MATLAB EXPO

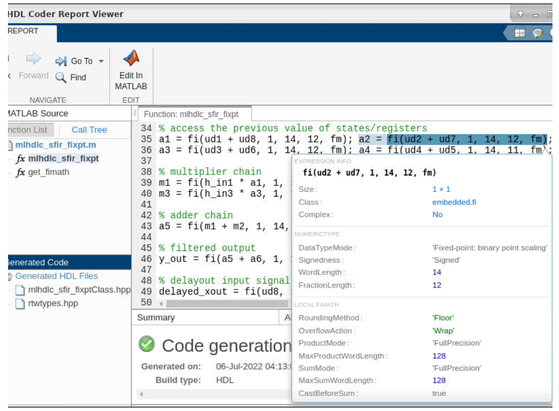
2024年5月28日 | 北京

MATLAB助力芯片研发：算法快速实现与硬件验证提效

赵恒, MathWorks

MathWorks

内容提要


HDL Coder Report Viewer

REPORT

Forward | Go To | Find | Edit In MATLAB

NAVIGATE | EDIT

MATLAB Source — Function List | Call Tree
- mlhdlc_sfir_fixpt.m
  - mlhdlc_sfir_fixpt
  - get_fimath

Generated Code — Generated HDL Files
- mlhdlc_sfir_fixptClass.hpp
- rtwtypes.hpp

Function: mlhdlc_sfir_fixpt

```
% access the previous value of states/registers
a1 = fi(ud1 + ud8, 1, 14, 12, fm); a2 = fi(ud2 + ud7, 1, 14, 12, fm);
a3 = fi(ud3 + ud6, 1, 14, 12, fm); a4 = fi(ud4 + ud5, 1, 14, 11, fm);

% multiplier chain
m1 = fi(h_in1 * a1, 1,
m3 = fi(h_in3 * a3, 1,

% adder chain
a5 = fi(m1 + m2, 1, 14,

% filtered output
y_out = fi(a5 + a6, 1,

% delayout input signal
delayed_xout = fi(ud8,
```

EXPRESSION INFO

fi(ud2 + ud7, 1, 14, 12, fm)

Size :	1 × 1
Class :	embedded.fi
Complex :	No

NUMERICTYPE

DataTypeMode :	'Fixed-point: binary point scaling'
Signedness :	'Signed'
WordLength :	14
FractionLength :	12

LOCAL FIMATH

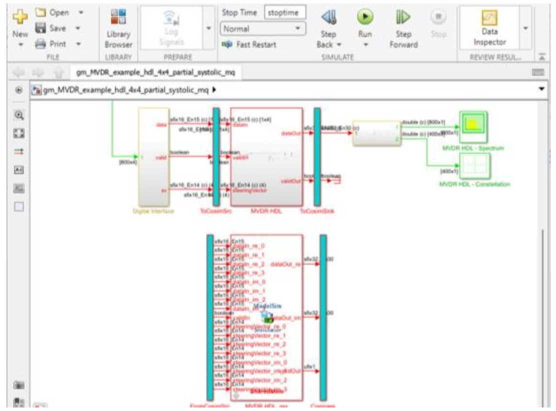
RoundingMethod :	'Floor'
OverflowAction :	'Wrap'
ProductMode :	'FullPrecision'
MaxProductWordLength :	128
SumMode :	'FullPrecision'
MaxSumWordLength :	128
CastBeforeSum :	true

Summary

Code generation

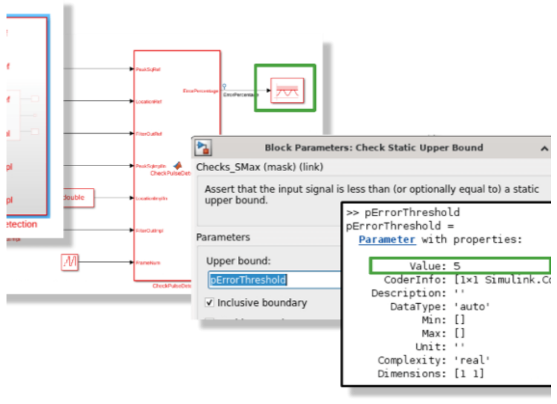
Generated on: 06-Jul-2022 04:13:

Build type: HDL



MATLAB HLS 加速
算法实现与 PPA 评估

自动生成 Testbench
加速验证和简化调试


Block Parameters: Check Static Upper Bound

Checks_SMax (mask) (link)

Assert that the input signal is less than (or optionally equal to) a static upper bound.

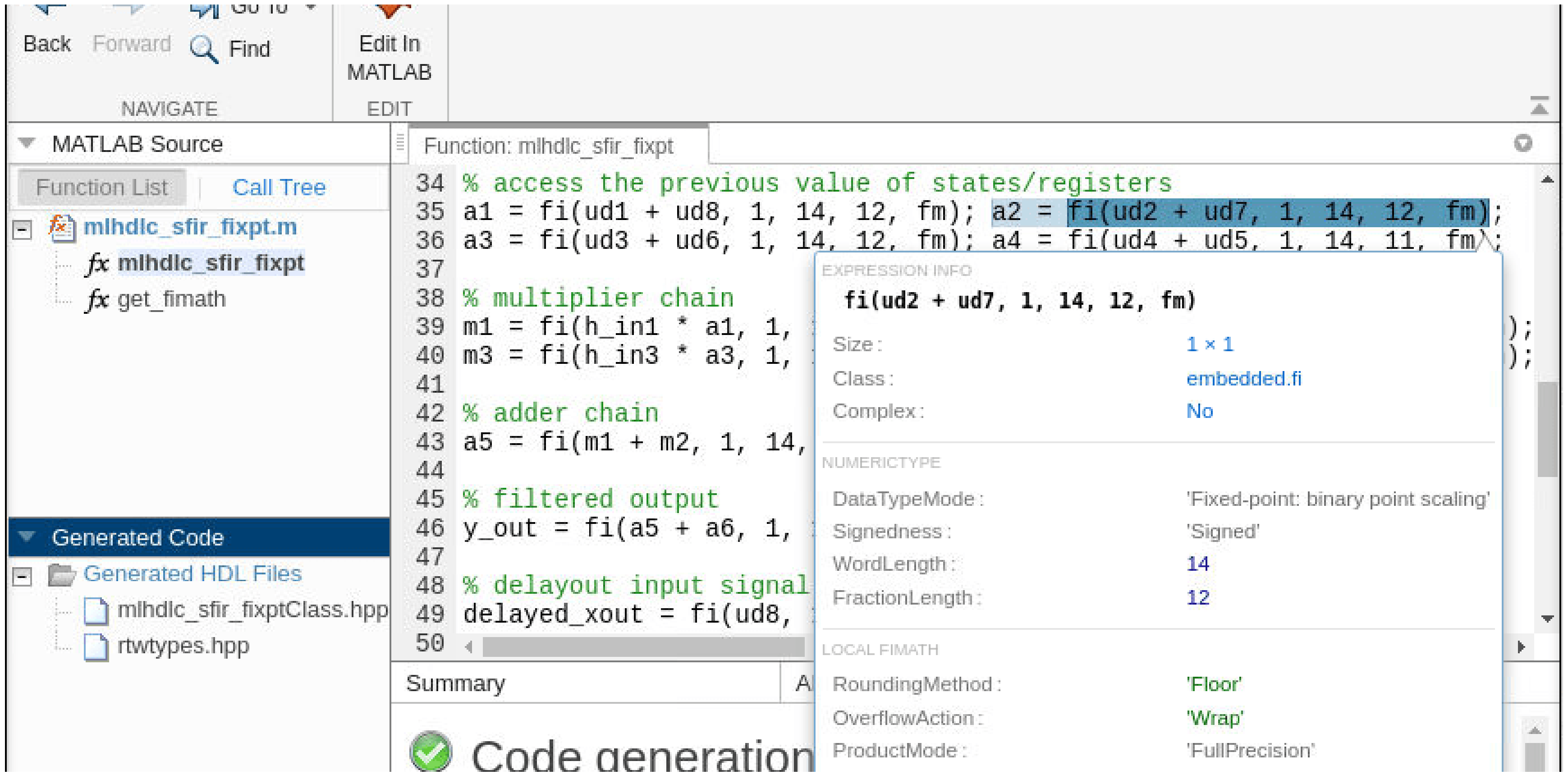
Parameters

Upper bound: pErrorThreshold

[✓] Inclusive boundary

```
>> pErrorThreshold
pErrorThreshold =
  Parameter with properties:

         Value: 5
     CoderInfo: [1×1 Simulink.Co
   Description: ''
      DataType: 'auto'
           Min: []
           Max: []
          Unit: ''
    Complexity: 'real'
    Dimensions: [1 1]
```

生成 SV DPI 组件
建立 UVM 测试环境



```
% access the previous value of states/registers
a1 = fi(ud1 + ud8, 1, 14, 12, fm); a2 = fi(ud2 + ud7, 1, 14, 12, fm);
a3 = fi(ud3 + ud6, 1, 14, 12, fm); a4 = fi(ud4 + ud5, 1, 14, 11, fm);

% multiplier chain
m1 = fi(h_in1 * a1, 1,                                              );
m3 = fi(h_in3 * a3, 1,                                              );

% adder chain
a5 = fi(m1 + m2, 1, 14,

% filtered output
y_out = fi(a5 + a6, 1,

% delayout input signal
delayed_xout = fi(ud8,
```

EXPRESSION INFO

`fi(ud2 + ud7, 1, 14, 12, fm)`

Size :	1 × 1
Class :	embedded.fi
Complex :	No

NUMERICTYPE

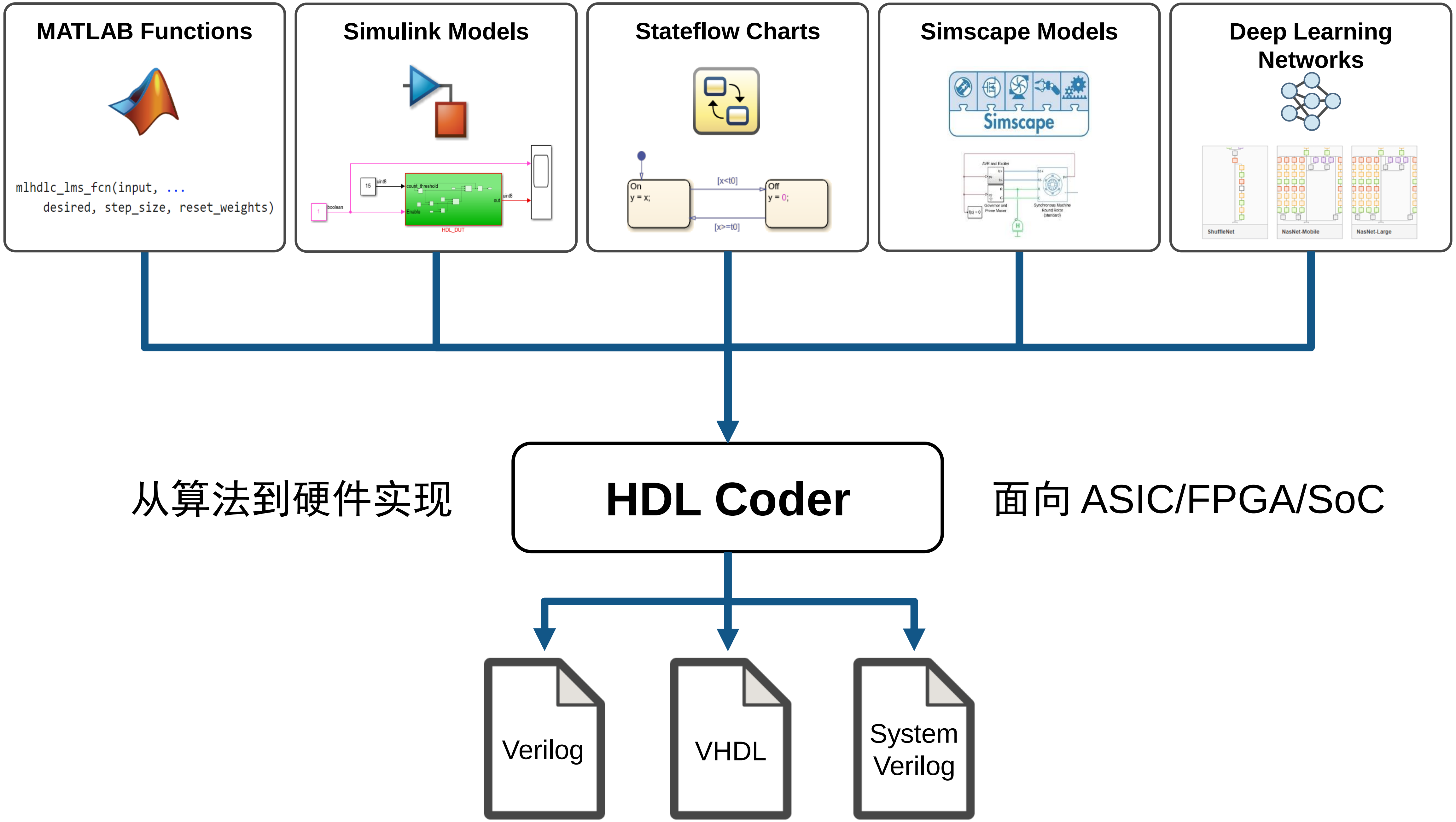
DataTypeMode :	'Fixed-point: binary point scaling'
Signedness :	'Signed'
WordLength :	14
FractionLength :	12

LOCAL FIMATH

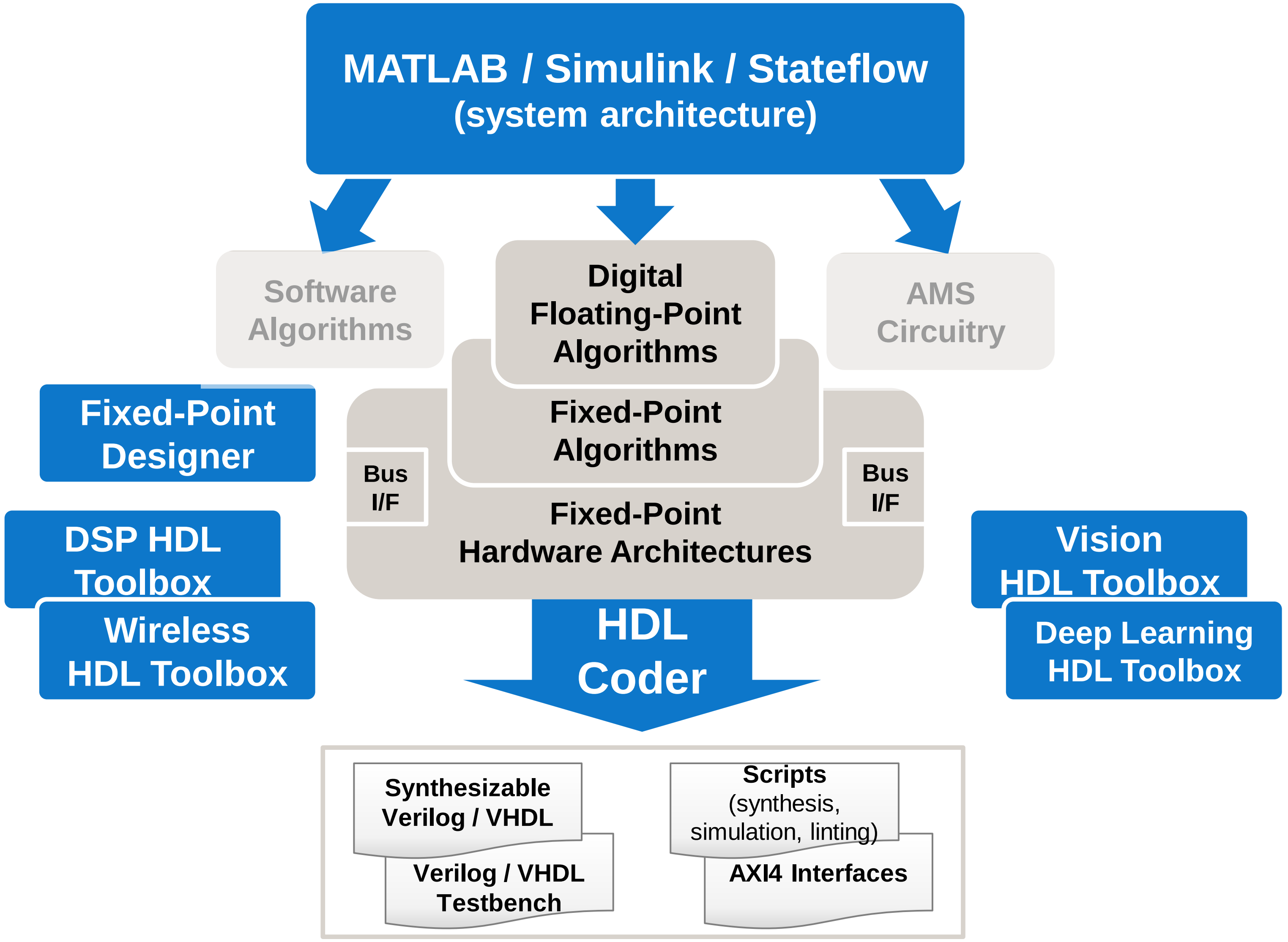
RoundingMethod :	'Floor'
OverflowAction :	'Wrap'
ProductMode :	'FullPrecision'

Code generation

MATLAB HLS 加速算法实现与 PPA 评估

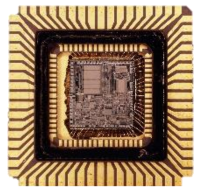


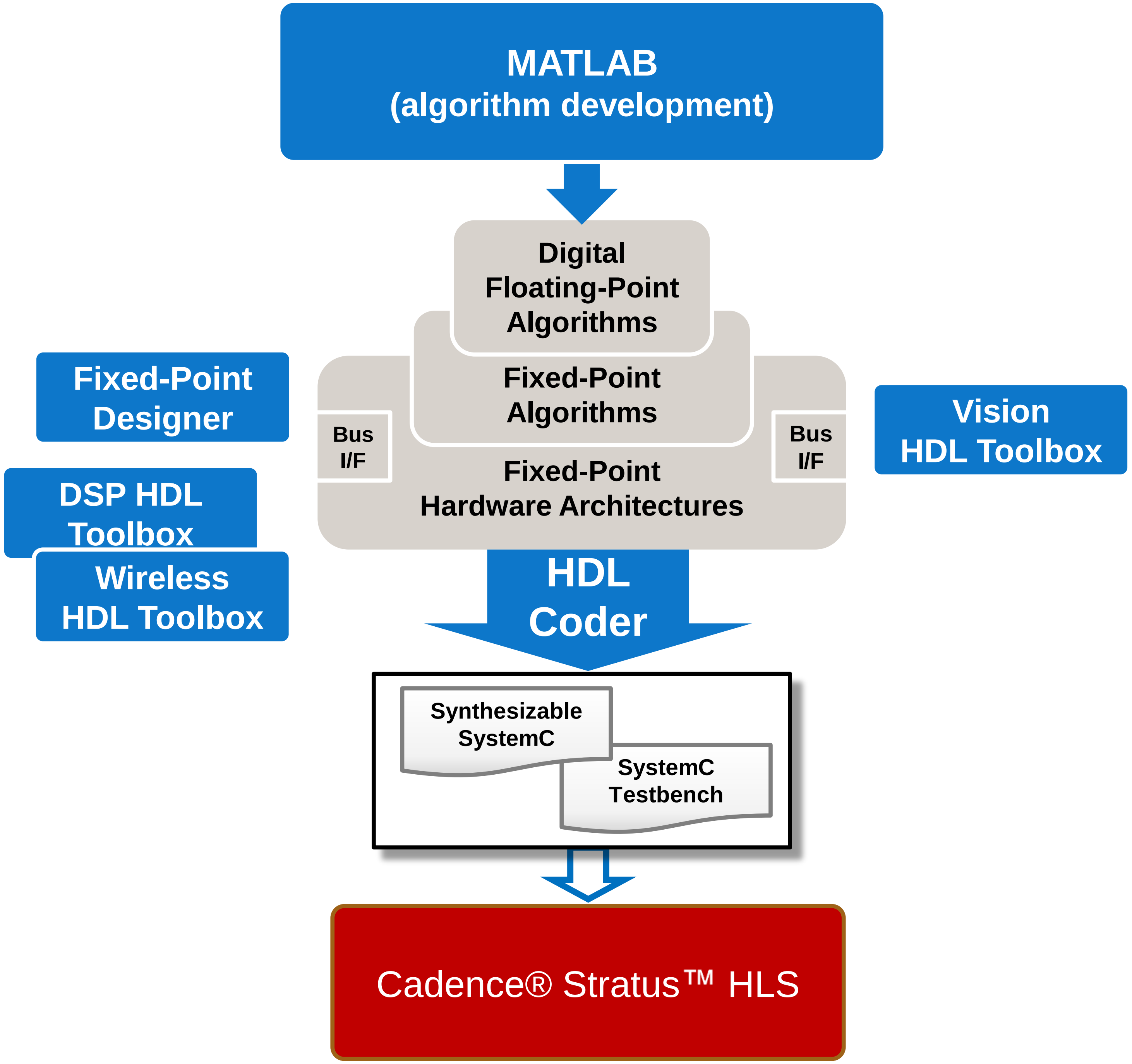
```
mlhdlc_lms_fcn(input, ...
    desired, step_size, reset_weights)
```


MATLAB / Simulink / Stateflow
(system architecture)
Software Algorithms
Digital Floating-Point Algorithms
AMS Circuitry
Fixed-Point Designer
Fixed-Point Algorithms
Bus I/F
Bus I/F
Fixed-Point Hardware Architectures
DSP HDL Toolbox
Wireless HDL Toolbox
Vision HDL Toolbox
Deep Learning HDL Toolbox
HDL Coder
Synthesizable Verilog / VHDL
Verilog / VHDL Testbench
Scripts (synthesis, simulation, linting)
AXI4 Interfaces

