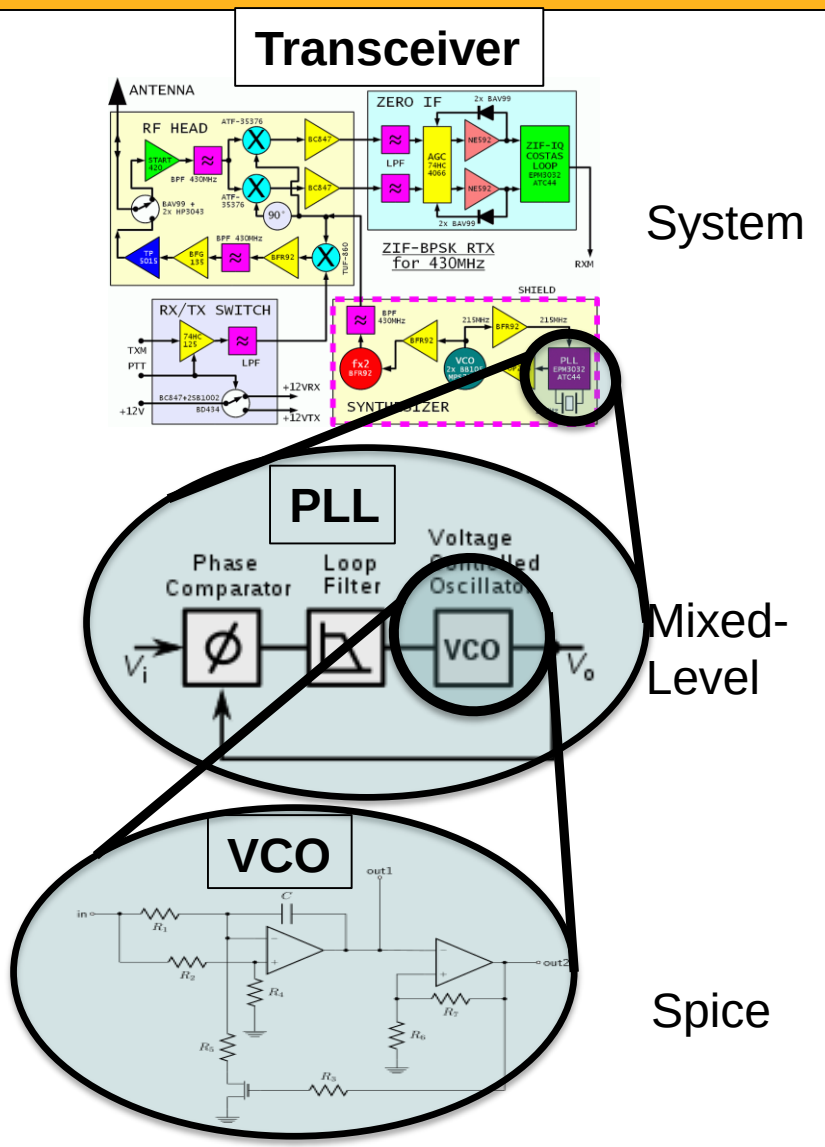
- Parallel simulation environment (supporting industry-standard SPICE simulators)
- Multitestbench-Support
- Circuit Constraint Feasibility Check
- Circuit Sensitivity Analysis for performances, gain, frequency, slewrates, power, stability, area, others
- Fast PVT & Operating Corner Analysis: influence of corner cases on given circuit performances
- Parameter Screening
- Fast & Enhanced Monte Carlo Analysis (3-5 sigma)
- High Sigma Worst Case Analysis (6-9 sigma)
- Importance Sampling & Robustness Verification
- Global and local (mismatch) variation and yield analysis
- Reliability & aging analysis for degradation effects
- Yield plot sweeps for global variation
- RSM Circuit Modelling & Model Generation

- **MunEDA WiCkeD™ Circuit Analysis & Verification: comprehensive and powerful tools for simulation-based analysis and verification of Custom IC**
- **Targets circuit constraints, design parameters, PVT, operating corners, specifications, sensitivities, correlations, process dependencies, sweeps, global and mismatch variation, Monte-Carlo, Worst-Case, High Sigma, reliability and more !**

Motivation

Complexity

- Analysis and optimization requires multiple simulations
- Works well with simulation times in the range of seconds/minutes
- Successfully applied with spice simulations at block level (OpAmps, Bandgaps, Memory cells, standard cells, etc.)
- For more complex circuits with simulation times of several hours (i.e. PLL, ADC) → no fun

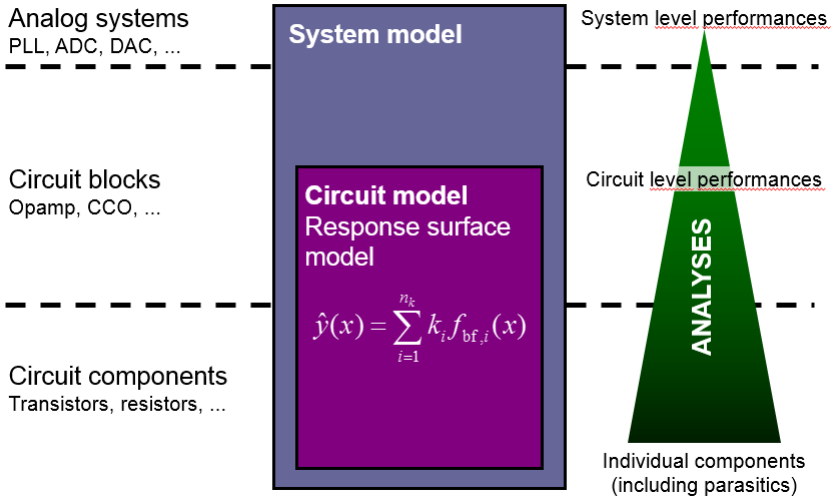


Source of graphics: Wikipedia, http://readingrat.net/

Motivation

Approaches

- Methodologies to create behavioural models of complex circuits
  → stuck in tradeoff between effort to create the model vs. accuracy
- Methodology to break up in blocks, model and simulate mixed-mode
  → cumbersome, mixed-tool environment



