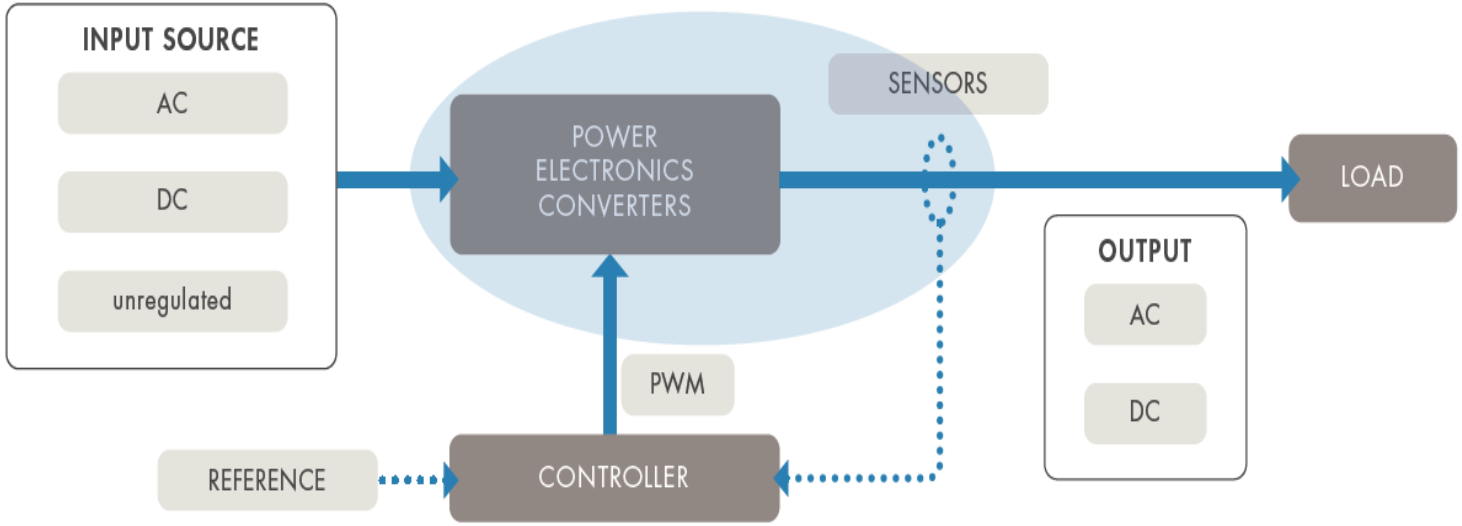
MATLAB EXPO 2019

Design Efficient DC-to-DC Power Converters

Vasco Lenzi, Application Engineer

Sebastien Dupertuis, Senior Application Engineer

Power Electronic Systems



Power Electronics Applications

Electric vehicles and charging stations

Rail

Renewable energy

Lighting

Our Project Today

DC/DC LED Developer's Kit

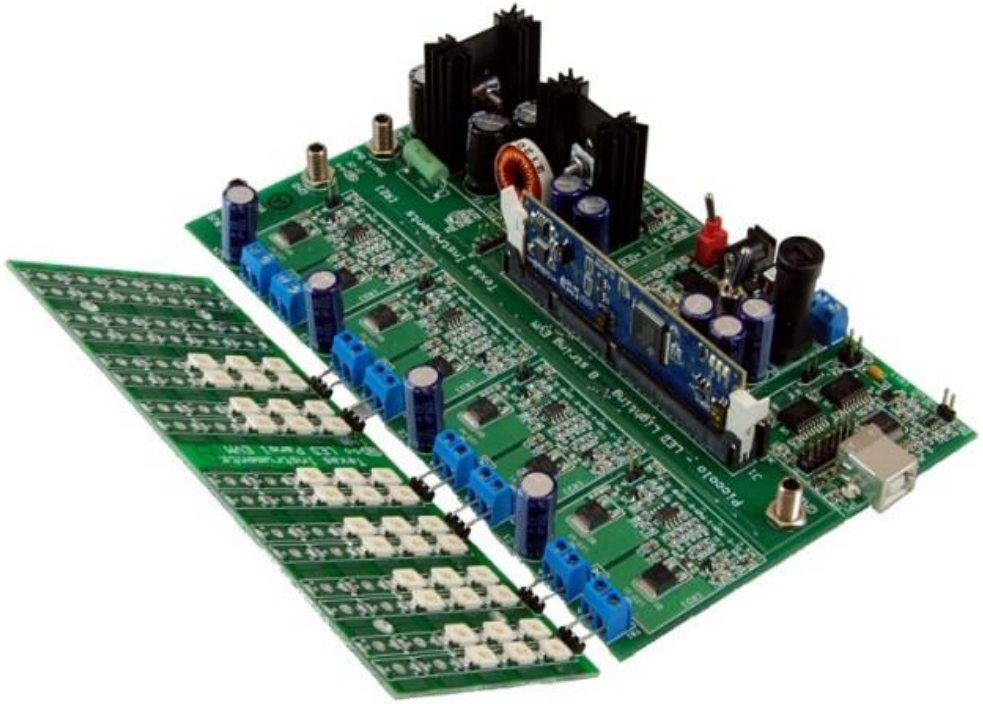


Fig 1: TMDSDCDCLEDKIT

LED Head Lamp

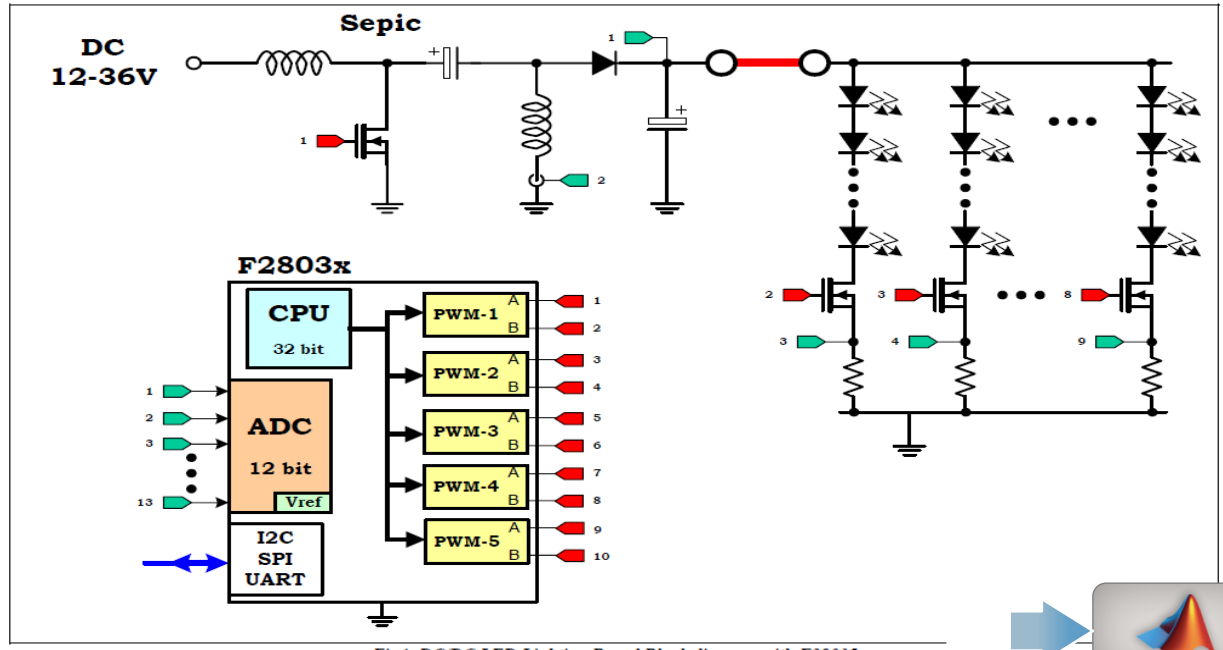
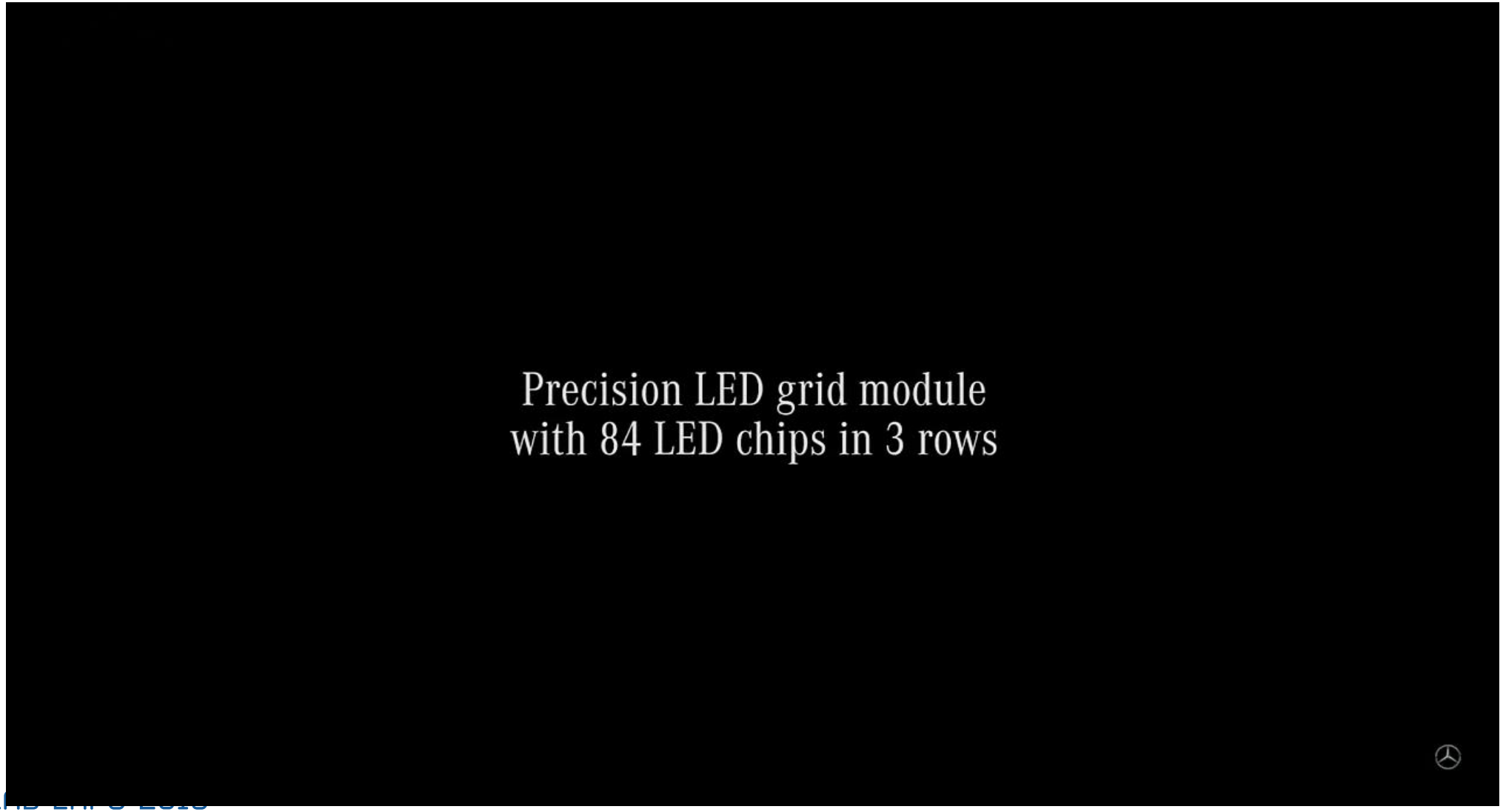


Fig4: DC/DC LED Lighting Board Block diagram with F28035

Mercedes headlamp technology – video

(Ref : Mercedes Benz Website)

https://www.mercedes-benz.com/en/mercedes-benz/innovation/multibeam-led-headlamps-in-the-new-e-class-video/



Why Simulink for Power Electronics Control?

- Extensive library of sources and loads
  - PV arrays, batteries, motors
- Broad range of power electronics models
  - Average value, fast ideal switching, physics-based
- Advanced control design capabilities
  - Auto-tuning in time & frequency domains for single and multiple loops
- Generation of readable, compact and fast code from models
  - C for microprocessors, HDL for FPGAs

Customers routinely report 50% faster time to market

ABB Accelerates the Delivery of Large-Scale, Grid-Connected Inverter Products with Model-Based Design

Challenge

Accelerate the design and delivery of large, grid-connected power inverter products

Solution

Use Model-Based Design to model, simulate, and generate control software for modular, scalable power electronic building blocks

Results

- Prototypes delivered in two weeks, not three months
- Defect-free, optimized code generated
- Potential damage to test equipment mitigated

Link to user story

A cabinet of Power Electronic Building Blocks (PEBBs).

"Simulink and Embedded Coder enabled us to open the door to new markets. With increased productivity from extensive simulation and efficient code generation, we have confidence in our ability to produce the systems that larger customers are asking for in the time frames they want."