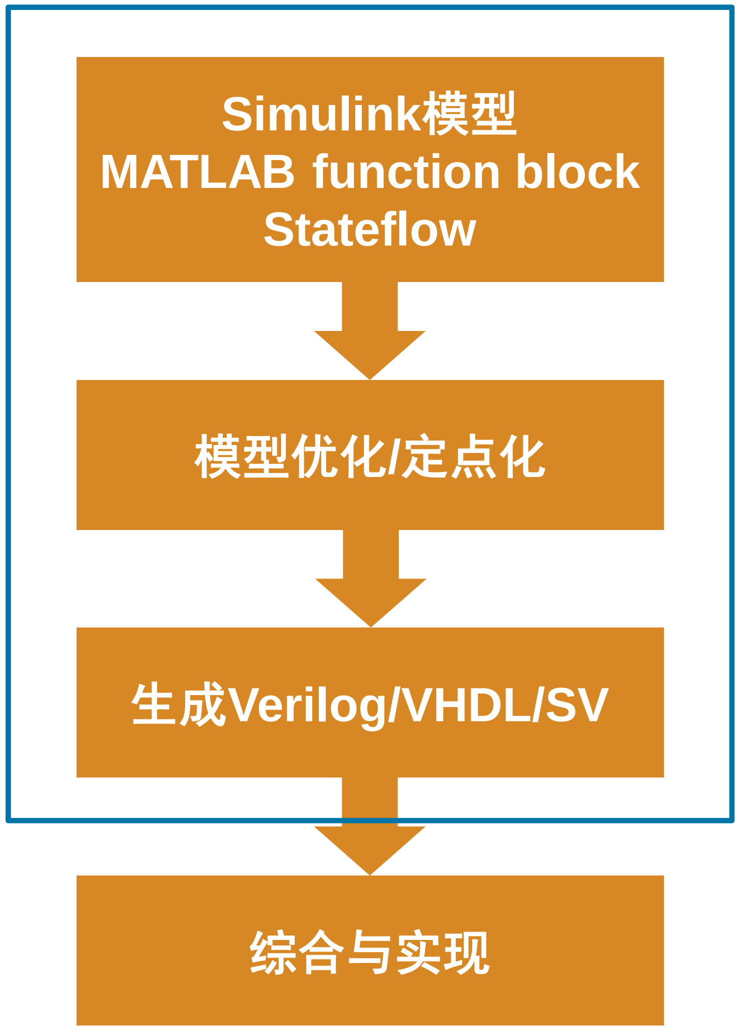
FPGA

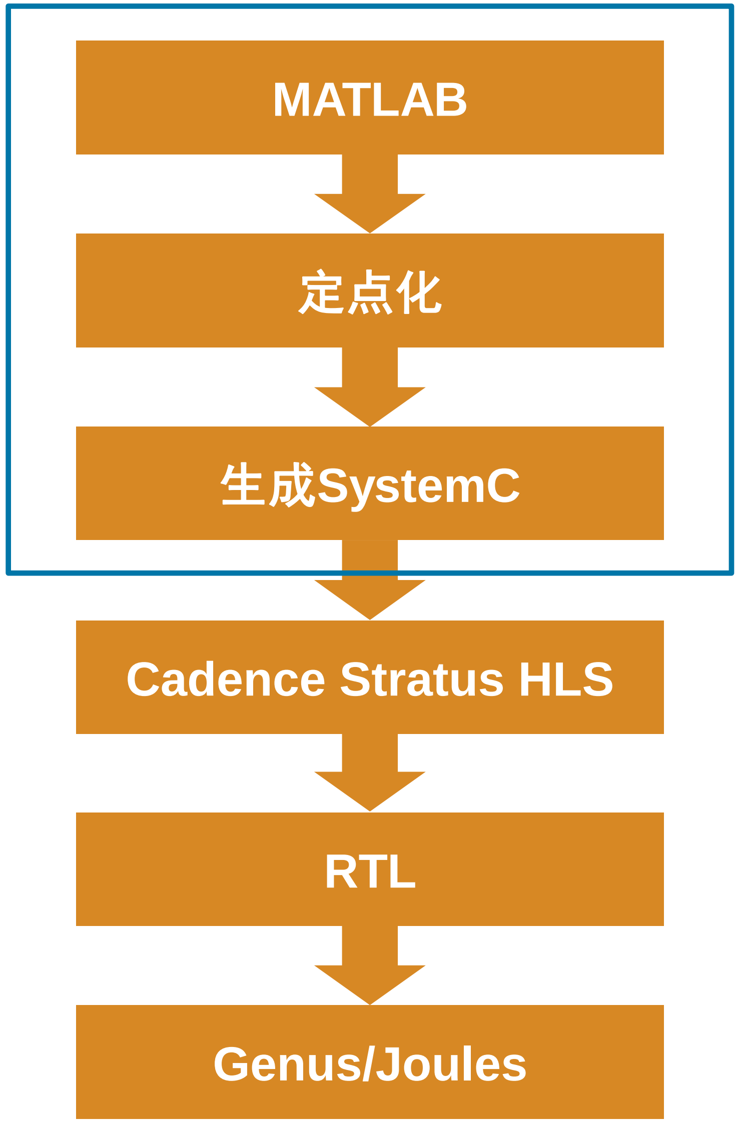
ASIC




MATLAB
(algorithm development)
Digital
Floating-Point
Algorithms
Fixed-Point
Algorithms
Fixed-Point
Hardware Architectures
Bus
I/F
Bus
I/F
Fixed-Point
Designer
DSP HDL
Toolbox
Wireless
HDL Toolbox
Vision
HDL Toolbox
HDL
Coder
Synthesizable
SystemC
SystemC
Testbench
Cadence® Stratus™ HLS

基于模型的 ASIC/FPGA/SoC 流程 & MATLAB High Level Synthesis 流程

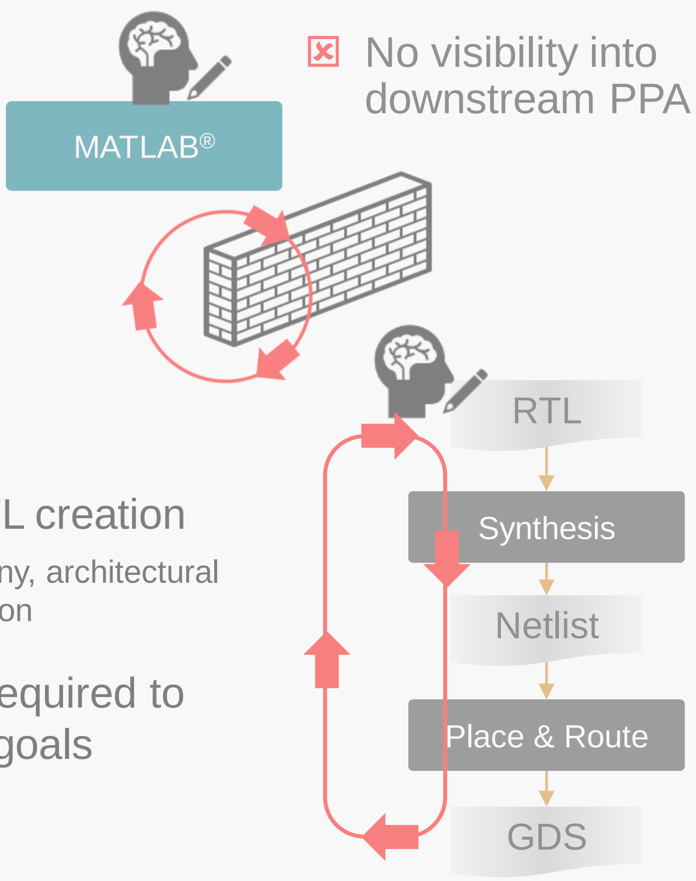




MATLAB EXPO

MATLAB/Stratus Flow

Conventional Flow

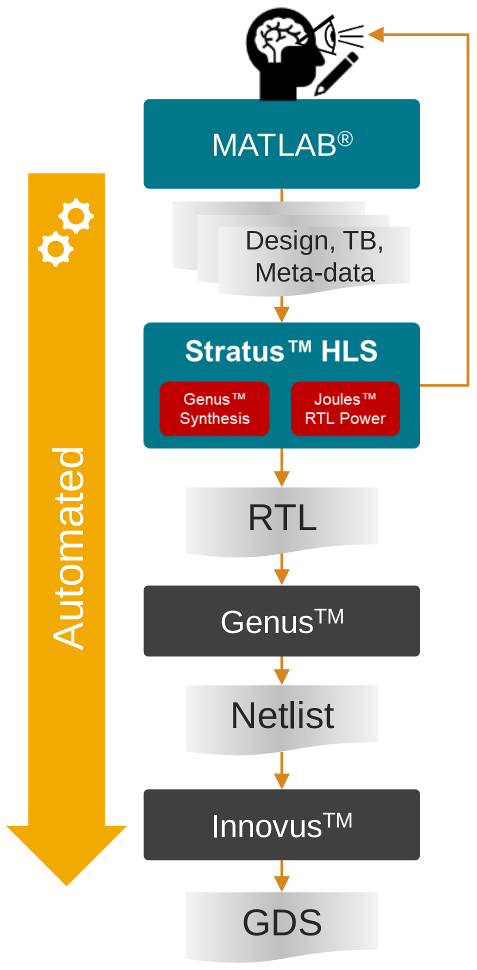


☒ No visibility into downstream PPA

☒ Manual RTL creation
– Little if any, architectural exploration

☒ Iterations required to meet PPA goals

New Flow



- Automated creation of synthesizable SystemC
- Automated import and Stratus project creation
  - Includes design and TB

Benefits

- Early PPA visibility
- Productivity force multiplier
  A single engineer produces results previously requiring a team

From MATLAB to Optimized RTL in Minutes with HDL Coder and Cadence Stratus HLS

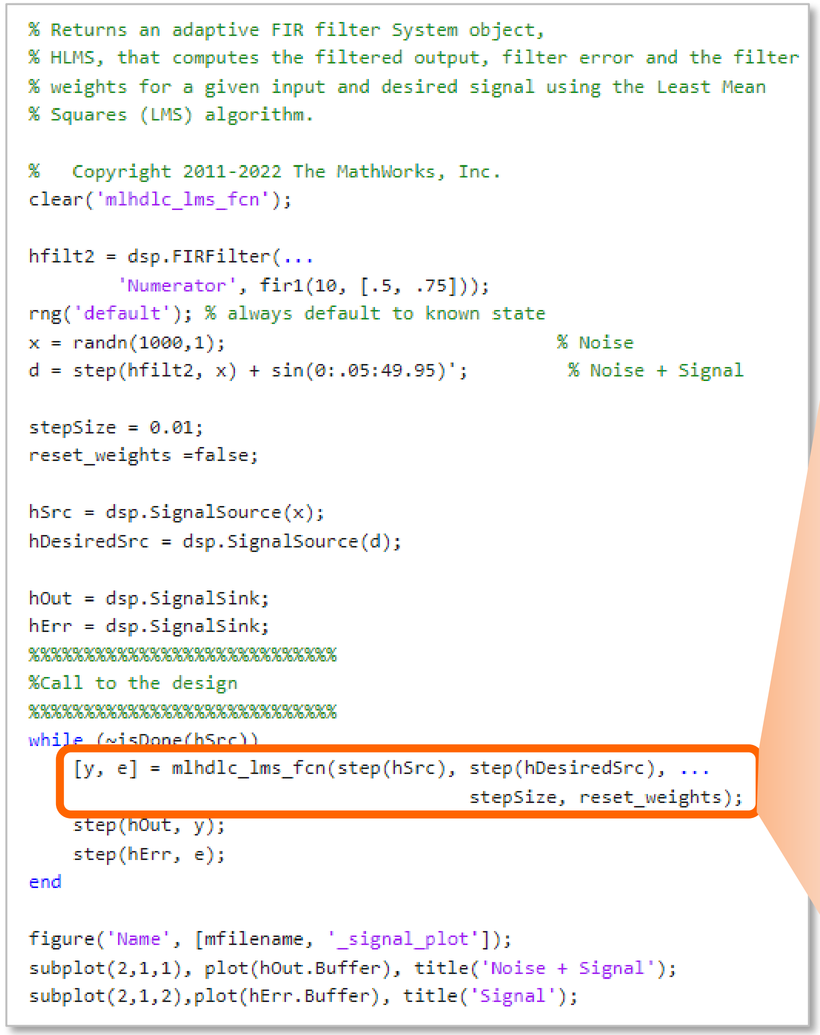
示例

- AES encryption
- Hand-coded RTL, Stratus-coded SystemC, and MATLAB code
- GlobalFoundries 12nm at 500MHz
- Hand-coded RTL area = 20,694

Pipeline Throughput	Hand-Coded SystemC		MATLAB-generated SystemC	
	Area	Power	Area	Power
1	**19,927**	**11.4**	**19,475**	**10.7**
2	**11,063**	**7.1**	**13,194**	**10.1**
4	6,572	4.6	9,877	7.5
6	4,427	2.9	8,139	5.5
11	2,355	1.3	6,339	4.5

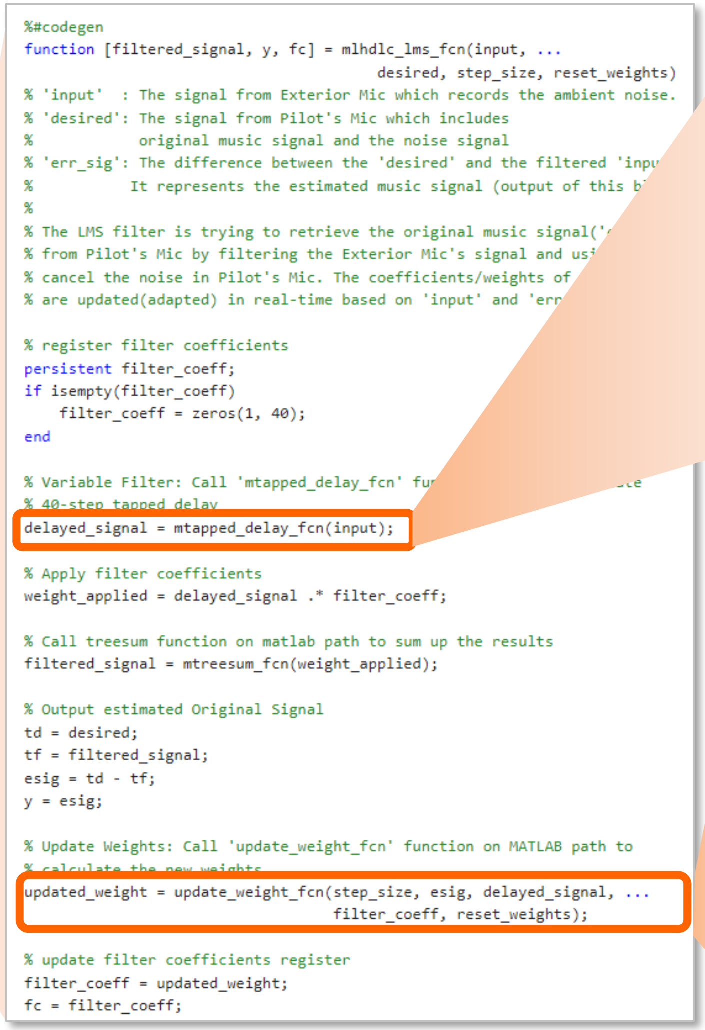
Cadence and MathWorks Announce Flow from MATLAB to RTL

MATLAB HLS 代码构成



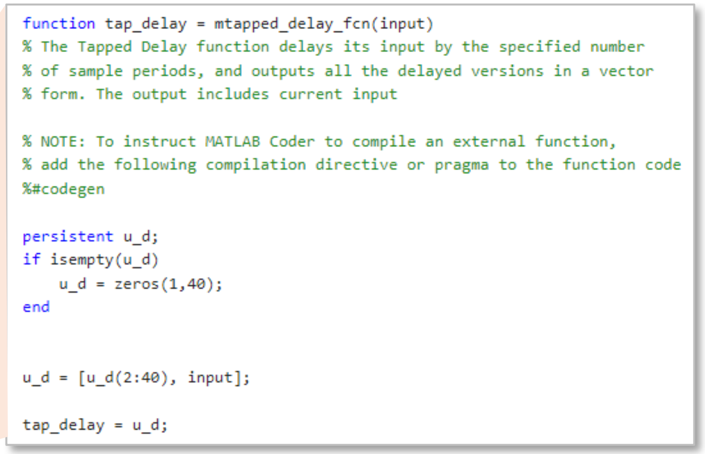
```
% Returns an adaptive FIR filter System object,
% HLMS, that computes the filtered output, filter error and the filter
% weights for a given input and desired signal using the Least Mean
% Squares (LMS) algorithm.

%   Copyright 2011-2022 The MathWorks, Inc.
clear('mlhdlc_lms_fcn');

hfilt2 = dsp.FIRFilter(...
        'Numerator', fir1(10, [.5, .75]));
rng('default'); % always default to known state
x = randn(1000,1);                            % Noise
d = step(hfilt2, x) + sin(0:.05:49.95)';        % Noise + Signal

stepSize = 0.01;
reset_weights =false;

hSrc = dsp.SignalSource(x);
hDesiredSrc = dsp.SignalSource(d);

hOut = dsp.SignalSink;
hErr = dsp.SignalSink;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Call to the design
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
while (~isDone(hSrc))
    [y, e] = mlhdlc_lms_fcn(step(hSrc), step(hDesiredSrc), ...
                                    stepSize, reset_weights);
    step(hOut, y);
    step(hErr, e);
end

figure('Name', [mfilename, '_signal_plot']);
subplot(2,1,1), plot(hOut.Buffer), title('Noise + Signal');
subplot(2,1,2),plot(hErr.Buffer), title('Signal');
```

MATLAB Testbench

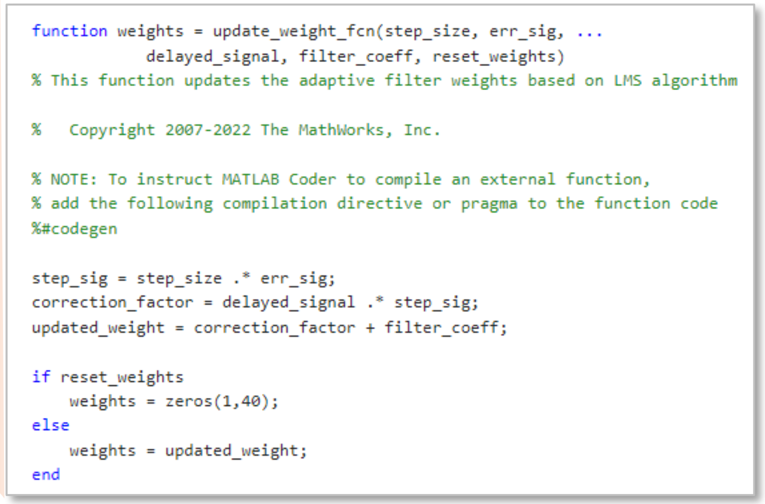


```
%#codegen
function [filtered_signal, y, fc] = mlhdlc_lms_fcn(input, ...
                                       desired, step_size, reset_weights)
% 'input'  : The signal from Exterior Mic which records the ambient noise.
% 'desired': The signal from Pilot's Mic which includes
%            original music signal and the noise signal
% 'err_sig': The difference between the 'desired' and the filtered 'inpu
%           It represents the estimated music signal (output of this b
%
% The LMS filter is trying to retrieve the original music signal('
% from Pilot's Mic by filtering the Exterior Mic's signal and usi
% cancel the noise in Pilot's Mic. The coefficients/weights of
% are updated(adapted) in real-time based on 'input' and 'err

% register filter coefficients
persistent filter_coeff;
if isempty(filter_coeff)
    filter_coeff = zeros(1, 40);
end

% Variable Filter: Call 'mtapped_delay_fcn' fu
% 40-step tapped delay
delayed_signal = mtapped_delay_fcn(input);

% Apply filter coefficients
weight_applied = delayed_signal .* filter_coeff;

% Call treesum function on matlab path to sum up the results
filtered_signal = mtreesum_fcn(weight_applied);

% Output estimated Original Signal
td = desired;
tf = filtered_signal;
esig = td - tf;
y = esig;

% Update Weights: Call 'update_weight_fcn' function on MATLAB path to
% calculate the new weights
updated_weight = update_weight_fcn(step_size, esig, delayed_signal, ...
                                   filter_coeff, reset_weights);

% update filter coefficients register
filter_coeff = updated_weight;
fc = filter_coeff;
```

MATLAB Design



```
function tap_delay = mtapped_delay_fcn(input)
% The Tapped Delay function delays its input by the specified number
% of sample periods, and outputs all the delayed versions in a vector
% form. The output includes current input

% NOTE: To instruct MATLAB Coder to compile an external function,
% add the following compilation directive or pragma to the function code
%#codegen

persistent u_d;
if isempty(u_d)
    u_d = zeros(1,40);
end


u_d = [u_d(2:40), input];

tap_delay = u_d;
```

