MATLAB EXPO 2019

Design and Verification of Mixed-Signal and SerDes Systems

Aniruddha Dayalu
Principal Application Engineer
Analog/Mixed-Signal and Signal Integrity

Agenda

- Motivation
- System Level Design of Mixed-Signal and SerDes Blocks
- Linking System Design to Circuit Design
- Conclusion

Mixed-Signal Integrated Circuits Require New Methodologies!

- Analog and digital circuits integrated on the same chip
- Large analog + large digital
  - ADC/DAC, PLL, Power supplies, MEMs, Sensors, …

  (control systems based on feedback loops + digital signal processing)
- Analog content in shipped semiconductor is growing

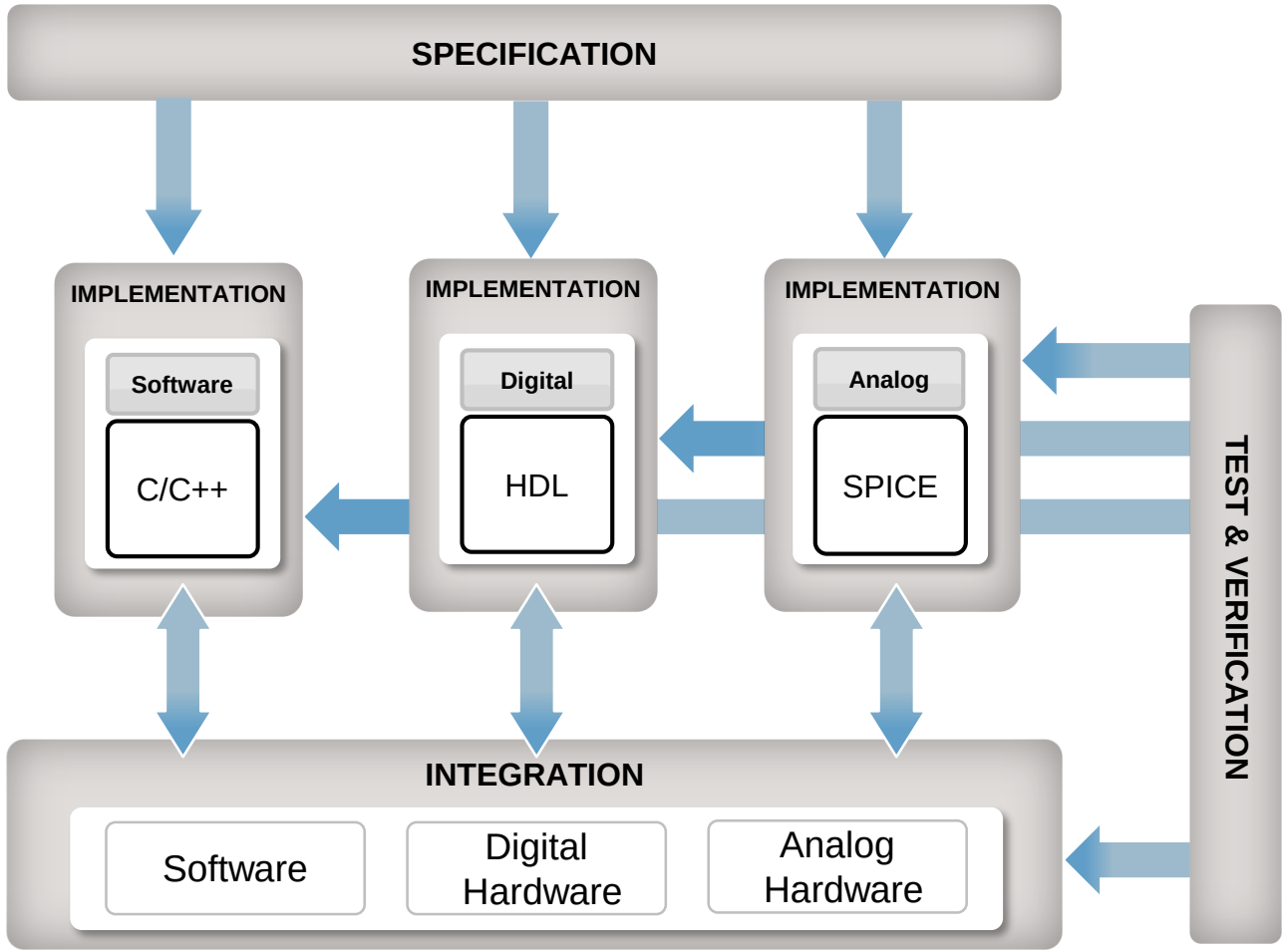
Challenges in ASIC Workflows

Design

Limited design abstractions

Design trade-offs difficult

Slow design iterations



Verification

Specification isolated from verification

Multiple, disconnected tools

Disconnected teams

Agenda

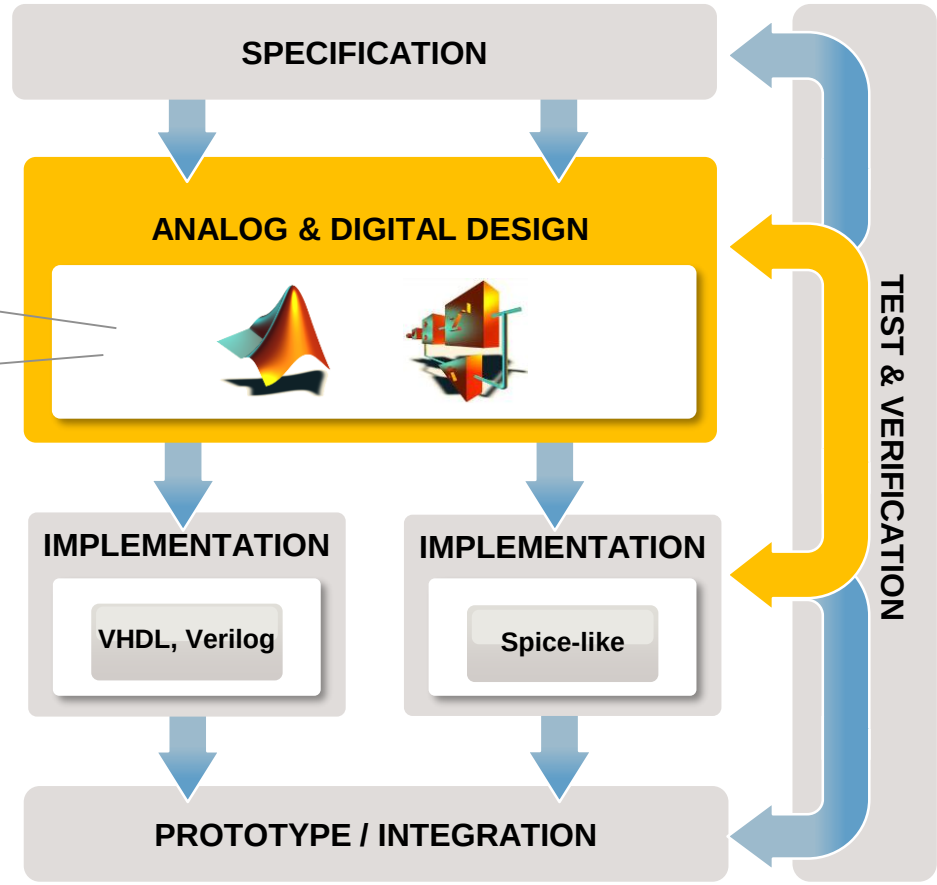
- Motivation
- System Level Design of Mixed-Signal and SerDes Blocks
- Linking System Design to Circuit Design
- Conclusion