Requirements

- Simplified creation of AMS Models
- Mixed-Level Simulation for optimization, verification, and debugging

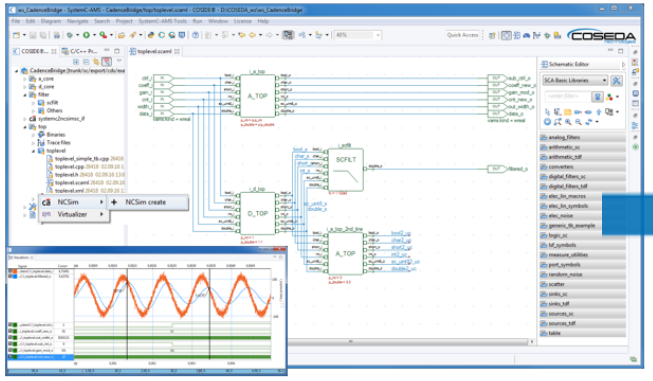
Solution via Coside

- System C / System C AMS enables simulation of complex circuits X times faster than spice level
- Coside practical tool for modeling, mixed-simulations, and debugging

↗ Mixed-level analysis / optimization

COSEDA-Cadence Bridge

SystemC/SystemC AMS Model created in COSIDE®



COSEDA technologies

Confidential

9

Initial export is system level, but as soon as individual blocks get implemented, optimization on mixed level is possible

Note: Mixed level is possible within Coside via SystemC / SystemC AMS Spice Integration

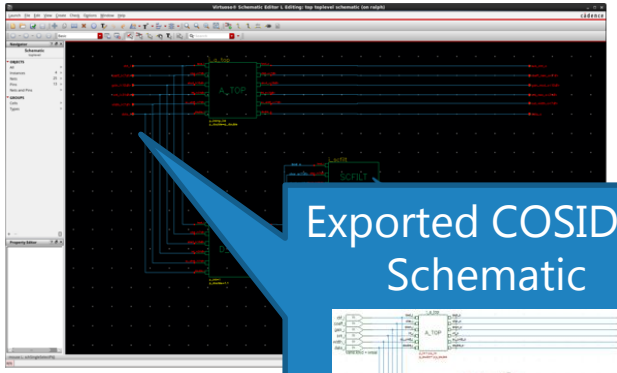
WiCkeD

Circuit Design Optimization

Optimize with WiCkeD in Virtuoso on exported netlist from Coside

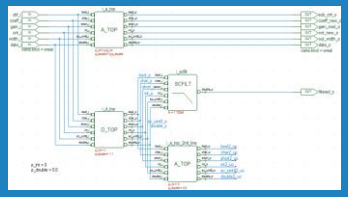
COSEDA-Cadence Bridge

Exported Toplevel Netlist from COSIDE® to Cadence Virtuoso



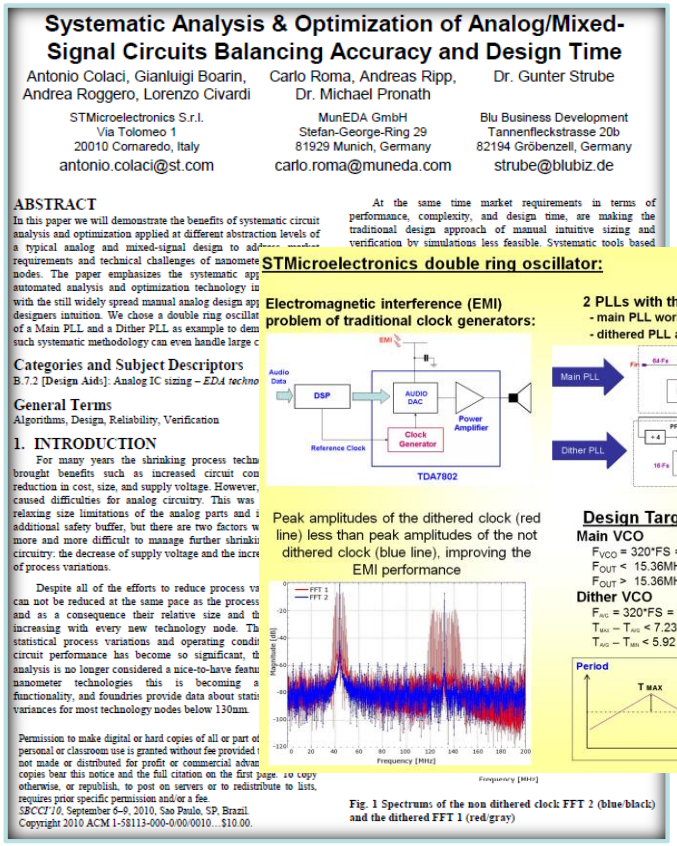
COSEDA technologies

Exported COSIDE® Schematic



Mixed-Level Optimization

- **Mixed-level optimization within normal EDA design framework**
- **STM paper in 2010**
  →optimization of VCO within Dithered PLL