- Dr. Robert Turner, ABB

Challenges for Power Electronics Engineer

- Examine effect of source and load on power converter operation
- Test embedded software for complete range of operation and fault conditions
- Design and implement digital controls in SPICE simulator tools only
- Identify errors during software-hardware integration testing
- Qualify designs to meet regulatory and industry standards

Power Converter Control Design Workflow Tasks

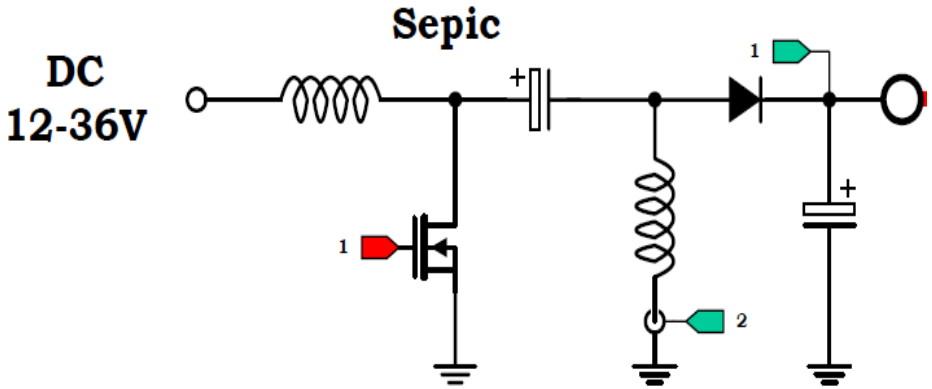
- Size inductor, capacitor and understand the behaviour in continuous and discontinuous mode
- Determine power losses and the thermal behaviour of the converter
- Design control algorithm based on time/frequency domain specification
- Implement power electronic controls on an embedded platform

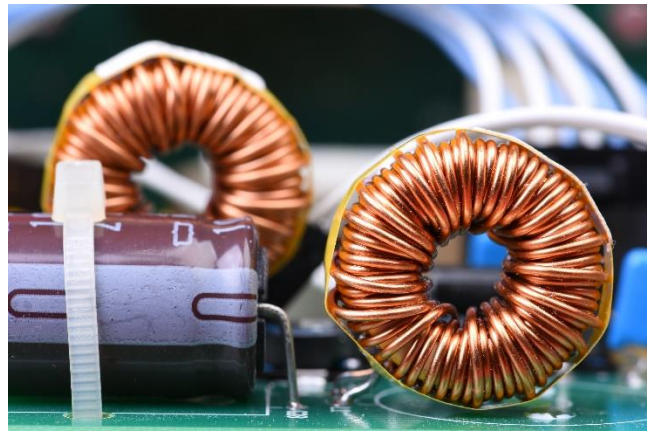
Let’s get to it!

Power Converter Control Design Workflow Tasks

- **Size inductor, capacitor and understand the behaviour in continuous and discontinuous mode**
- Determine power losses and the thermal behaviour of the converter
- Design control algorithm based on time/frequency domain specification
- Implement power electronic controls on an embedded platform

Recap: Size Inductor, Capacitor and Understand the Behaviour in Continuous and Discontinuous mode.





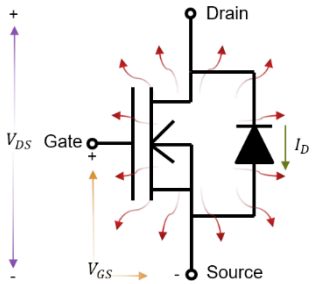
What we did:

- Use simulation to design DC to DC converters
- Optimize component sizing using simulation driven analysis

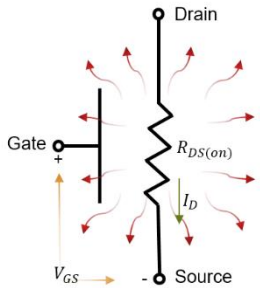
Power Converter Control Design Workflow Tasks

- Size inductor, capacitor and understand the behaviour in continuous and discontinuous mode
- **Determine power losses and the thermal behaviour of the converter**
- Design control algorithm based on time/frequency domain specification
- Implement power electronic controls on an embedded platform

Recap: Determine Power Losses and Simulate Thermal Behaviour of the Converter.

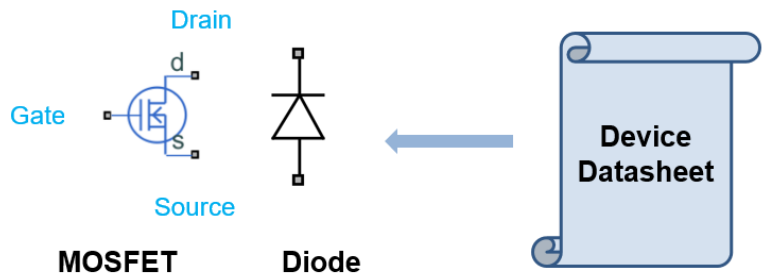


Switching loss



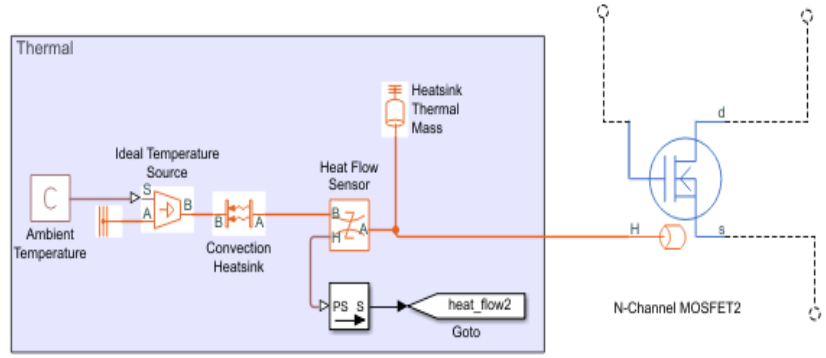
Conduction loss

Device Blocks



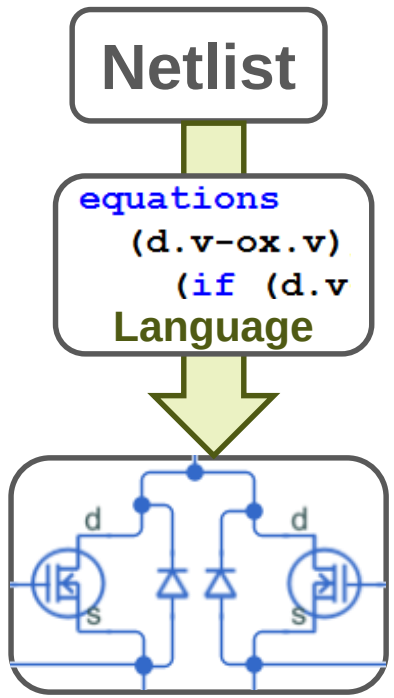
What we did

- Use semiconductor blocks from Simscape Electrical to model the non-linear switching behavior of SEPIC converter
- Leverage the multi-domain simulation capability of Simscape in understanding the thermal dynamics



New: Convert SPICE models into Simscape components

- Incorporate manufacturer specific behavior into simulation
- Easily parameterize the model
- Combine existing electronic models with other domains (such as thermal), control algorithms, signal processing, all in a single environment



testMosfetNetlist.txt

```
.FUNC  Idiode(Usd,Tj,Iss)  {exp(min(l
.FUNC  Idiod(Usd,Tj)       {a*Idiode(

.FUNC  Pr(Vss0,Vssp)   {Vss0*Vss0/Rm+

.FUNC J1(d,g,T,da,s,x)  {a*(s*(exp(mi

.FUNC QCds(x)  {Cds3*min(x,x1)+Cds0*m
```

subcircuit2ssc

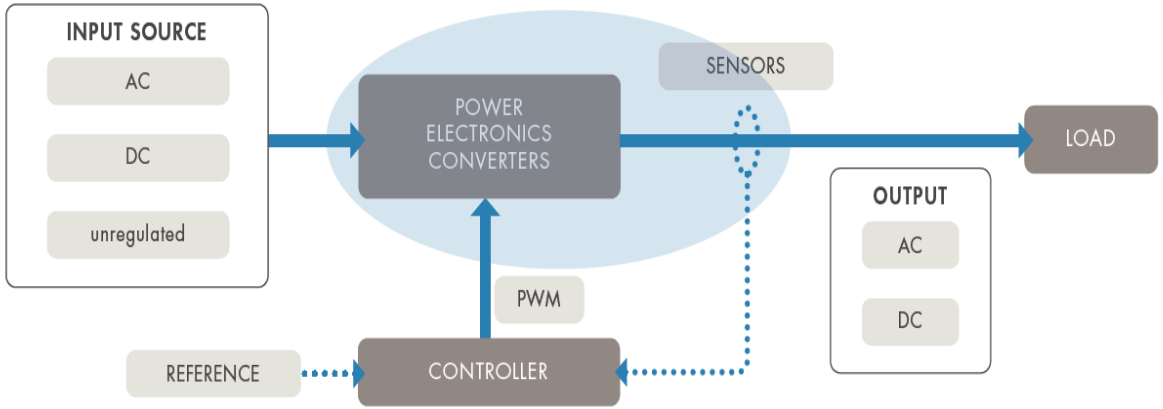
testMosfetNetlist.txt | ipt015n10n5_l1.ssc

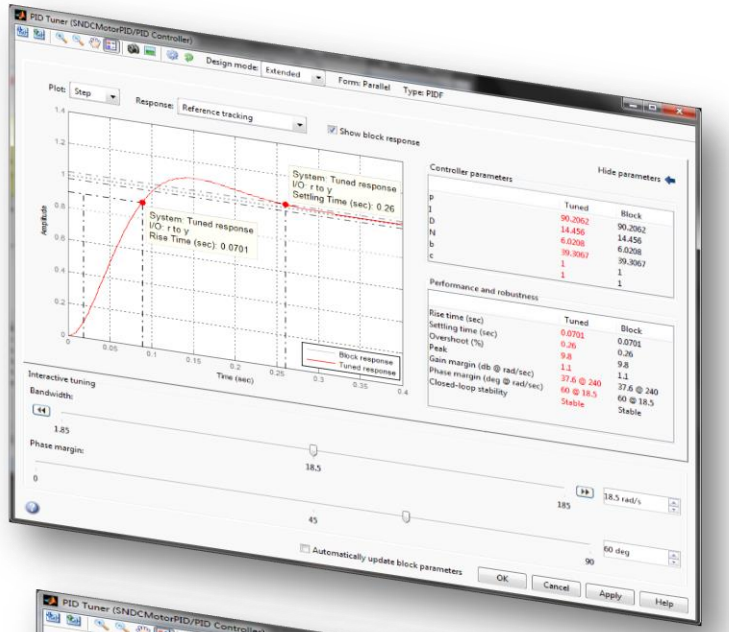
```
components(ExternalAccess=observe)
    X1 = test.s5_100_f_var(a=act,rsp=r
        rs=rs,rp=rd,dc=dc,rm=rm);
    RG = elec.passive.instrumented_res
    LG = foundation.electrical.element
        i_L.priority=priority.none);
    RSA = elec.passive.instrumented_re
    LS = foundation.electrical.element
```

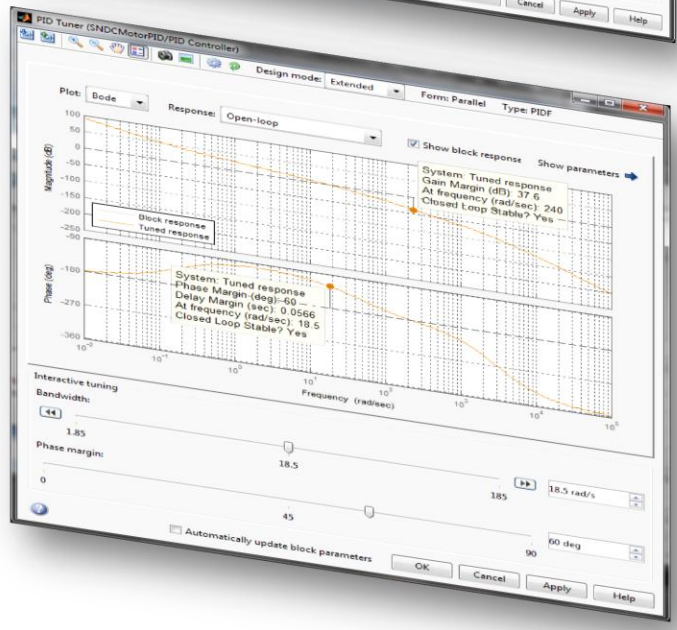
Power Converter Control Design Workflow Tasks

- Size inductor, capacitor and understand the behaviour in continuous and discontinuous mode
- Determine power losses and the thermal behaviour of the converter
- **Design control algorithm based on time/frequency domain specification**
- Implement power electronic controls on an embedded platform

Recap: Design Control Algorithm Based on Time/Frequency Domain Specifications







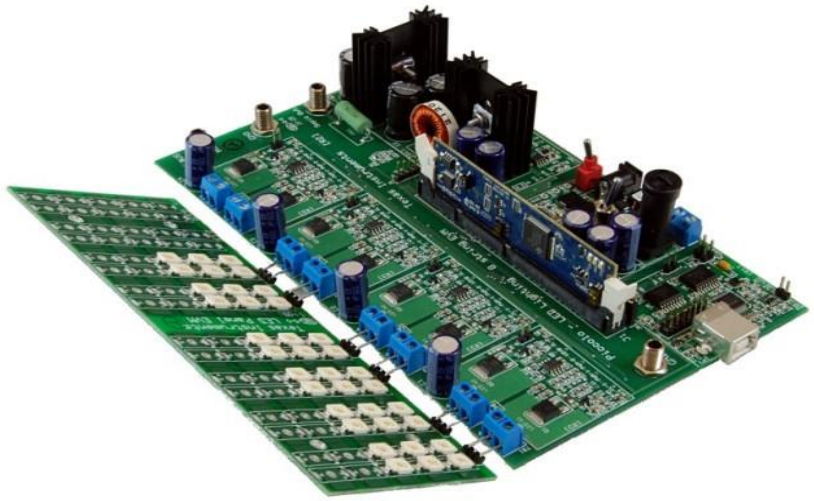
What we did

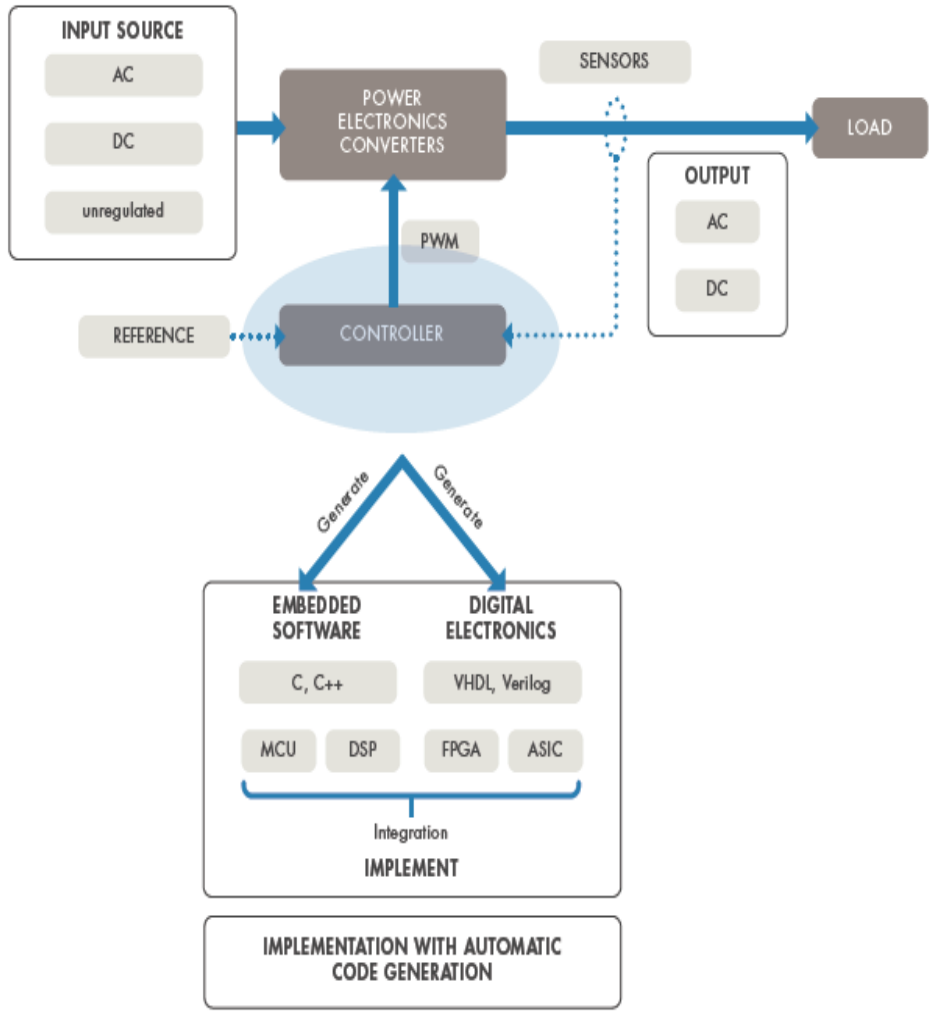
- Identify plant model from input output simulation data
- Use auto tuning algorithms to tune the control gains