Mixed-Signal and SerDes system design

R2019a

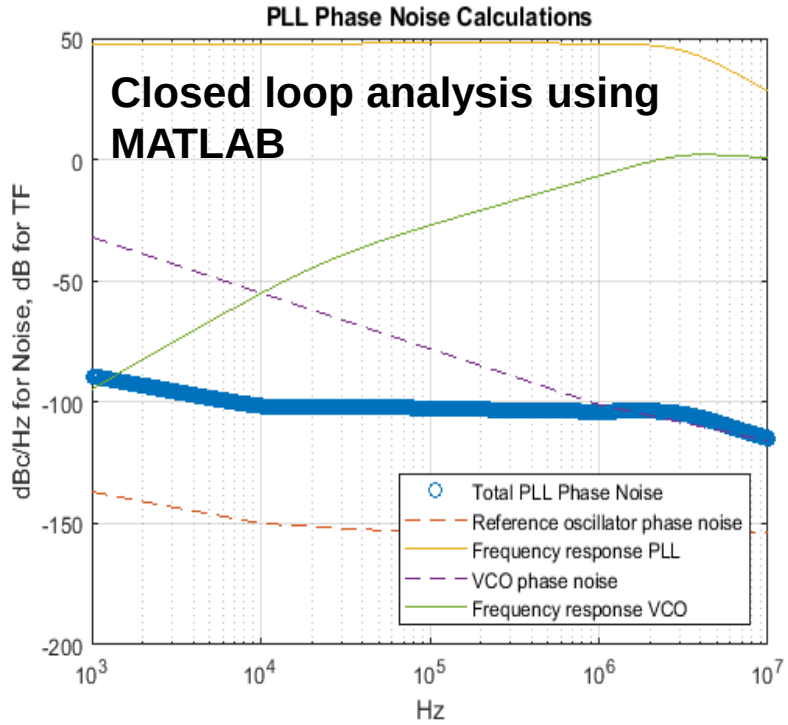
Mixed-Signal Blockset

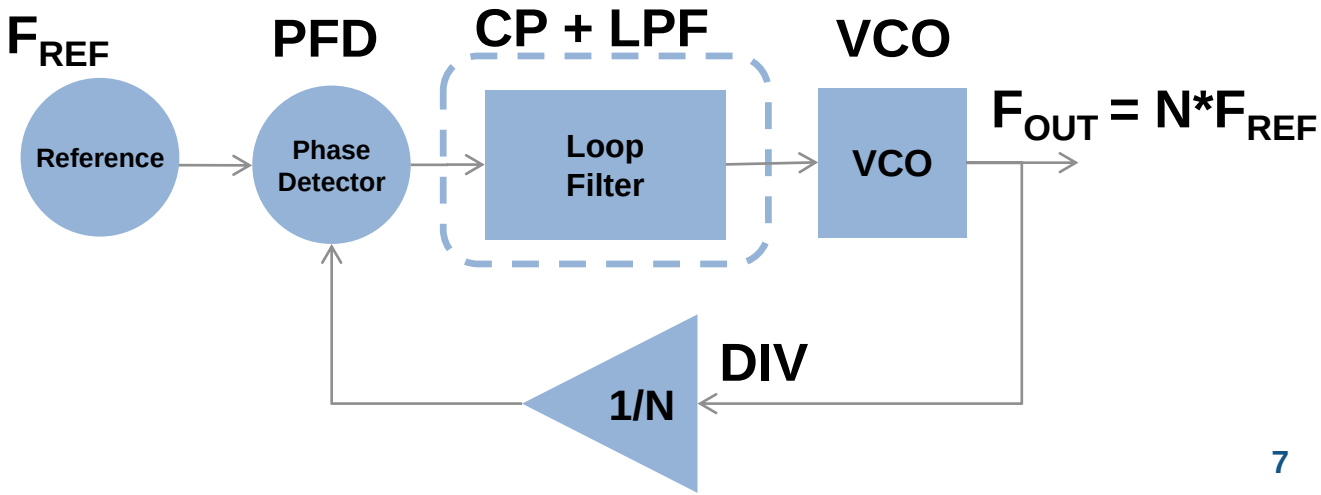
SerDes Toolbox



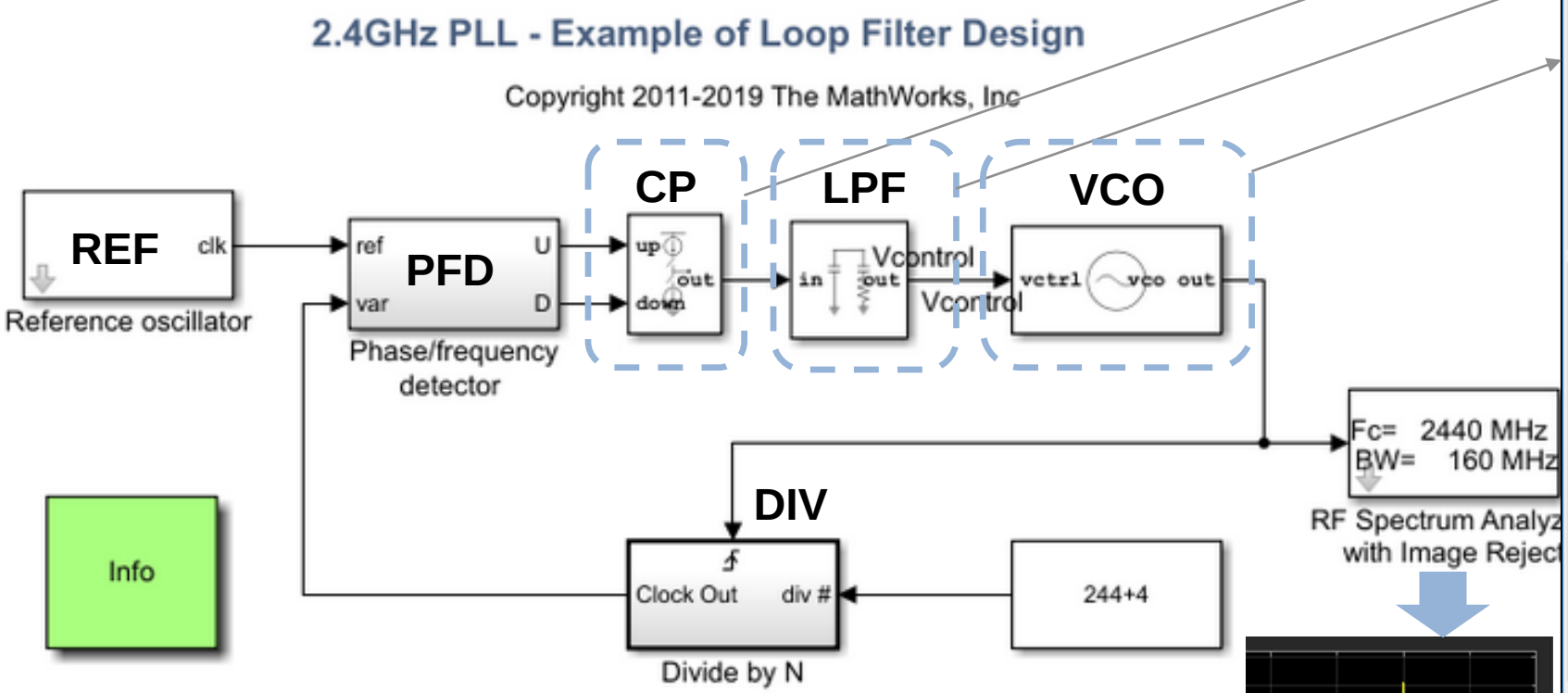
Phase-Locked Loop

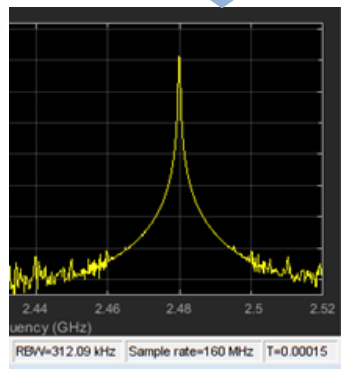
- Feedback control system
  - Which generates a signal with a fixed relation to a reference
  - It is used for frequency synthesis, synchronization
- Focus on loop filter system design
  - Design should target minimal phase noise
  - Start with a PLL behavioral model
  - Use circuit level verification to build better models for CP
  - Loop filter final design

