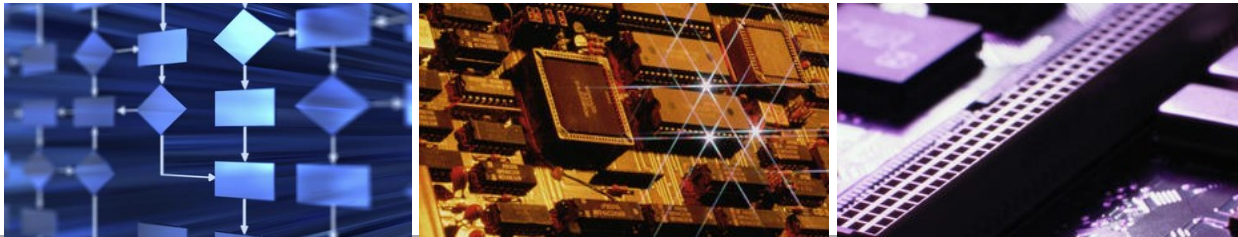


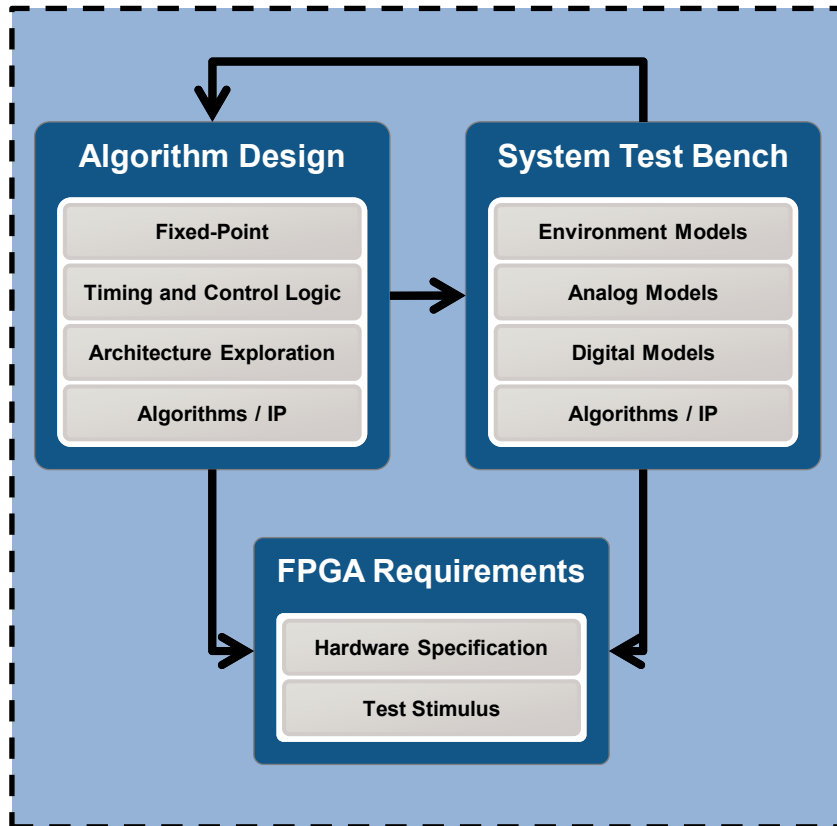
# Model-Based Design for Altera