Power Converter Control Design Workflow Tasks

- Size inductor, capacitor and understand the behaviour in continuous and discontinuous mode
- Determine power losses and the thermal behaviour of the converter
- Design control algorithm based on time/frequency domain specification
- **Implement power electronic controls on an embedded platform**

Implement Power Electronics Control on an Embedded Platform





What do we want to do:

- Verify the controller for various test cases on real HW
- Generate code from MATLAB and Simulink models optimized for embedded platforms (DSP and FPGA)
- System-level test using Hardware-in-the-Loop

Challenges of a Software Engineer

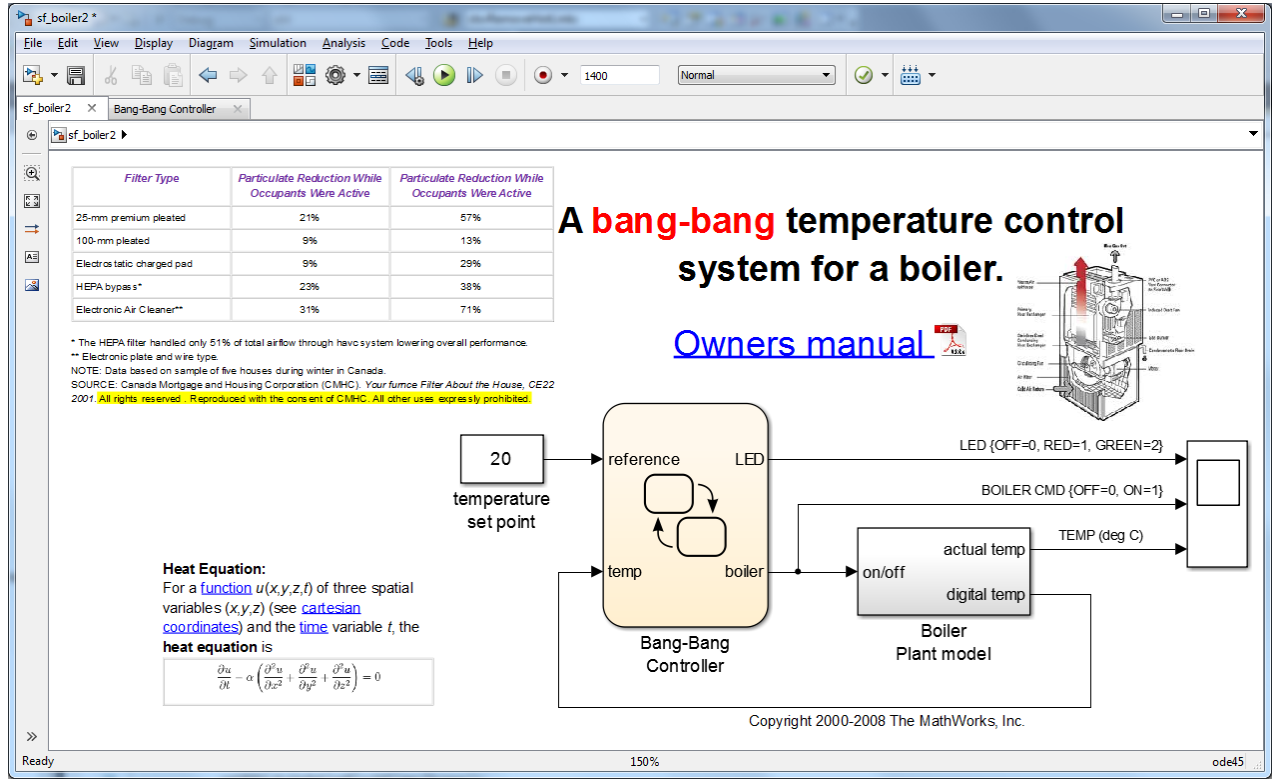
- Collaborate in multidisciplinary teams
- Create working code
- Achieve required efficiency
- Respect project timeline

Integrated Development Environment

Rich Environment Editor: Simulink Editor, Rich Annotations

Add images, tables, and bulleted lists to your model

- Rich text with:
  - Bold, italic, font name, font size, highlight etc.
  - Bulleted and enumerated lists
  - Tables
  - Hyperlinks (proper URLs)
  - Images
- Image annotations with callbacks
- Author your content directly in the editor or paste clipboard content copied for Word, Excel, HTML, etc.



Integrated Development Environment

Simulink Projects Becomes MATLAB and Simulink Projects

Project - TIC2000_DCDC_LED

Views

- Files
- Dependency Analysis

Labels

- Classification
  - Artifact
  - Convenience
  - Derived
  - Design
  - None
  - Other
  - Test

All | Project (70) | Modified (2) Layout: Tree

Name	Status	SVN	Revision	Classification
Cache	✓		963	
Code	✓			
Configurations	✓			
Data	✓		963	
Documentation	✓		963	
Images	✓			
Libraries	✓		963	
PWM_SFCN	✓		963	
PWM_sfcn.c	✓		963	Design
PWM_sfcn.mexw64	✓		963	Derived
Models	✓		963	
Scripts	✓		963	
cleanup.m	✓		963	Design
elec_gate_charge_char.m	✓		963	Design
elec_mosfet_plot_transfer_c...	✓		963	Design
elec_mosfet_steadystate_ch...	✓		963	Design
operatingPointSimscape.m	✓		963	Design
processingLogData.m	✓		963	Design
retrieving_results_from_figu...	✓		963	Design
Sepic_Demo_Script.m	✓		963	Design
setup.m	✓		963	Design
Sessions	✓		963	
Work	✓			

Models (Folder) 0 labels

Integrated Development Environment

Manage projects

git

SVN

Integrate with
Source Control

+ Your Company Library v1.3
+ Your Project Libraries

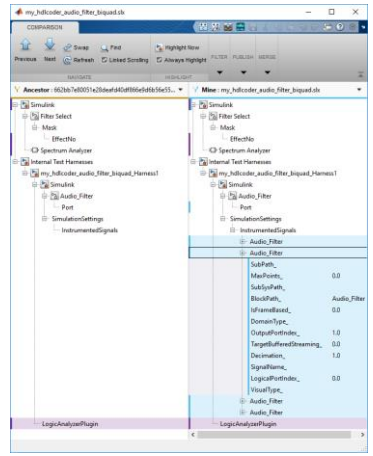
Standardize team environment

Execute dependency
analysis

Projects

Automate tasks

Build & share best-
practices

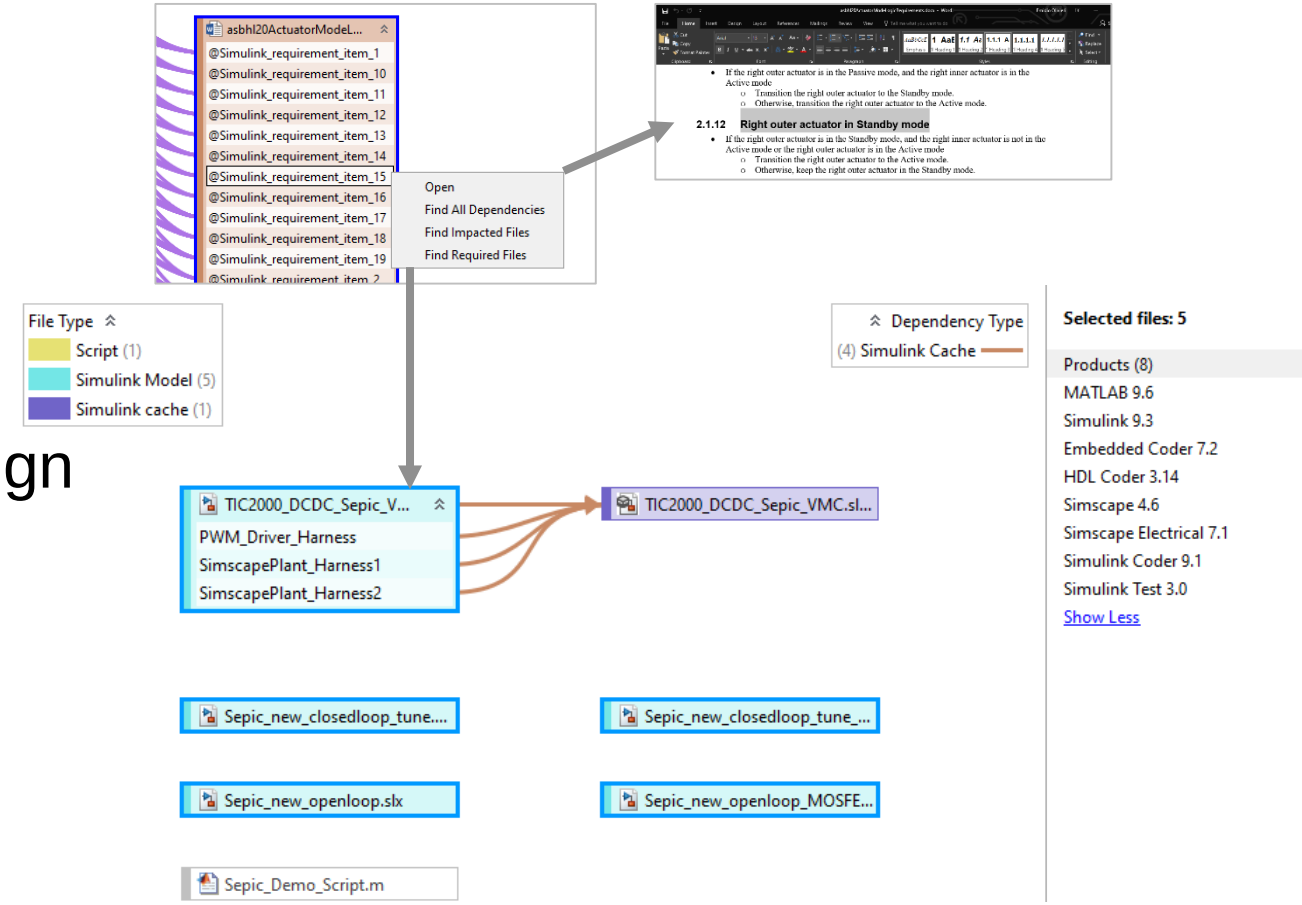


Support
for change
management

Integrated Development Environment

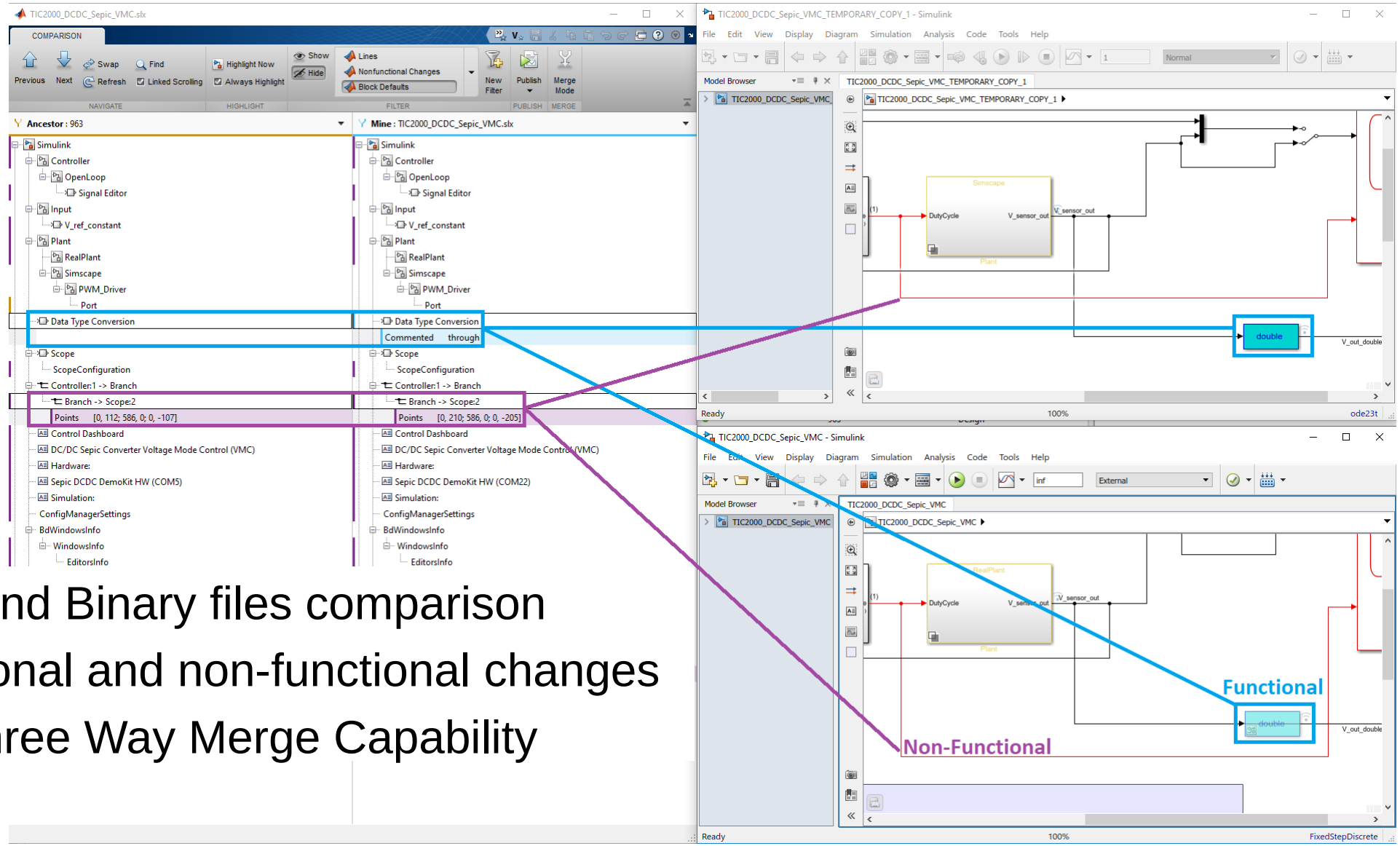
Dependency Analysis

- Visualize file dependencies, associations, and missing files
- Visualize the impact of changes to the files within your project
- Understand the impact of requirements changes on your design
- Store and access previous dependency analysis results