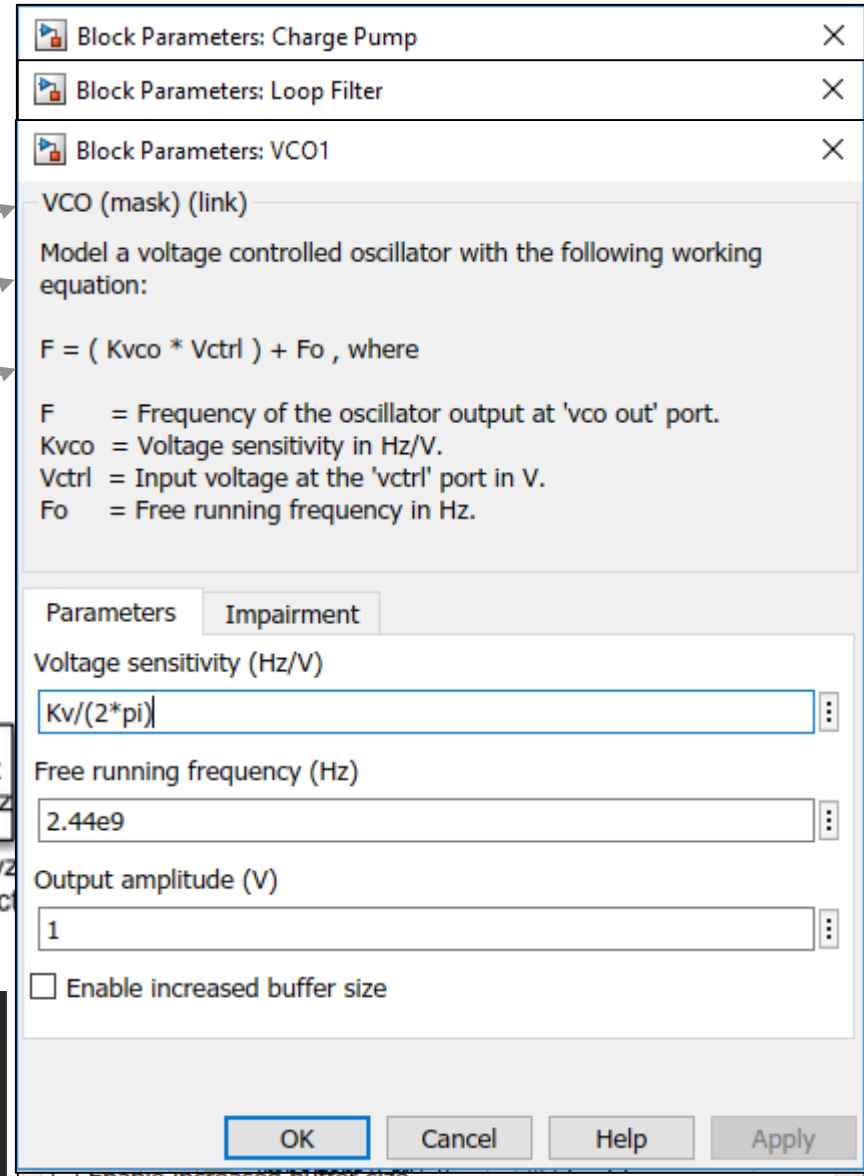




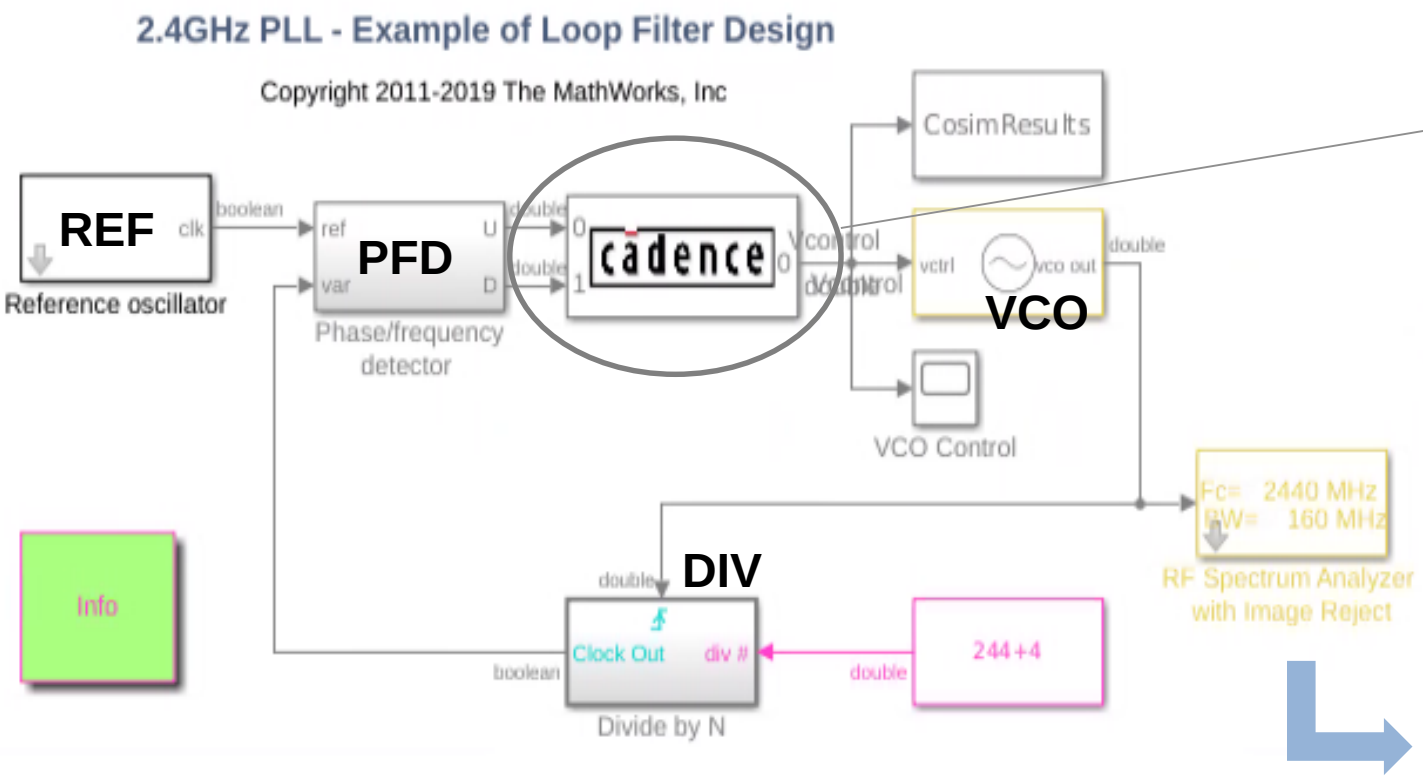
Behavioral Model of Phase-Locked Loop

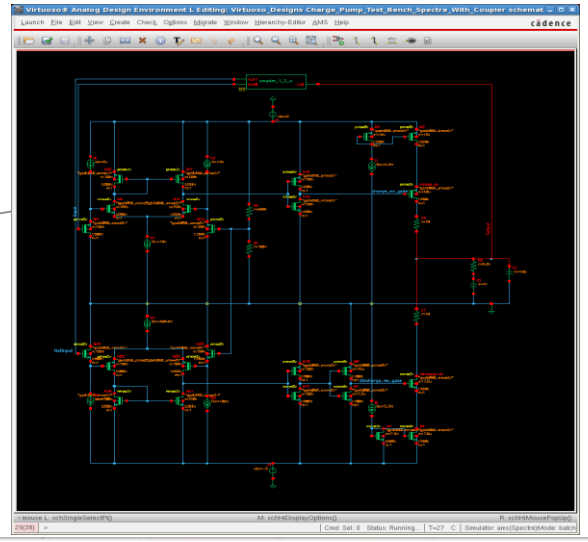


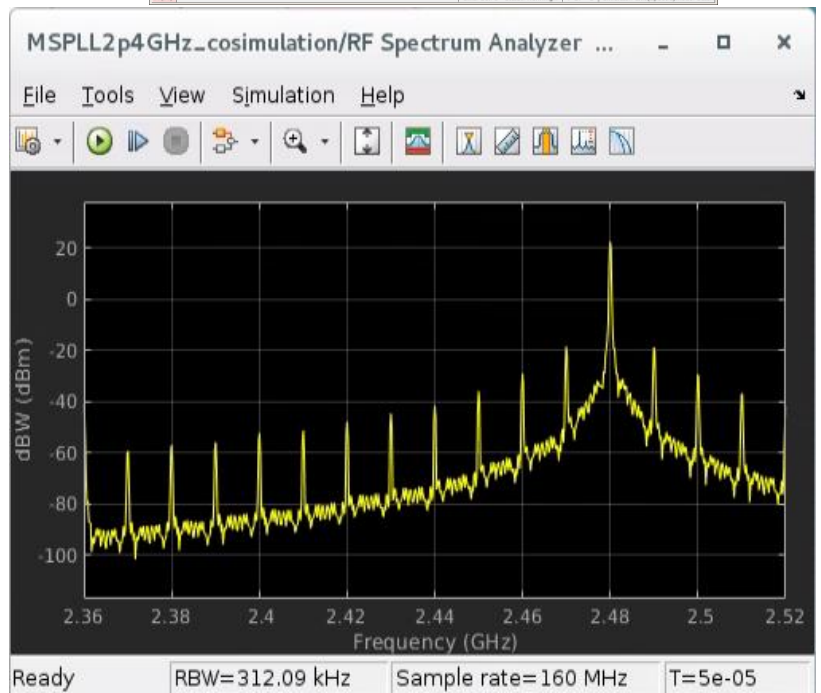




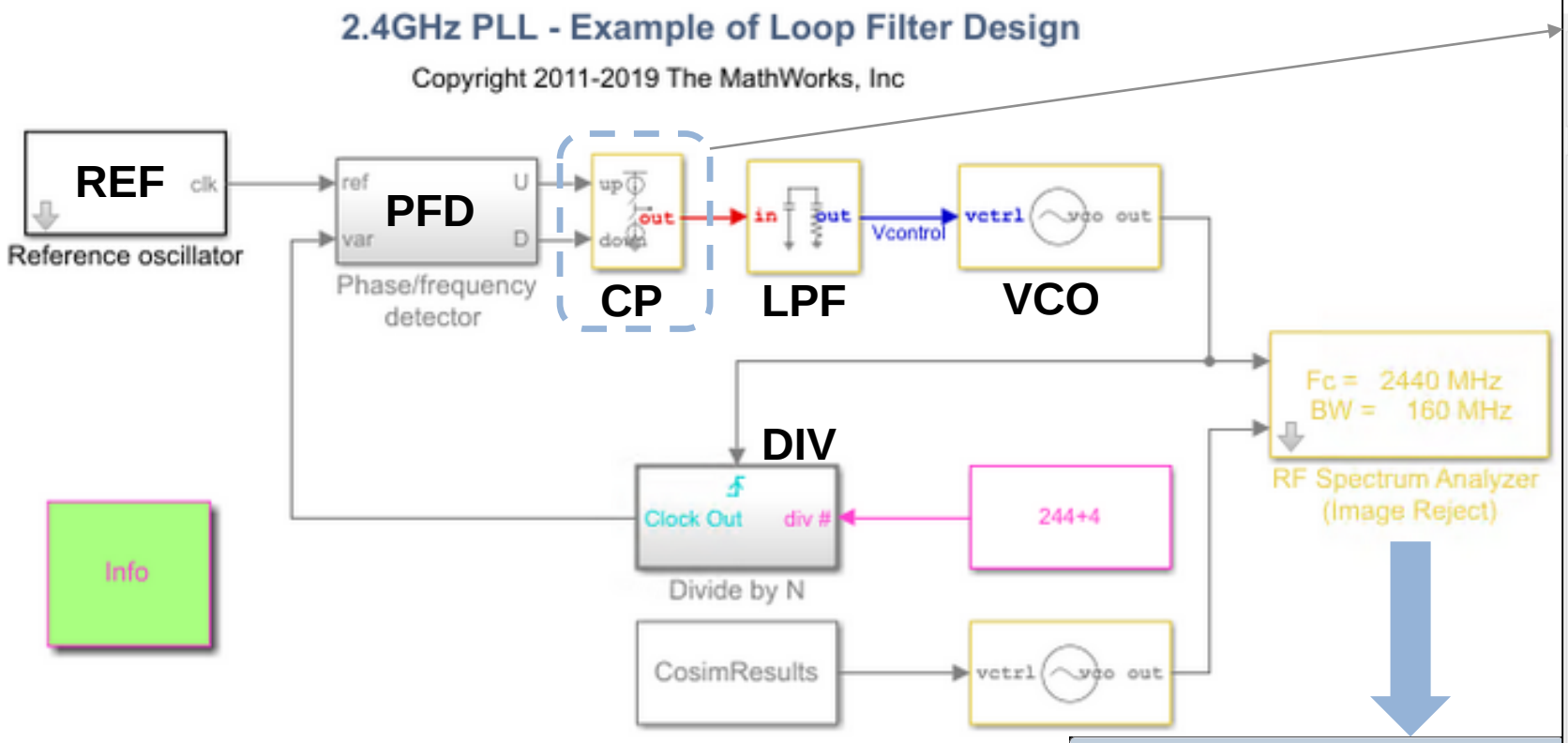
PLL Going Beyond Behavioral – Using Analog Co-simulation