FPGAs Using HDL Code Generation



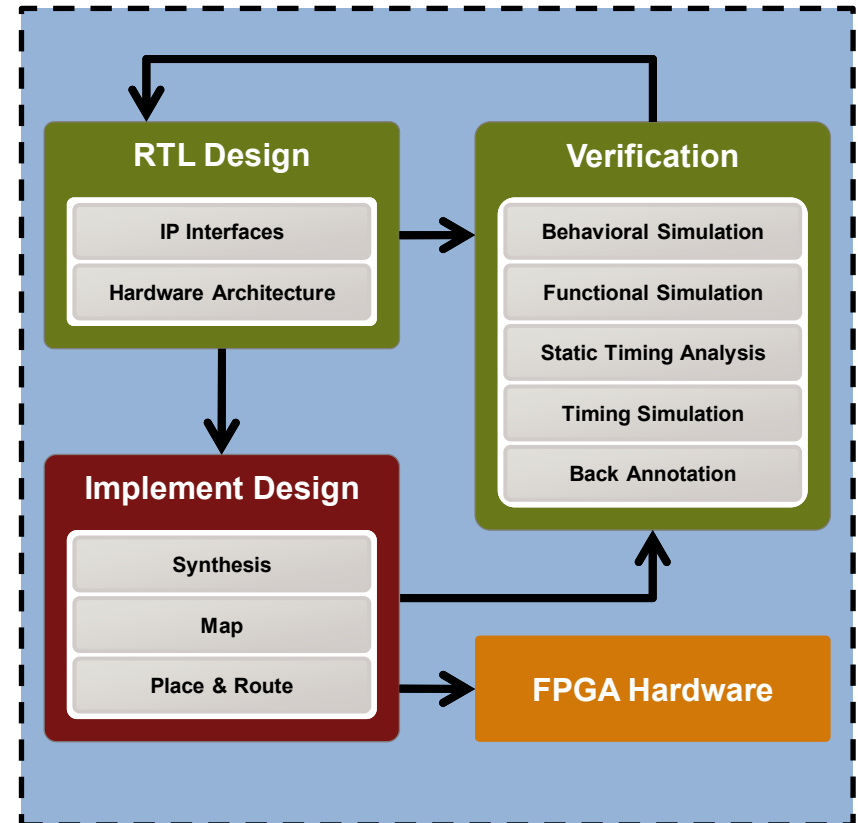
z

# Separate Views of DSP Implementation