Integrated Development Environment

Graphical Model Comparison

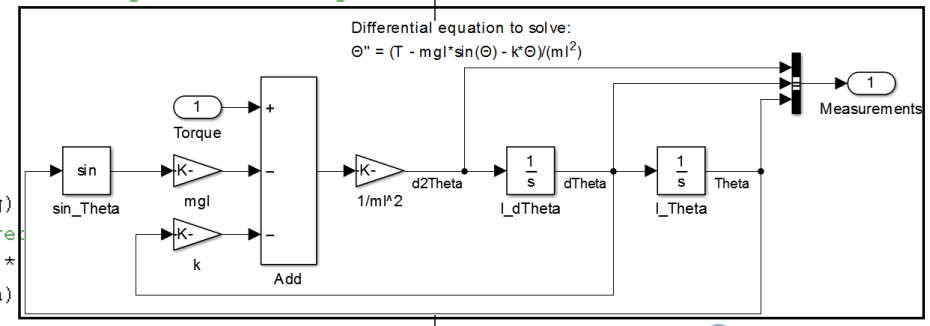


- Graphical and Binary files comparison
- View functional and non-functional changes
- Two and Three Way Merge Capability

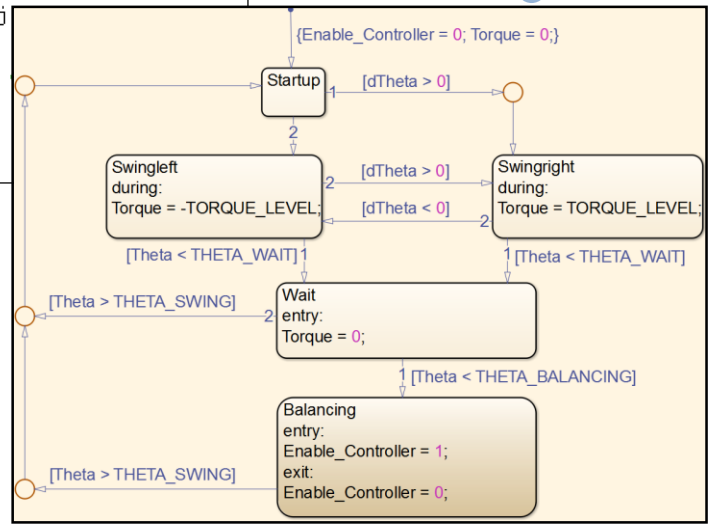
Replace Hand Coding with Code Generation

```
function [symbols, weights] = gainctrl(rxsig, train)
% 1-tap adaptive equalizer using LMS or RLS algorithm

% Equalizer settings
lambda = 0.99;
Delta = 0.1+0i;
weights = 0+0i;

for n = 1:length(rxsig)
    u = rxsig(n);  % rec
    y = conj(weights) *
    if n<=length(train)
        d = train(n);
    else
        d = detect(real(y)) + 1j*detect(imag(y));
    end
    % Single-tap RLS
    Delta = 1/(lambda/Delta + u*conj
    G = Delta * u;
    e = d - y;  % symbol estimation
    weights = weights + G*conj(e);
    symbols(n) = y;
end
```

MATLAB



Simulink



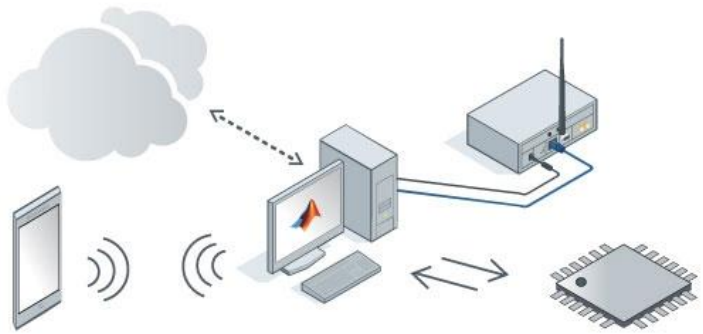
Stateflow

Optimizations

Unified representation

- C Code
- C++ Code
- HDL Code
- PLC Code
- CUDA Code

Hardware support



Browse Support for:

Lab Instruments

Examples: Visa Support from Instrument Control Toolbox

Data Acquisition Systems

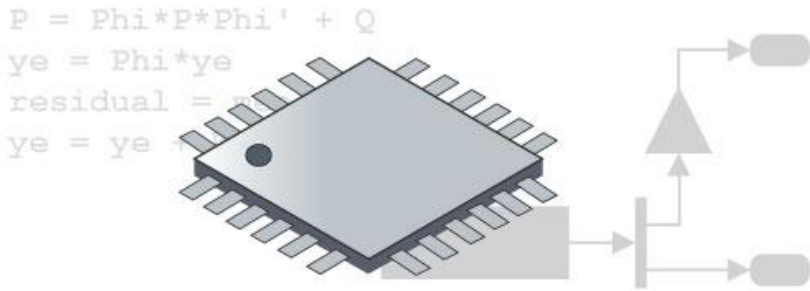
Examples: NI-DAQmx Support from Data Acquisition Toolbox

Image and Video Acquisition and Camera Applications

Examples: Web Cam I/O with MATLAB and Simulink, Microsoft Kinect Support from MATLAB and Simulink

Streaming Audio with MATLAB and Simulink

FPGA-in-the-Loop Platforms



Browse Support for:

Programming Microprocessors with C/C++

Examples: Low Cost Simulink Targets, ARM Cortex Targets, Embedded Coder Hardware Support Packages

Programming FPGAs

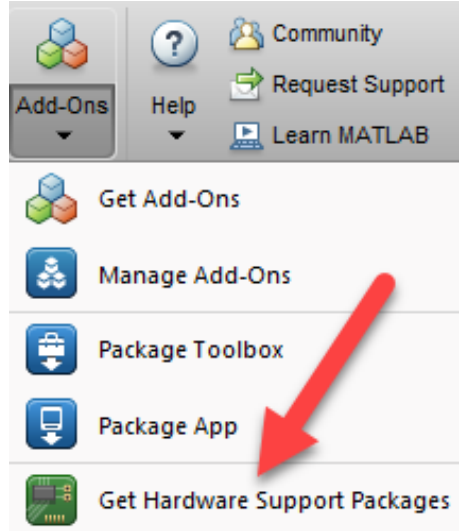
Examples: Programming SDR Algorithms with HDL, Motor Control with Zynq

Programming PLCs

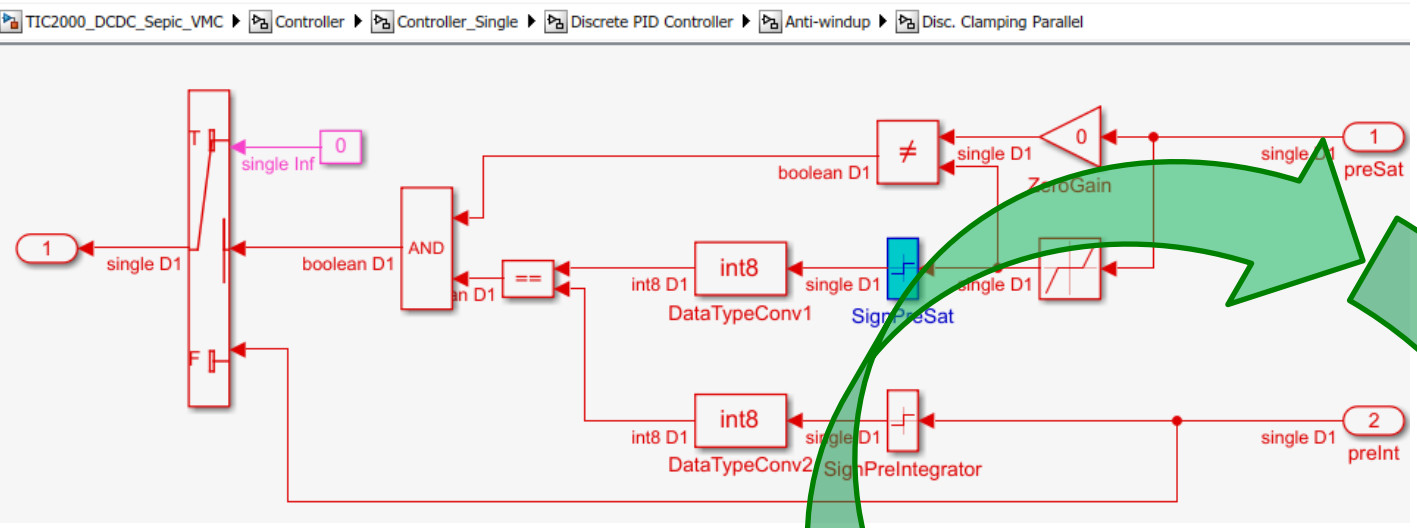
Real-Time Simulation, Testing, and Hardware-in-the-Loop

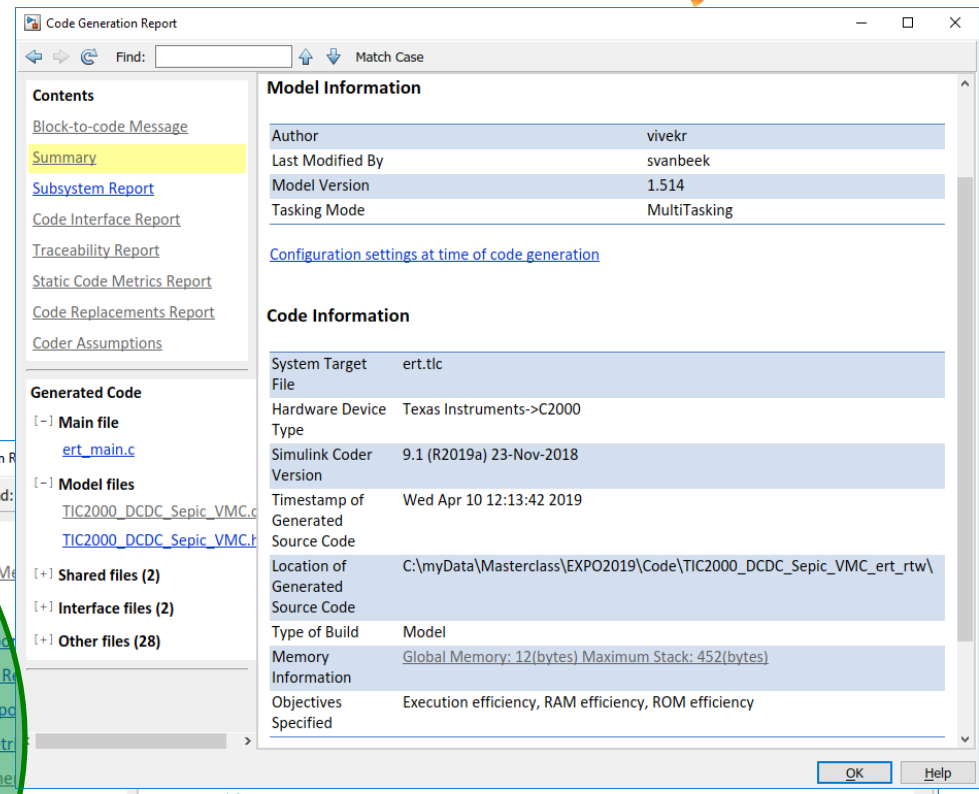
Examples: Simulink Real-Time Solutions with Speedgoat