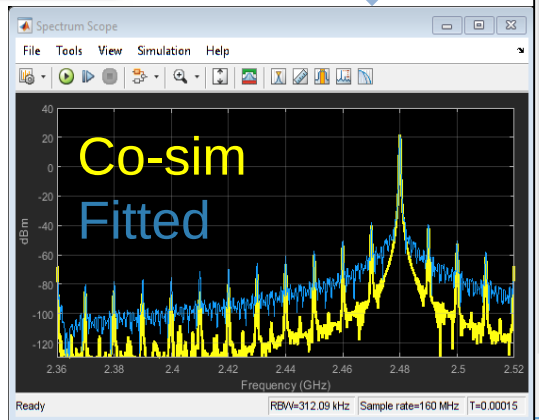


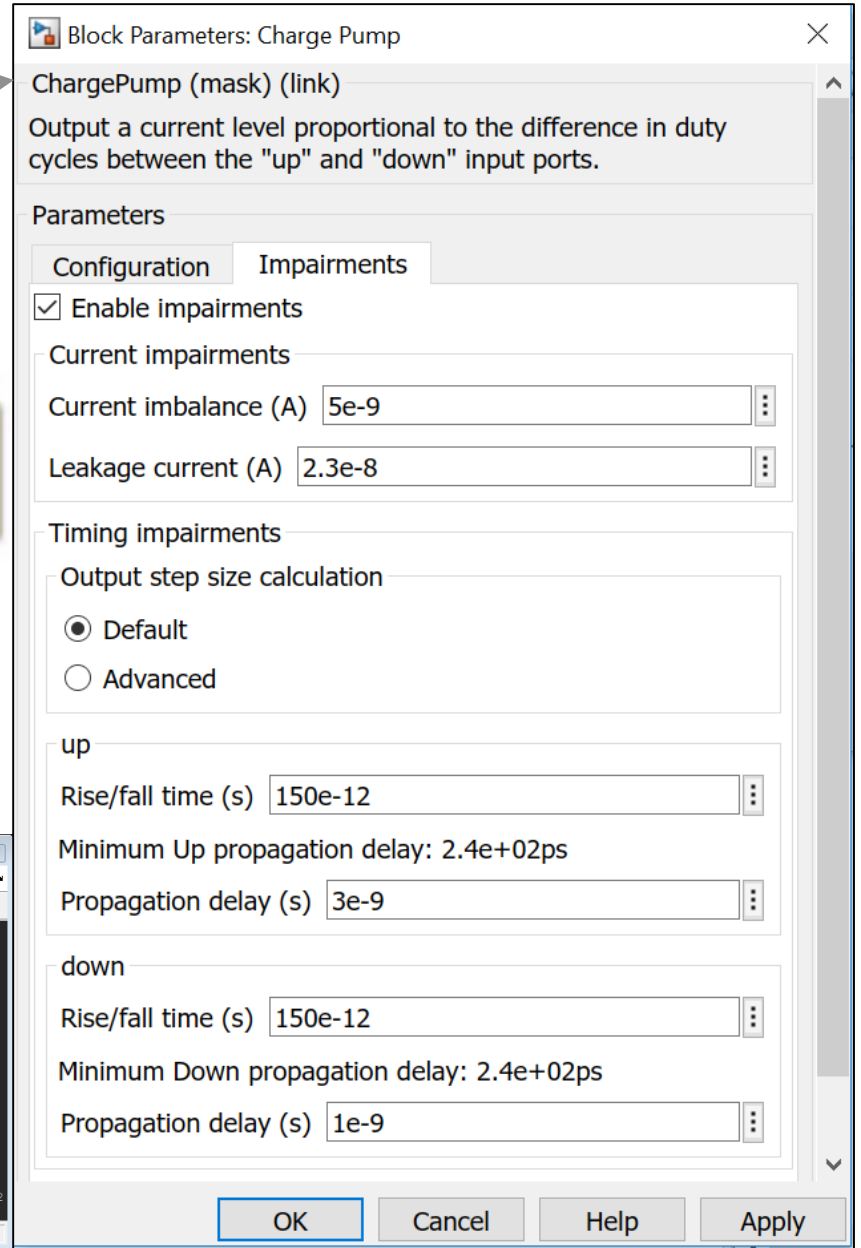




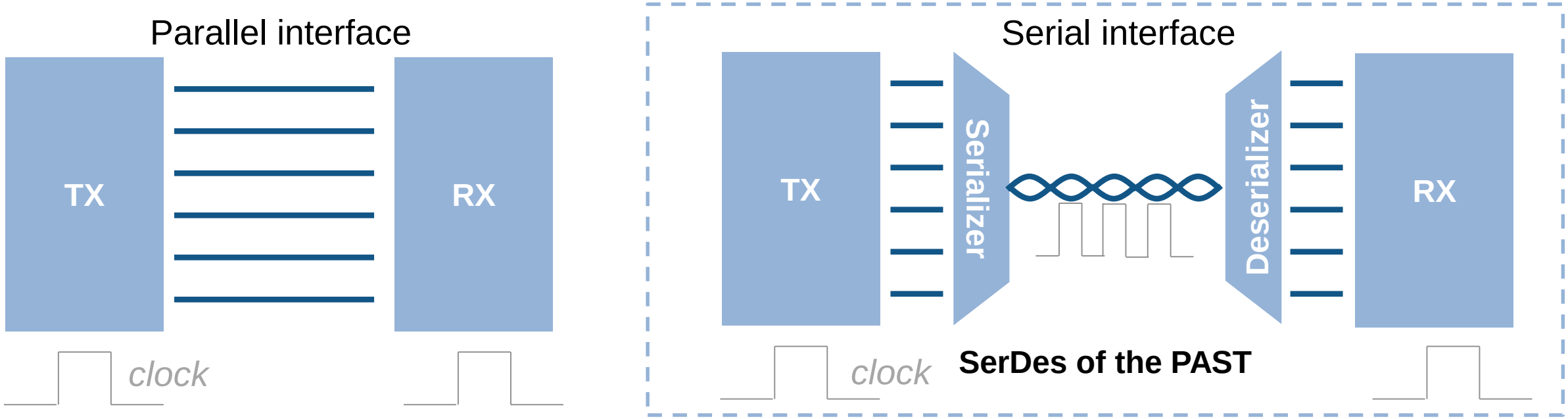
Fitting of PLL Model With Circuit Non-idealities

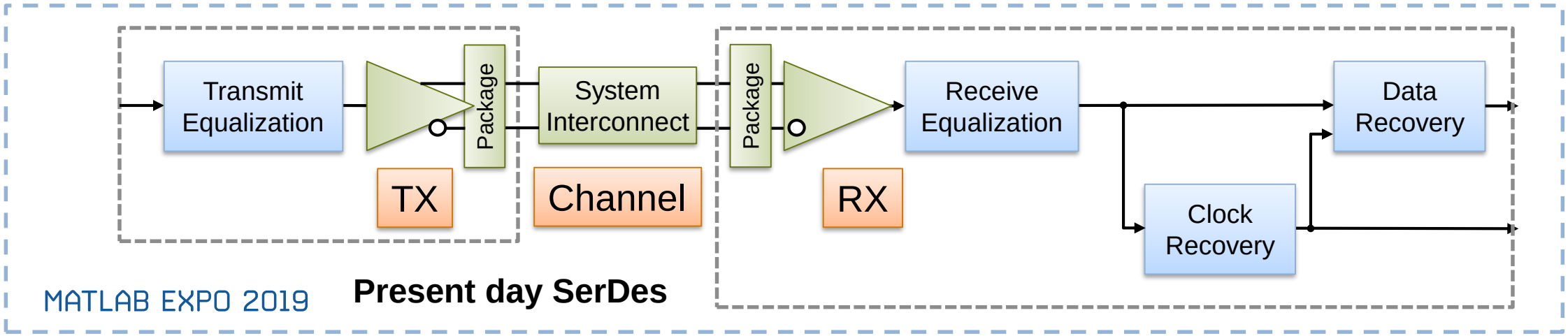






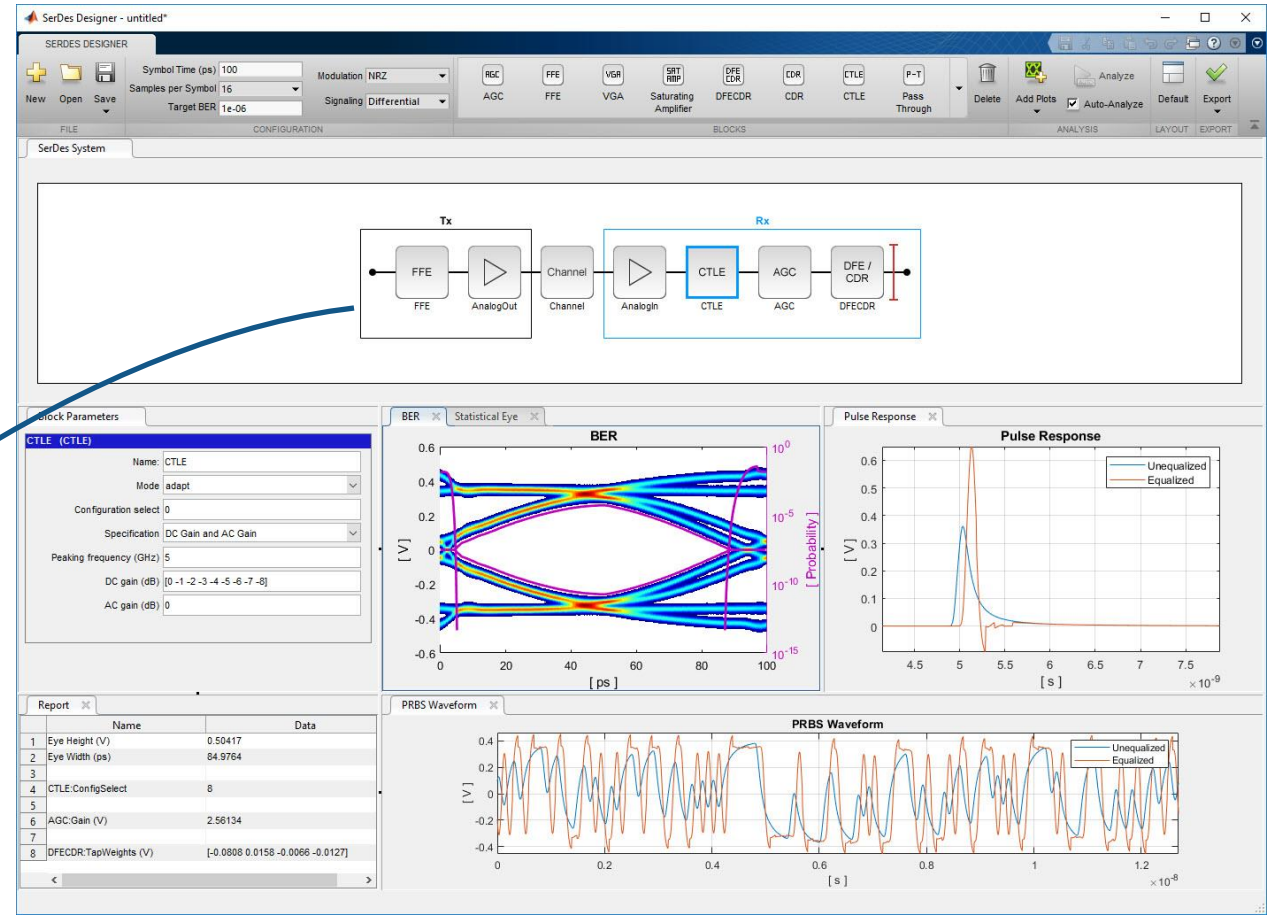
SerDes System is a Specialized Mixed Signal Circuit