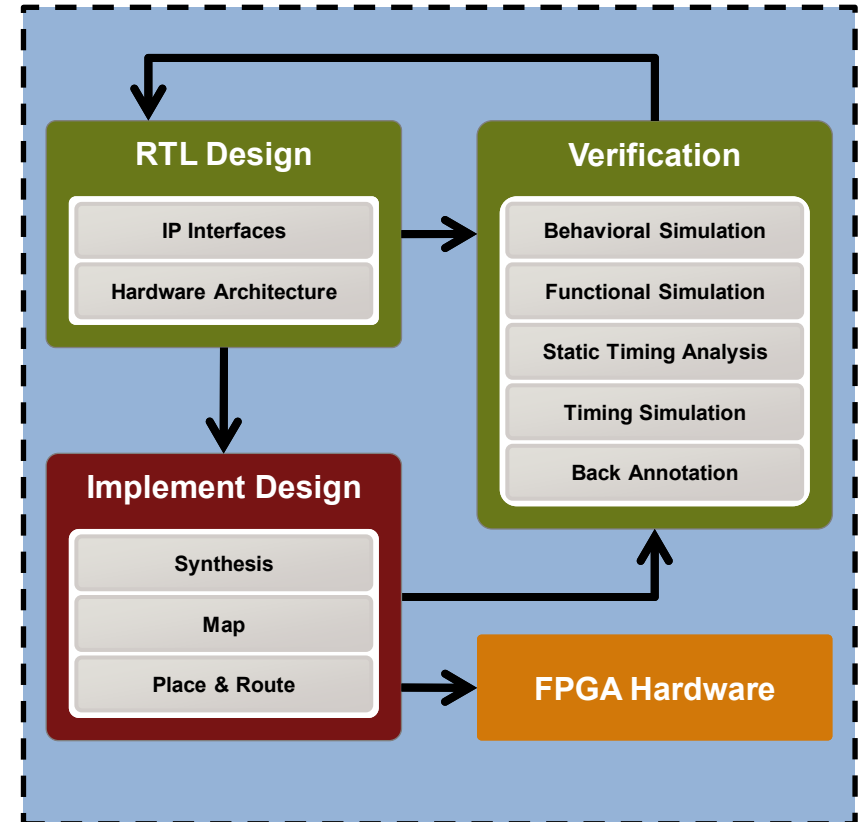
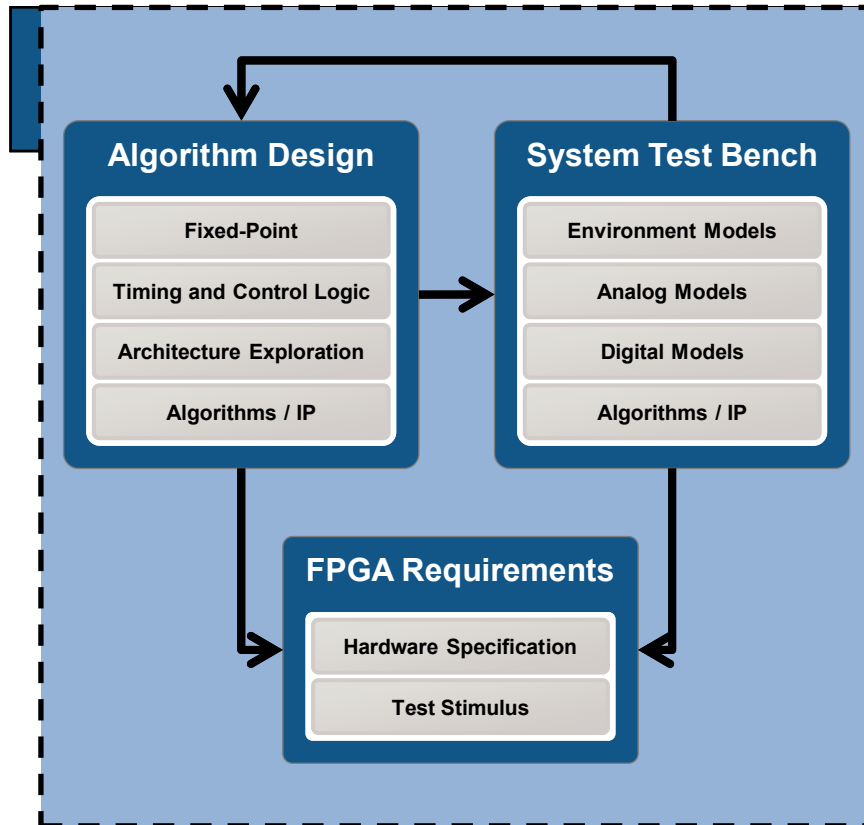
## System Designer