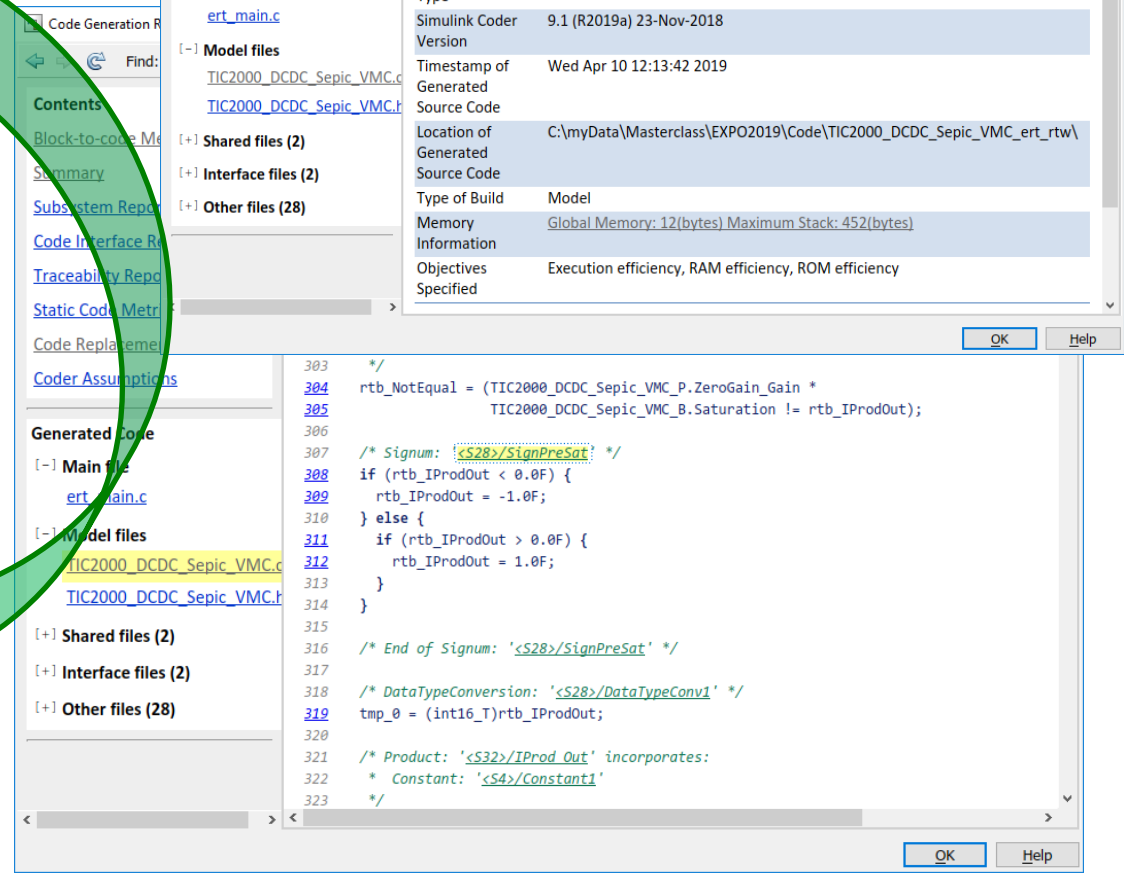
Audio Related Targeting



Automatically Generate C/C++ Code



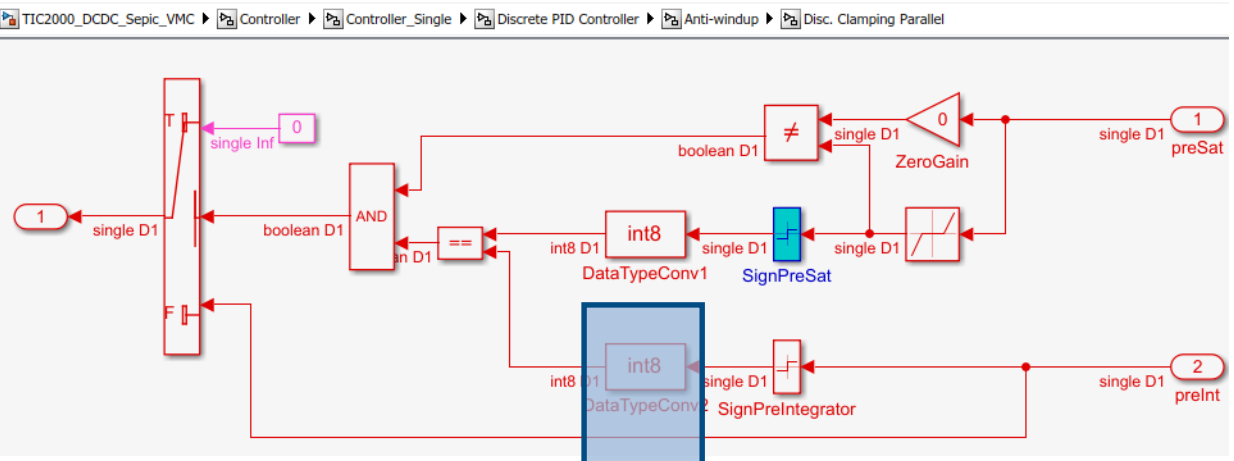


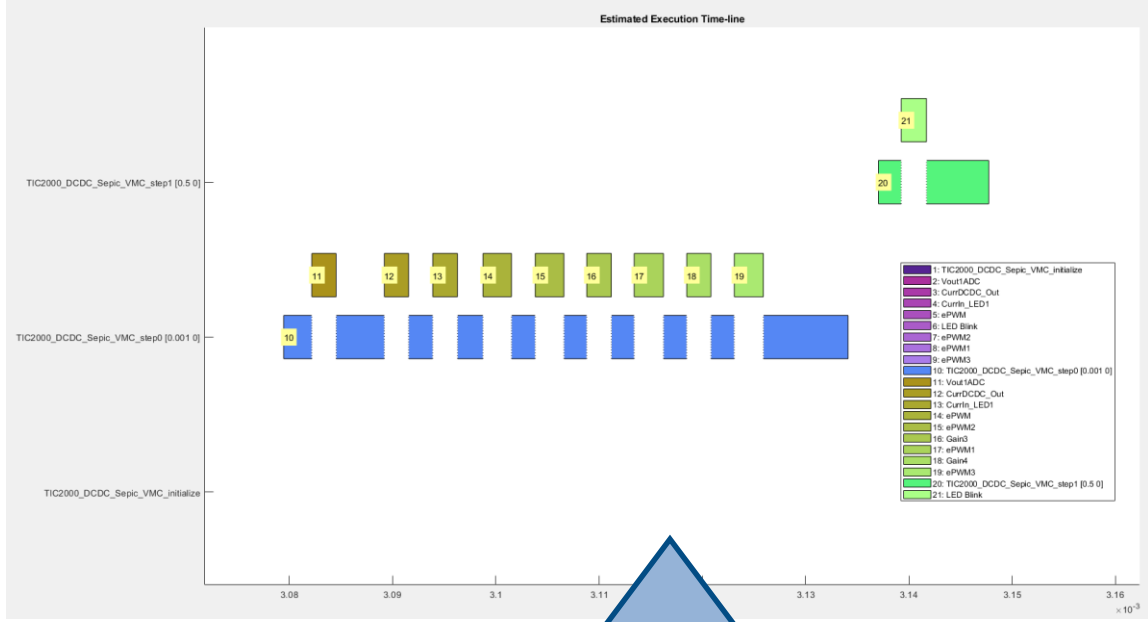


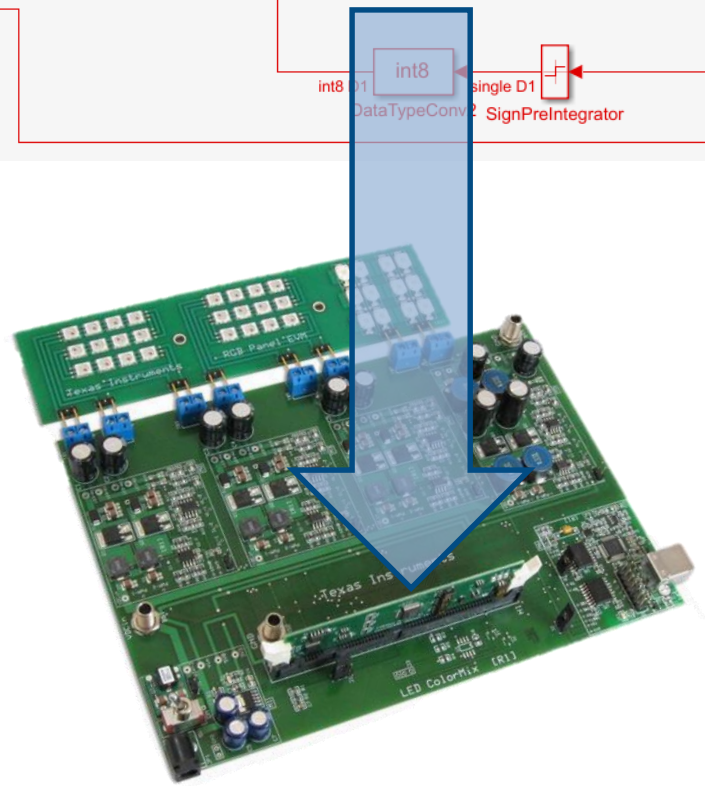
Full bi-directional traceability!!

Requirements

Measure On-Target Code Performance

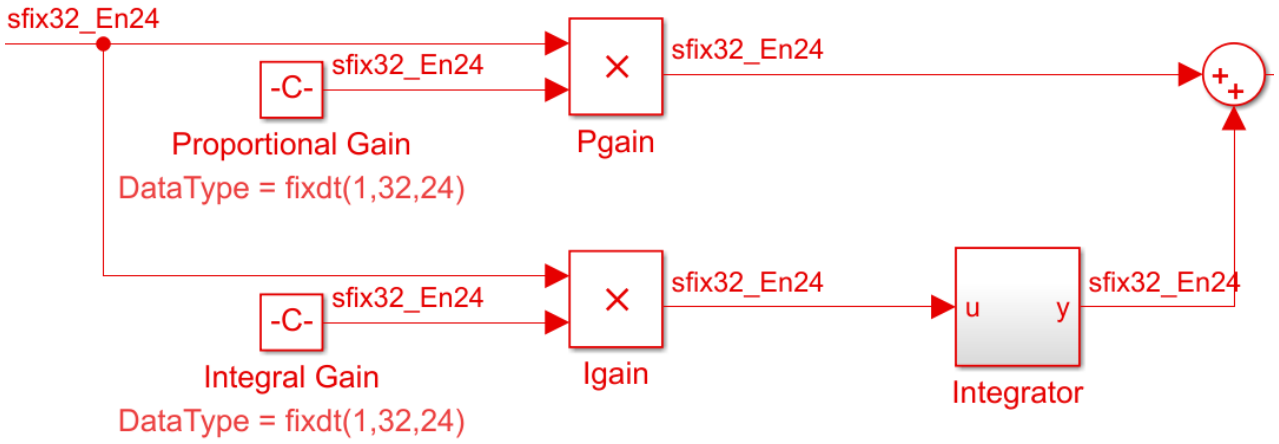


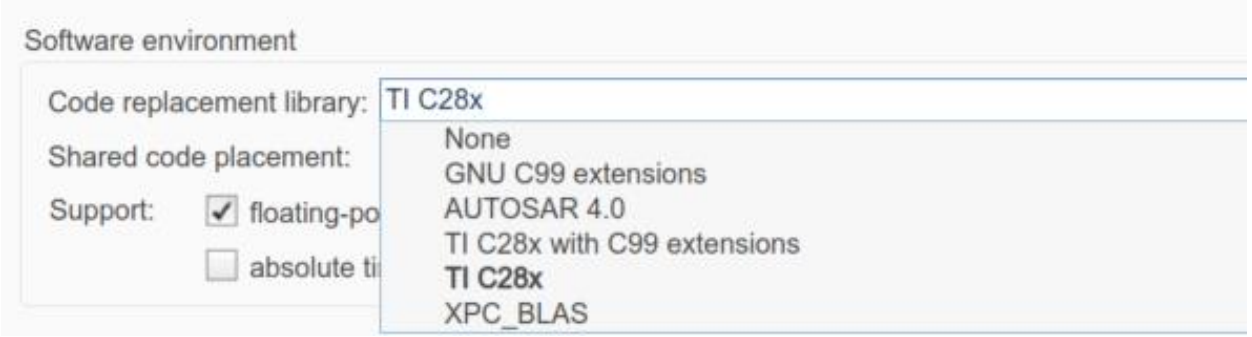




Instrument generated code for On-Target Execution Profiling

Optimize for C2000 to Improve Execution Performance





Code replacements for library 'TI C28x'. The library comprises:

- TI C28x
  - IQmath_tfl_table.mat
  - TI_C28x_addsub_tfl_table.mat
- TI C28x with C99 extensions
  - TI_C28x_iso_tfl_table.mat
  - TI_C28x_tfl_table.mat

To see the replacements and misses in the Code Replacement Viewer, look here.

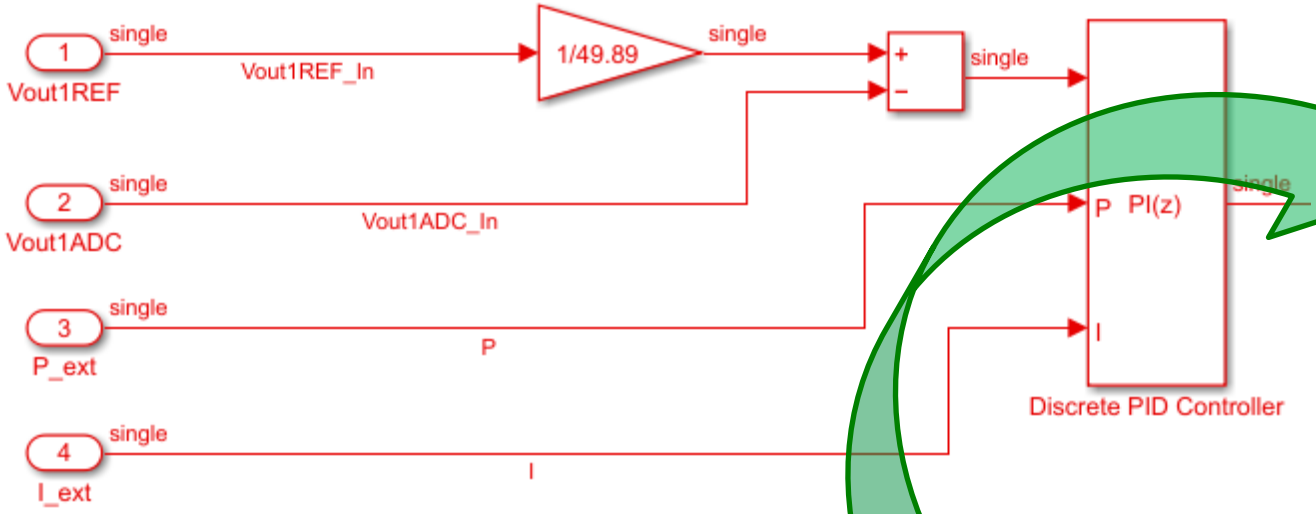
1. Multiplication operator replacements [hide]