Systematic Analysis & Optimization of Analog/Mixed-Signal Circuits Balancing Accuracy and Design Time

Antonio Colaci, Gianluigi Boarin, Andrea Roggero, Lorenzo Civardi
STMicroelectronics S.r.l.
Via Tolomeo 1
20010 Cornaredo, Italy
antonio.colaci@st.com

Carlo Roma, Andreas Ripp, Dr. Michael Pronath
MunEDA GmbH
Stefan-George-Ring 29
81929 Munich, Germany
carlo.roma@muneda.com

Dr. Gunter Strube
Blu Business Development
Tannenfleckstrasse 20b
82194 Gröbenzell, Germany
strube@blubiz.de

ABSTRACT

In this paper we will demonstrate the benefits of systematic circuit analysis and optimization applied at different abstraction levels of a typical analog and mixed-signal design to ad... requirements and technical challenges of nanometer nodes. The paper emphasizes the systematic ap... automated analysis and optimization technology in... with the still widely spread manual analog design ap... designers intuition. We chose a double ring oscillat... of a Main PLL and a Dither PLL as example to dem... such systematic methodology can even handle large c...

At the same time market requirements in terms of performance, complexity, and design time, are making the traditional design approach of manual intuitive sizing and verification by simulations less feasible. Systematic tools based

Categories and Subject Descriptors
B.7.2 [Design Aids]: Analog IC sizing – *EDA techno...*

General Terms
Algorithms, Design, Reliability, Verification

1. INTRODUCTION

For many years the shrinking process techn... brought benefits such as increased circuit com... reduction in cost, size, and supply voltage. However, ... caused difficulties for analog circuitry. This was ... relaxing size limitations of the analog parts and i... additional safety buffer, but there are two factors w... more and more difficult to manage further shrinki... circuitry: the decrease of supply voltage and the incre... of process variations.

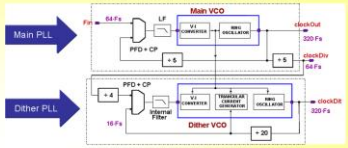
Despite all of the efforts to reduce process va... can not be reduced at the same pace as the process ... and as a consequence their relative size and th... increasing with every new technology node. Th... statistical process variations and operating condit... circuit performance has become so significant, th... analysis is no longer considered a nice-to-have featu... nanometer technologies this is becoming a... functionality, and foundries provide data about stati... variances for most technology nodes below 130nm.

SBCCI'10, September 6–9, 2010, Sao Paulo, SP, Brazil.

STMicroelectronics double ring oscillator:

Electromagnetic interference (EMI) problem of traditional clock generators:

2 PLLs with the same ring oscillator:
- main PLL working at fixed clock frequency
- dithered PLL at modulated clock frequency



Peak amplitudes of the dithered clock (red line) less than peak amplitudes of the not dithered clock (blue line), improving the EMI performance

Design Targets:

Main VCO
F_{VCO} = 320*FS =15.360MHz (closed loop)
F_{OUT} < 15.36MHz for VF=2.0V
F_{OUT} > 15.36MHz for VF=3.0V

Dither VCO
F_{AVG} = 320*FS = 15.360MHz
$T_{MAX} - T_{AVG} < 7.23$ ns
$T_{AVG} - T_{MIN} < 5.92$ ns

Well suited for:
1) Systematic sizing
2) Usage of different abstraction levels (Hierarchical approach)