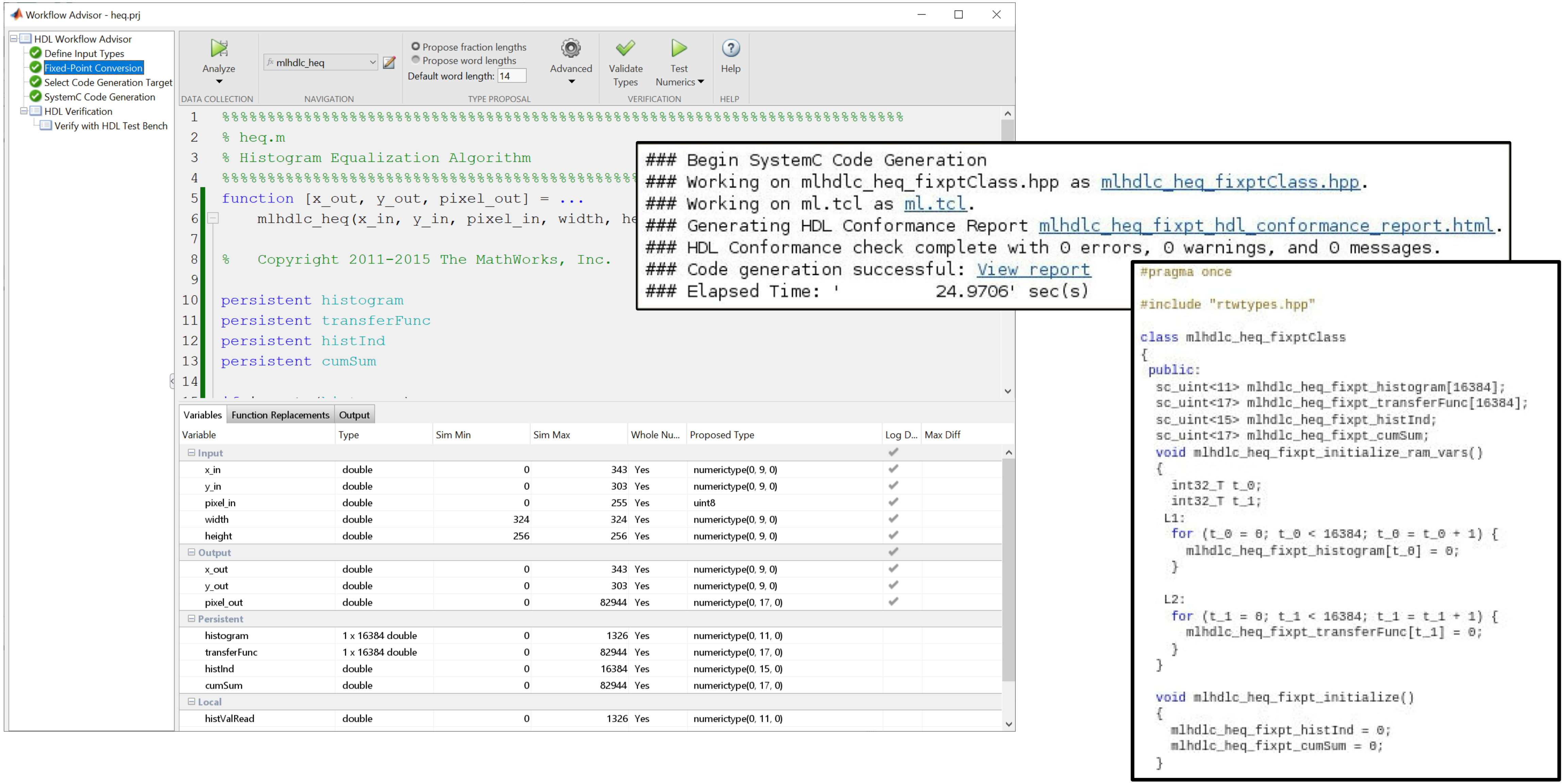


```
function weights = update_weight_fcn(step_size, err_sig, ...
            delayed_signal, filter_coeff, reset_weights)
% This function updates the adaptive filter weights based on LMS algorithm

%   Copyright 2007-2022 The MathWorks, Inc.

% NOTE: To instruct MATLAB Coder to compile an external function,
% add the following compilation directive or pragma to the function code
%#codegen

step_sig = step_size .* err_sig;
correction_factor = delayed_signal .* step_sig;
updated_weight = correction_factor + filter_coeff;

if reset_weights
    weights = zeros(1,40);
else
    weights = updated_weight;
end
```

MATLAB functions

SystemC 代码生成流程

Generate C++ with Timing (Leverage FXPD, HDLC, MLC)
Connectivity with Cadence ASIC toolchain (Xcelium, Genus, Joules)



```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% heq.m
% Histogram Equalization Algorithm
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [x_out, y_out, pixel_out] = ...
    mlhdlc_heq(x_in, y_in, pixel_in, width, he

%   Copyright 2011-2015 The MathWorks, Inc.

persistent histogram
persistent transferFunc
persistent histInd
persistent cumSum

```

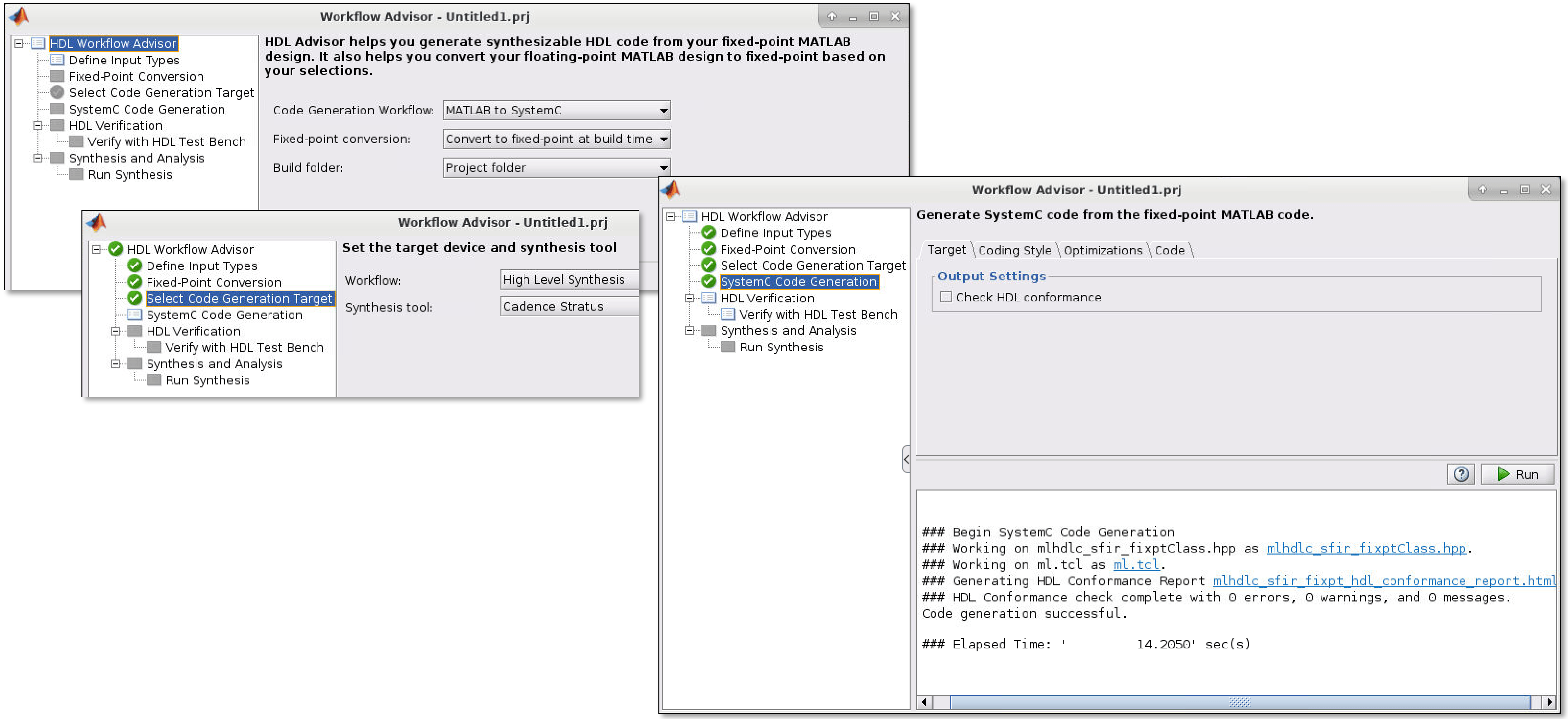
Variables | Function Replacements | Output

Variable	Type	Sim Min	Sim Max	Whole Nu...	Proposed Type	Log D...	Max Diff
⊟ Input						✓	
x_in	double	0	343	Yes	numerictype(0, 9, 0)	✓	
y_in	double	0	303	Yes	numerictype(0, 9, 0)	✓	
pixel_in	double	0	255	Yes	uint8	✓	
width	double	324	324	Yes	numerictype(0, 9, 0)	✓	
height	double	256	256	Yes	numerictype(0, 9, 0)	✓	
⊟ Output						✓	
x_out	double	0	343	Yes	numerictype(0, 9, 0)	✓	
y_out	double	0	303	Yes	numerictype(0, 9, 0)	✓	
pixel_out	double	0	82944	Yes	numerictype(0, 17, 0)	✓	
⊟ Persistent							
histogram	1 x 16384 double	0	1326	Yes	numerictype(0, 11, 0)		
transferFunc	1 x 16384 double	0	82944	Yes	numerictype(0, 17, 0)		
histInd	double	0	16384	Yes	numerictype(0, 15, 0)		
cumSum	double	0	82944	Yes	numerictype(0, 17, 0)		
⊟ Local							
histValRead	double	0	1326	Yes	numerictype(0, 11, 0)		

```
### Begin SystemC Code Generation
### Working on mlhdlc_heq_fixptClass.hpp as mlhdlc_heq_fixptClass.hpp.
### Working on ml.tcl as ml.tcl.
### Generating HDL Conformance Report mlhdlc_heq_fixpt_hdl_conformance_report.html.
### HDL Conformance check complete with 0 errors, 0 warnings, and 0 messages.
### Code generation successful: View report
### Elapsed Time: '         24.9706' sec(s)
```

```
#pragma once

#include "rtwtypes.hpp"

class mlhdlc_heq_fixptClass
{
 public:
  sc_uint<11> mlhdlc_heq_fixpt_histogram[16384];
  sc_uint<17> mlhdlc_heq_fixpt_transferFunc[16384];
  sc_uint<15> mlhdlc_heq_fixpt_histInd;
  sc_uint<17> mlhdlc_heq_fixpt_cumSum;
  void mlhdlc_heq_fixpt_initialize_ram_vars()
  {
    int32_T t_0;
    int32_T t_1;
   L1:
    for (t_0 = 0; t_0 < 16384; t_0 = t_0 + 1) {
      mlhdlc_heq_fixpt_histogram[t_0] = 0;
    }

   L2:
    for (t_1 = 0; t_1 < 16384; t_1 = t_1 + 1) {
      mlhdlc_heq_fixpt_transferFunc[t_1] = 0;
    }
  }

  void mlhdlc_heq_fixpt_initialize()
  {
    mlhdlc_heq_fixpt_histInd = 0;
    mlhdlc_heq_fixpt_cumSum = 0;
  }
```

MATLAB HDL Workflow Advisor



Workflow Advisor - Untitled1.prj

- HDL Workflow Advisor
  - Define Input Types
  - Fixed-Point Conversion
  - Select Code Generation Target
  - SystemC Code Generation
  - HDL Verification
    - Verify with HDL Test Bench
  - Synthesis and Analysis
    - Run Synthesis

HDL Advisor helps you generate synthesizable HDL code from your fixed-point MATLAB design. It also helps you convert your floating-point MATLAB design to fixed-point based on your selections.

Code Generation Workflow: MATLAB to SystemC
Fixed-point conversion: Convert to fixed-point at build time
Build folder: Project folder

Workflow Advisor - Untitled1.prj

Set the target device and synthesis tool

Workflow: High Level Synthesis
Synthesis tool: Cadence Stratus

Workflow Advisor - Untitled1.prj

Generate SystemC code from the fixed-point MATLAB code.

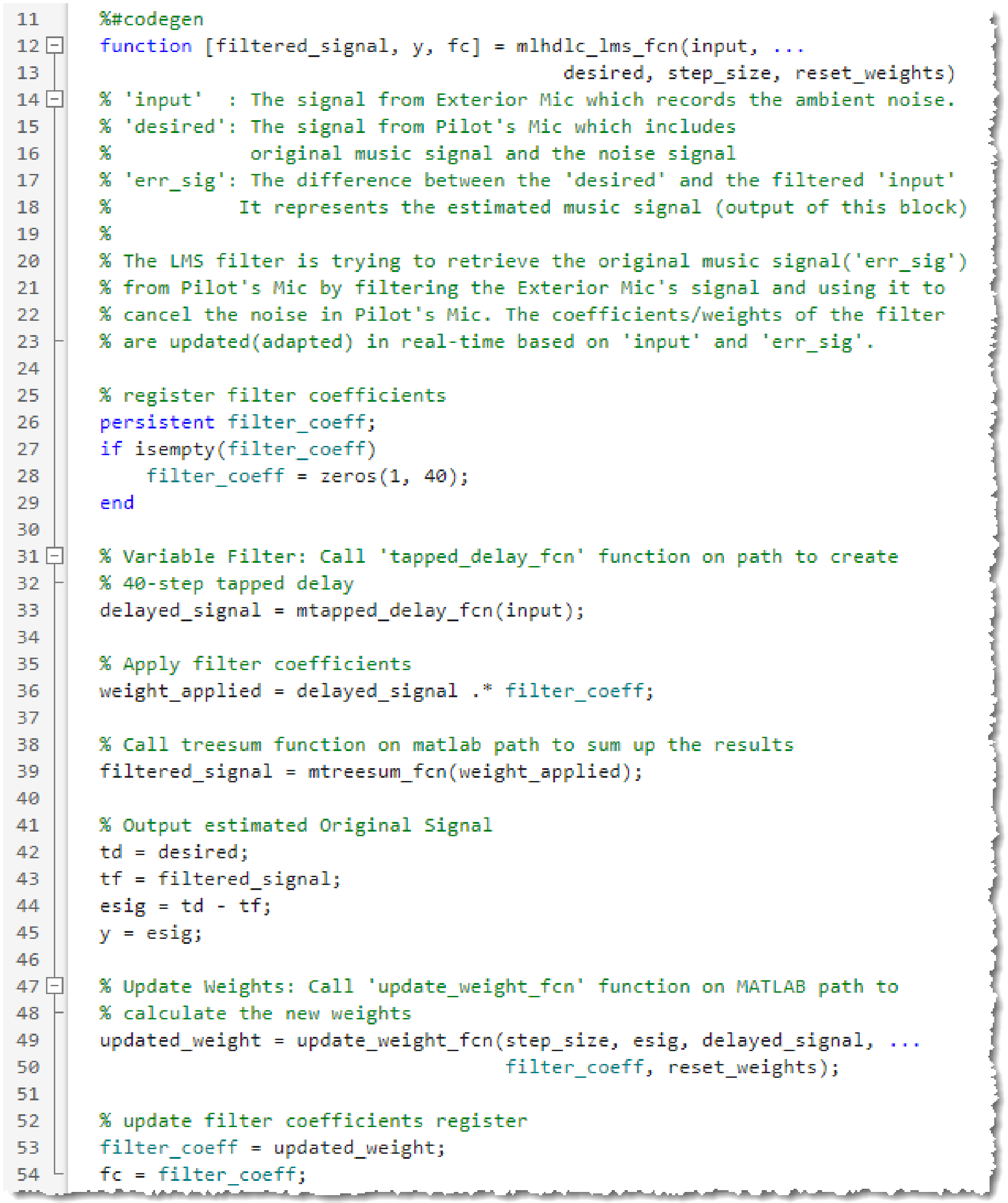
Target | Coding Style | Optimizations | Code

Output Settings
☐ Check HDL conformance

```
### Begin SystemC Code Generation
### Working on mlhdlc_sfir_fixptClass.hpp as mlhdlc_sfir_fixptClass.hpp.
### Working on ml.tcl as ml.tcl.
### Generating HDL Conformance Report mlhdlc_sfir_fixpt_hdl_conformance_report.html
### HDL Conformance check complete with 0 errors, 0 warnings, and 0 messages.
Code generation successful.

### Elapsed Time: '         14.2050' sec(s)
```

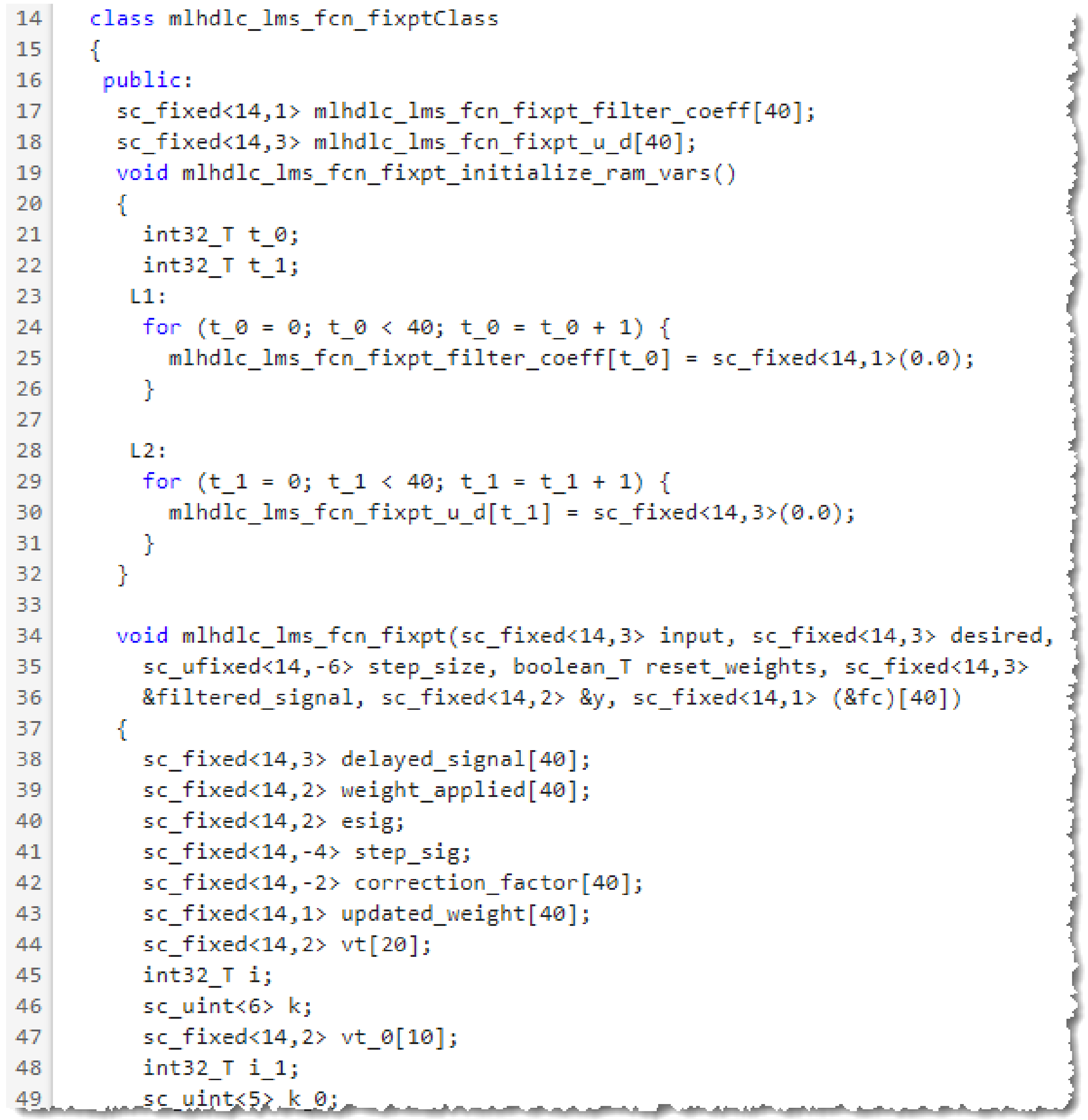
MATLAB HLS 代码特点

- 支持200多个函数和运算符
  - 包括DSP HDL Toolbox和Vision HDL Toolbox中的函数
- 流式数据
  - 标量I/O – 不支持矩阵/结构体
- 静态变量/矩阵维度/类型
- 持久变量用于保存状态 (寄存器或RAM)
  - 先读后写
  - 持久数组可映射到RAM
- 用coder.load从MAT文件加载常量
- 特定函数
  - coder.hdl.loopspec 循环展开
  - coder.hdl.constrainlatency 延迟范围

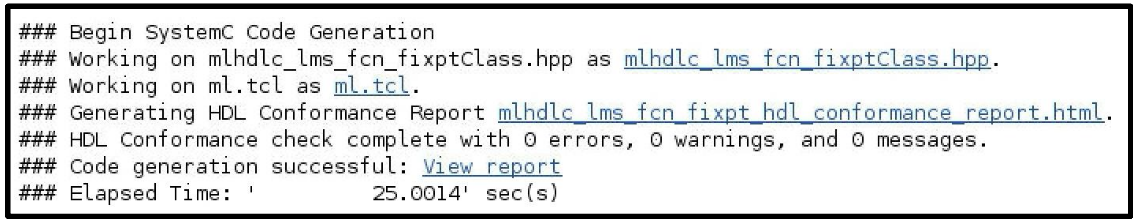


```
%#codegen
function [filtered_signal, y, fc] = mlhdlc_lms_fcn(input, ...
                                        desired, step_size, reset_weights)
% 'input'  : The signal from Exterior Mic which records the ambient noise.
% 'desired': The signal from Pilot's Mic which includes
%            original music signal and the noise signal
% 'err_sig': The difference between the 'desired' and the filtered 'input'
%           It represents the estimated music signal (output of this block)
%
% The LMS filter is trying to retrieve the original music signal('err_sig')
% from Pilot's Mic by filtering the Exterior Mic's signal and using it to
% cancel the noise in Pilot's Mic. The coefficients/weights of the filter
% are updated(adapted) in real-time based on 'input' and 'err_sig'.

% register filter coefficients
persistent filter_coeff;
if isempty(filter_coeff)
    filter_coeff = zeros(1, 40);
end

% Variable Filter: Call 'tapped_delay_fcn' function on path to create
% 40-step tapped delay
delayed_signal = mtapped_delay_fcn(input);

% Apply filter coefficients
weight_applied = delayed_signal .* filter_coeff;

% Call treesum function on matlab path to sum up the results
filtered_signal = mtreesum_fcn(weight_applied);

% Output estimated Original Signal
td = desired;
tf = filtered_signal;
esig = td - tf;
y = esig;

% Update Weights: Call 'update_weight_fcn' function on MATLAB path to
% calculate the new weights
updated_weight = update_weight_fcn(step_size, esig, delayed_signal, ...
                                   filter_coeff, reset_weights);

% update filter coefficients register
filter_coeff = updated_weight;
fc = filter_coeff;
```