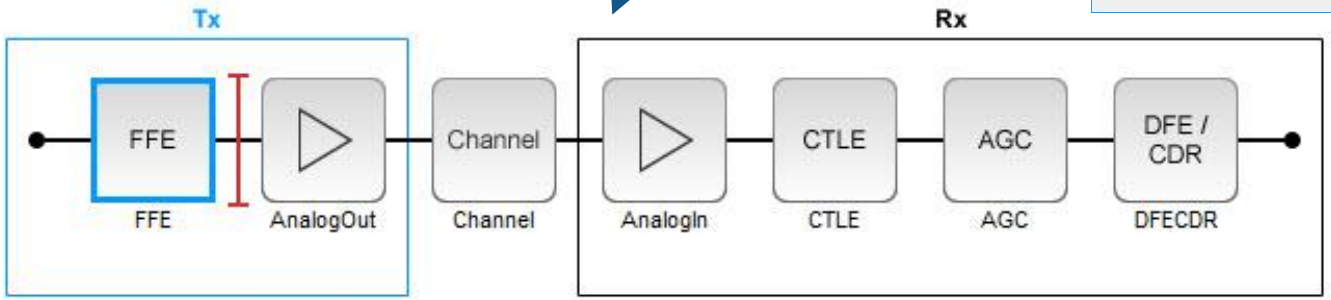




SerDes Designer app: No Need to be a SerDes Expert

- Design and analyze SerDes systems including transmitters and receivers with arbitrary configuration
- Use parameterized building blocks
- Perform statistical analysis: eye diagram, BER, bathtub, pulse response….



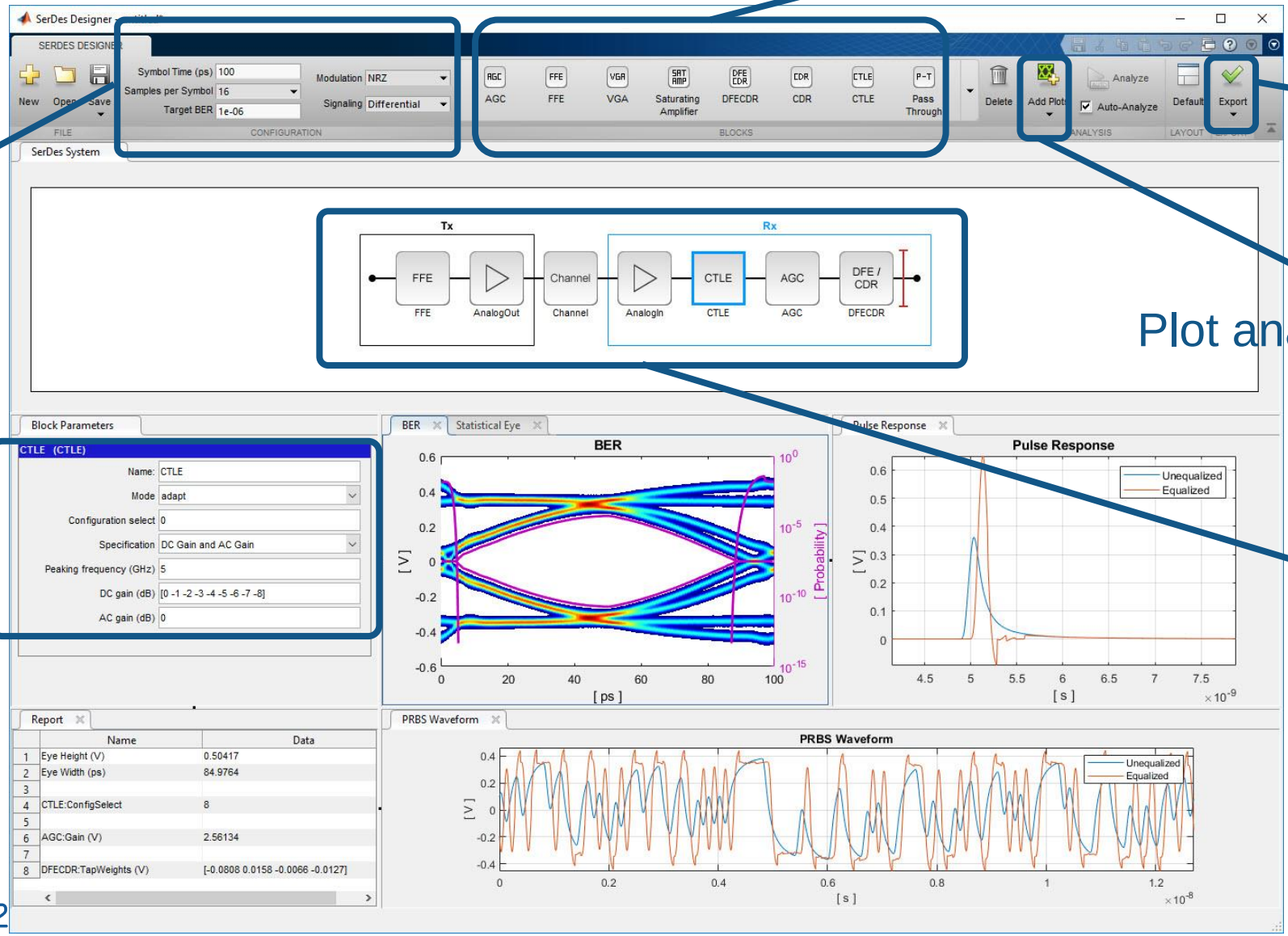


SerDes Design: Where to Start?