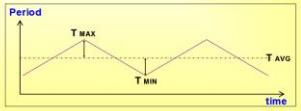
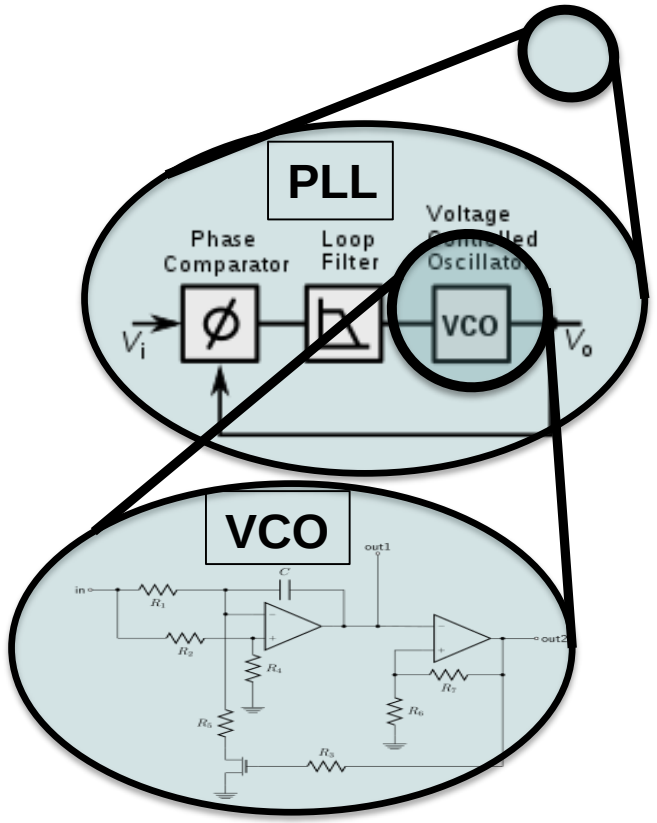
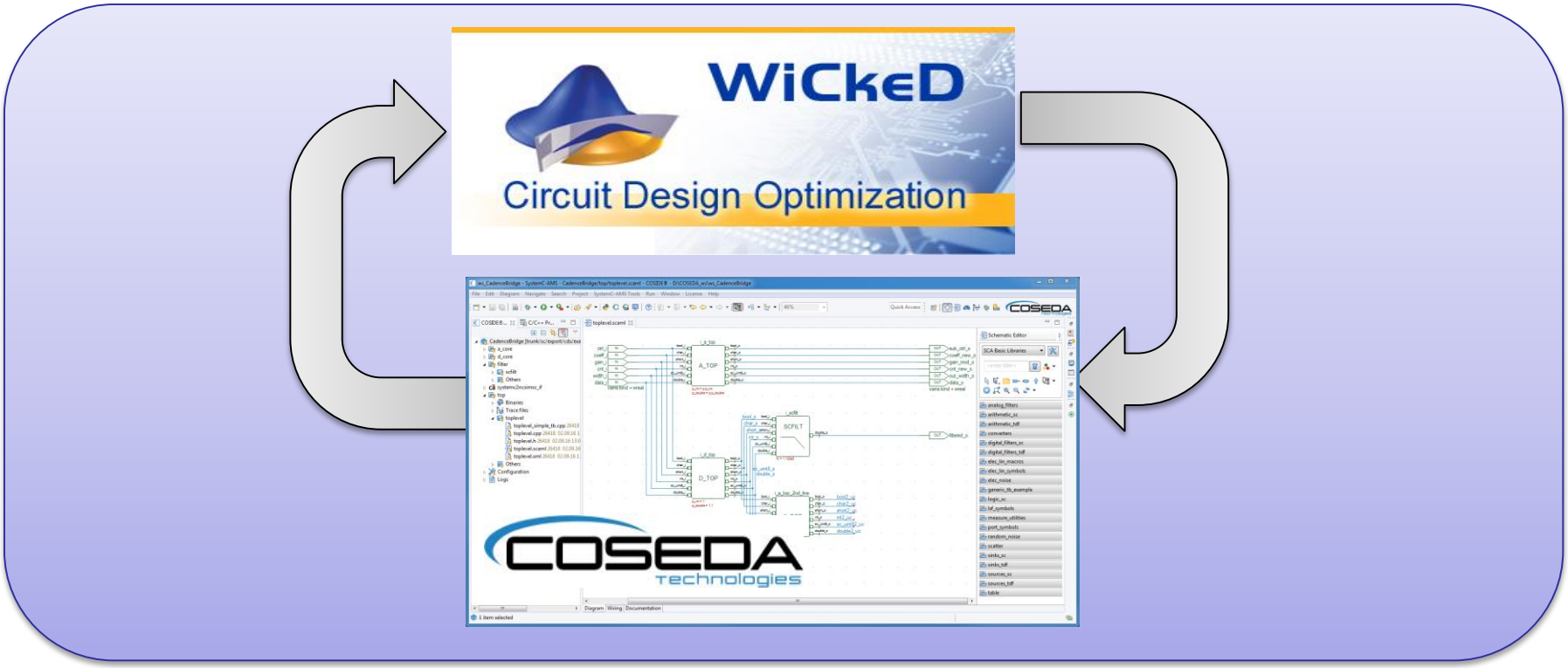


Fig. 1 Spectrums of the non dithered clock FFT 2 (blue/black) and the dithered FFT 1 (red/gray)



Using Coside to Apply Optimization to System

- **Full hierarchcial flow: use WiCkeD @ system level**
- **WiCkeD / Coside interface → optimize using SystemC simulation**



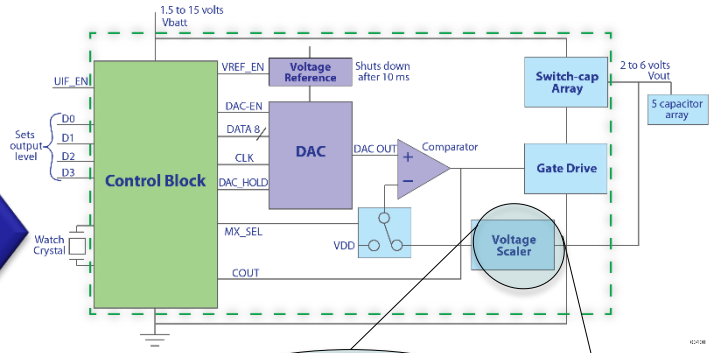
- Mixed level is possible by Coside SystemC / SystemC AMS Spice Integration
- Optimization can include „other components“ (mechanical, etc.)