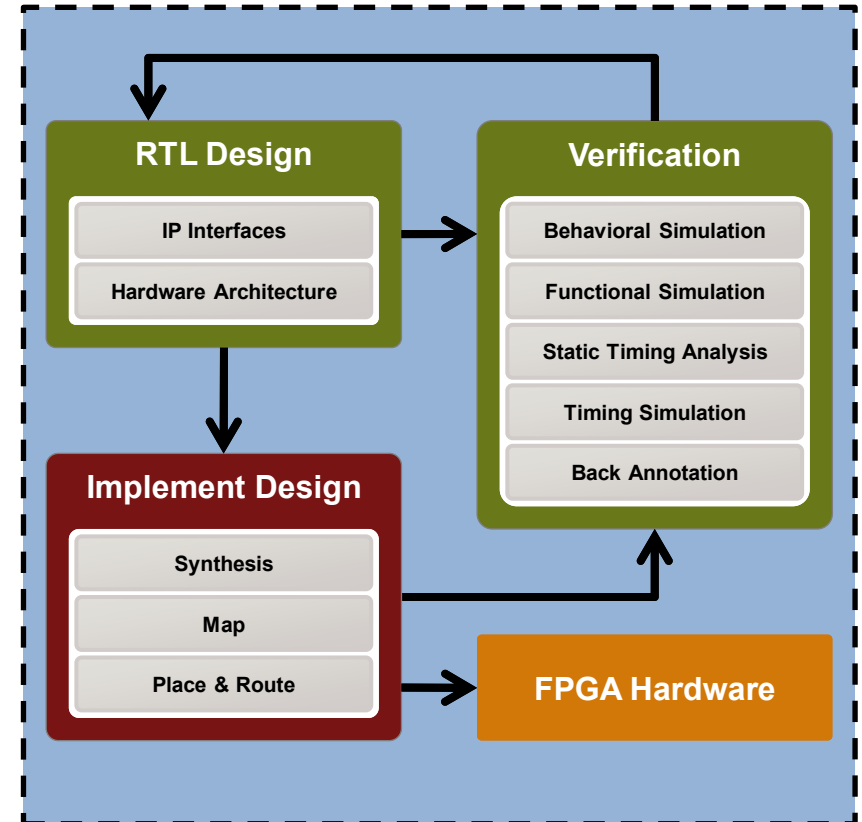
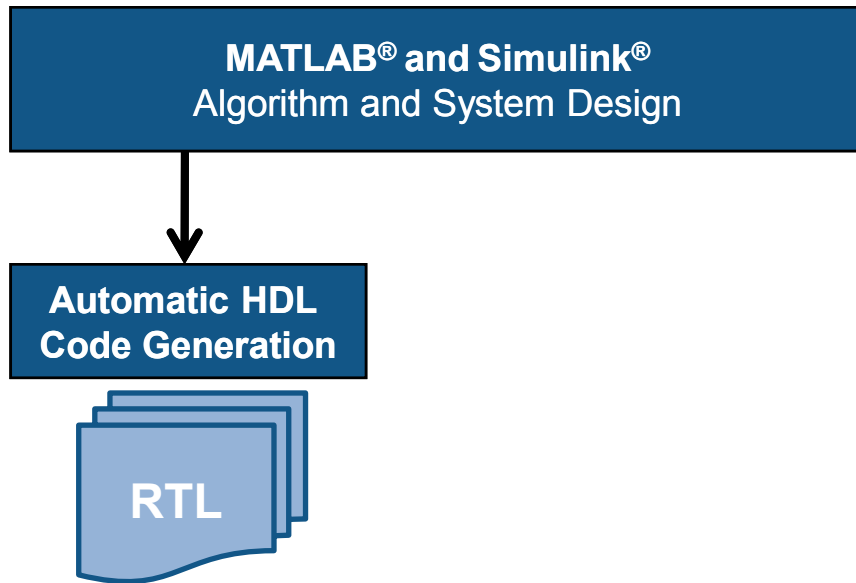
## FPGA Designer