Add SerDes components

- Modulation
- Sample rate
- Signaling

Export to:
- MATLAB
- Simulink
- IBIS-AMI



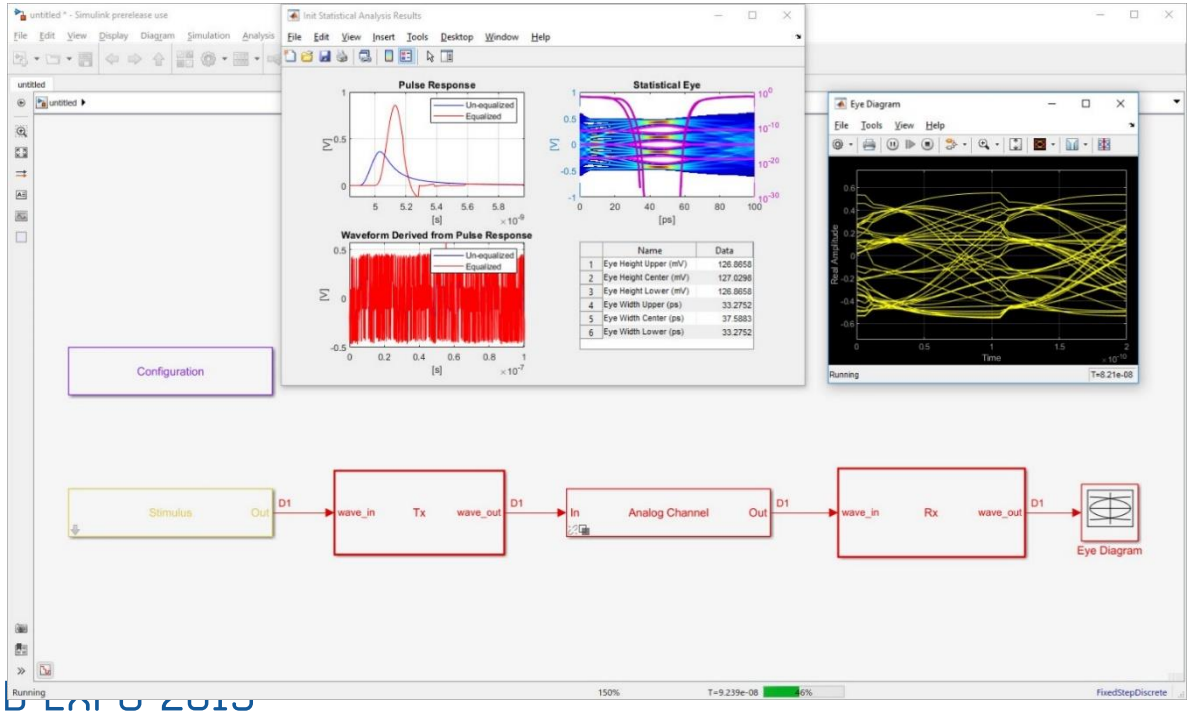
Component specifications

Plot analysis results

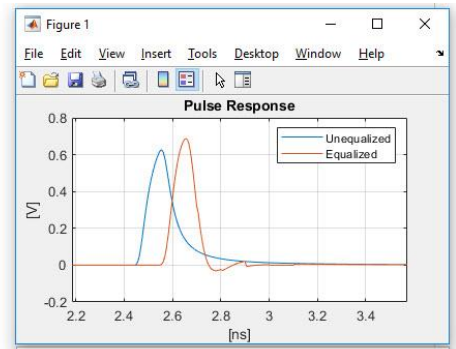
High-speed link

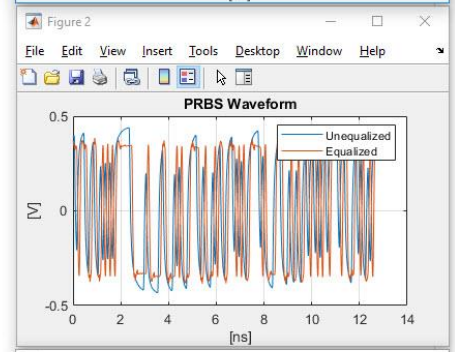
SerDes Top Down Design

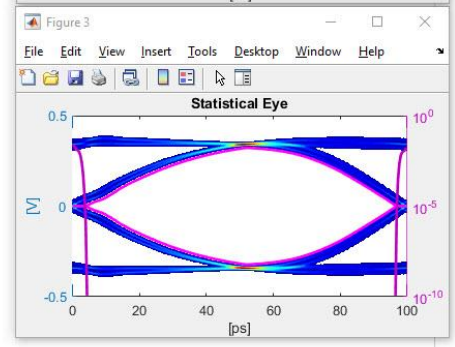
- Create a MATLAB script for automation and design space exploration
- Export to Simulink models for time-domain simulation
- Create dual IBIS-AMI models




```
% ------------------------------------
% MATLAB script to build SerDes System
% ------------------------------------

% Build cell array of Tx blocks:
txBlocks{1} = serdes.FFE;
txBlocks{1}.BlockName = 'FFE';
txBlocks{1}.Mode = 1;
txBlocks{1}.TapWeights = [0 1 0 0 0];
txBlocks{1}.Normalize = true;


% Build cell array of Rx blocks:
rxBlocks{1} = serdes.CTLE;
rxBlocks{1}.BlockName = 'CTLE';
rxBlocks{1}.Mode = 2;
rxBlocks{1}.ConfigSelect = 0;
rxBlocks{1}.Specification = 'DC Gain and Peaking Gain';
rxBlocks{1}.PeakingFrequency = 5000000000;
rxBlocks{1}.DCGain = [0 -1 -2 -3 -4 -5 -6 -7 -8];
rxBlocks{1}.PeakingGain = [0 1 2 3 4 5 6 7 8];

rxBlocks{2} = serdes.AGC;
rxBlocks{2}.BlockName = 'AGC';
rxBlocks{2}.Mode = 1;
rxBlocks{2}.TargetRMSVoltage = 0.3;

rxBlocks{3} = serdes.DFECDR;
rxBlocks{3}.BlockName = 'DFECDR';
rxBlocks{3}.Mode = 2;
rxBlocks{3}.TapWeights = [0 0 0 0];
rxBlocks{3}.MinimumTap = -1;
rxBlocks{3}.MaximumTap = 1;


% Build txModel:
txAnalogModel = AnalogModel( ...
    'R',50, ...
    'C',1.000000e-13);
tx = Transmitter( ...
'Blocks',txBlocks, ...
    'AnalogModel',txAnalogModel, ...
    'RiseTime',1.000000e-11, ...
    'VoltageSwingIdeal',1, ...
    'Name','TX');

% Build rxModel:
rxAnalogModel = AnalogModel( ...
    'R',50, ...
    'C',2.000000e-13);
rx = Receiver( ...
'Blocks',rxBlocks, ...
    'AnalogModel',rxAnalogModel, ...
    'Name','RX');

% Build ChannelData:
channel = ChannelData( ...
    'ChannelLossdB',8, ...
    'ChannelLossFreq',5000000000, ...
    'ChannelDifferentialImpedance',100);

% Build SerDes System:
SymbolTime = 1e-10;
SamplesPerSymbol = 16;
ModulationLevels = 2;
BERtarget = 1e-06;
```

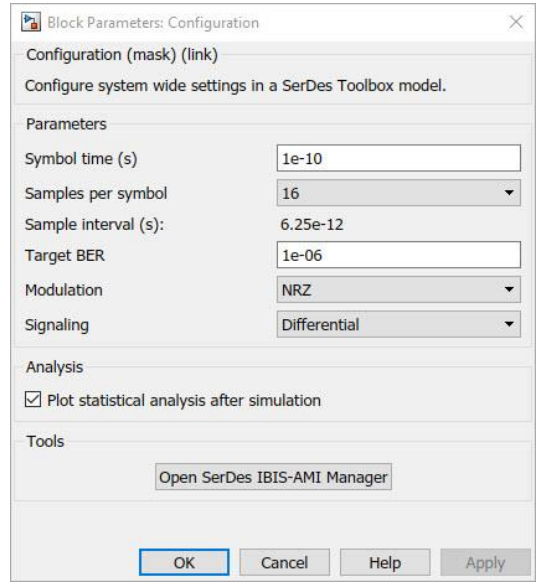


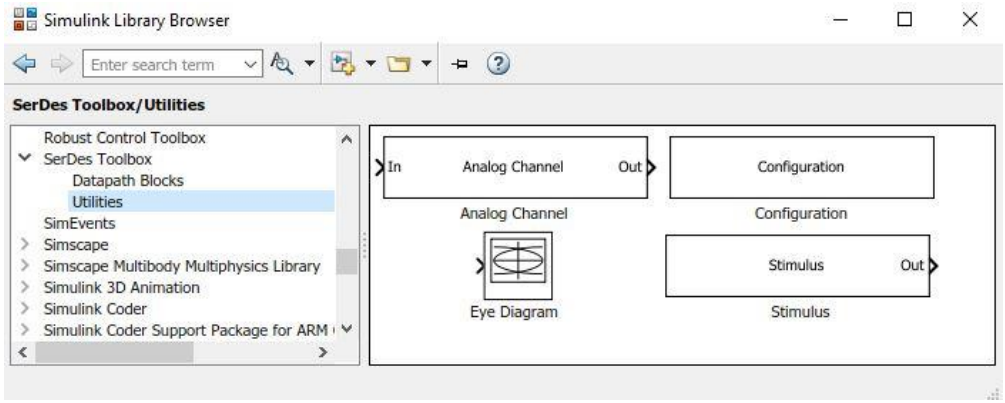


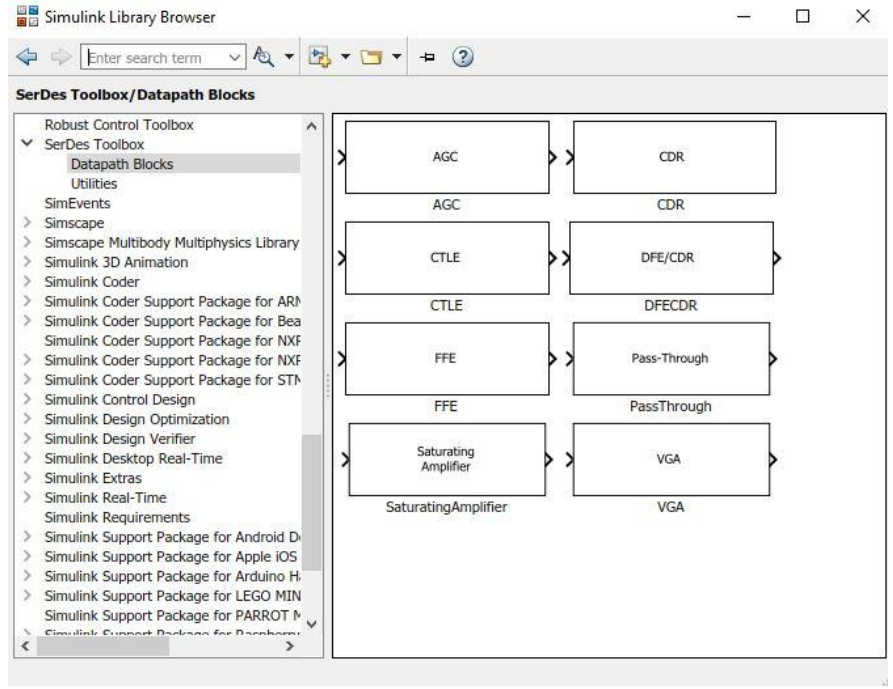


SerDes Toolbox: Simulink Models

- Develop adaptive equalizers using white-box models such as DFE, CTLE, AGC, and CDR
- Use parametrized blocks and algorithms for single-ended and differential signals
- Generate PRBS and custom stimulus patterns supporting PAM4 and NRZ modulation

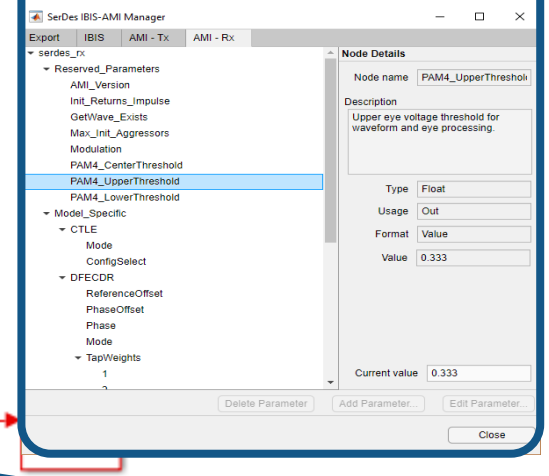


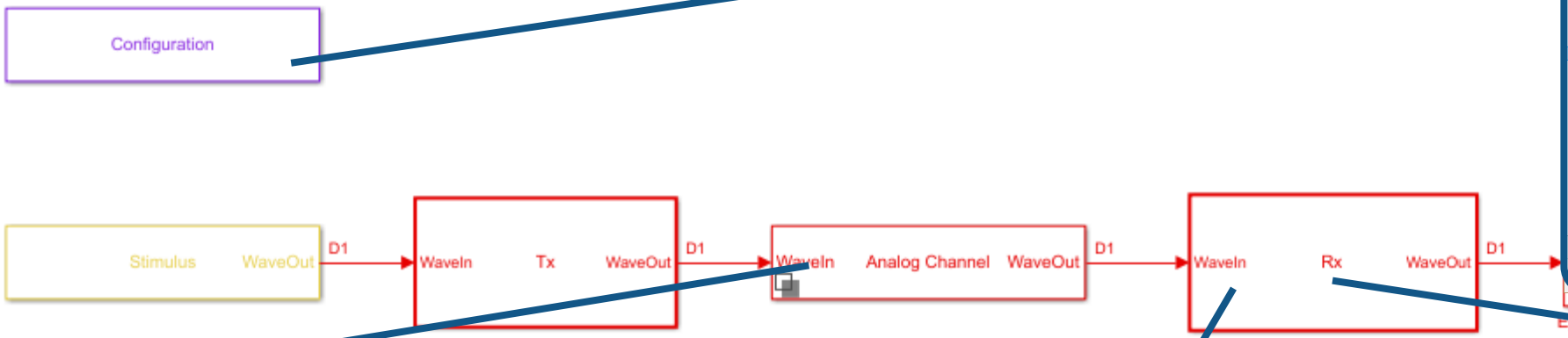




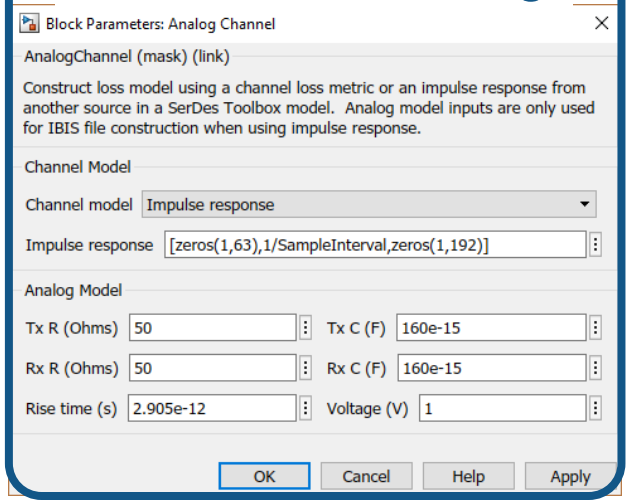
SerDes Simulation and Architecture Exploration