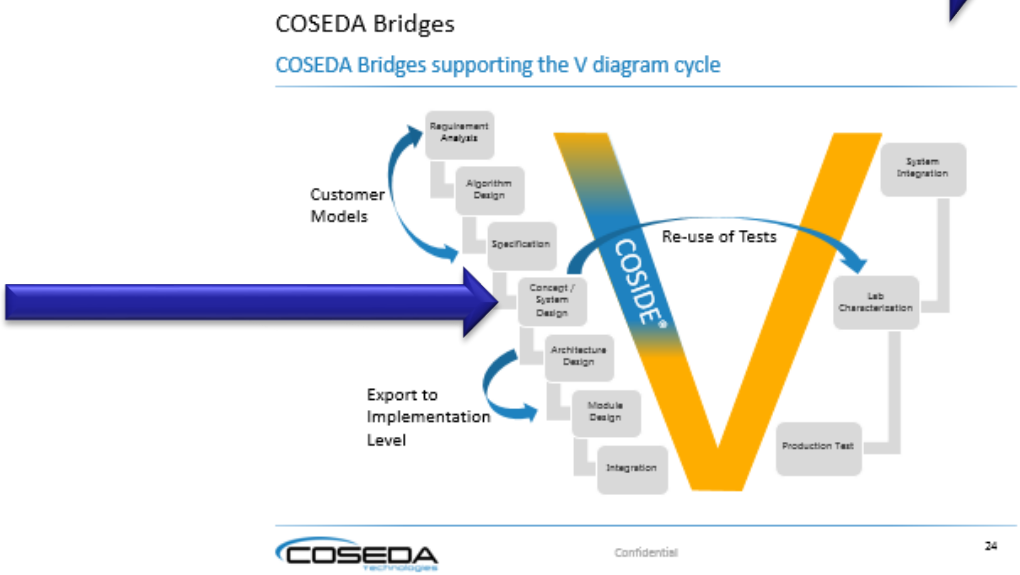
System-Level Optimization

- **System-level optimization with SystemC AMS**
- **Conceptual test: DC-DC converter**

Analysis and optimization done at system level

(prior to implementation)





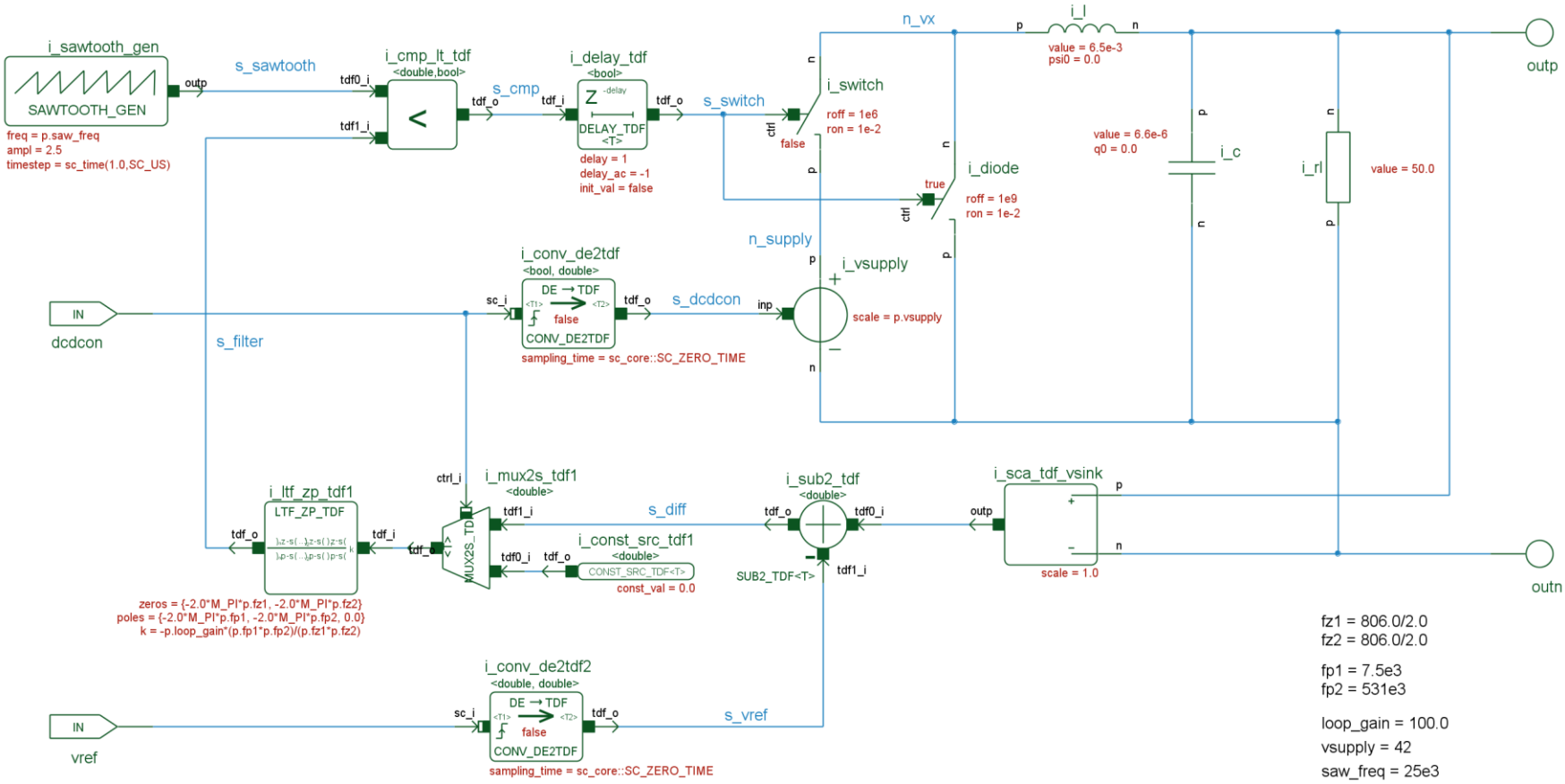
Component implementation to be done later