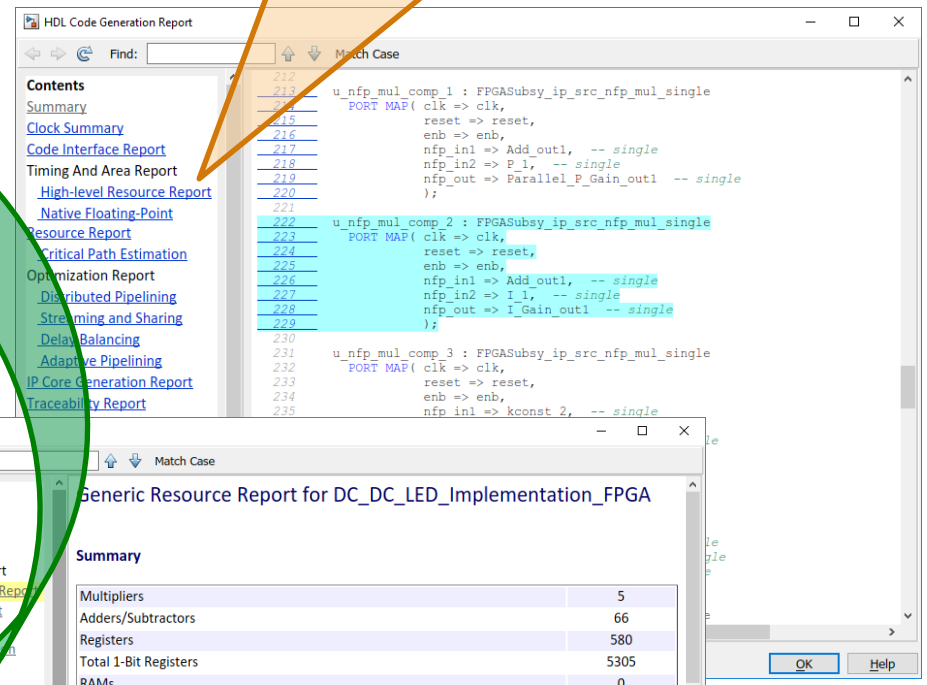
The following table provides a mapping from the multiplication operators used from the selected Code Replacement Library to the blocks in the model that triggered the replacement.

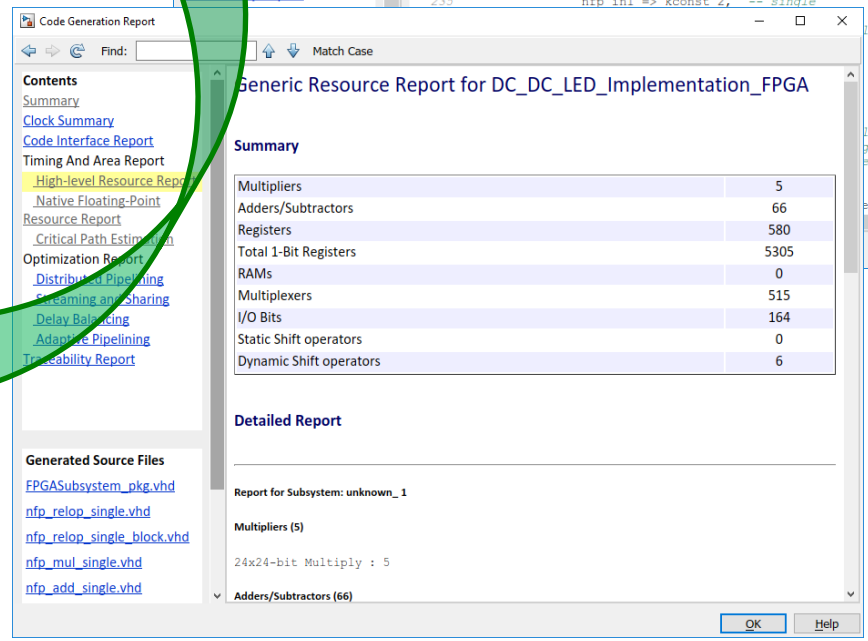
Function	Block
_IQ22mpyI32	<S1>/Gain2 <S1>/Gain3 <S1>/Gain4
__IQmpy	<S2>/Gain
__IQmpy	<S4>/Gain <S4>/Gain1 <Root>/Gain5

Automatically Generate VHDL/Verilog Code

Mix Fixed-Point and Native Floating Point



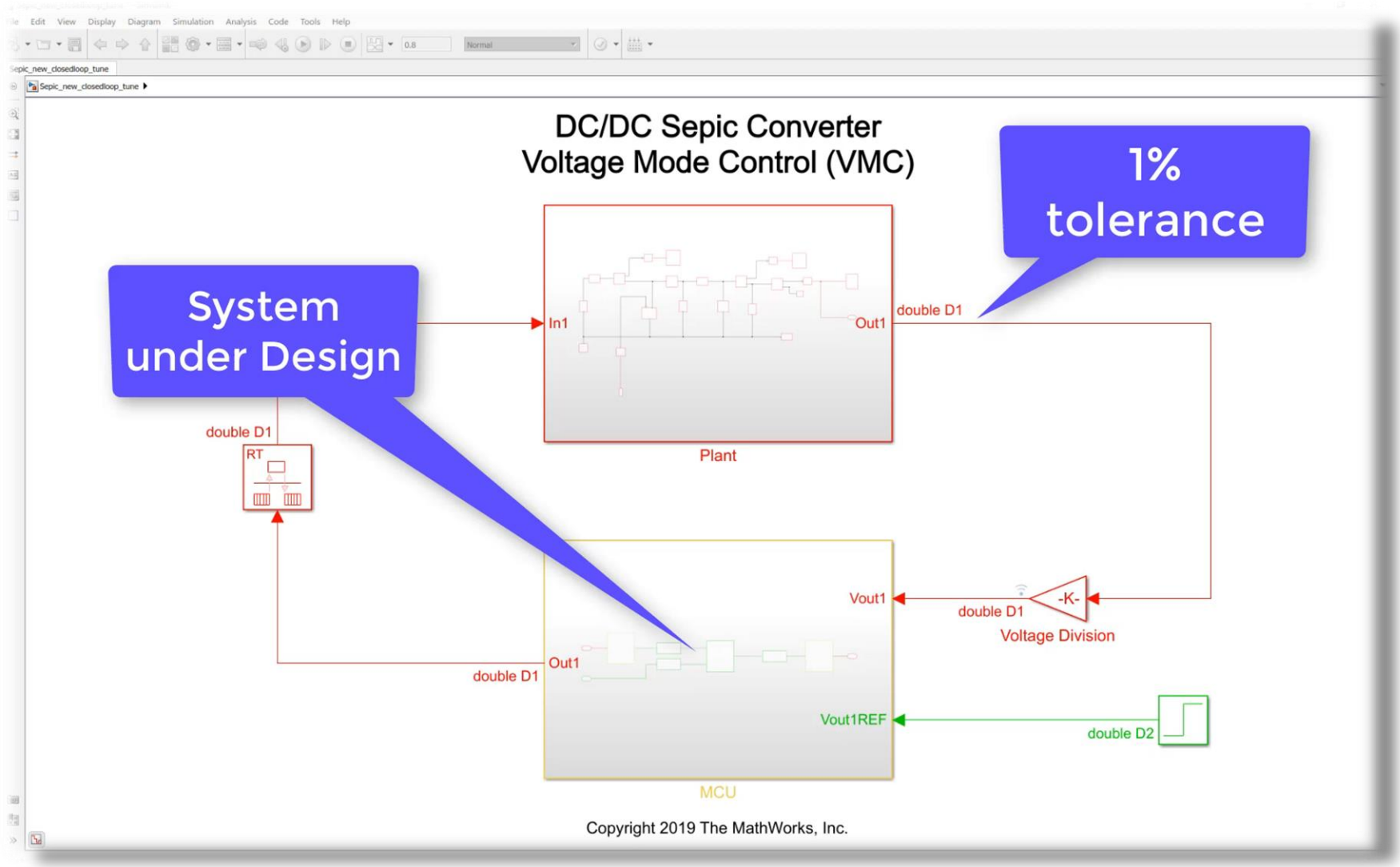




Full bi-directional traceability!!

Requirements

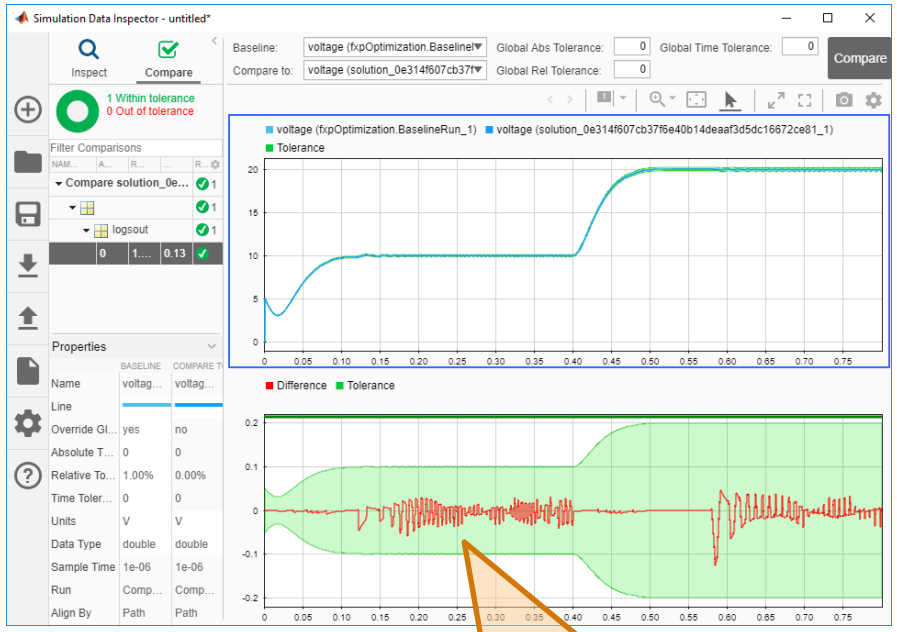
Automated Fixed-Point Optimization and Analysis



Conversion goals:

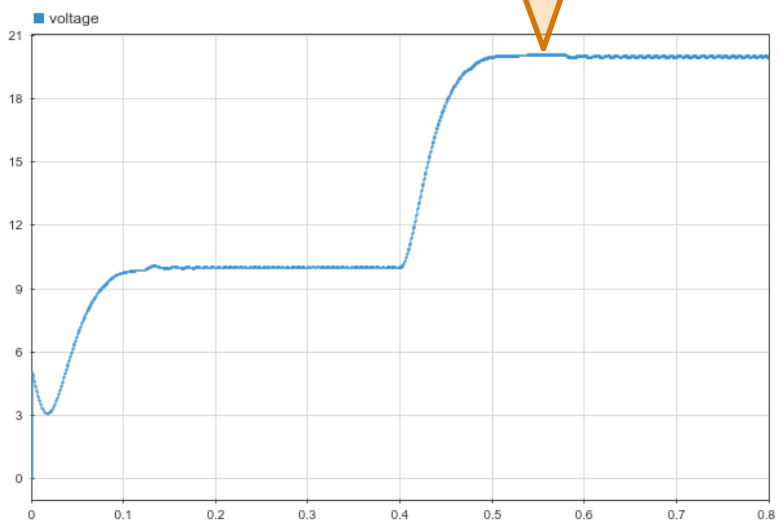
- 1% relative tolerance -- Vout1
- Stable controller behavior
- Manage fixed-point bit growth
- Automated workflow

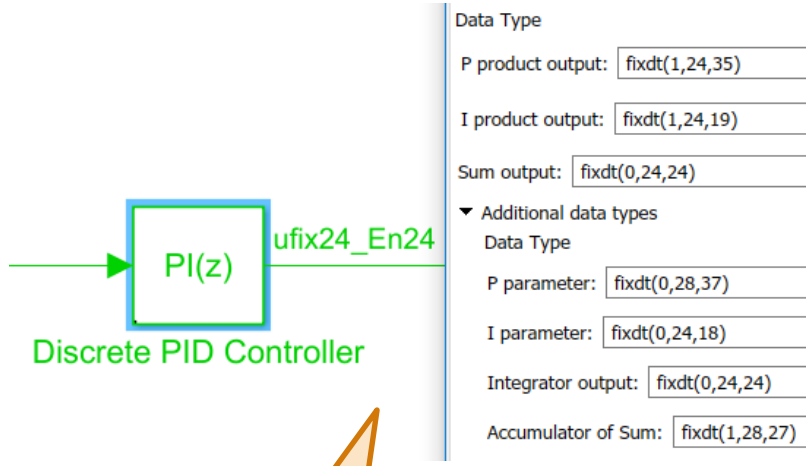
Automated Fixed-Point Optimization and Analysis