Customize AMI parameters

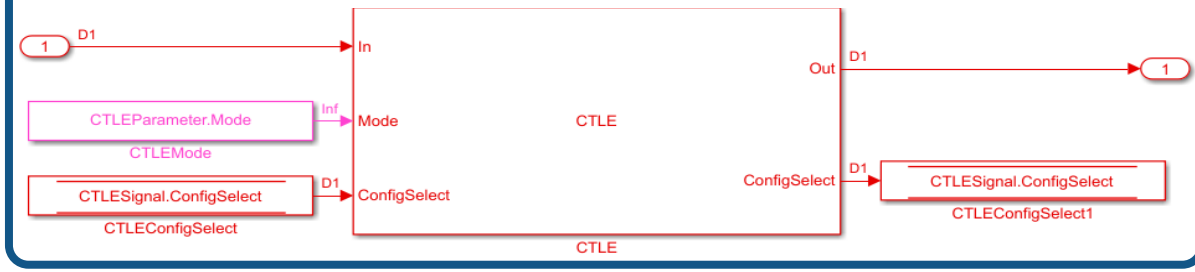




Channel modeling



Simulate and customize adaptive equalizers



White-box (customizable) models

```
%Note that step or time domain adaptation of the CTLE must be
%done in an exterior block.

ixed(obj)
 = obj.ConfigSelect;

.privConfigInitialFlag
j.privConfigInitial = obj.ConfigSelect;
j.privConfigInitialFlag = false;

j.privConfigInitial = obj.ConfigSelect;

 = obj.privConfigInitial;

bj.FilterCoefficients.np(Config+1)+1;
bj.FilterCoefficients.nz(Config+1)+1;
```

Agenda

- Motivation
- System Level Design of Mixed-Signal and SerDes Blocks
- Linking System Design to Circuit Design
- Conclusion

Cadence Integration

- Co-simulation

Debugging
Validation of behavioral models

→ Analog: between Simulink and AMS Designer

→ Digital: between MATLAB/Simulink and Incisive

HDL Verifier

- Code-generation

IC Implementation
Testbench generation

→ Synthesizable: VHDL or Verilog

HDL Coder

→ Behavioral: real-number SystemVerilog (DPI-C)

HDL Verifier

- Data post-processing

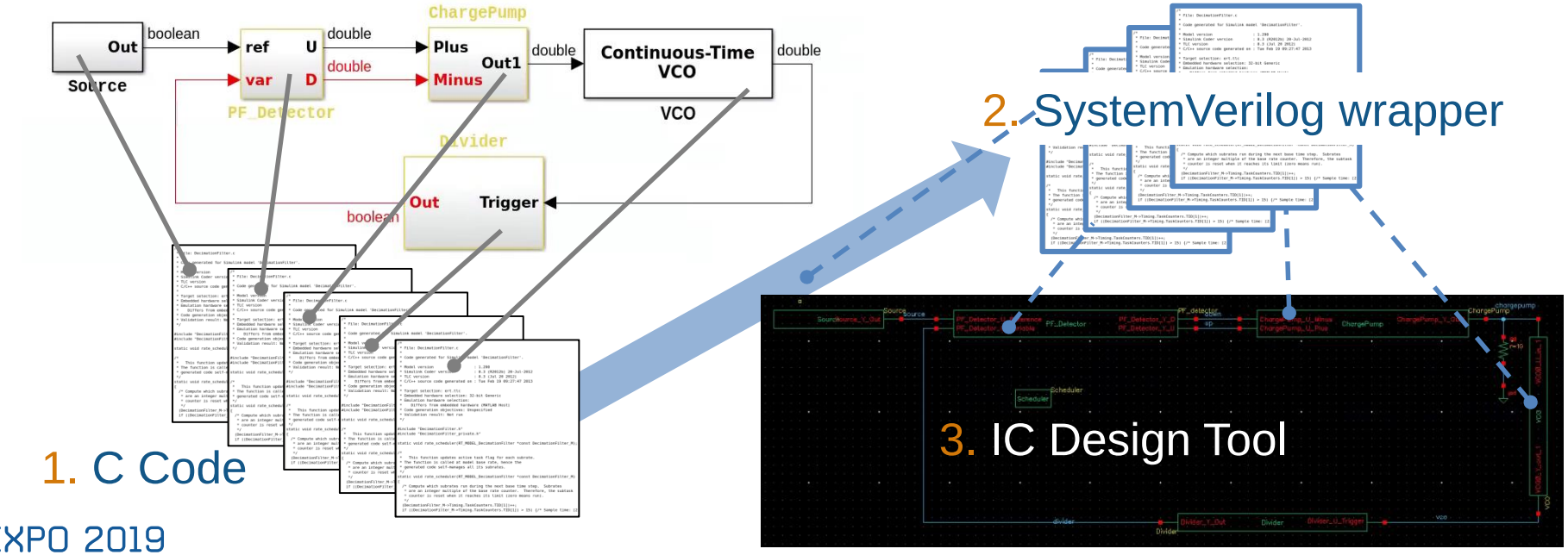
Result analysis (statistical)
Model fitting

→ File: import data files in MATLAB for post-processing

→ ADE Explorer & Assembler: Invoke MATLAB functions

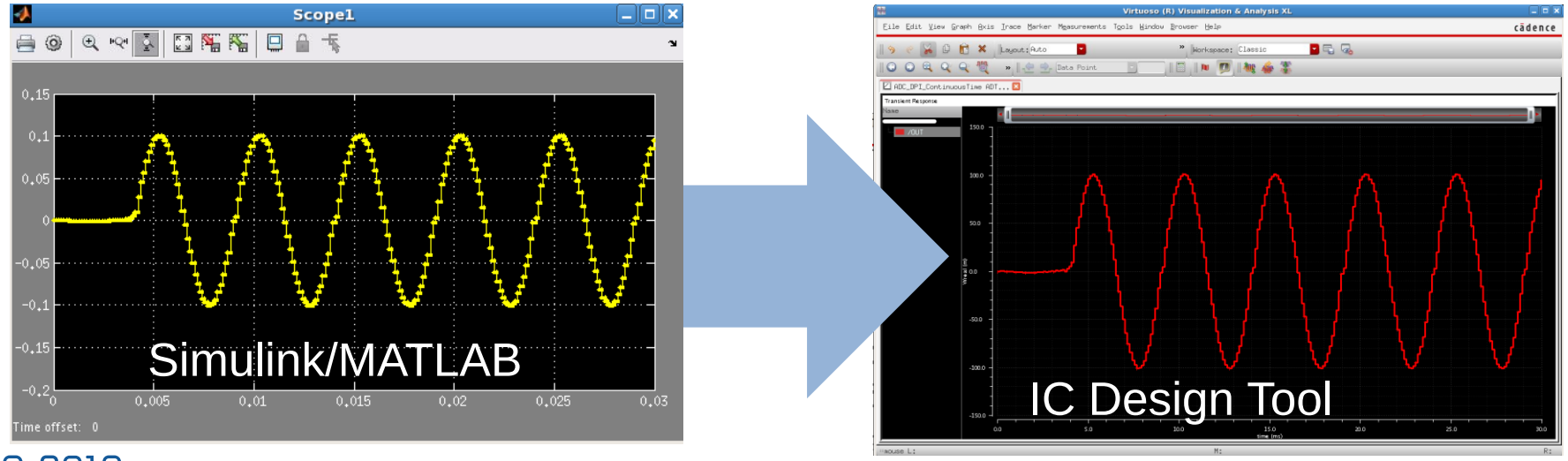
Generate SystemVerilog Module for EDA Tools

1. Make the Simulink model / MATLAB code compliant with C code generation
2. Generate C code
3. Automatically wrap the C code using the DPI-C interface (UVM Compliant Models)
4. Import, build and simulate an equivalent behavioral SystemVerilog model in your IC design tool



Benefits of C Code Generation and DPI-C Export

- Fast simulation using the native SystemVerilog API
- IC design tool independent
- Uses mature C code generation technology to provide real-number models
- Most suitable for testbench generation and IC verification (regression tests)
- Supports discrete and continuous time signals
- Supports code generation from MATLAB and Simulink



Endorsed Workflow by STARC

A Next-Generation Workflow for System-Level Design of Mixed-Signal Integrated Circuits

By Kunihiko Tsuboi and Nobutaka Okumura, STARC

In the competitive world of mixed-signal design, project delays are deadly for profitability. Design cycles are short, and the market moves very quickly, giving a competitive edge to any company that manages to accelerate its design process. At the Semiconductor Technology Academic Research Center (STARC), we have been given the mission of finding a way to cut design time in half and eliminate costly respins for our supporting companies. We have achieved this goal with our new system-level design flow.