LMS Filter 算法生成的 SystemC 代码片段

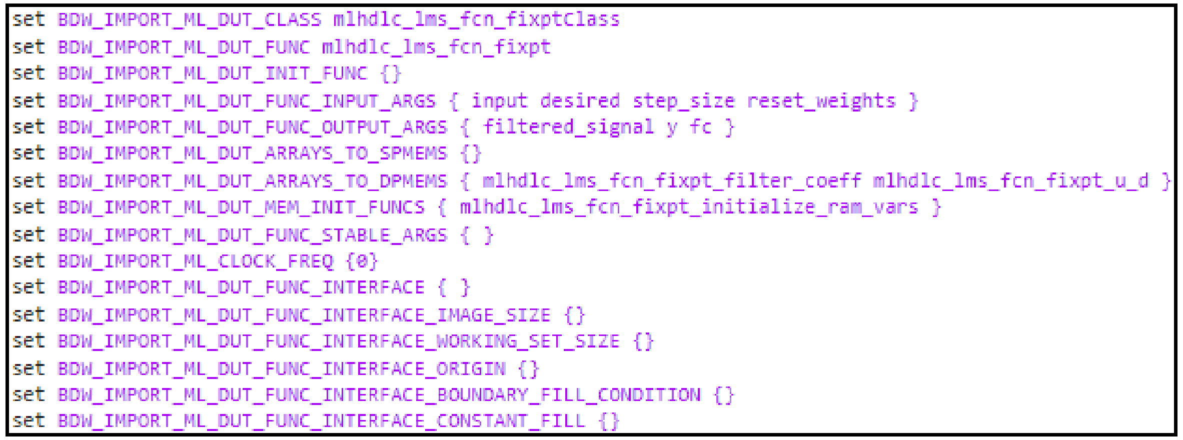


```
class mlhdlc_lms_fcn_fixptClass
{
 public:
  sc_fixed<14,1> mlhdlc_lms_fcn_fixpt_filter_coeff[40];
  sc_fixed<14,3> mlhdlc_lms_fcn_fixpt_u_d[40];
  void mlhdlc_lms_fcn_fixpt_initialize_ram_vars()
  {
    int32_T t_0;
    int32_T t_1;
   L1:
    for (t_0 = 0; t_0 < 40; t_0 = t_0 + 1) {
      mlhdlc_lms_fcn_fixpt_filter_coeff[t_0] = sc_fixed<14,1>(0.0);
    }

   L2:
    for (t_1 = 0; t_1 < 40; t_1 = t_1 + 1) {
      mlhdlc_lms_fcn_fixpt_u_d[t_1] = sc_fixed<14,3>(0.0);
    }
  }

  void mlhdlc_lms_fcn_fixpt(sc_fixed<14,3> input, sc_fixed<14,3> desired,
    sc_ufixed<14,-6> step_size, boolean_T reset_weights, sc_fixed<14,3>
    &filtered_signal, sc_fixed<14,2> &y, sc_fixed<14,1> (&fc)[40])
  {
    sc_fixed<14,3> delayed_signal[40];
    sc_fixed<14,2> weight_applied[40];
    sc_fixed<14,2> esig;
    sc_fixed<14,-4> step_sig;
    sc_fixed<14,-2> correction_factor[40];
    sc_fixed<14,1> updated_weight[40];
    sc_fixed<14,2> vt[20];
    int32_T i;
    sc_uint<6> k;
    sc_fixed<14,2> vt_0[10];
    int32_T i_1;
    sc_uint<5> k_0;
```



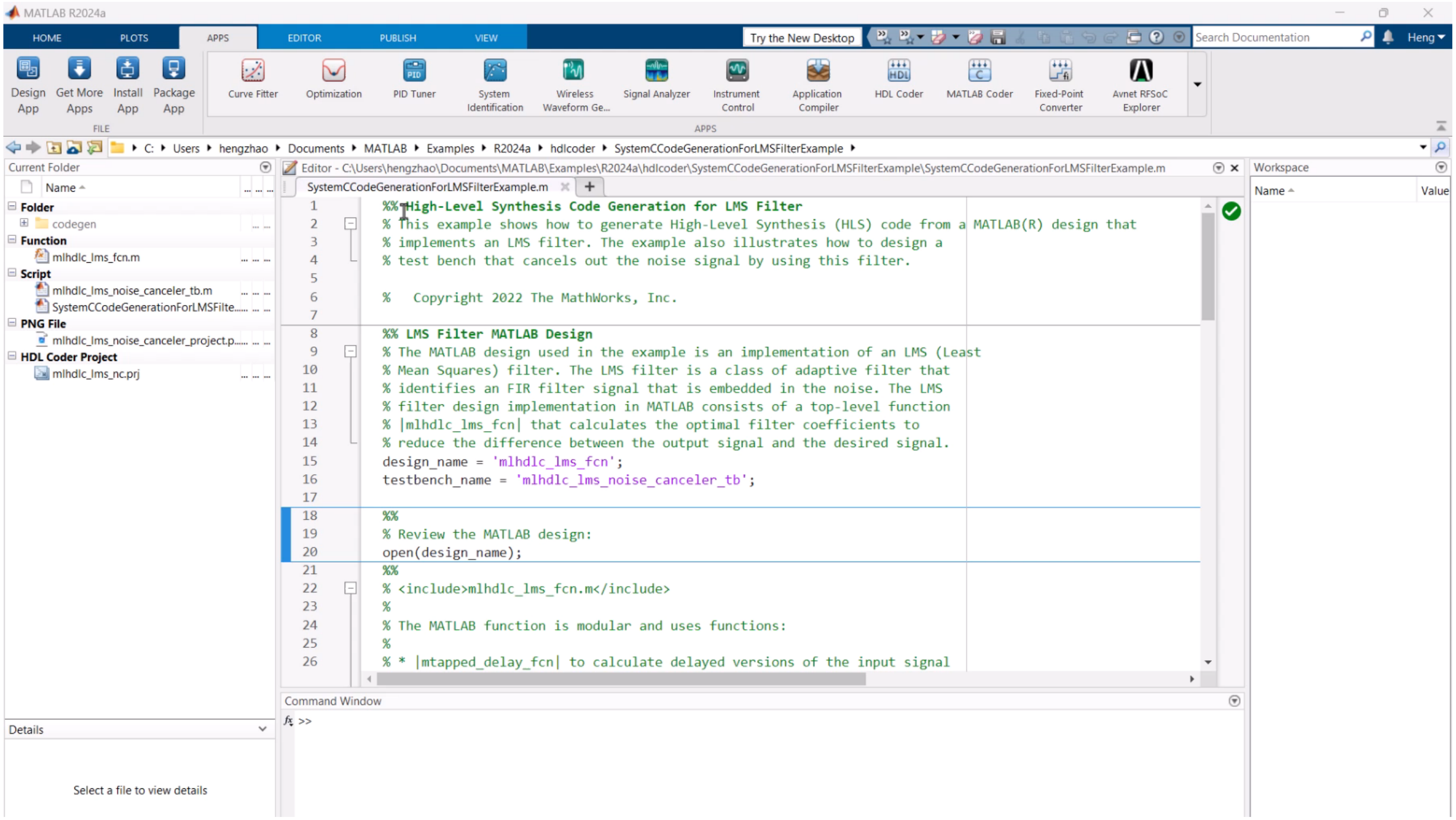
```
### Begin SystemC Code Generation
### Working on mlhdlc_lms_fcn_fixptClass.hpp as mlhdlc_lms_fcn_fixptClass.hpp.
### Working on ml.tcl as ml.tcl.
### Generating HDL Conformance Report mlhdlc_lms_fcn_fixpt_hdl_conformance_report.html.
### HDL Conformance check complete with 0 errors, 0 warnings, and 0 messages.
### Code generation successful: View report
### Elapsed Time: '          25.0014' sec(s)
```

ml.tcl



```
set BDW_IMPORT_ML_DUT_CLASS mlhdlc_lms_fcn_fixptClass
set BDW_IMPORT_ML_DUT_FUNC mlhdlc_lms_fcn_fixpt
set BDW_IMPORT_ML_DUT_INIT_FUNC {}
set BDW_IMPORT_ML_DUT_FUNC_INPUT_ARGS { input desired step_size reset_weights }
set BDW_IMPORT_ML_DUT_FUNC_OUTPUT_ARGS { filtered_signal y fc }
set BDW_IMPORT_ML_DUT_ARRAYS_TO_SPMEMS {}
set BDW_IMPORT_ML_DUT_ARRAYS_TO_DPMEMS { mlhdlc_lms_fcn_fixpt_filter_coeff mlhdlc_lms_fcn_fixpt_u_d }
set BDW_IMPORT_ML_DUT_MEM_INIT_FUNCS { mlhdlc_lms_fcn_fixpt_initialize_ram_vars }
set BDW_IMPORT_ML_DUT_FUNC_STABLE_ARGS { }
set BDW_IMPORT_ML_CLOCK_FREQ {0}
set BDW_IMPORT_ML_DUT_FUNC_INTERFACE { }
set BDW_IMPORT_ML_DUT_FUNC_INTERFACE_IMAGE_SIZE {}
set BDW_IMPORT_ML_DUT_FUNC_INTERFACE_WORKING_SET_SIZE {}
set BDW_IMPORT_ML_DUT_FUNC_INTERFACE_ORIGIN {}
set BDW_IMPORT_ML_DUT_FUNC_INTERFACE_BOUNDARY_FILL_CONDITION {}
set BDW_IMPORT_ML_DUT_FUNC_INTERFACE_CONSTANT_FILL {}
```

MATLAB EXPO



MATLAB R2024a
HOME PLOTS APPS EDITOR PUBLISH VIEW
Try the New Desktop
Search Documentation
Heng
Design App
Get More Apps
Install App
Package App
FILE
Curve Fitter
Optimization
PID Tuner
System Identification
Wireless Waveform Ge...
Signal Analyzer
Instrument Control
Application Compiler
HDL Coder
MATLAB Coder
Fixed-Point Converter
Avnet RFSoC Explorer
APPS
C: ▸ Users ▸ hengzhao ▸ Documents ▸ MATLAB ▸ Examples ▸ R2024a ▸ hdlcoder ▸ SystemCCodeGenerationForLMSFilterExample ▸
Current Folder
Name
Folder
codegen
Function
mlhdlc_lms_fcn.m
Script
mlhdlc_lms_noise_canceler_tb.m
SystemCCodeGenerationForLMSFilte...
PNG File
mlhdlc_lms_noise_canceler_project.p...
HDL Coder Project
mlhdlc_lms_nc.prj
Details
Select a file to view details
Editor - C:\Users\hengzhao\Documents\MATLAB\Examples\R2024a\hdlcoder\SystemCCodeGenerationForLMSFilterExample\SystemCCodeGenerationForLMSFilterExample.m
SystemCCodeGenerationForLMSFilterExample.m
Workspace
Name
Value
Command Window

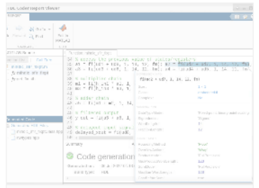
```
%% High-Level Synthesis Code Generation for LMS Filter
% This example shows how to generate High-Level Synthesis (HLS) code from a MATLAB(R) design that
% implements an LMS filter. The example also illustrates how to design a
% test bench that cancels out the noise signal by using this filter.

%   Copyright 2022 The MathWorks, Inc.

%% LMS Filter MATLAB Design
% The MATLAB design used in the example is an implementation of an LMS (Least
% Mean Squares) filter. The LMS filter is a class of adaptive filter that
% identifies an FIR filter signal that is embedded in the noise. The LMS
% filter design implementation in MATLAB consists of a top-level function
% |mlhdlc_lms_fcn| that calculates the optimal filter coefficients to
% reduce the difference between the output signal and the desired signal.
design_name = 'mlhdlc_lms_fcn';
testbench_name = 'mlhdlc_lms_noise_canceler_tb';

%%
% Review the MATLAB design:
open(design_name);
%%
% <include>mlhdlc_lms_fcn.m</include>
%
% The MATLAB function is modular and uses functions:
%
% * |mtapped_delay_fcn| to calculate delayed versions of the input signal
```

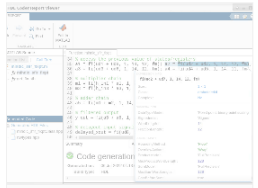
```
>>
```

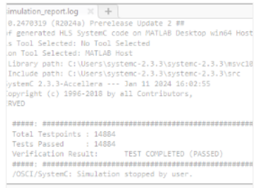
SystemC 代码生成示例

MATLAB HLS 代码风格; 信号处理和图像处理应用示例

起步入门


Get Started with MATLAB to High-Level Synthesis Workflow Using the

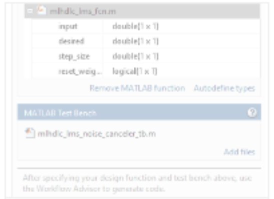
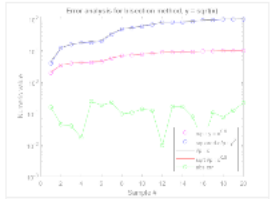

Get Started with MATLAB to High-Level Synthesis Workflow Using HDL Coder

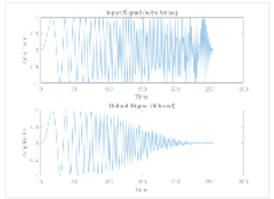


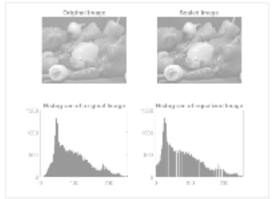
```
imulation_report.log
0.2470319 (R2024a) Prerelease Update 2 ##
f generated HLS SystemC code on MATLAB Desktop win64 Host
s Tool Selected: No Tool Selected
on Tool Selected: MATLAB Host
Library path: C:\Users\systemc-2.3.3\systemc-2.3.3\msvc10
Include path: C:\Users\systemc-2.3.3\systemc-2.3.3\src
ystemC 2.3.3-Accellera --- Jan 11 2024 16:02:55
opyright (c) 1996-2018 by all Contributors,
RVED

#####: ##################################################
Total Testpoints : 14884
Tests Passed     : 14884
Verification Result:     TEST COMPLETED (PASSED)
#####: ##################################################
/OSCI/SystemC: Simulation stopped by user.
```

Verify Generated HLS Code Using MATLAB Desktop Host

算法设计

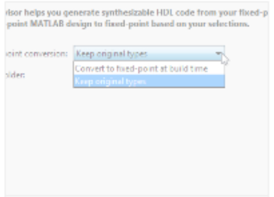


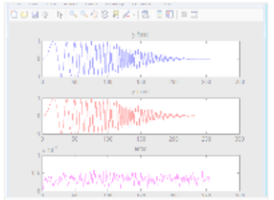


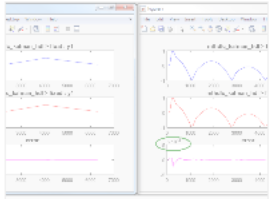
SystemC Code Generation for Contrast Adjustment

Generate SystemC code from a MATLAB® design that adjusts image contrast by linearly scaling pixel values.

定点化


Working with Fixed-Point Code

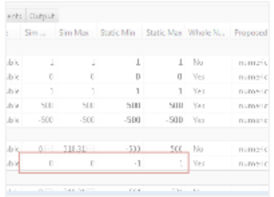

Floating-Point to Fixed-Point Conversion


Fixed-Point Type Conversion and Refinement

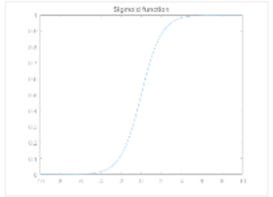
Achieve your desired numerical accuracy when converting fixed-point MATLAB® code to floating-point code using the HDL Workflow

Working with Generated Fixed-Point Files

Work with the files generated during floating-point to fixed-point conversion.

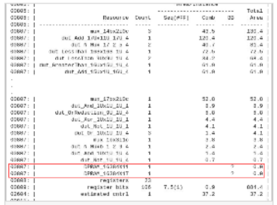

Fixed-Point Type Conversion and Derived Ranges

Achieve your desired numerical accuracy when converting fixed-point MATLAB® code to floating-point code using static range

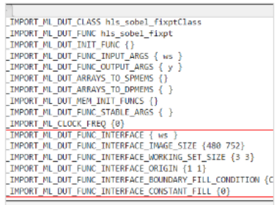

Generate HDL-Compatible Lookup Table Function Replacements Using...

MATLAB® code generation from a floating-point MATLAB design that is not ready for code generation. Use `coder.approximate` function to

速度/面积优化


Map Persistent Variables to RAM for Histogram Equalization

Use the RAM mapping optimization in HDL Coder to map persistent matrix variables to block RAMs in hardware.



```
_IMPORT_ML_DUT_CLASS hls_sobel_fixptClass
_IMPORT_ML_DUT_FUNC hls_sobel_fixpt
_IMPORT_ML_DUT_INIT_FUNC {}
_IMPORT_ML_DUT_FUNC_INPUT_ARGS { ws }
_IMPORT_ML_DUT_FUNC_OUTPUT_ARGS { y }
_IMPORT_ML_DUT_ARRAYS_TO_SPMEMS {}
_IMPORT_ML_DUT_ARRAYS_TO_DPMEMS { }
_IMPORT_ML_DUT_MEM_INIT_FUNCS {}
_IMPORT_ML_DUT_FUNC_STABLE_ARGS { }
_IMPORT_ML_CLOCK_FREQ {0}
_IMPORT_ML_DUT_FUNC_INTERFACE { ws }
_IMPORT_ML_DUT_FUNC_INTERFACE_IMAGE_SIZE {480 752}
_IMPORT_ML_DUT_FUNC_INTERFACE_WORKING_SET_SIZE {3 3}
_IMPORT_ML_DUT_FUNC_INTERFACE_ORIGIN {1 1}
_IMPORT_ML_DUT_FUNC_INTERFACE_BOUNDARY_FILL_CONDITION {C
_IMPORT_ML_DUT_FUNC_INTERFACE_CONSTANT_FILL {0}
```

Create a Line Buffer Interface for SystemC Code Generation

Generate HLS code from MATLAB code for Sobel filter.

MATLAB EXPO

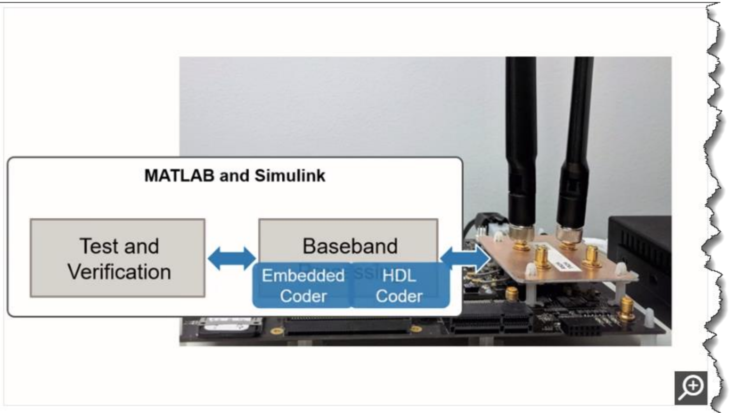
FPGA、ASIC 和 SoC 开发

总览 | 原型设计 | 产品级设计与验证 | 快速入门 | 合作伙伴解决方案 ▾

无线应用

您可以在设计中逐步添加实时硬件元素，无论是使用实时无线 (OTA) 输入/输出对算法进行仿真，还是在 FPGA 或 SoC 软件无线电平台或自定义板上进行完整部署。

Wireless HDL Toolbox 提供经硬件验证的无线设计 IP 模块和子系统，帮助您快速上手。IP 包含若干示例，演示如何将 MATLAB 中的算法设计逐步转化为 Simulink 中的无线系统实现模型。所有 IP 都经过定点量化，因此，您可以使用 Fixed-Point Designer™ 管理您添加的自定义逻辑的量化，然后再使用 HDL Coder 进行部署。

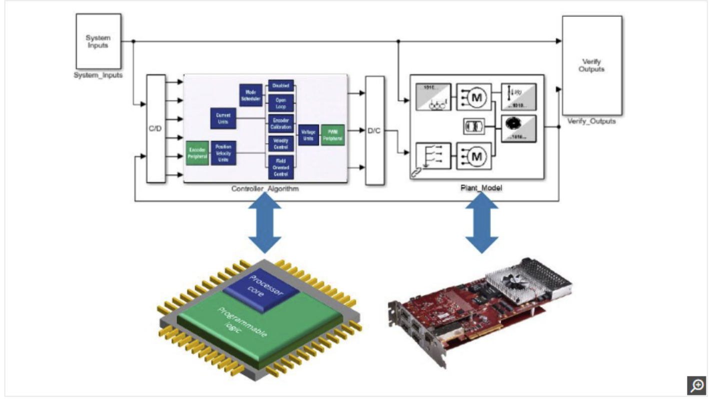


先进行系统级设计和仿真，然后逐步添加真实硬件元素，直到实现完整部署，以用于现场测试。

电机和电力电子控制应用

您可以研究基于 FPGA 的硬件上所运行的控制算法的性能，或者使用基于 FPGA 的硬件在环对被控对象模型进行加速。HDL Coder 广泛支持数学及三角函数的定点或本机浮点 (9:19) HDL 代码生成，让您能够从 Simulink 模型直达硬件。

如果您正在研究如何进行算法分区以用于 SoC 部署，您可以搜索和仿真分区策略以评估性能，之后再部署到原型平台。您可以部署到预配置套件、Speedgoat 硬件 (7:53)或您的自定义板。