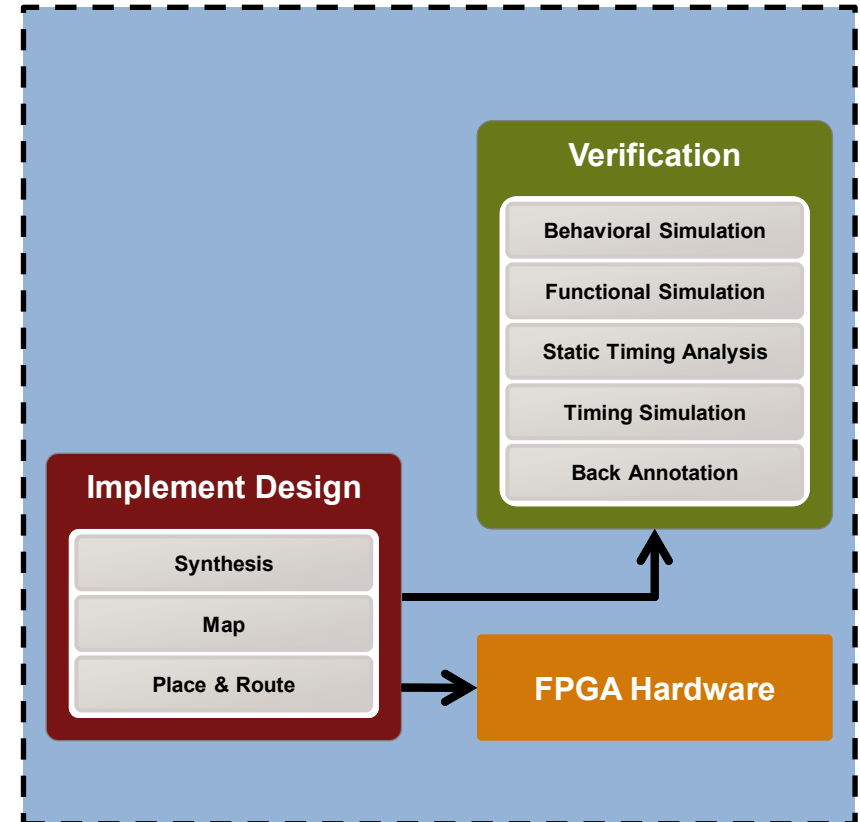
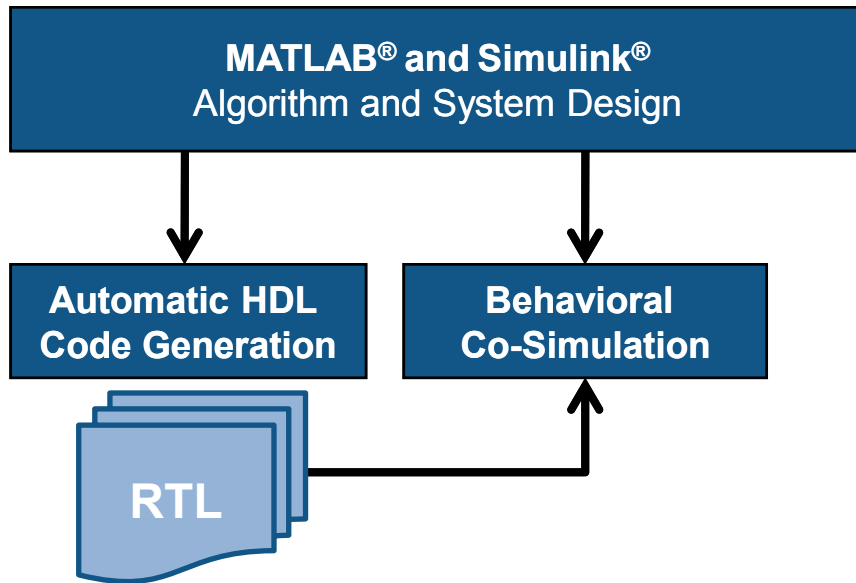
# Model Based Design for Implementation