Block implementation to be done later

Source of graphic: https://www.ridgetopgroup.com/products/semiconductors-for-critical-applications/instacell-ip-core-library/dcdc-converter/

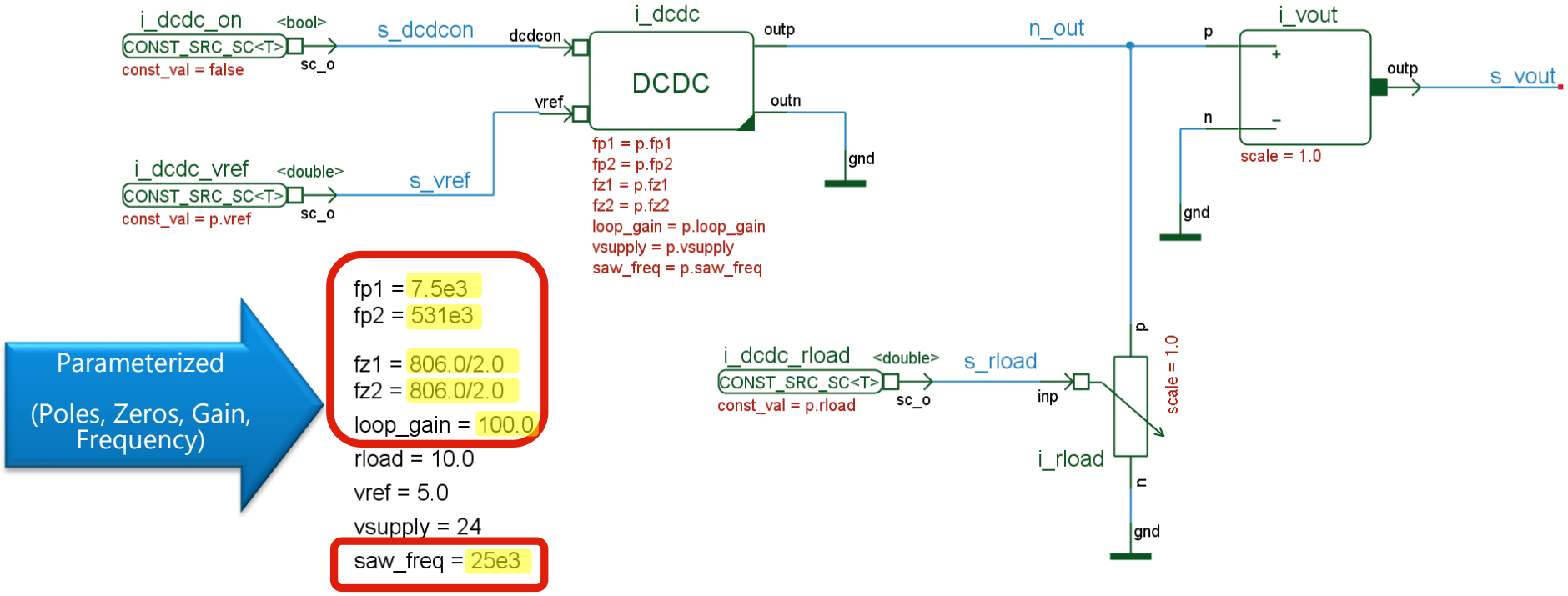
DC-DC converter example

DCDC Systemlevel model



DC-DC converter example

Testbench



DC-DC converter example

Input parameter file

```
//parameter for optimization
fp1 = 7.5e3; //loop filter poles and zeros
fp2 = 531e3;
fz1 = 403.0;
fz2 = 403.0;
loop_gain = 10000.0;  //loop gain
saw_freq = 25e3;      //sawtooth generator frequency

//configuration parameter
rload = 10.0;  //load resitance
vref = 5.0;    //reference voltage
vsupply = 24;  //supply voltage

trace_file="onchar.dat";  //trace file

//measurement configuration
settling_tolerance=0.1;  // +/- vref to be consdered as inbound
settle_time=5e-3;        // inbound time to be considered as settled
max_settle_time=50e-3;   //maximum measurement time -> if not
settled within this time -> unsettled
```