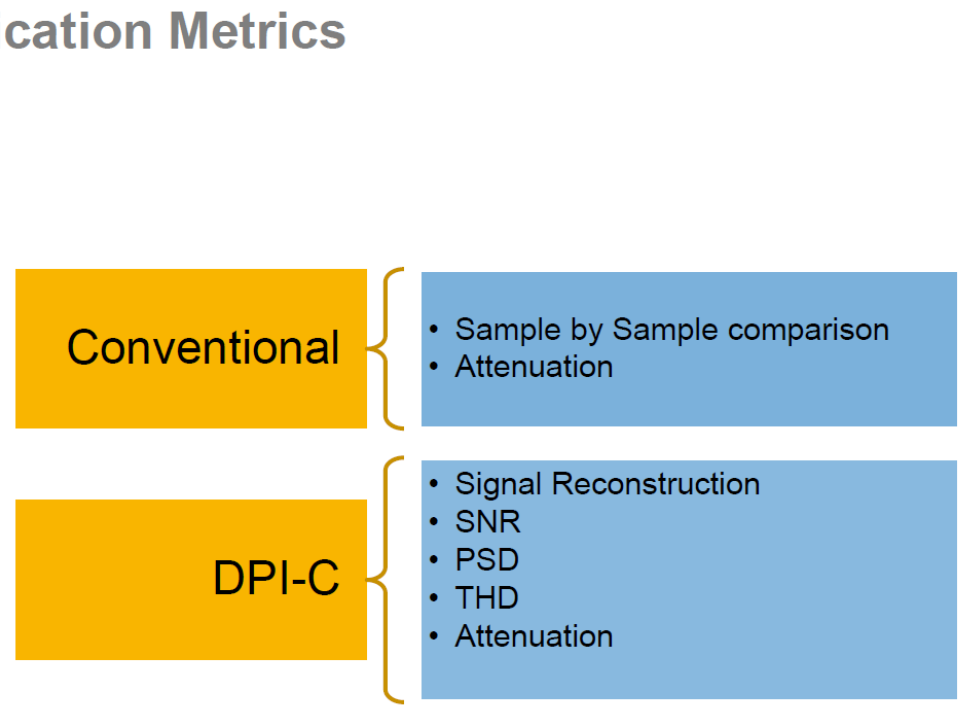
Our system-level design flow (which we call STARCAD-AMS) starts with rapid and extensive behavioral modeling in Simulink®. Once we have a design that works at the system level, we generate C code from our Simulink models and import it into Cadence® Virtuoso®, where it is simulated using AMS Designer. We use our C code to verify the correctness of our circuit-level designs. We have benchmarked our STARCAD-AMS flow using a sigma-delta analog-to-digital converter (ADC) design. Our results show that design time is cut in half.

(*)

(*) http://www.mathworks.com/company/newsletters/articles/a-next-generation-workflow-for-system-level-design-of-mixed-signal-integrated-circuits.html

DPI-C Customer Successes – NXP Semiconductors, India

- Leveraged DPI-C to perform advanced verification using MATLAB functions
- Moved to a more efficient verification
- Sign off tool leveraged due to UVM com of DPI-C model generation



(*)

(*) *Verifying hardware implementation of automotive radar signal processing with MATLAB, Sainath K, Shashank V*

Agenda

- Motivation
- System Level Design of Mixed-Signal and SerDes Blocks
- Linking System Design to Circuit Design
- Conclusion and Next Steps

Conclusion

- Rapid system level design of mixed-signal IC’s using architectures or from building blocks
- Build SerDes systems like never before using SerDes designer app
- Perform advanced analysis using measurement test benches and visualization capabilities
- Link with IC design tools to model implementation impairments and reuse testbenches
- Deliver DPI-C compliant SV models and IBIS-AMI to customers (internal and external)

Call to Action – MathWorks Resources

- **Mixed Signal and SerDes Product Pages**
- **User stories**

Allegro MicroSystems Reduces Anti-Lock Braking System Sensor Development Time

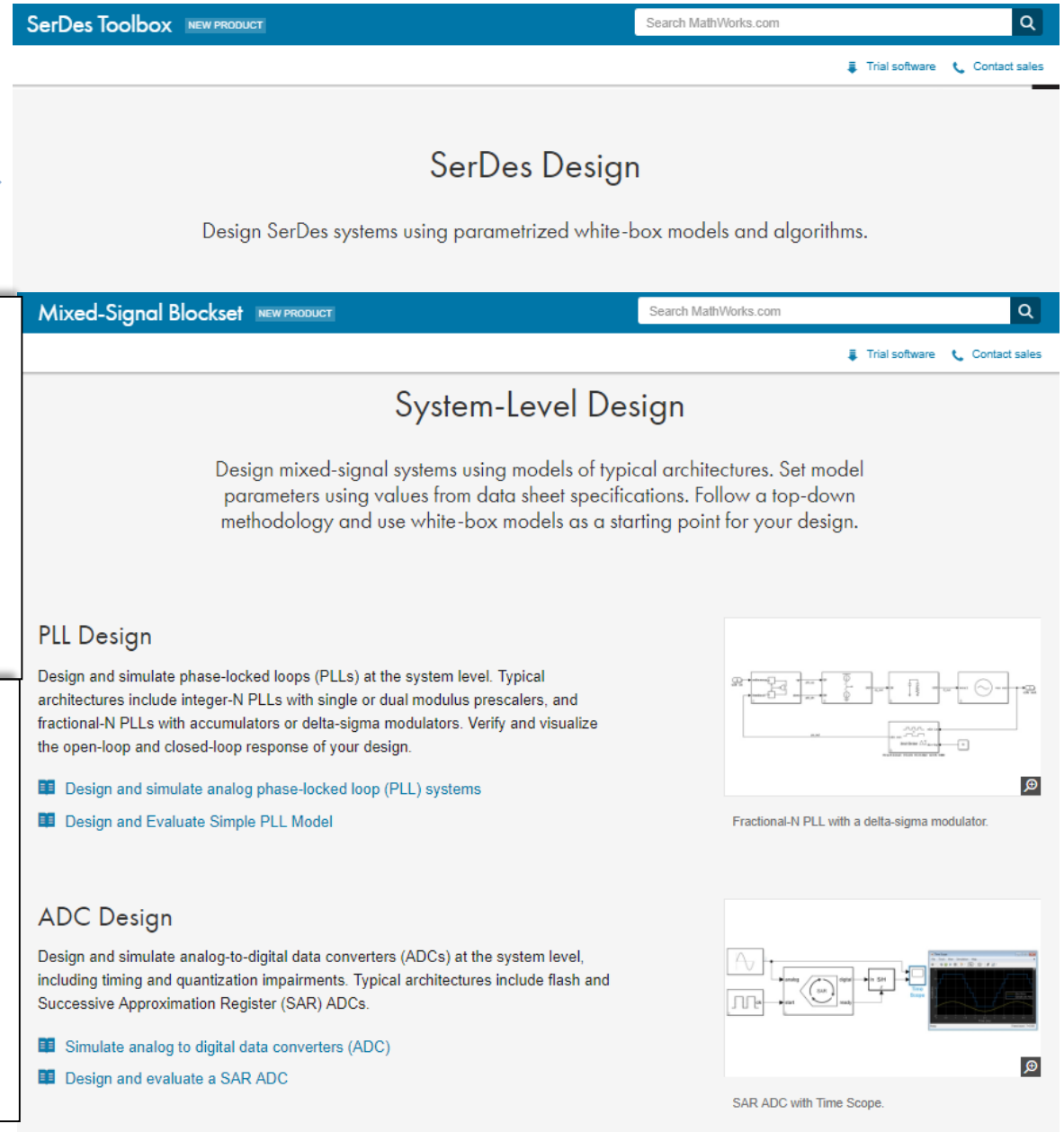
"Using MathWorks tools, we identified the best algorithm choice. Because the model ran much faster than our circuit simulator, we caught implementation errors much quicker and shortened our time to market."

— Cory Voisine, Allegro MicroSystems

Fujitsu Develops and Tests State-of-the-Art 40 Gbps Optical Transponder

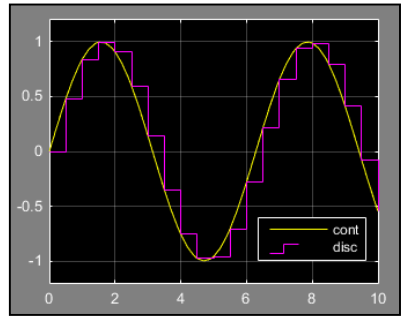
"By including circuit-level simulation results in our Simulink models we can simulate millions of cycles with the accuracy needed to account for noise and other transient effects. Simulink is the only tool fast enough for our jitter-tolerance simulations."

— William Walker, Fujitsu Laboratories of America

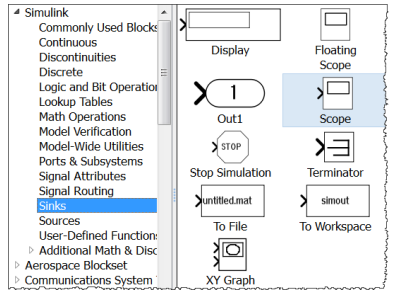


MathWorks® | Training Services

System Design and Modeling with Simulink

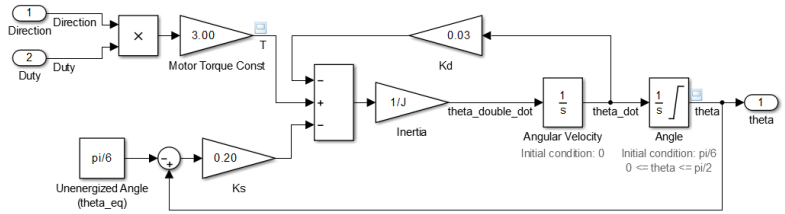


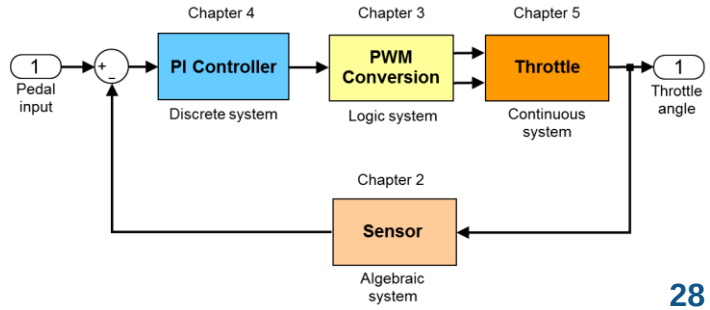
This three-day course uses basic modeling techniques and tools to demonstrate how to develop Simulink block diagrams for signal processing applications.



Topics include:

- Modeling single-channel and multi-channel discrete dynamic systems
- Implementing sample-based and frame-based processing
- Modeling mixed-signal (hybrid) systems
- Developing custom blocks and libraries
- Performing spectral analysis with Simulink
- Integrating filter designs into Simulink
- Modeling multirate systems
- Incorporating external code





Please provide feedback for this block of sessions

- Scan this QR Code or log onto link below (link also sent to your phone and email)
- http://bit.ly/expo19-feedback
- Enter the registration id number displayed on your badge
- Provide feedback for this session

Email: Aniruddha.Dayalu@mathworks.in

LinkedIn: https://www.linkedin.com/in/aniruddhadayalu