将电机和电力电子控制算法部署到 FPGA 硬件，并使用 FPGA 加速器（例如 Speedgoat I/O 模块）对硬件在环被控对象模型进行加速。

视频和图像处理应用

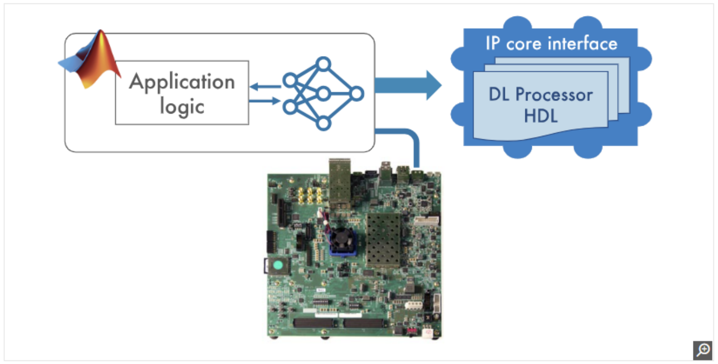
将基于 FPGA 的平台连接到 MATLAB 和 Simulink 后，您可以为平台上运行的算法自动生成 HDL 和 C 代码，以进行视觉算法原型设计。同样，您可以使用经硬件验证的视觉处理模块构建实现模型，以仿真硬件行为，例如像素流、基于近邻的算法、外部内存访问以及控制信号。

您可以将模型部署到带摄像头的现成 FPGA 评估套件。或者，如果您的硬件团队可以为平台提供支持，您也可以直接从 MATLAB 和 Simulink 部署原型。

FPGA 原型板上运行的去雾算法。

深度学习推断

只需几条 MATLAB 命令，即可在 FPGA 和 SoC 板上设计网络原型以加速深度学习推断。随后，您可以分析 FPGA 推断的性能、调整网络、量化为定点、重新部署，从而在 MATLAB 内进行网络迭代。最终，您可以生成不依赖于特定目标的 HDL IP 核，交给硬件团队进行实现。

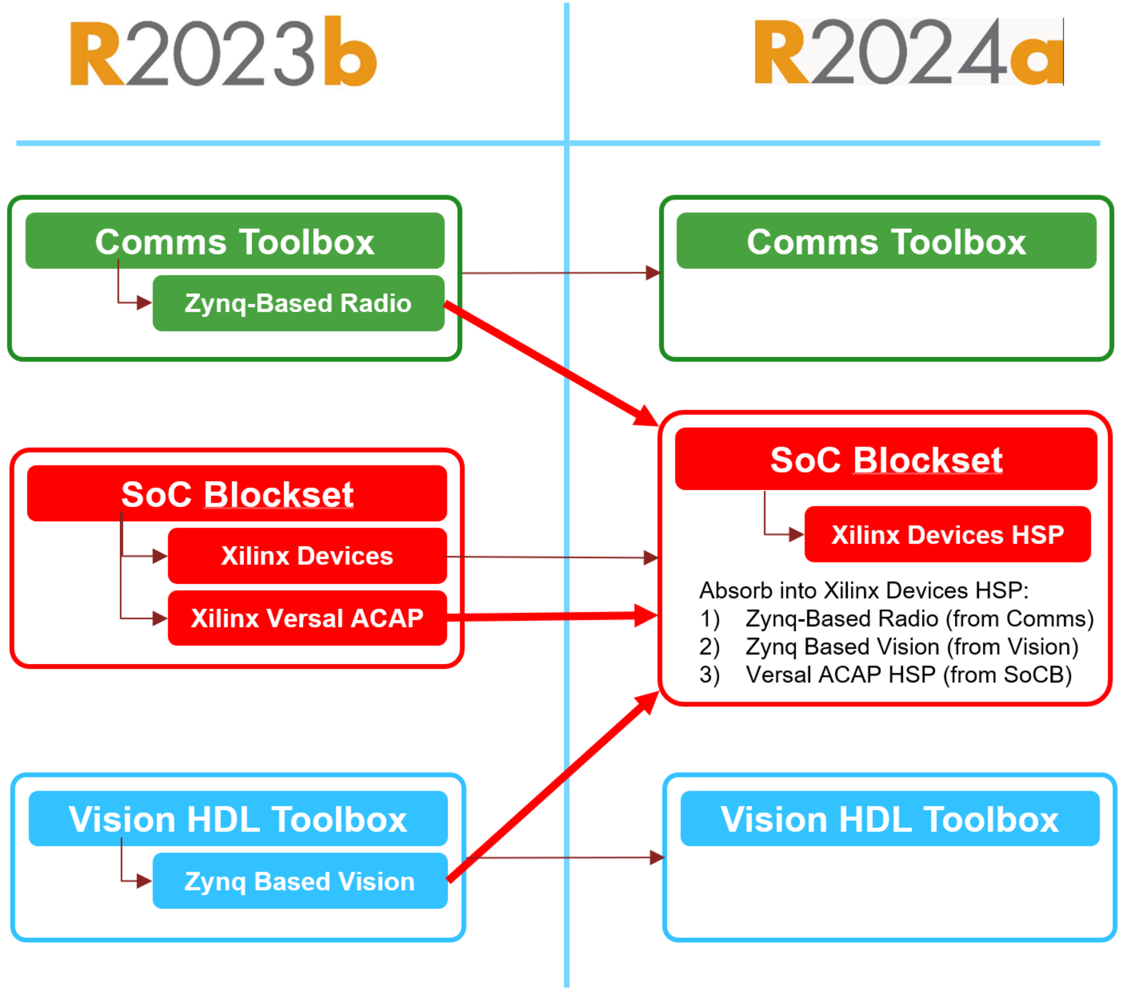


直接从 MATLAB 对原型硬件运行基于 FPGA 的深度学习推断，随后生成深度学习 HDL IP 核，以在任意 FPGA 或 ASIC 上进行部署。

使用 MATLAB 进行 FPGA、ASIC 及 SoC 开发

合并简化 AMD-Xilinx 硬件支持包

- 将以下支持包整合到现有的 SoC Blockset Support Package for Xilinx Devices 支持包：
  - Communications Toolbox Support Package for Xilinx Zynq-Based Radio
  - Vision HDL Toolbox Support Package for Xilinx Zynq-Based Hardware
  - SoC Blockset Support Package for AMD-Xilinx Versal ACAP Devices
- 支持的每种 Xilinx SoC 板卡使用单一 Linux 镜像
  - R2024a之前：同一硬件板卡需要3个不同的 Linux 镜像文件，支持3种不同的硬件支持包（HSPs）
  - R2024a之后：每个板卡一个单一镜像，无需更换 SD 卡



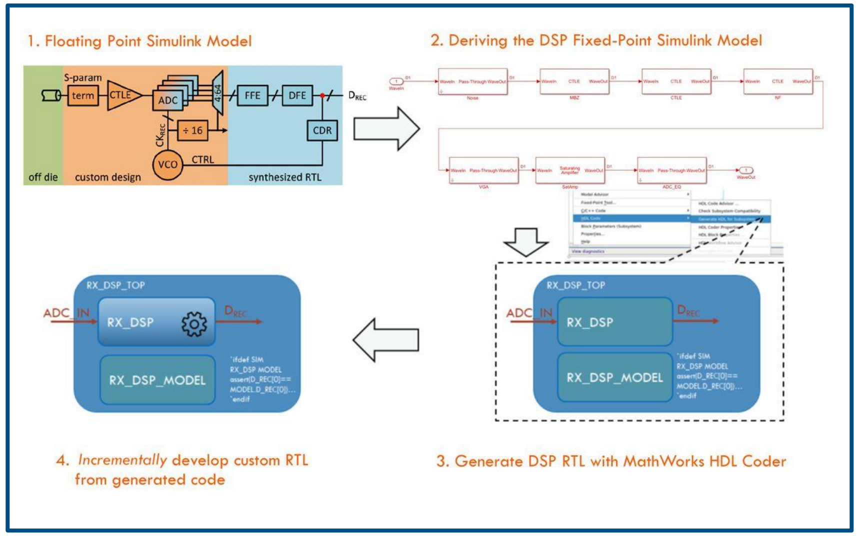
Rambus 使用 HDL Coder 高级综合开发 ASIC 数字信号处理模块

使用 Simulink 和 HDL Coder，Rambus 为 DSP 芯片生成了可综合的 RTL 代码。随后以该 RTL 代码为参考，从中逐步开发 PPA 优化的定制实现。采用这种方法，验证和后端工程师得以提前数月开始他们的工作。

关键成果和优势：

- 设计流程从一年缩短到三个月，并且通过使用 HDL 代码生成，开发效率得到显著提升，在最终实现中保留了80%的生成代码。
- 验证环境的开发从自动生成的 RTL 代码开始，与定制 RTL 开发同时并行进行。
- 在设计的不同迭代之间或某个迭代的不同实现之间切换，对于 Simulink 模型所需的更改非常少。

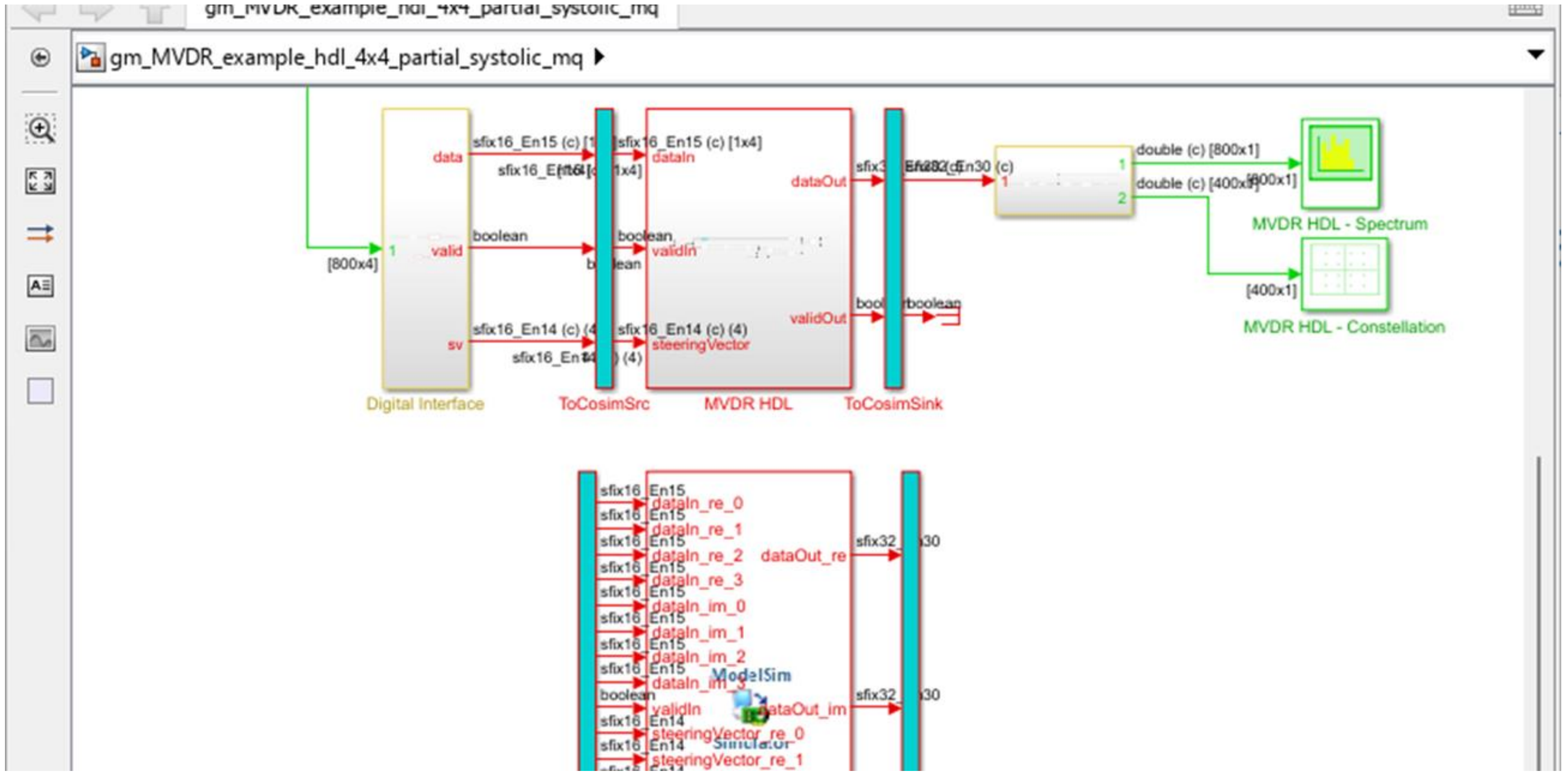
Rambus Develops DSP Blocks for ASICs Using High-Level Synthesis with HDL Coder


1. Floating Point Simulink Model
S-param
term
CTLE
ADC
FFE
DFE
D_REC
CK_REC
÷ 16
CDR
VCO
CTRL
off die
custom design
synthesized RTL
2. Deriving the DSP Fixed-Point Simulink Model
RX_DSP_TOP
ADC_IN
RX_DSP
RX_DSP_MODEL
D_REC
4. Incrementally develop custom RTL from generated code
3. Generate DSP RTL with MathWorks HDL Coder

使用 HDL 代码生成、RTL 验证和定制化的
数字信号处理 ASIC 工作流程

“通常情况下，在基于规范的第一个版本 RTL 准备好之前，验证和后端团队无法开始集成，而这个初始版本可能还存在缺陷。采用基于 HDL Coder 的工作流程，可以从系统级验证的模型中一键生成 HDL，这样不仅可以节省数月的时间，而且还能为团队提供一个高质量的起点。”

- Ehud Nir, Rambus 数字工程部总监



自动生成 Testbench加速验证和简化调试

验证带来的挑战需要更高效的生产力工具

66% | of ASIC projects behind schedule

76% | of ASIC projects require re-spins

56% | of project time is consumed by verification for ASIC projects (on average)

70% | of FPGA projects are behind Schedule

84% | of FPGA designs have bugs that escape into production

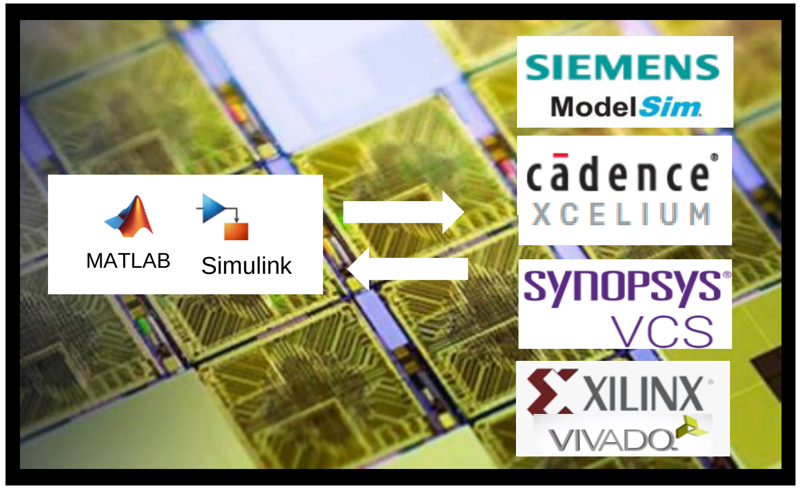
~1:1 | Ratio of Design vs Verification Engineers

- 尽管在验证上投入了巨大的努力和成本，但错误、返工和项目延误的问题仍然司空见惯。
- 解决这一挑战需要更好的工具和工作流程。

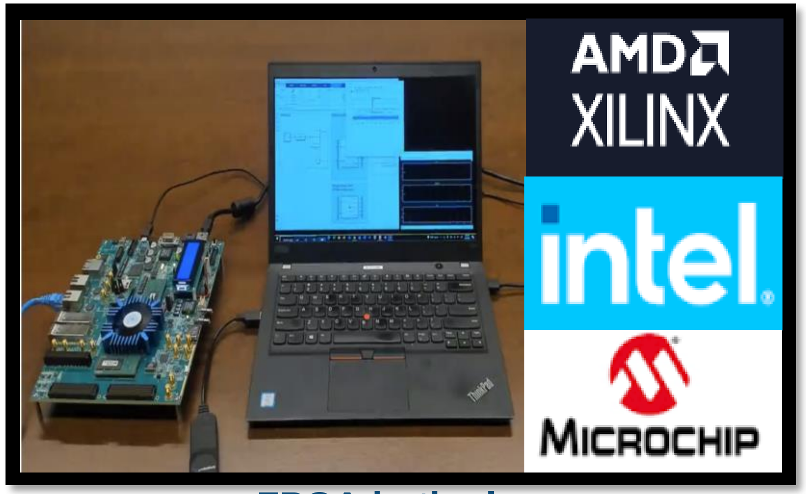
Source: 2022 Wilson Research Group Functional Verification Study

HDL Verifier

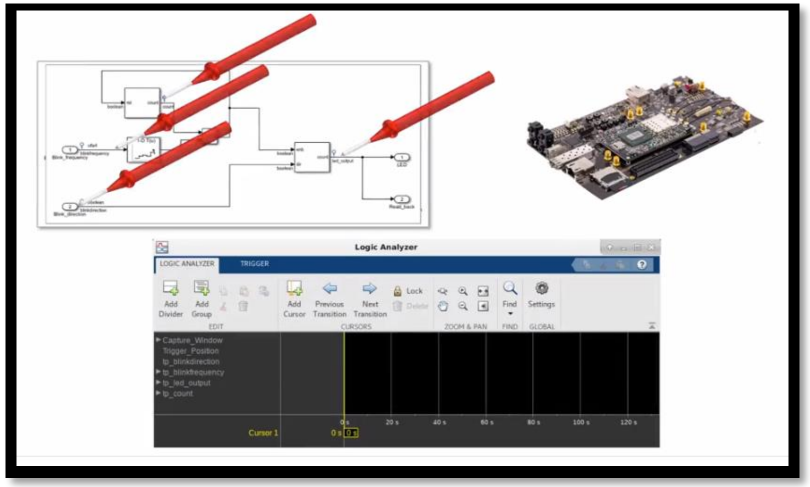
Test and verify Verilog and VHDL using HDL simulators and FPGA boards



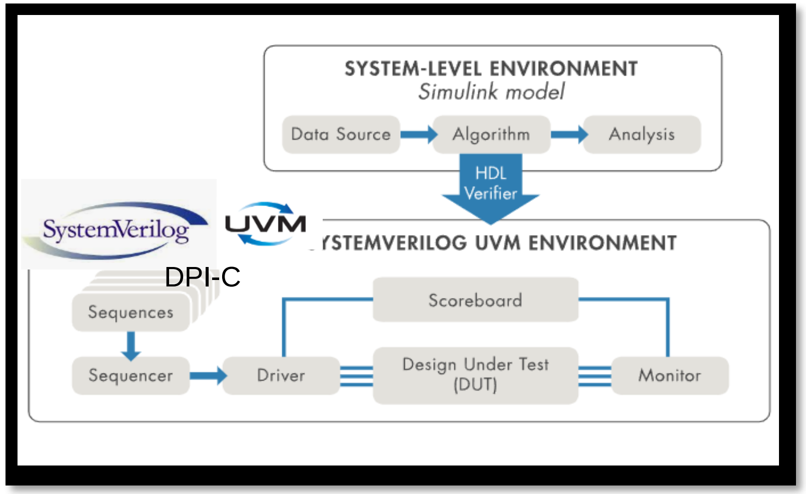
HDL Co-simulation



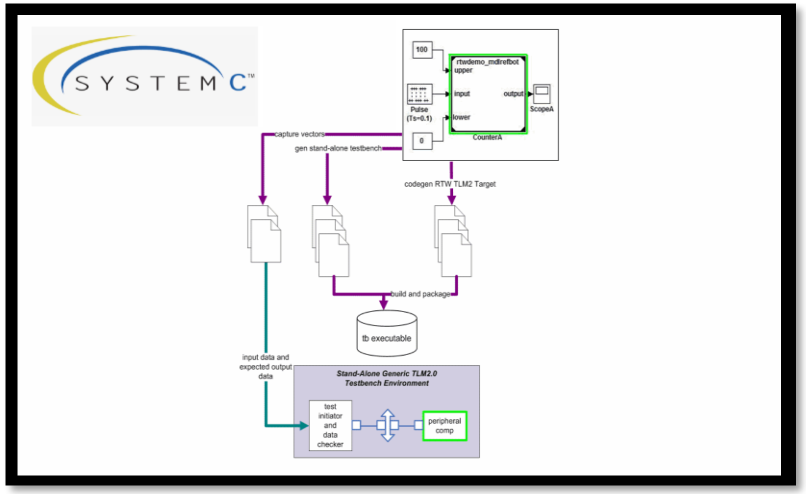
FPGA-in-the-loop



FPGA Debugging (Data Capture and AXI Manager)

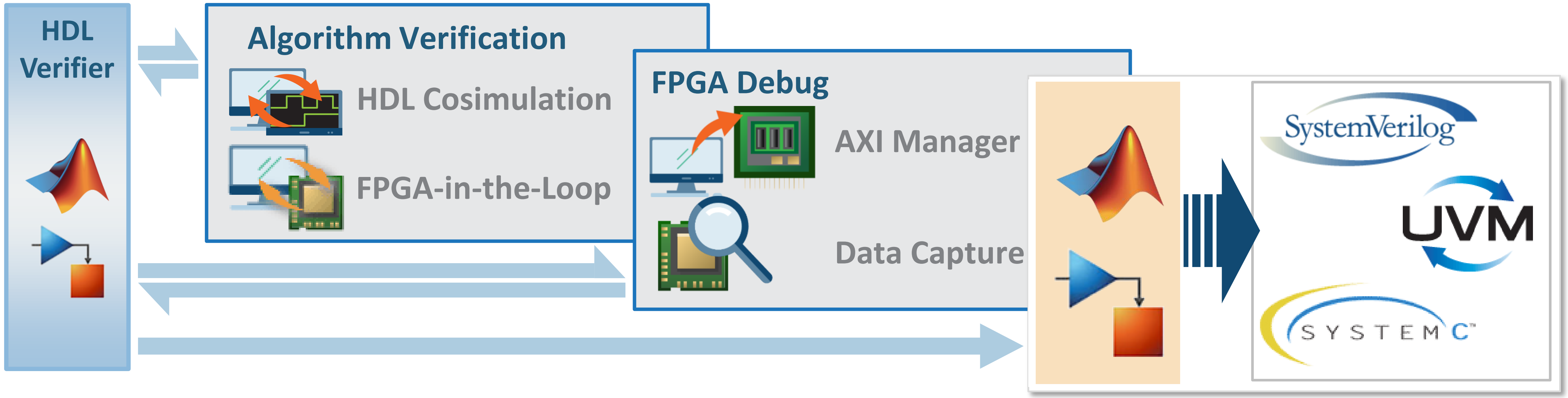


UVM Testbench / SystemVerilog DPI-C Test Components Generation


SYSTEM C
capture vectors
gen stand-alone testbench
codegen RTW TLM2 Target
build and package
tb executable
input data and expected output data
Stand-Alone Generic TLM2.0 Testbench Environment
test initiator and data checker
peripheral comp
CounterA
ScopeA