Analyze fixed-point quantization impact to defined tolerances

Simulate dynamic behavior of fixed-point model

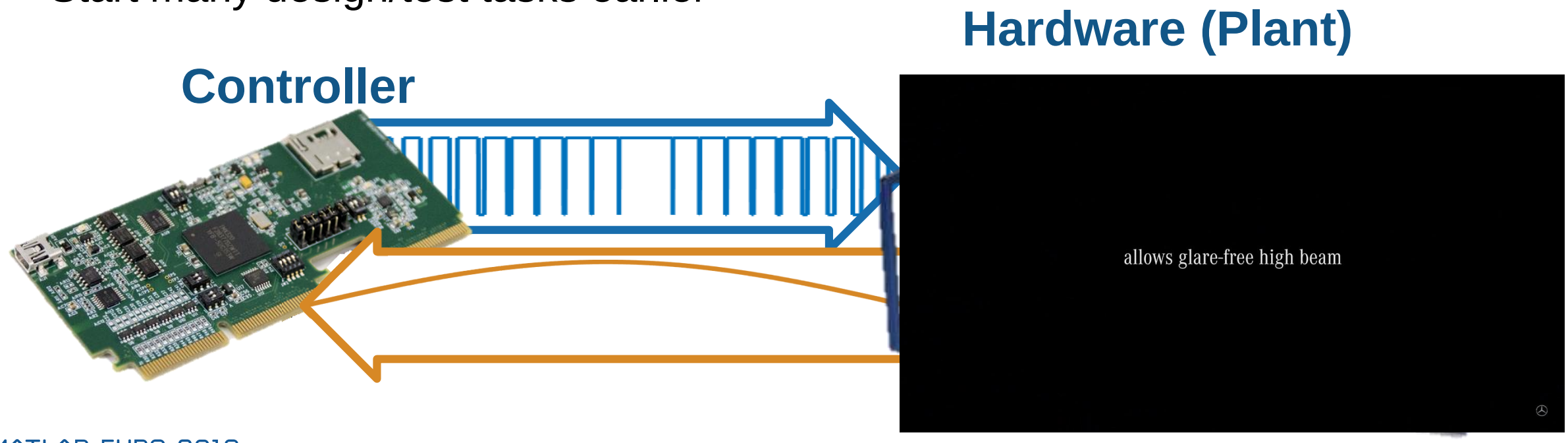




Analyze bit-growth after fixed-point conversion

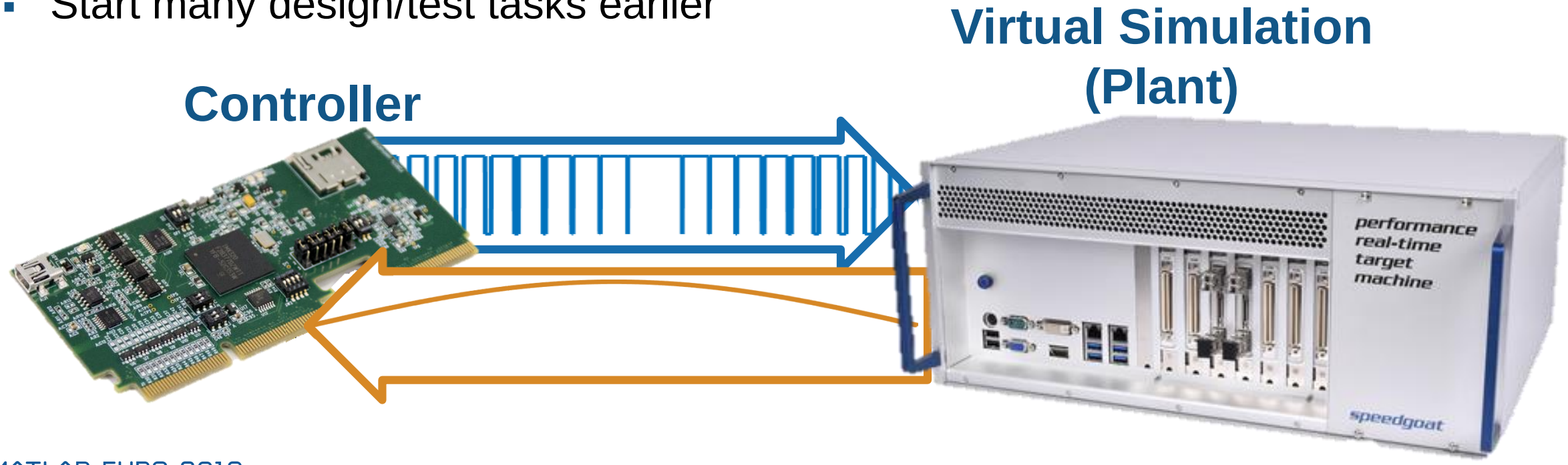
Why do we need Hardware-in-the-Loop (HIL) Testing

- Can replace prototypes or production hardware with a real-time system
- Safer than most power electronics hardware
- Easier to automate testing and test fault conditions
- Start many design/test tasks earlier



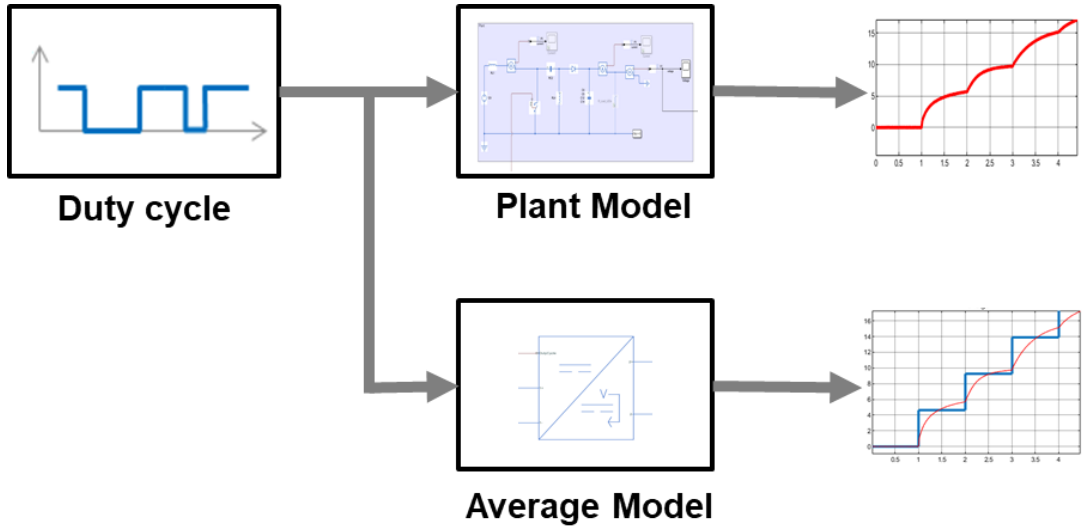
Why do we need Hardware-in-the-Loop (HIL) Testing

- Can replace prototypes or production hardware with a real-time system
- Safer than most power electronics hardware
- Easier to automate testing and test fault conditions
- Start many design/test tasks earlier



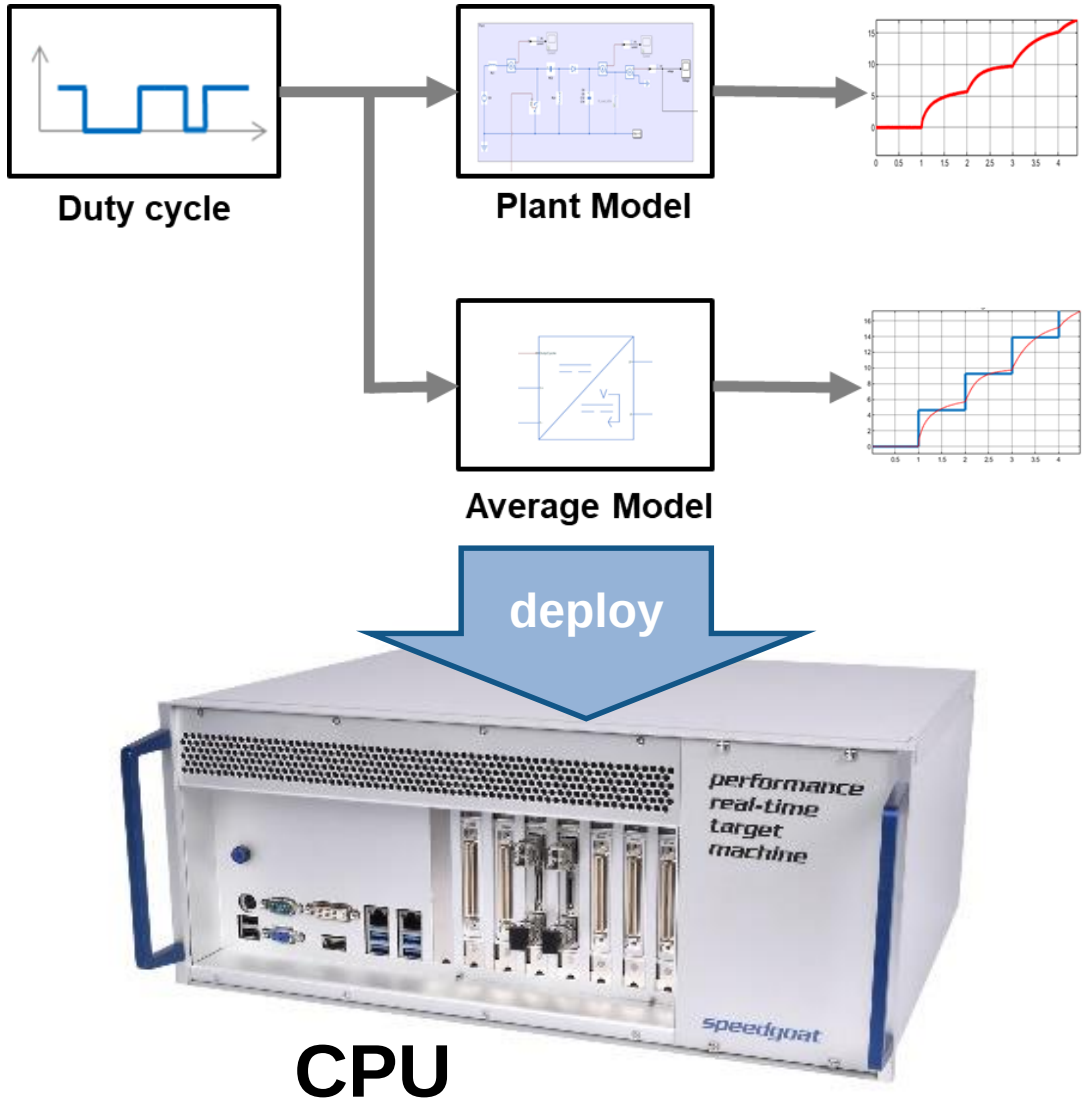
Strategies Hardware-in-the-Loop (HIL) Testing

- Deploy Simscape models on CPUs
  - Using Average-value models to validate controller behaviour
  - To validate electronic switching converters
    - <=2kHz switching frequency
    - Where step-sizes >=10us are sufficient
- Deploy Simscape models on FPGAs
  - To validate electronic switching converters
    - >2kHz switching frequency
    - Where step-sizes <10us are required



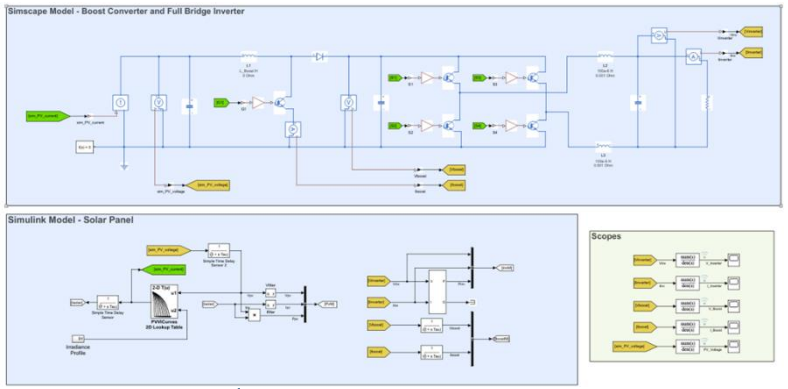
Using Average-value Model for HIL

- A simplified model with acceptable tradeoff between fidelity and performance
- Average models could be obtained using
  - Simscape
  - Small signal analysis
  - Frequency response estimation
  - System identification

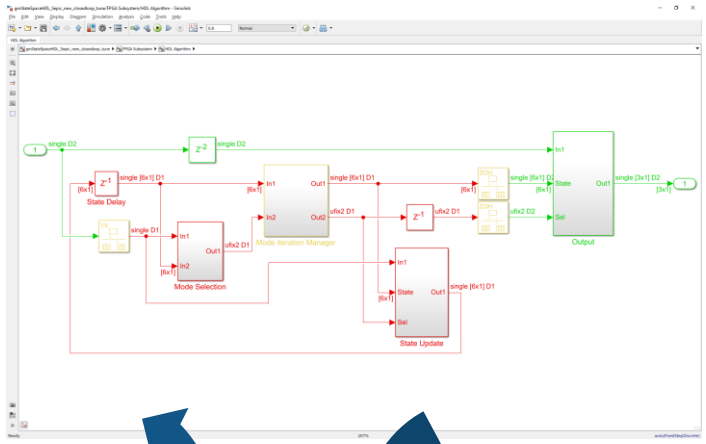


Using Simscape (FPGA) for HIL

Simscape model

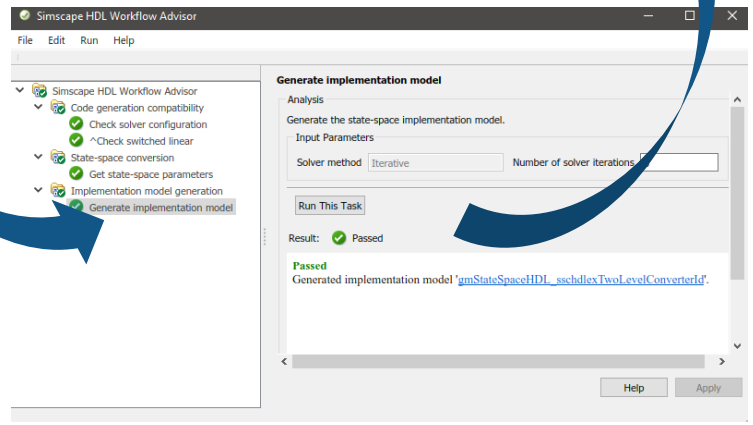


Simulink state-space model

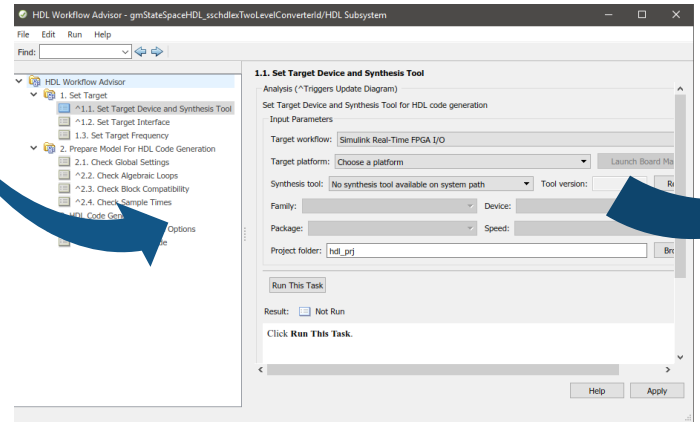


FPGA

```
Delay13_process : PROCESS (clk)
BEGIN
  IF clk'EVENT AND clk = '1' THEN
    IF reset = '1' THEN
      Delay13_out1 <= '0';
    ELSIF enb = '1' THEN
      Delay13_out1 <= Relational_Operator_relop1;
    END IF;
  END IF;
END PROCESS Delay13_process;
```