DC-DC converter example

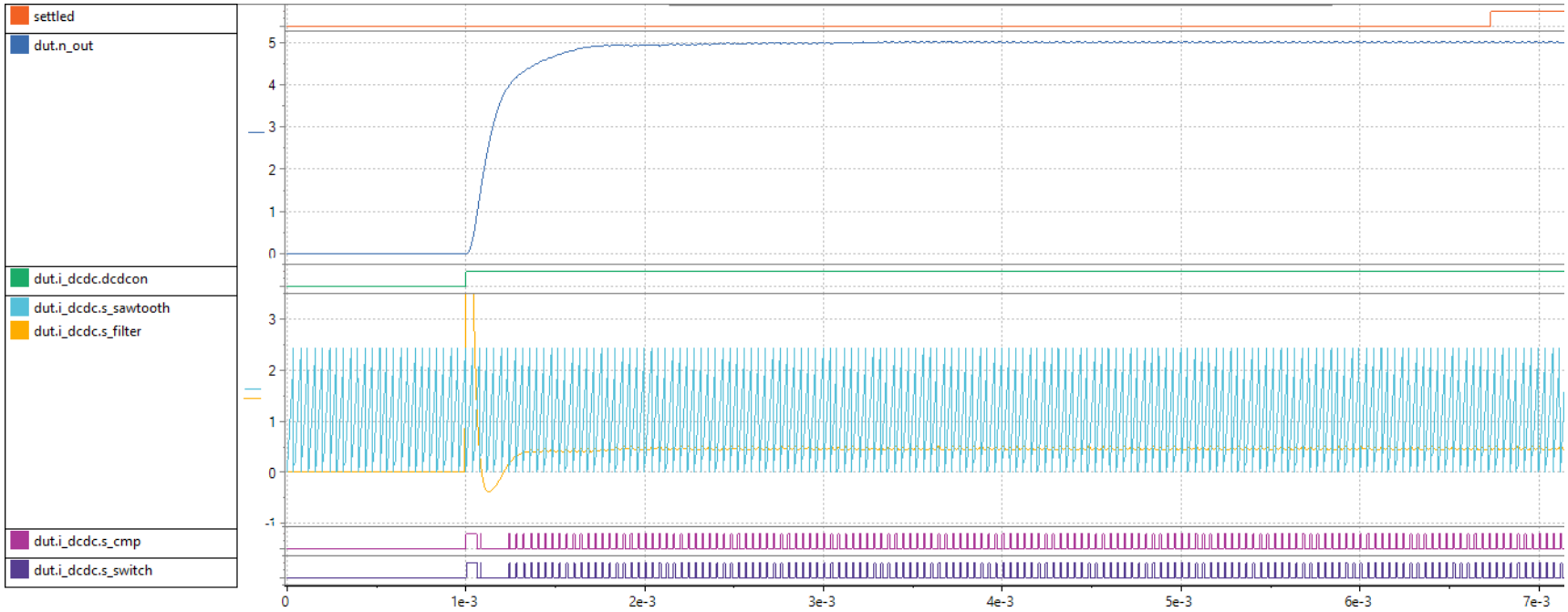
Testsequence (to calculate settling time)

```
void dcdc_tb_stim_onchar::stimulus_sequence()
{
    std::cout << sc_core::sc_time_stamp() << " Stimulus sequence started..." << std::endl;

    const sc_core::sc_event& change_ev=s.vout.get_new_value_event(); /* get event for dcdc out*/

    wait(1.0,SC_MS);

    s.dcon.set_value(true);  //switch converter on
    double dcon_time=sc_time_stamp().to_seconds();

    check_state cstate=OUTBAND;

    double last_outtime=0.0;
    double settling_time=1e9;

    double max_overshot=0.0;
    double max_undershot=0.0;
    bool finished=false;
    bool shot=false;

    while(true)
    {
       wait(change_ev); /* continue if next dcdc out value available*/
       double vout= s.vout.get_value();

       if(vout > dut_p->vref) shot=true; //if once over vref -> start shot measurement

       if(shot)
       {
         if(vout-dut_p->vref>max_overshot) max_overshot=vout-dut_p->vref;
         if(vout-dut_p->vref < max_undershot) max_undershot=vout-dut_p->vref;
       }

       double ctime=sc_time_stamp().to_seconds();

       switch(cstate)
       {
         case OUTBAND:
               last_outtime=ctime;
             if(fabs(vout-dut_p->vref)<s.settling_tolerance)  cstate=INBAND;
        break;
```

```
        case INBAND:
            if(fabs(vout-dut_p->vref)>s.settling_tolerance)  cstate=OUTBAND;


          if((ctime-last_outtime)>s.settle_time) //settled
          {
            settling_time=ctime-dcon_time-s.settle_time;
            s.settled=true;
            finished=true;
          }
          break;
        }

        if(finished)
        {
        std::cout << "Settling time: " << settling_time << std::endl;
        std::cout << "overshot: " << max_overshot << std::endl;
        std::cout << "undershot: " << max_undershot << std::endl;
        break;
        }



        if(sc_time_stamp().to_seconds()-dcon_time > s.max_settle_time)
        {
        std::cout << "Not settled" << std::endl;
        break;
        }
        }


        std::ofstream fstr("dcdcon_result.dat");
        fstr << std::setprecision(15);
        if(!finished)
        {
         fstr << "Settling time = " << "  " << ";" << std::endl;
         fstr << "overshot = " << max_overshot << std::endl;
        fstr << "undershot = " << max_undershot << std::endl;
        }
        else
        {
      fstr << "Settling time = " << settling_time << ";"<< std::endl;
        fstr << "overshot = " << max_overshot << ";"<< std::endl;
        fstr << "undershot = " << max_undershot << ";"<< std::endl;
        }

        fstr.close();

        std::cout << sc_core::sc_time_stamp() << " Stimulus sequence finished."  << std::endl;

        wait(1.0,SC_MS);
        sc_core::sc_stop();
}
```