System-C TLM 2.0 Components Generation

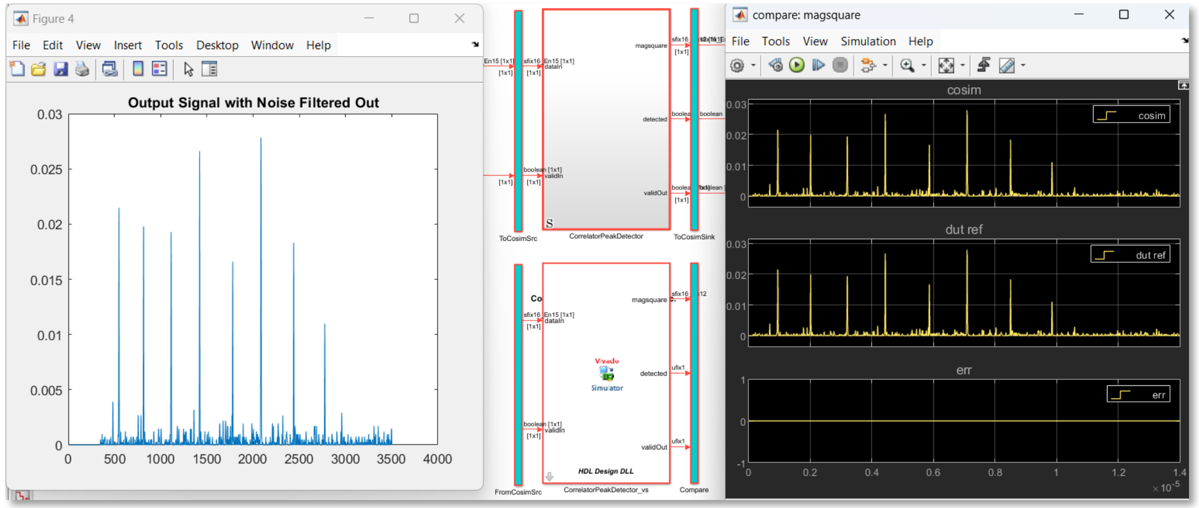
ASIC Testbench for HDL Verifier

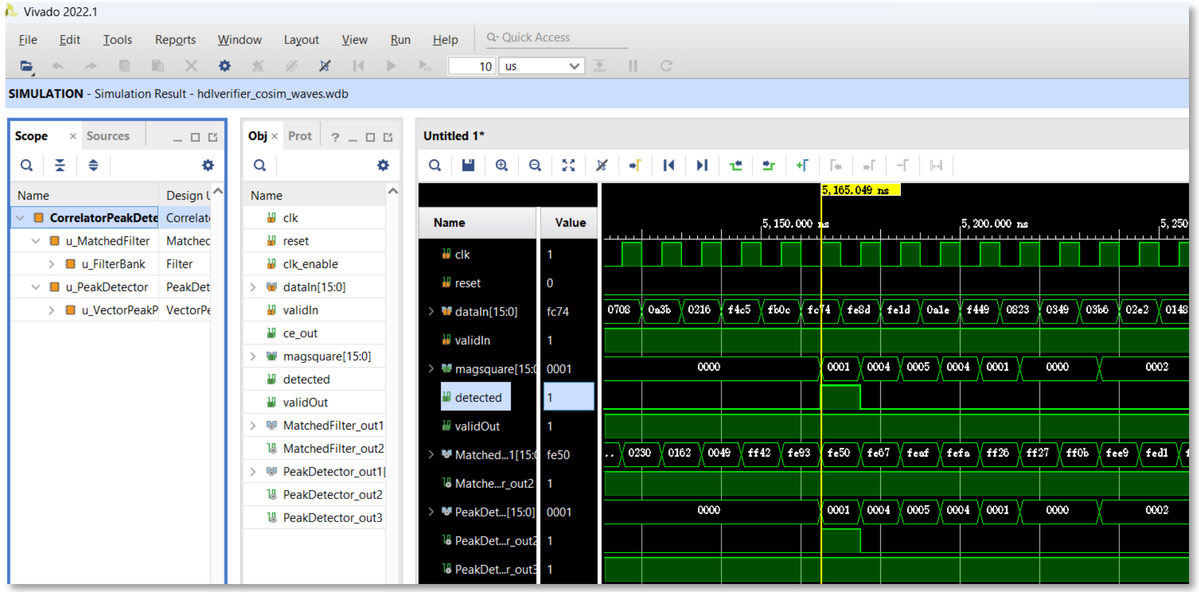


23b: ASIC Testbench for HDL Verifier

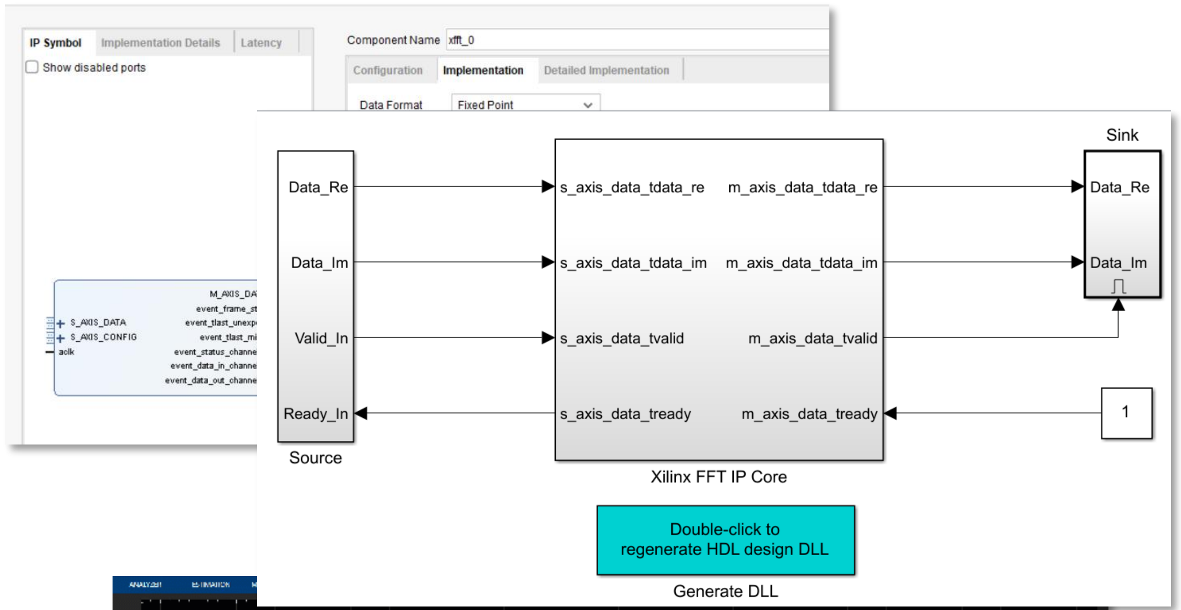
Vivado 联合仿真

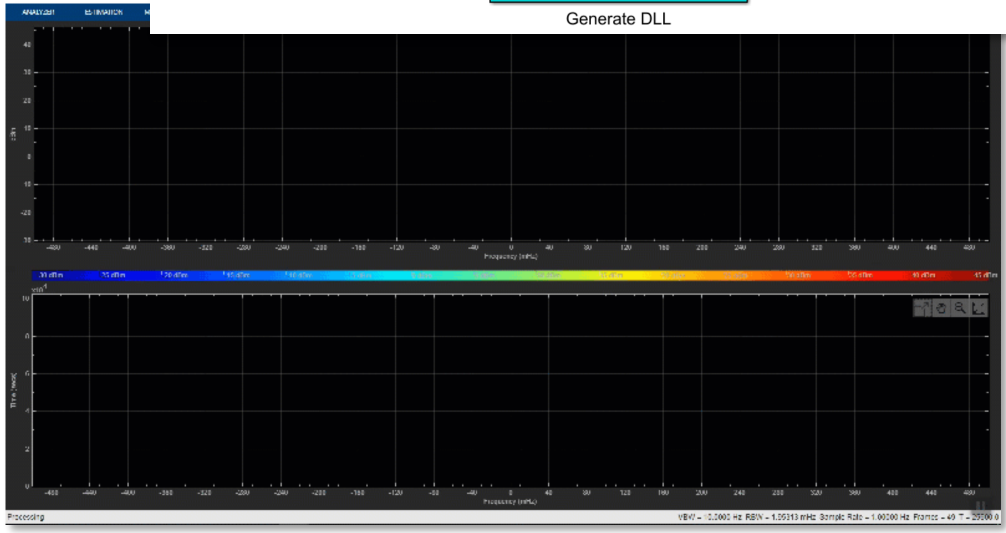
与 Vivado Simulator 联合仿真





在 Simulink 模型中使用 Vivado IP 核仿真



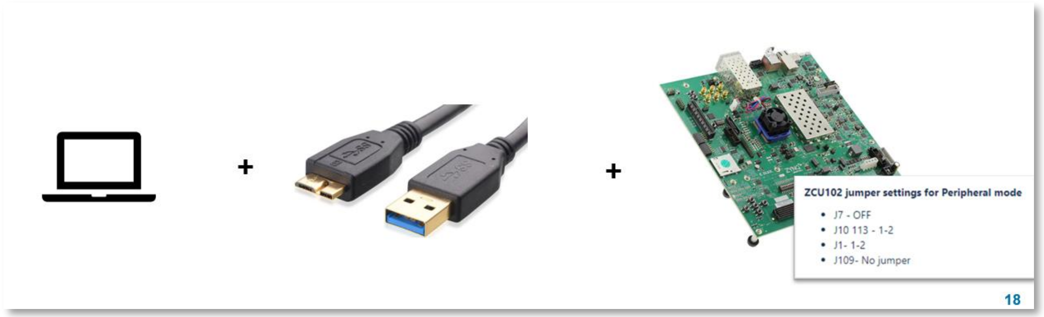


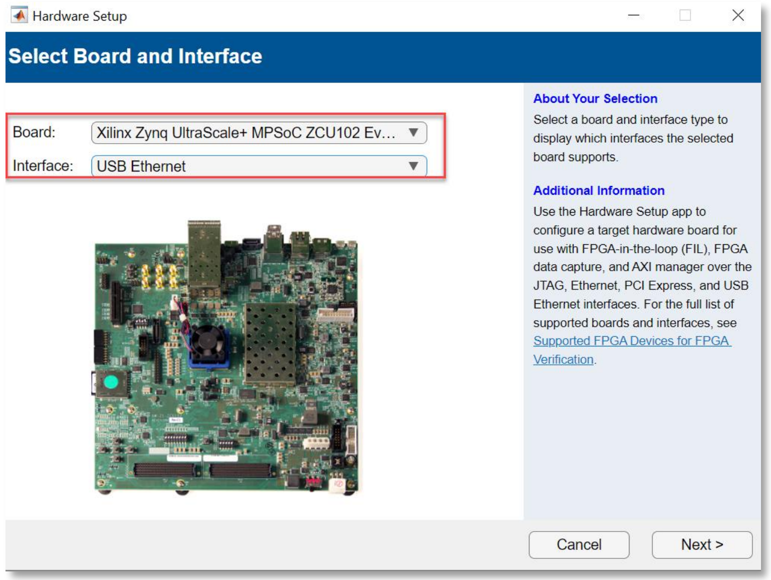
MATLAB EXPO

MATLAB 通过高速 USB 接口连接开发板通信

支持超高速 USB 3.0 和高速 2.0 接口

- 即插即用
- USB 以太网 DHCP 网络自动配置
- 显著的性能提升
- 集成到硬件配置向导




Hardware Setup

Select Board and Interface

Board: Xilinx Zynq UltraScale+ MPSoC ZCU102 Ev...

Interface: USB Ethernet

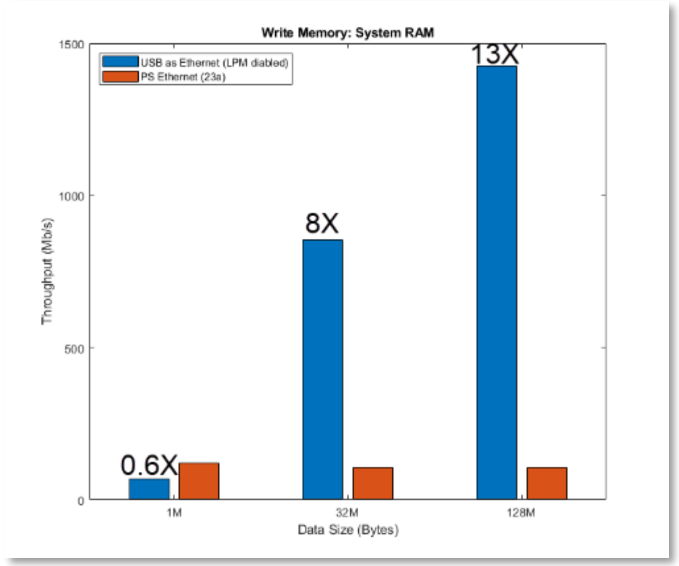
About Your Selection

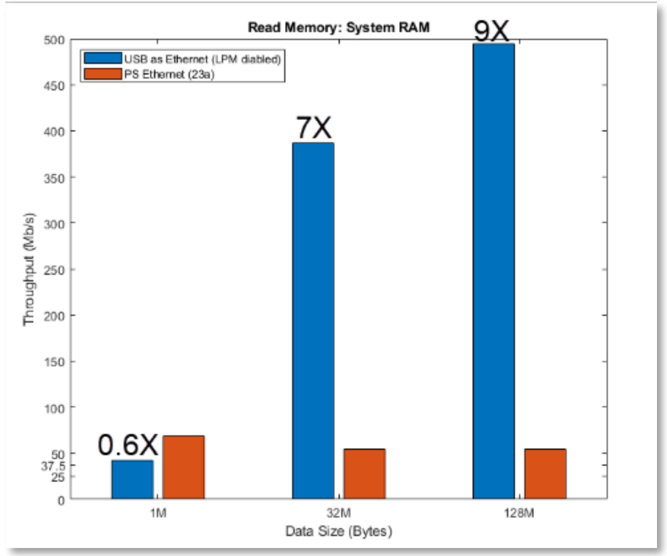
Select a board and interface type to display which interfaces the selected board supports.

Additional Information

Use the Hardware Setup app to configure a target hardware board for use with FPGA-in-the-loop (FIL), FPGA data capture, and AXI manager over the JTAG, Ethernet, PCI Express, and USB Ethernet interfaces. For the full list of supported boards and interfaces, see Supported FPGA Devices for FPGA Verification.

Cancel | Next >

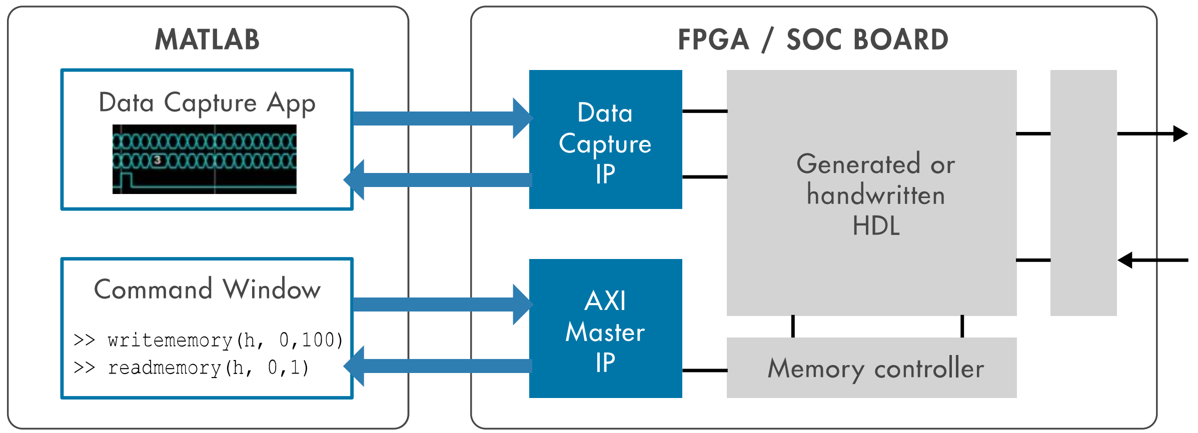


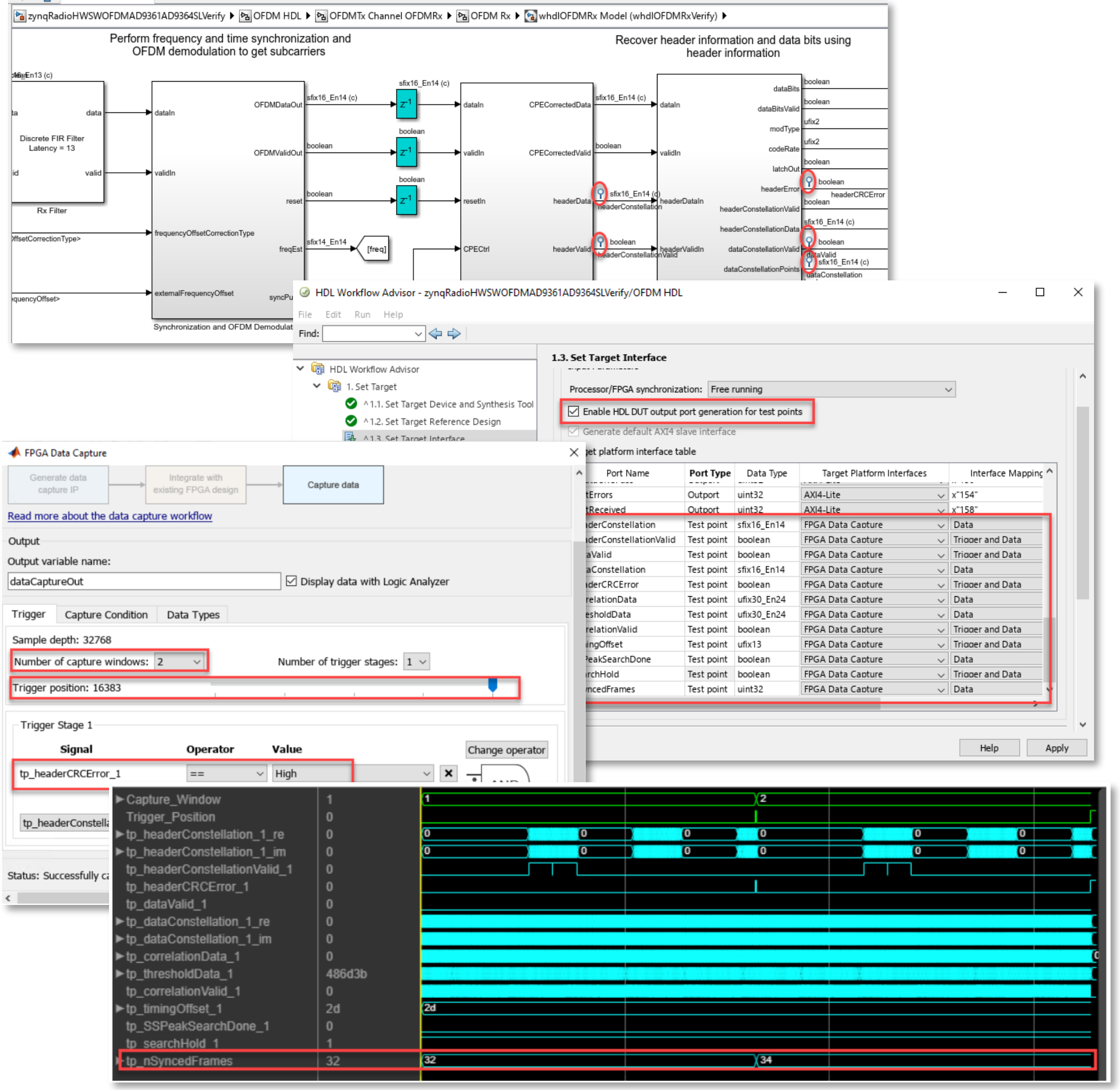


For large data up to 13x Faster performance

MATLAB 连接 FPGA/SoC 调试

- FPGA 正常模式运行下直接在 MATLAB 或 Simulink 中调试 FPGA 内部信号
- 在 MATLAB 或 Simulink 中读写开发板上的 DDR
- 在 MATLAB 或 Simulink 中进行数据可视化并进一步分析




zynqRadioHWSWOFDMAD9361AD9364SLVerify ▸ OFDM HDL ▸ OFDMTx Channel OFDMRx ▸ OFDM Rx ▸ whdlOFDMRx Model (whdlOFDMRxVerify)
Perform frequency and time synchronization and OFDM demodulation to get subcarriers
Recover header information and data bits using header information
Discrete FIR Filter Latency = 13
Rx Filter
Synchronization and OFDM Demodulation

HDL Workflow Advisor - zynqRadioHWSWOFDMAD9361AD9364SLVerify/OFDM HDL

- HDL Workflow Advisor
  - 1. Set Target
    - ✓ 1.1. Set Target Device and Synthesis Tool
    - ✓ 1.2. Set Target Reference Design
    - 1.3. Set Target Interface