Simscape HDL Advisor



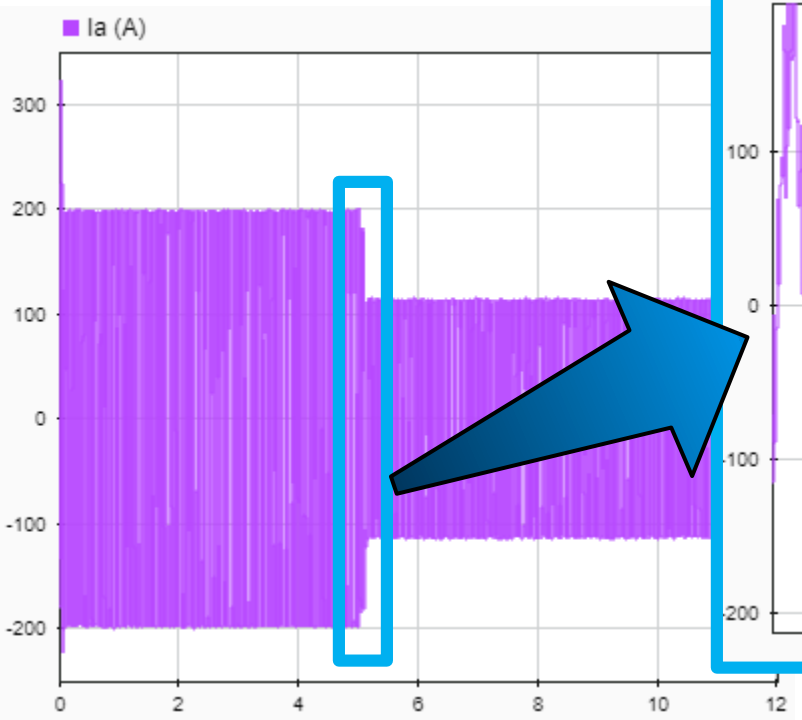
HDL Workflow Advisor

Why is FPGA-based HIL Relevant to Power Electronics?

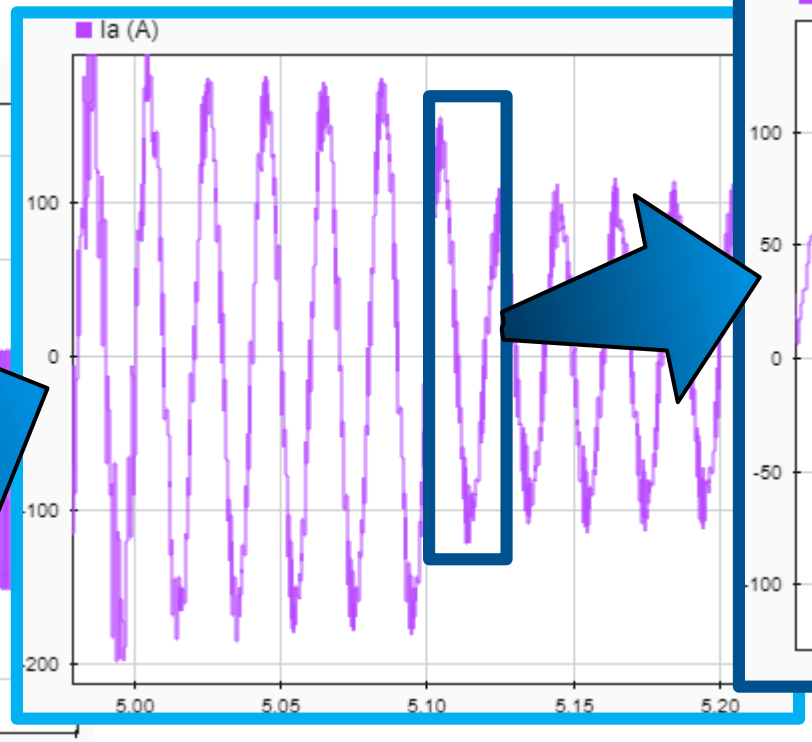
The Need for Small Time Step Simulations

- High sample rates (small time steps) are required to capture fast transients in systems like power electronics

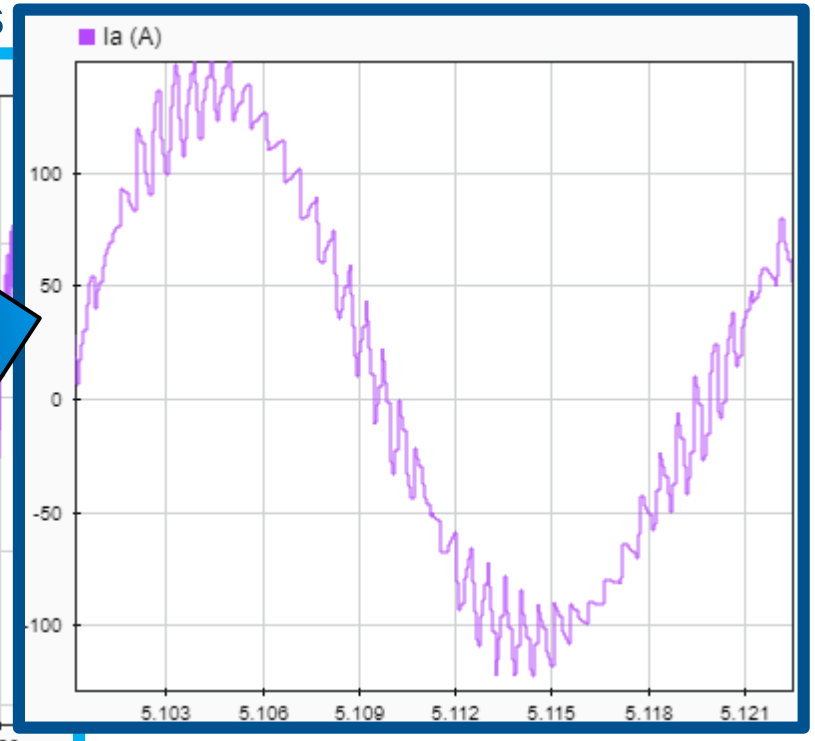
Resolution: seconds



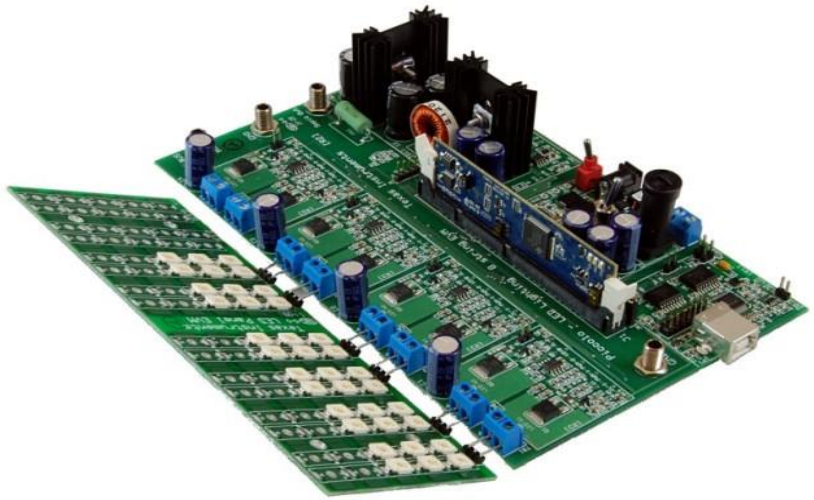
Resolution: milliseconds

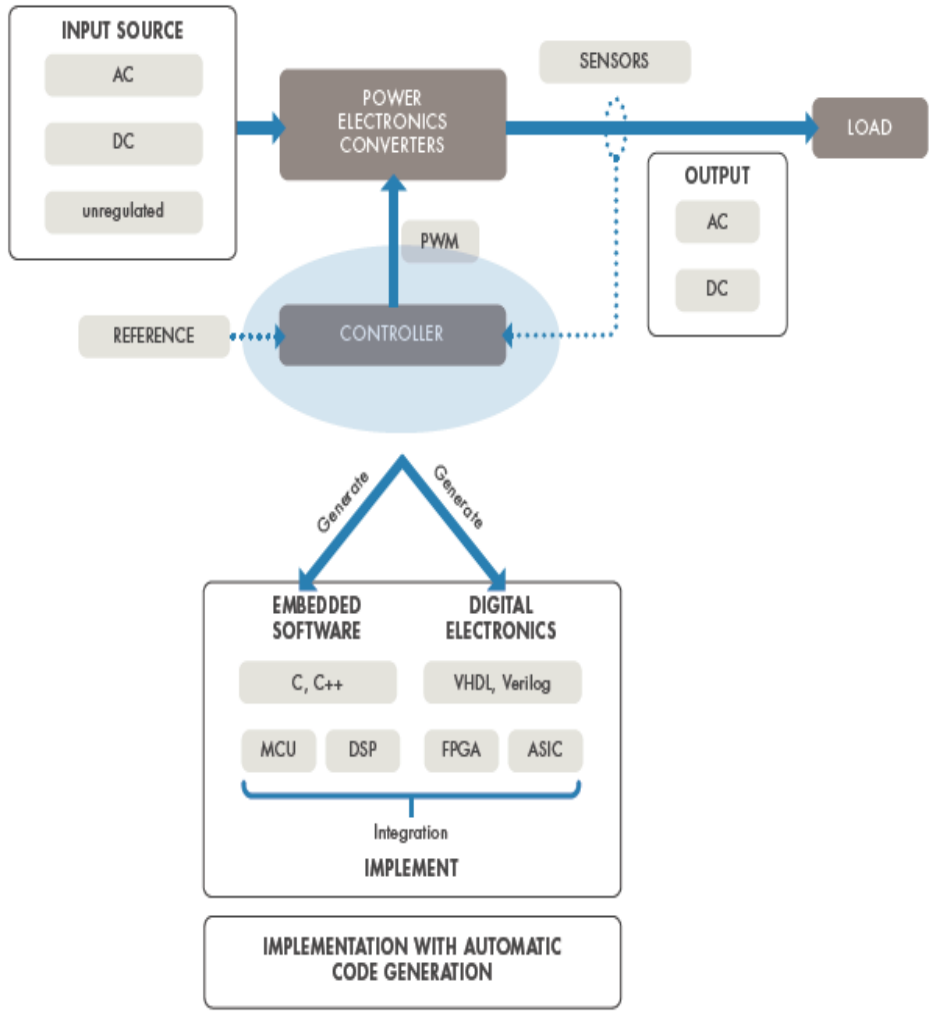


Resolution: microseconds



Recap: Implement Power Electronics Control on an Embedded Platform





What we did:

- Verify the controller for various test cases on real HW
- Generate code from MATLAB and Simulink models optimized for embedded platforms (DSP and FPGA)
- System-level test using Hardware-in-the-Loop

Power Converter Control Design Workflow Tasks

- Size inductor, capacitor and understand the behaviour in continuous and discontinuous mode
- Determine power losses and the thermal behaviour of the converter
- Design control algorithm based on time/frequency domain specification
- Implement power electronic controls on an embedded platform

Alstom Generates Production Code for Safety-Critical Power Converter Control Systems

Challenge

Design and implement real-time power conversion and control systems for trams, metros, and railways

Solution

Use MathWorks tools for Model-Based Design to design, simulate, and automatically generate production code for safety-critical transportation systems

Results

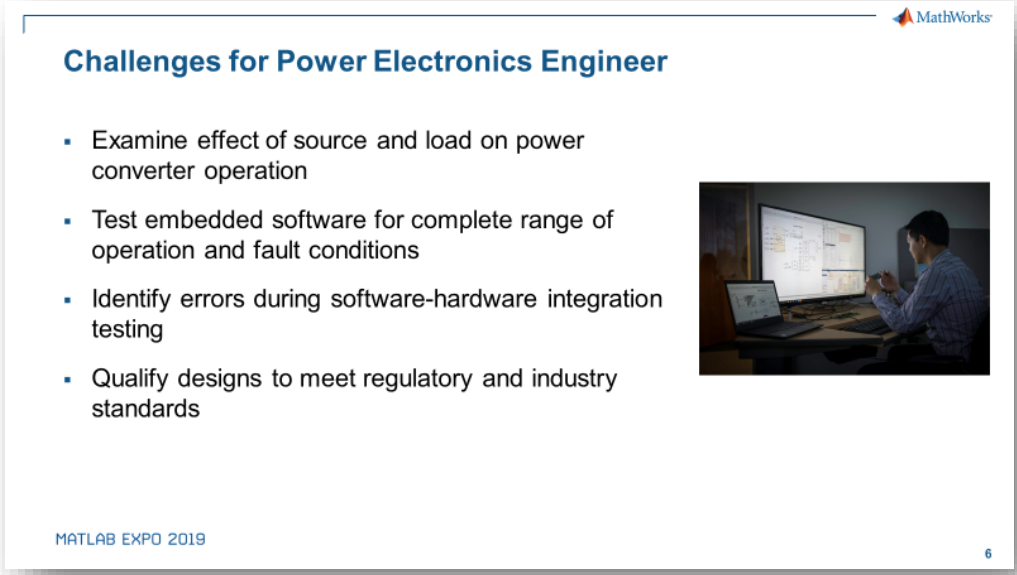
- Development time cut by 50%
- Defect-free, safety-critical code generated and certified
- Common language established

Pendolino tilting train.

"MathWorks tools enable us to control every line of code, and the generated code is readable, fast, and compact. Also, MathWorks tools are industry-standard, with extensive packages and broad support for embedded targets."

- Han Geerligs, senior engineer, Alstom

Summary



- Simulation to validate assumptions early
- Common language to enable collaboration
- Verification from start to end:
  Virtual Prototypes ➔ Physical Prototypes ➔ Hardware-in-the-Loop
- Code generation for rapid design iterations

Related Trainings

- Simscape
  - Modeling Physical Systems with Simscape
  - Modeling Electrical Power Systems with Simscape

- Embedded Code Generation
  - Embedded Coder for Production Code Generation

- HDL Code Generation
  - Generating HDL Code from Simulink

https://ch.mathworks.com/services/training.html