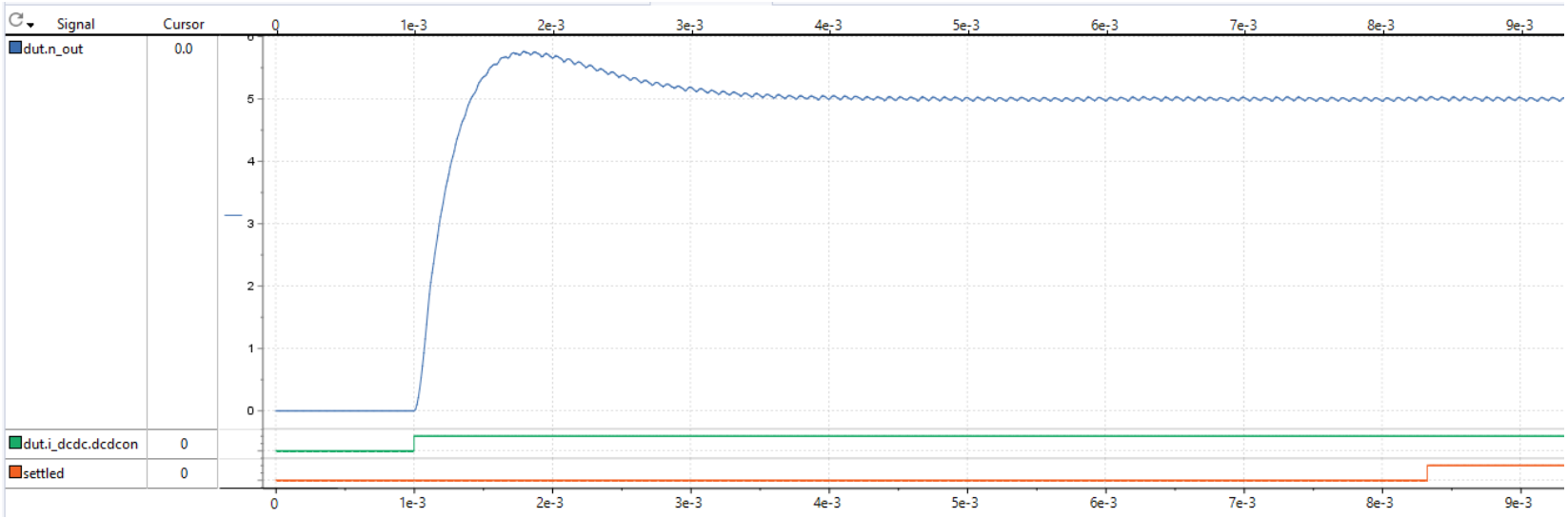
DC-DC converter example

Example for simulation result



DC-DC converter example

Simulation Result



Optimization Example

Parameters:

- two poles (fp1, fp2)
- two zeros (fz1, fz2)
- loop gain
- saw frequency

Performances:

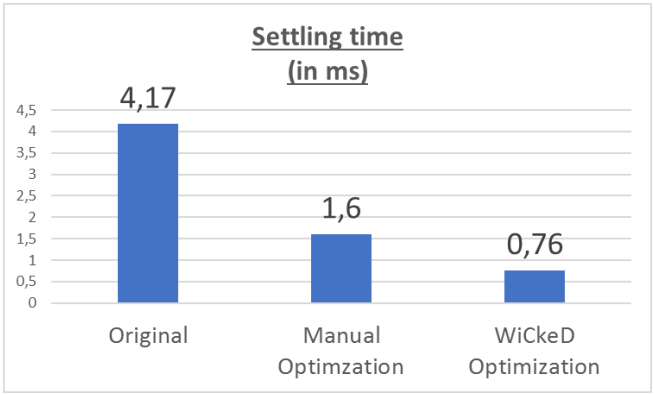
Settling Time

Optimzation Target:

- Minimize Settling Time

149 Simulations,
55s run time
(single core machine,
no parallel simulation)

vs. manual: ~1hour

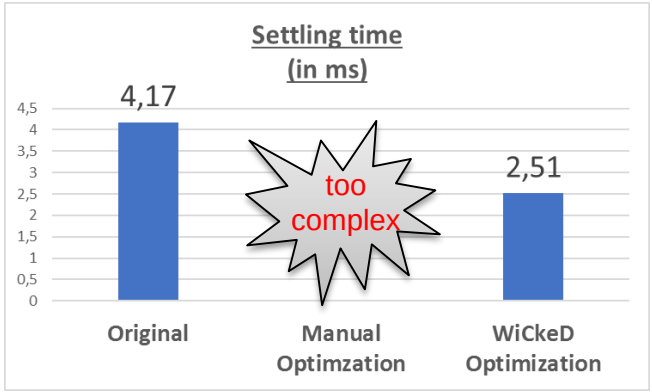


- Minimize Settling Time
- Over full operating range:

(2Ω ≤ load resistance ≤ 200Ω)

170 Simulations,
59s run time
(single core machine,
no parallel simulation)

manual estim.: >1 day



Summary

- **Simple demonstrator showed tool coupling example**
- **Optimization worked fast and showed good results (applicable to far higher complexity than this simple demonstrator)**
- **Interesting learnings / observations from this simple example:**
  - Difference between implementation and system level not just different abstraction: Parameters at system level are not physical dimensions (as transistor widths)
  - Parameters at system level can be performances (results) of sub-blocks → completely new field of applications opening up
- **Outlook:**
  - Same environment can be extended to more complex circuits
  - Interfaces at system level enable cross domain simulation (analog/digital, hardware, software, embedded, mechanical, etc.)

Thank You !