1.3. Set Target Interface

Processor/FPGA synchronization: Free running

☑ Enable HDL DUT output port generation for test points

☑ Generate default AXI4 slave interface

Target platform interface table

Port Name	Port Type	Data Type	Target Platform Interfaces	Interface Mapping
tErrors	Outport	uint32	AXI4-Lite	x"154"
tReceived	Outport	uint32	AXI4-Lite	x"158"
aderConstellation	Test point	sfix16_En14	FPGA Data Capture	Data
aderConstellationValid	Test point	boolean	FPGA Data Capture	Trigger and Data
aValid	Test point	boolean	FPGA Data Capture	Trigger and Data
aConstellation	Test point	sfix16_En14	FPGA Data Capture	Data
aderCRCError	Test point	boolean	FPGA Data Capture	Trigger and Data
relationData	Test point	ufix30_En24	FPGA Data Capture	Data
esholdData	Test point	ufix30_En24	FPGA Data Capture	Data
relationValid	Test point	boolean	FPGA Data Capture	Trigger and Data
ingOffset	Test point	ufix13	FPGA Data Capture	Trigger and Data
PeakSearchDone	Test point	boolean	FPGA Data Capture	Data
rchHold	Test point	boolean	FPGA Data Capture	Trigger and Data
ncedFrames	Test point	uint32	FPGA Data Capture	Data

FPGA Data Capture

Generate data capture IP → Integrate with existing FPGA design → Capture data

Read more about the data capture workflow

Output

Output variable name: dataCaptureOut ☑ Display data with Logic Analyzer

Trigger | Capture Condition | Data Types

Sample depth: 32768

Number of capture windows: 2   Number of trigger stages: 1

Trigger position: 16383

Trigger Stage 1

Signal	Operator	Value
tp_headerCRCError_1	==	High

Status: Successfully ca

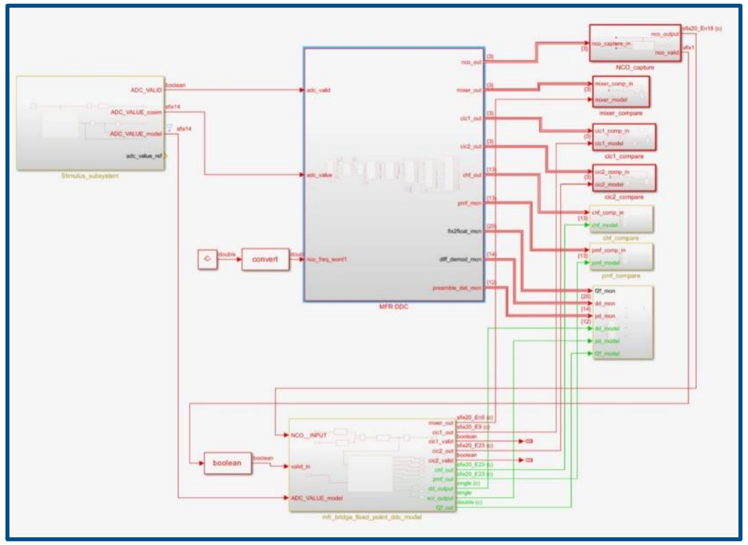
```
Capture_Window                    1
Trigger_Position                  0
tp_headerConstellation_1_re       0
tp_headerConstellation_1_im       0
tp_headerConstellationValid_1     0
tp_headerCRCError_1               0
tp_dataValid_1                    0
tp_dataConstellation_1_re         0
tp_dataConstellation_1_im         0
tp_correlationData_1              0
tp_thresholdData_1                486d3b
tp_correlationValid_1             0
tp_timingOffset_1                 2d
tp_SSPeakSearchDone_1             0
tp_searchHold_1                   1
tp_nSyncedFrames                  32
```

Meteorcomm 使用多个联合仿真模块进行 RTL 逐周期验证

利用 HDL Verifier 的联合仿真功能，Meteorcomm 对 MATLAB 参考模型和 HDL 仿真结果进行逐周期比较，验证了其数字下变频器设计，节省了测试验证时间。

关键成果和优势：

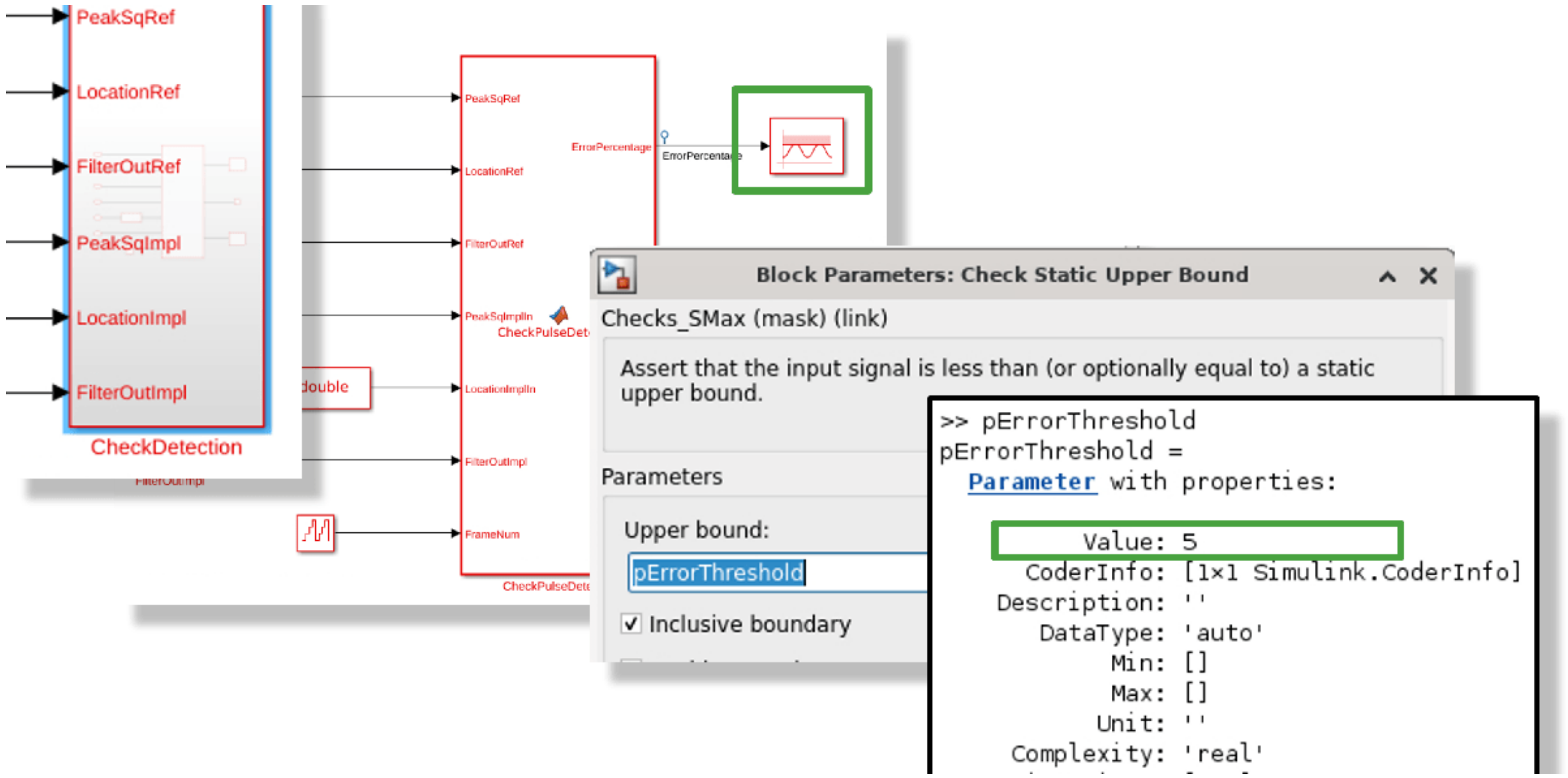
- 通过在 HDL 联合仿真中重用 Simulink 测试平台，节省了时间并减少了错误，无需手写 SystemVerilog 即可验证 HDL 实现。
- 在联合仿真中对 HDL 设计和参考模型逐周期直接比较。
- 在 Simulink 中使用逻辑分析仪，查看和比较系统模型与 RTL 联合仿真模块之间的信号值，提高了调试效率。


数字下变频器模型

“通过使用联合仿真模型，一名FPGA工程师在不到一个小时内完成了三组测试激励。我用了三周来创建联合仿真测试平台。总体开发时间从九周减少到了三周。无论是系统模型的调整还是 RTL 的修改，联合仿真平台都可以很好的适配。”

- Frank Xiao, Meteorcomm

[Meteorcomm Uses Simulink Testbench for Cycle-by-Cycle RTL Verification Using Multiple Cosimulation Blocks]



Assert that the input signal is less than (or optionally equal to) a static upper bound.

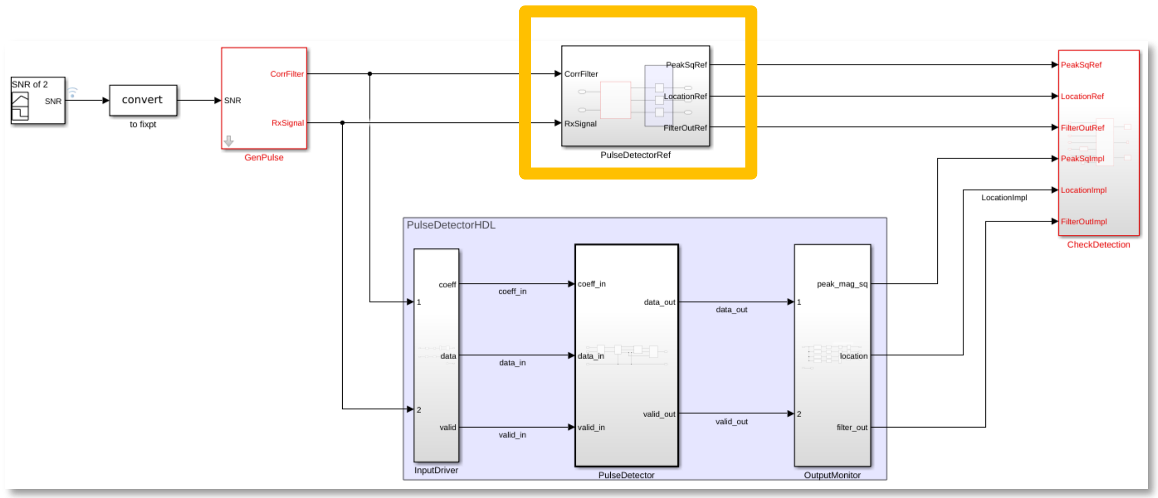
```
>> pErrorThreshold
pErrorThreshold =
  Parameter with properties:

        Value: 5
    CoderInfo: [1×1 Simulink.CoderInfo]
  Description: ''
     DataType: 'auto'
          Min: []
          Max: []
         Unit: ''
   Complexity: 'real'
```

生成 SV DPI 组件建立 UVM 测试环境

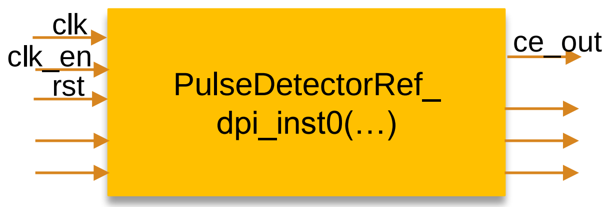
生成独立的 SystemVerilog 组件

将 MATLAB 或 Simulink 算法或测试平台导出为 SystemVerilog 组件

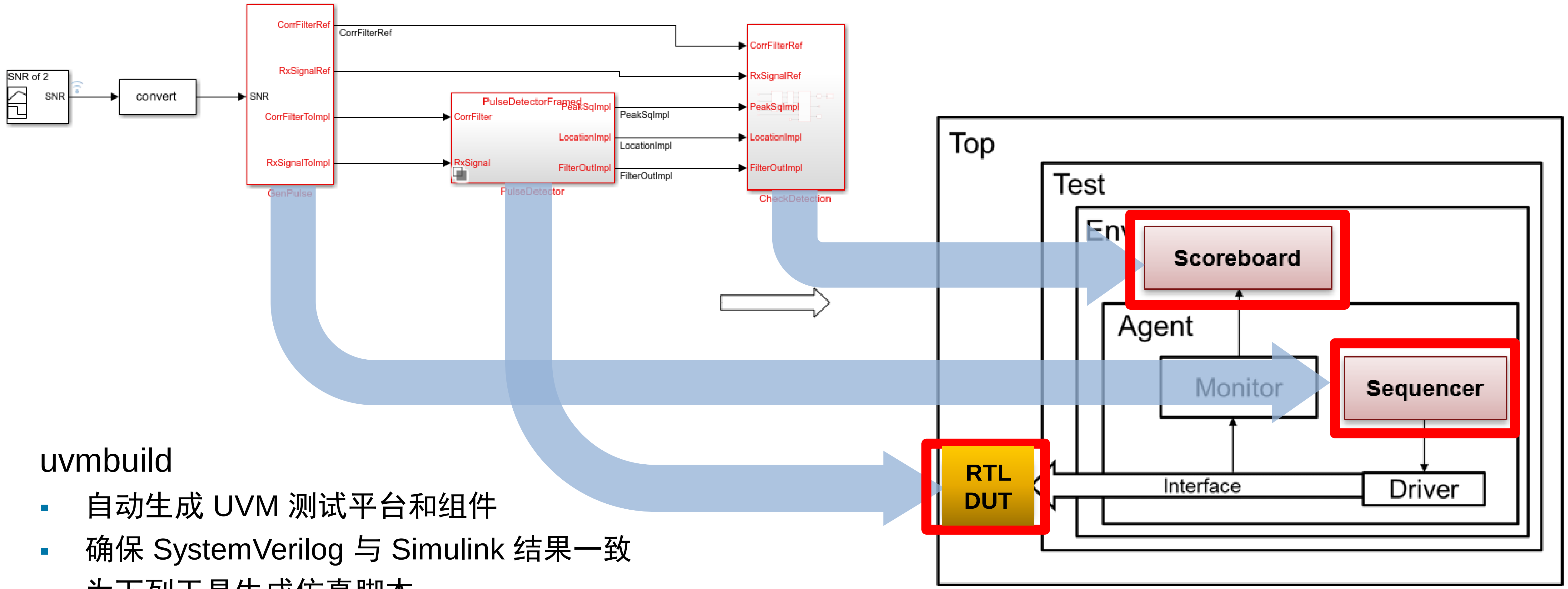
- 生成 C 代码并通过 DPI-C 封装为 SystemVerilog 组件
- 模型中的可调参数生成 setparam 函数，提供模板定制生成的 SystemVerilog 代码
- 可在 Windows 上生成 SystemVerilog 组件，然后在 Linux 上进行编译
- 选择不同的 SystemVerilog 数据类型映射、接口风格及展平处理，满足不同的集成需求



```
>> dpigen
```



生成完整的 UVM 验证环境



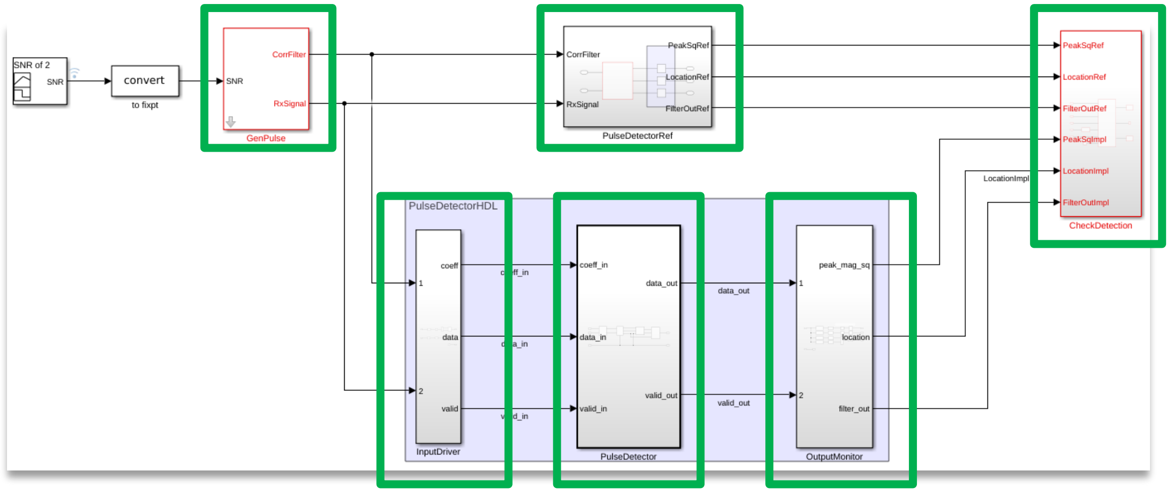
uvmbuild

- 自动生成 UVM 测试平台和组件
- 确保 SystemVerilog 与 Simulink 结果一致
- 为下列工具生成仿真脚本：
  - Siemens EDA (Mentor) ModelSim/Questa
  - Cadence Incisive/Xcelium
  - Synopsys VCS

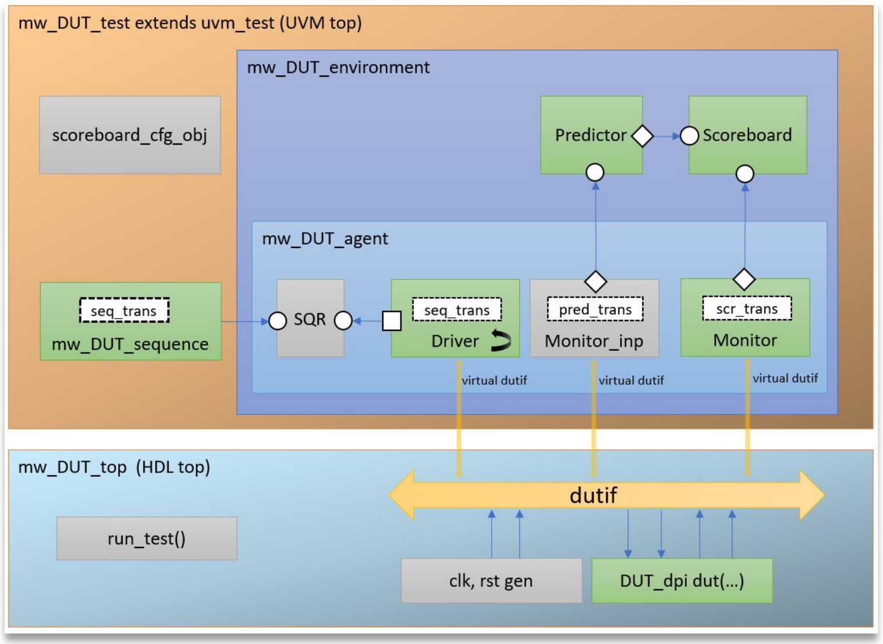
从 Simulink 生成参数化的 UVM 测试平台

在 Simulink 中为 DUT 开发一套完整的测试平台并在 UVM 中复用

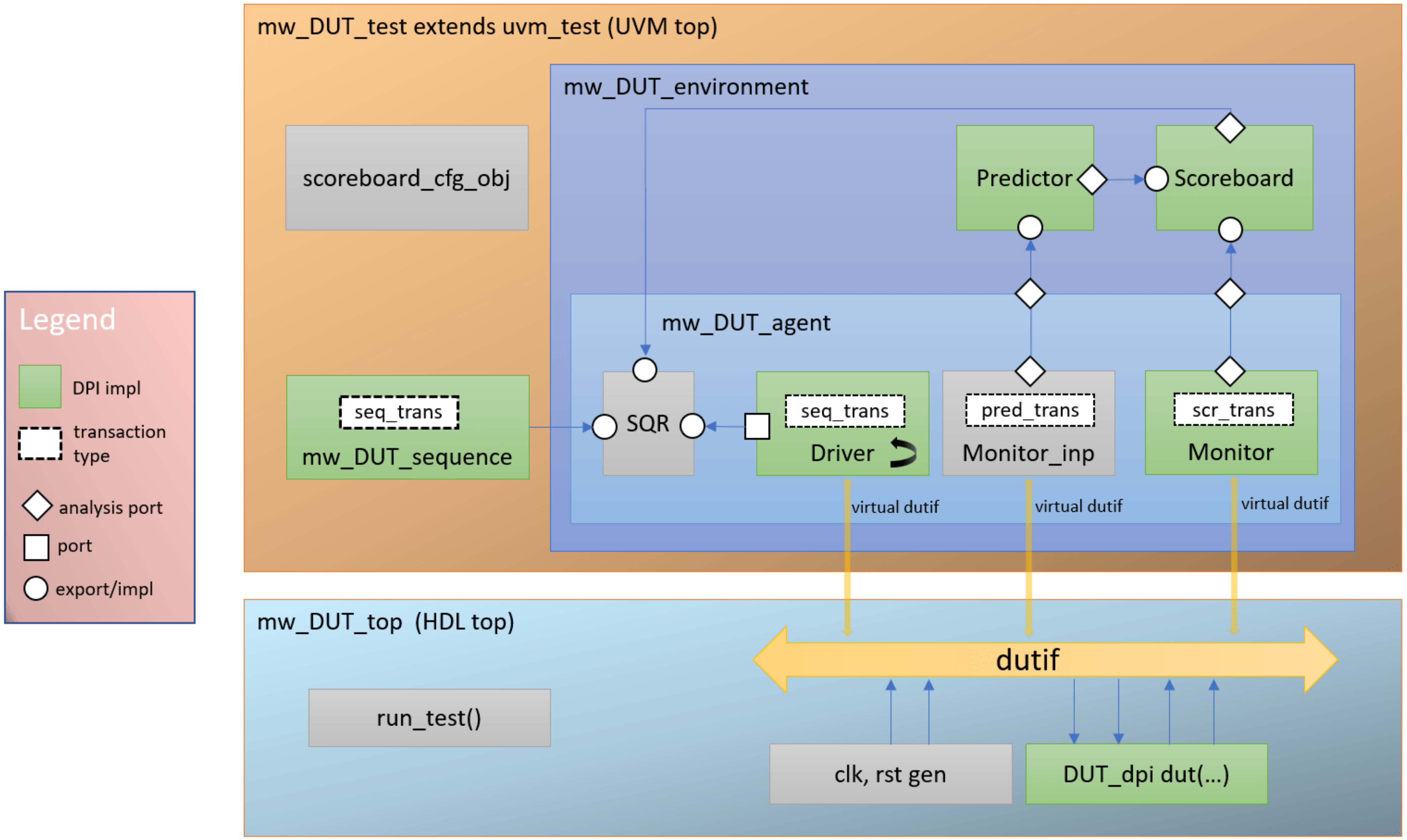
- 实现从系统设计到 HDL 开发的清晰、可执行的交付
- 一整套测试流程，包括 stimulus、prediction、check
- 基于基础 UVM 类和测试平台架构
- 可扩展、便于复用
- 支持快速设计迭代



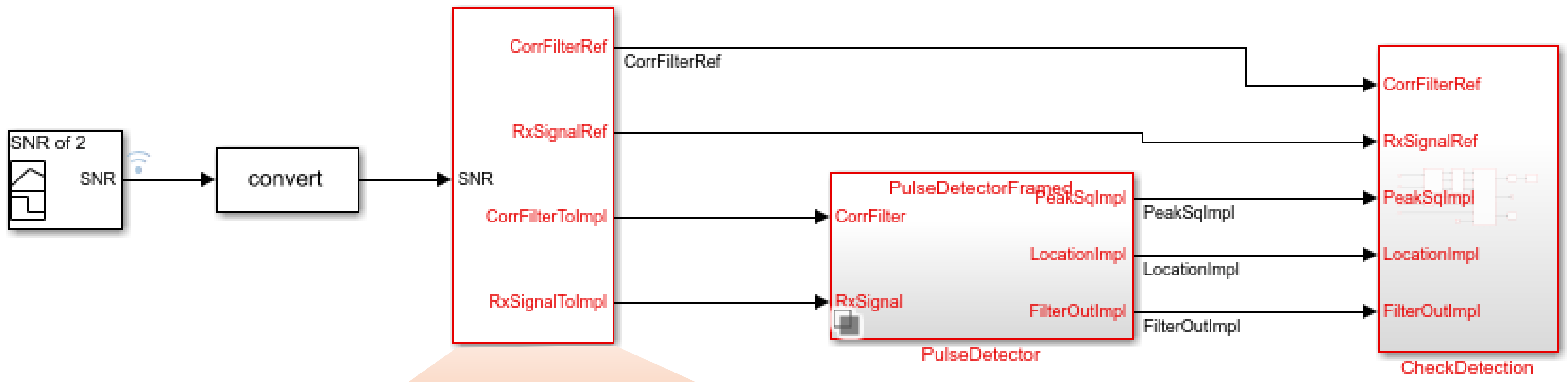
>> uvmbuild



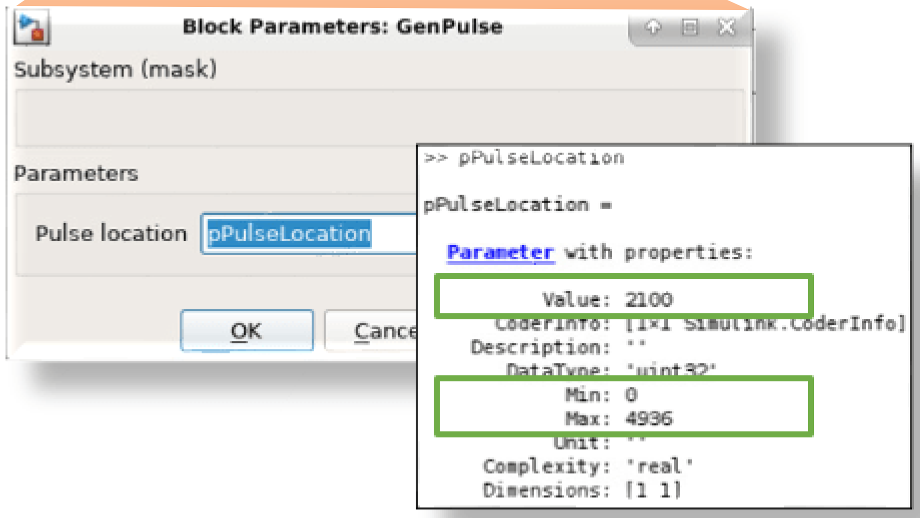
生成的 UVM 测试平台结构



在 UVM 测试平台中添加约束随机验证功能

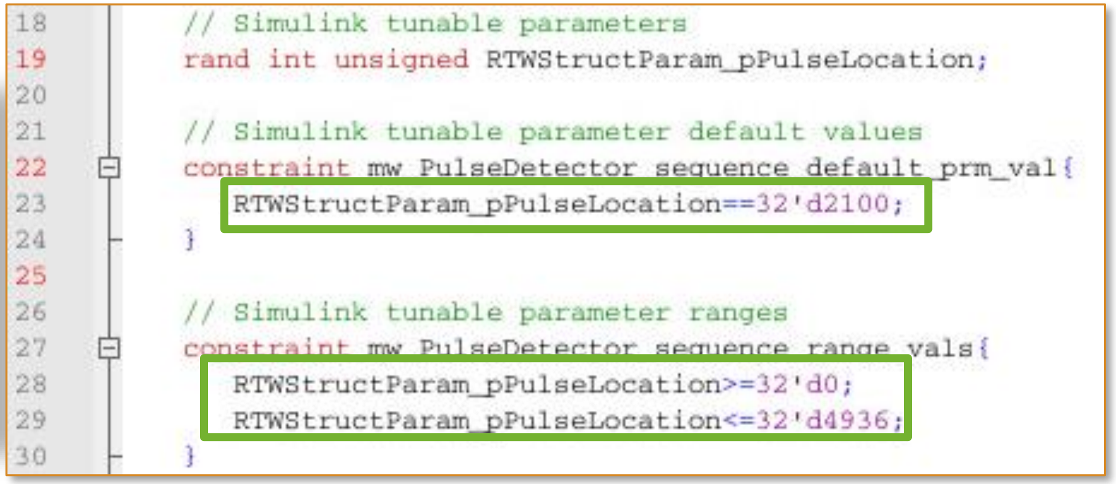


在5000个信号序列中更改64个脉冲的位置



Pulse location: pPulseLocation

```
>> pPulseLocation

pPulseLocation =

  Parameter with properties:

          Value: 2100
      CoderInfo: [1×1 Simulink.CoderInfo]
    Description: ''
       DataType: 'uint32'
            Min: 0
            Max: 4936
           Unit: ''
     Complexity: 'real'
     Dimensions: [1 1]
```

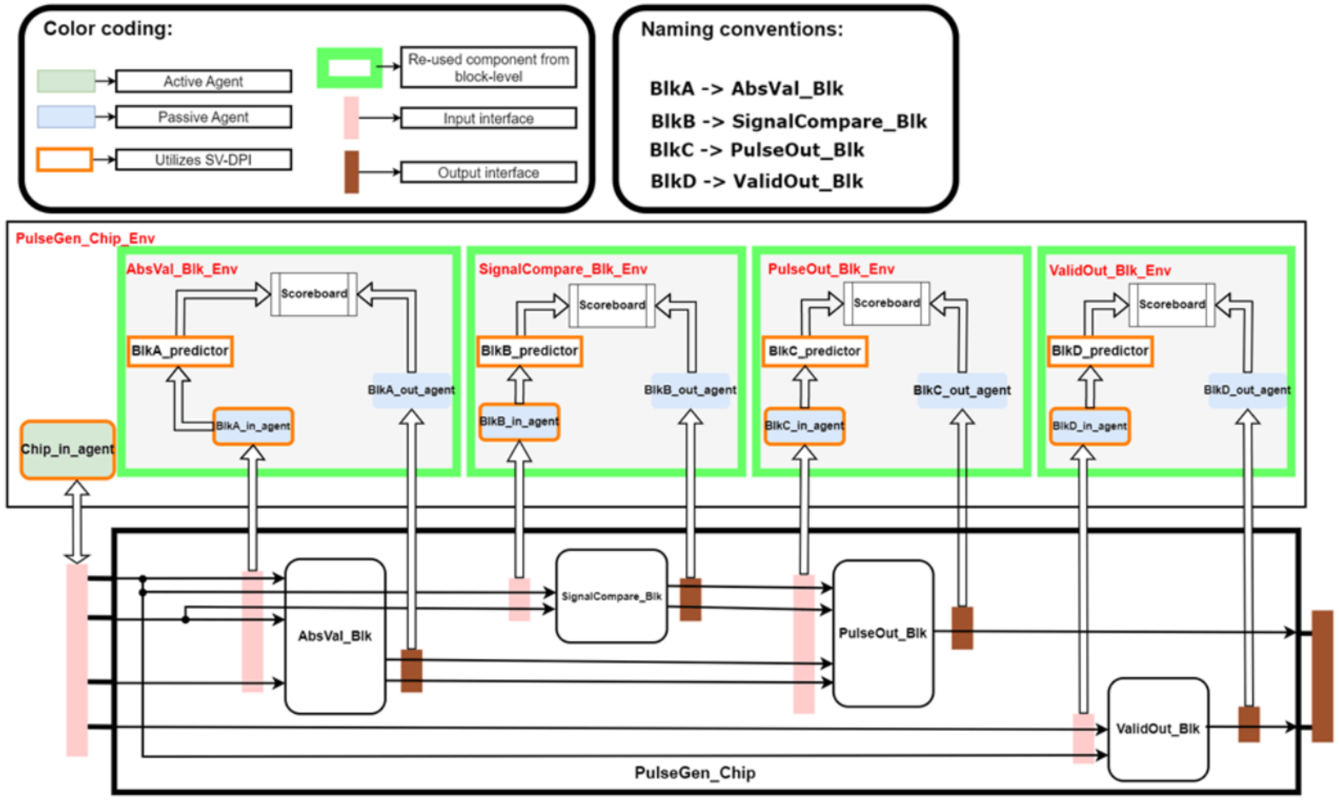


```
    // Simulink tunable parameters
    rand int unsigned RTWStructParam_pPulseLocation;

    // Simulink tunable parameter default values
    constraint mw_PulseDetector_sequence_default_prm_val{
        RTWStructParam_pPulseLocation==32'd2100;
    }

    // Simulink tunable parameter ranges
    constraint mw_PulseDetector_sequence_range_vals{
        RTWStructParam_pPulseLocation>=32'd0;
        RTWStructParam_pPulseLocation<=32'd4936;
    }
```

将生成的 SystemVerilog DPI 集成到 UVM Framework (UVMF)

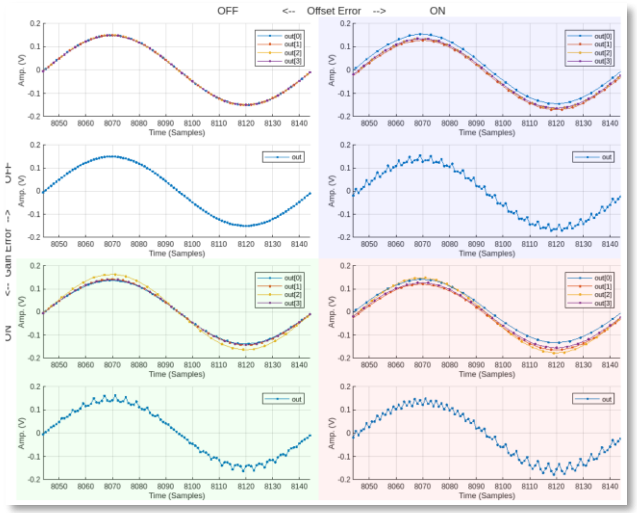
在芯片级验证环境中重用模块级 UVM 组件

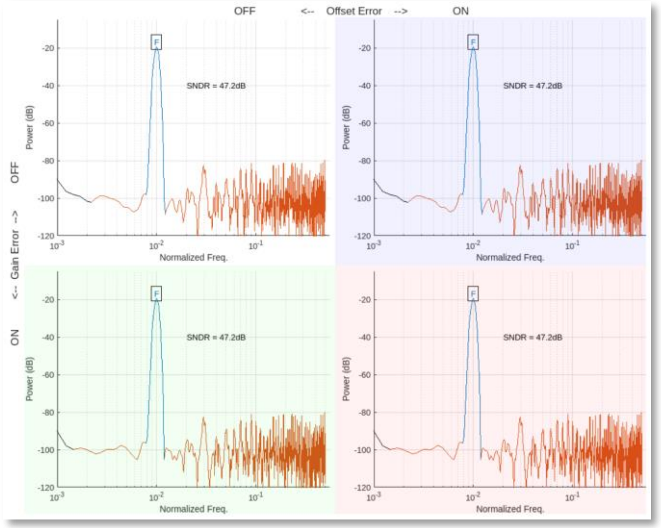
- Siemens UVM Framework 支持从模板生成测试平台
- 使用从 MATLAB 生成的 SV DPI-C 填充 UVMF 模板
- 提供 mtlb2uvmf 工作流
- 在顶层芯片级验证环境中重用单元级测试环境

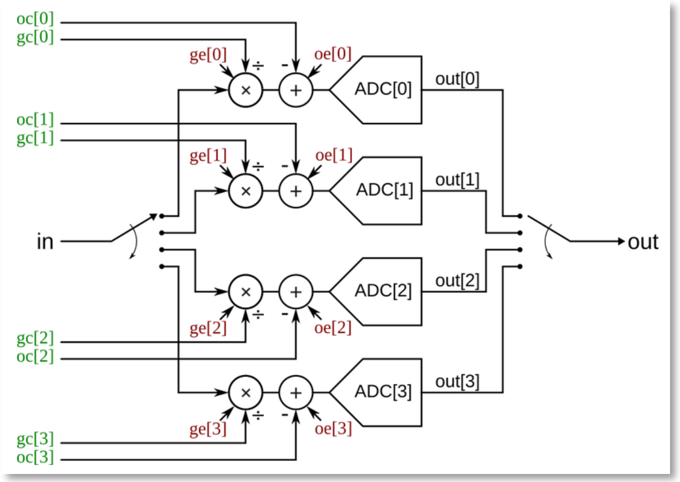


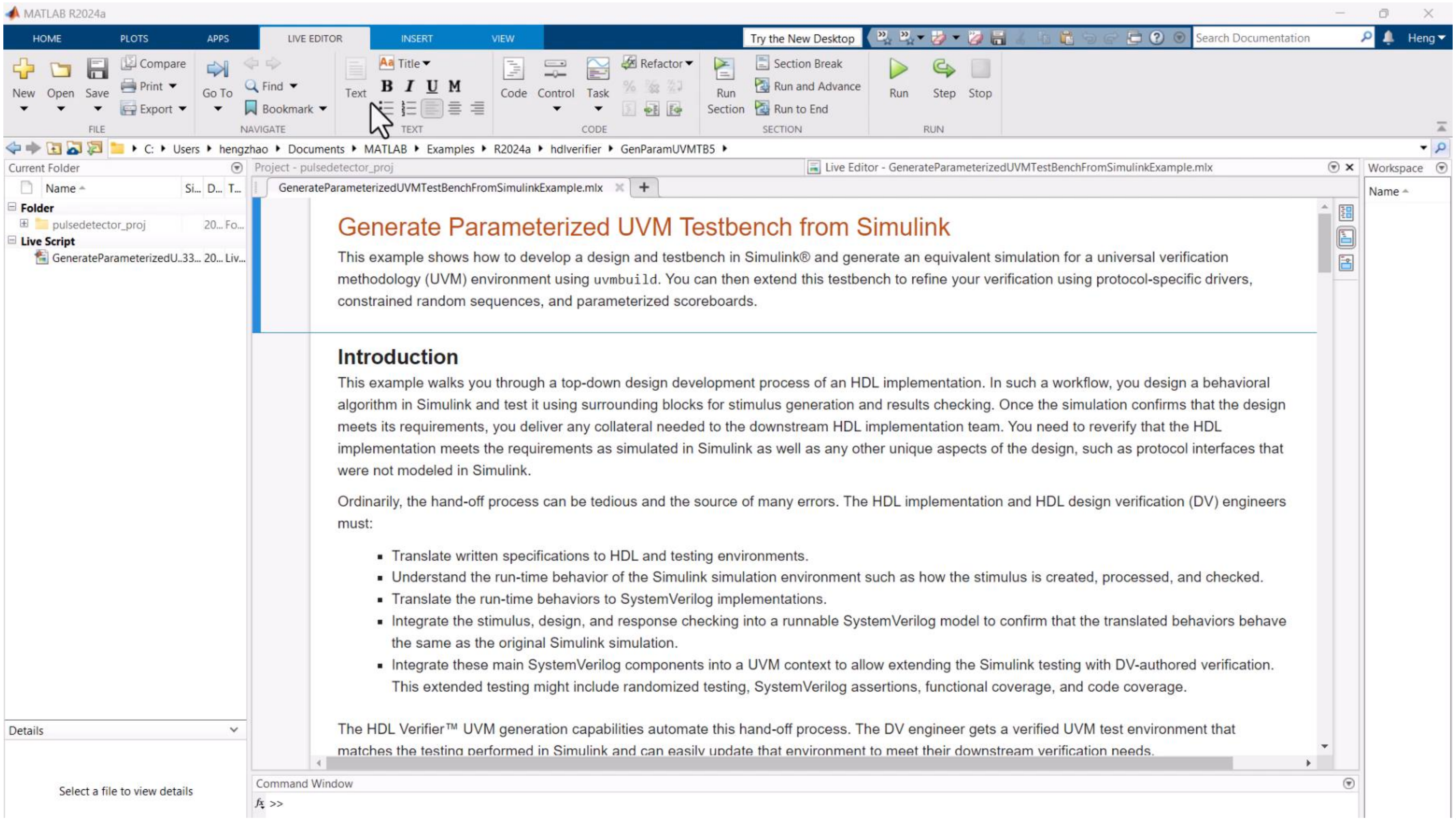
SystemVerilog DPI 用于 SerDes 和混合信号验证

- SerDes
  - 从 SerDes Designer app 创建模型
  - 生成用于混合信号验证的 SV 行为模型
  - 在模拟电路设计完成之前就能开始进行早期数字控制开发
  - 保持系统模型、行为模型和电路模型之间的一致
- 混合信号
  - 仿真时间交织 ADC 中的偏移和增益失配
  - 增益误差和偏移误差作为可调参数
  - 从 ADC 模型生成 SV-DPI 组件
  - 验证 ADC 偏移和增益补偿算法








MATLAB R2024a
HOME
PLOTS
APPS
LIVE EDITOR
INSERT
VIEW
Try the New Desktop
Search Documentation
Heng
New
Open
Save
Compare
Print
Export
FILE
Go To
Find
Bookmark
NAVIGATE
Text
Title
TEXT
Code
Control
Task
Refactor
CODE
Run Section
Section Break
Run and Advance
Run to End
SECTION
Run
Step
Stop
RUN
C: ▸ Users ▸ hengzhao ▸ Documents ▸ MATLAB ▸ Examples ▸ R2024a ▸ hdlverifier ▸ GenParamUVMTB5
Current Folder
Name
Si... D... T...
Folder
pulsedetector_proj 20... Fo...
Live Script
GenerateParameterizedU..33... 20... Liv...
Details
Select a file to view details
Project - pulsedetector_proj
Live Editor - GenerateParameterizedUVMTestBenchFromSimulinkExample.mlx
GenerateParameterizedUVMTestBenchFromSimulinkExample.mlx
Workspace
Name
Command Window
>>

Generate Parameterized UVM Testbench from Simulink

This example shows how to develop a design and testbench in Simulink® and generate an equivalent simulation for a universal verification methodology (UVM) environment using `uvmbuild`. You can then extend this testbench to refine your verification using protocol-specific drivers, constrained random sequences, and parameterized scoreboards.

Introduction

This example walks you through a top-down design development process of an HDL implementation. In such a workflow, you design a behavioral algorithm in Simulink and test it using surrounding blocks for stimulus generation and results checking. Once the simulation confirms that the design meets its requirements, you deliver any collateral needed to the downstream HDL implementation team. You need to reverify that the HDL implementation meets the requirements as simulated in Simulink as well as any other unique aspects of the design, such as protocol interfaces that were not modeled in Simulink.

Ordinarily, the hand-off process can be tedious and the source of many errors. The HDL implementation and HDL design verification (DV) engineers must:

- Translate written specifications to HDL and testing environments.
- Understand the run-time behavior of the Simulink simulation environment such as how the stimulus is created, processed, and checked.
- Translate the run-time behaviors to SystemVerilog implementations.
- Integrate the stimulus, design, and response checking into a runnable SystemVerilog model to confirm that the translated behaviors behave the same as the original Simulink simulation.
- Integrate these main SystemVerilog components into a UVM context to allow extending the Simulink testing with DV-authored verification. This extended testing might include randomized testing, SystemVerilog assertions, functional coverage, and code coverage.

The HDL Verifier™ UVM generation capabilities automate this hand-off process. The DV engineer gets a verified UVM test environment that matches the testing performed in Simulink and can easily update that environment to meet their downstream verification needs.

STMicroelectronics 通过重用 Simulink 组件缩短验证时间

- 挑战
  - 在系统级（Simulink）与 RTL 级（EDA）验证之间存在重复工作：需要在不同环境中实现两次 stimuli 和 scoreboard
- 解决方案
  - 使用 Simulink 来建模和仿真混合信号系统，并使用 HDL Verifier 自动生成 UVM 组件，集成到已有的 UVM 芯片级验证环境中
- 结果
  - 只需维护一个测试平台（stimuli 和 scoreboard）节省了时间
  - 提高了测试用例的可移植性和可重用性
  - 基于 Simulink 将验证时间缩短了一半

2021 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE OCTOBER 26-27, 2021

Reuse of System-Level Verification Components within Chip-Level UVM Environments

Diego Alagna, STMicroelectronics, Milan, Italy (*diego.alagna@st.com*)
Marzia Annovazzi, STMicroelectronics, Milan, Italy (*marzia.annovazzi@st.com*)
Alessandro Cannone, STMicroelectronics, Milan, Italy (*alessandro.cannone1@st.com*)
Marcello Raimondi, STMicroelectronics, Milan, Italy (*marcello.raimondi@st.com*)
Simone Saracino, STMicroelectronics, Milan, Italy (*simone.saracino@st.com*)

“将 RTL 验证时间缩短了一半。此外，生成 RTL 代码帮助工程师减少人为错误，系统仿真和 RTL 仿真可以使用相同的设计源，从而进一步减少集成电路设计的工作量。”

DVCON EUROPE conference paper

MATLAB EXPO

Thank you

MathWorks®