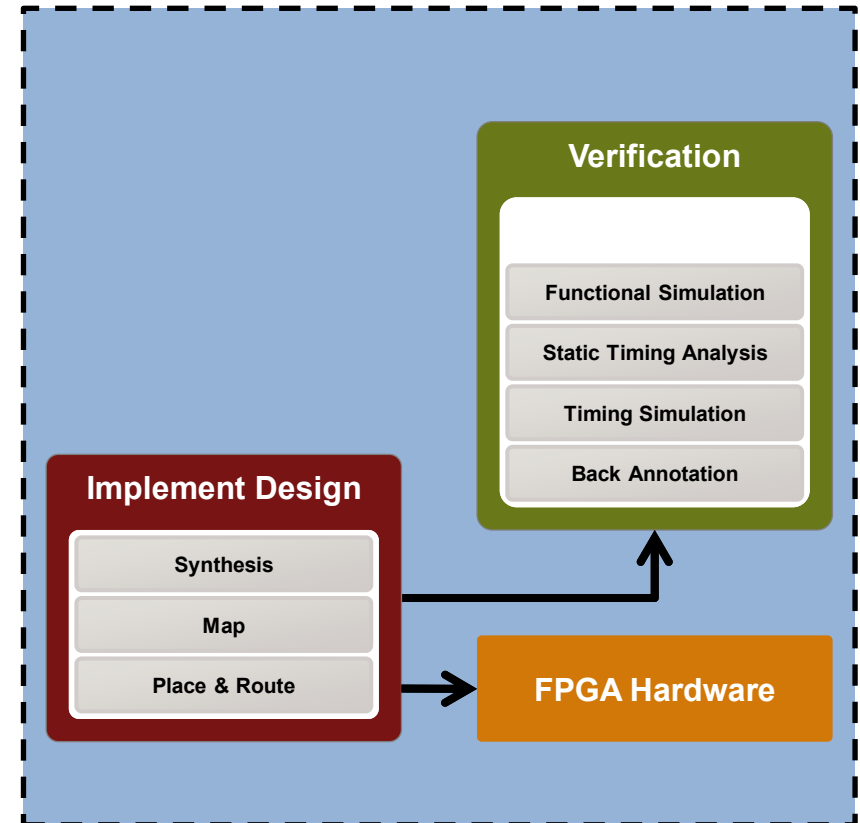
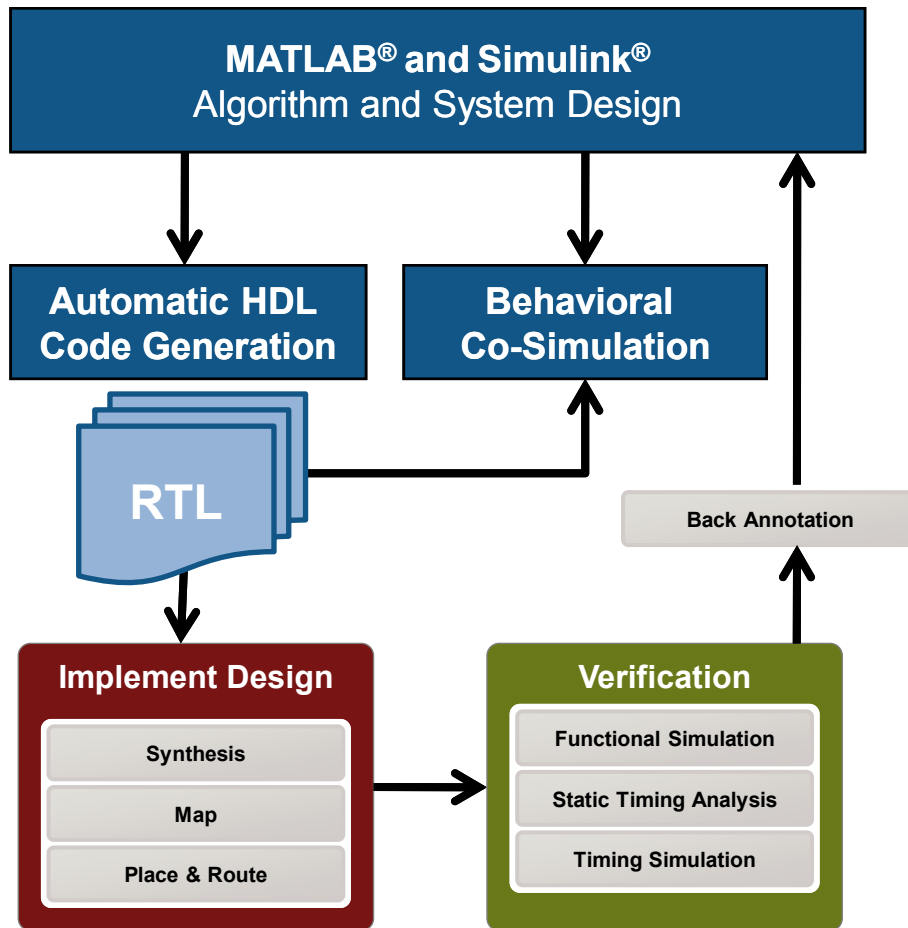
# Model Based Design for Implementation



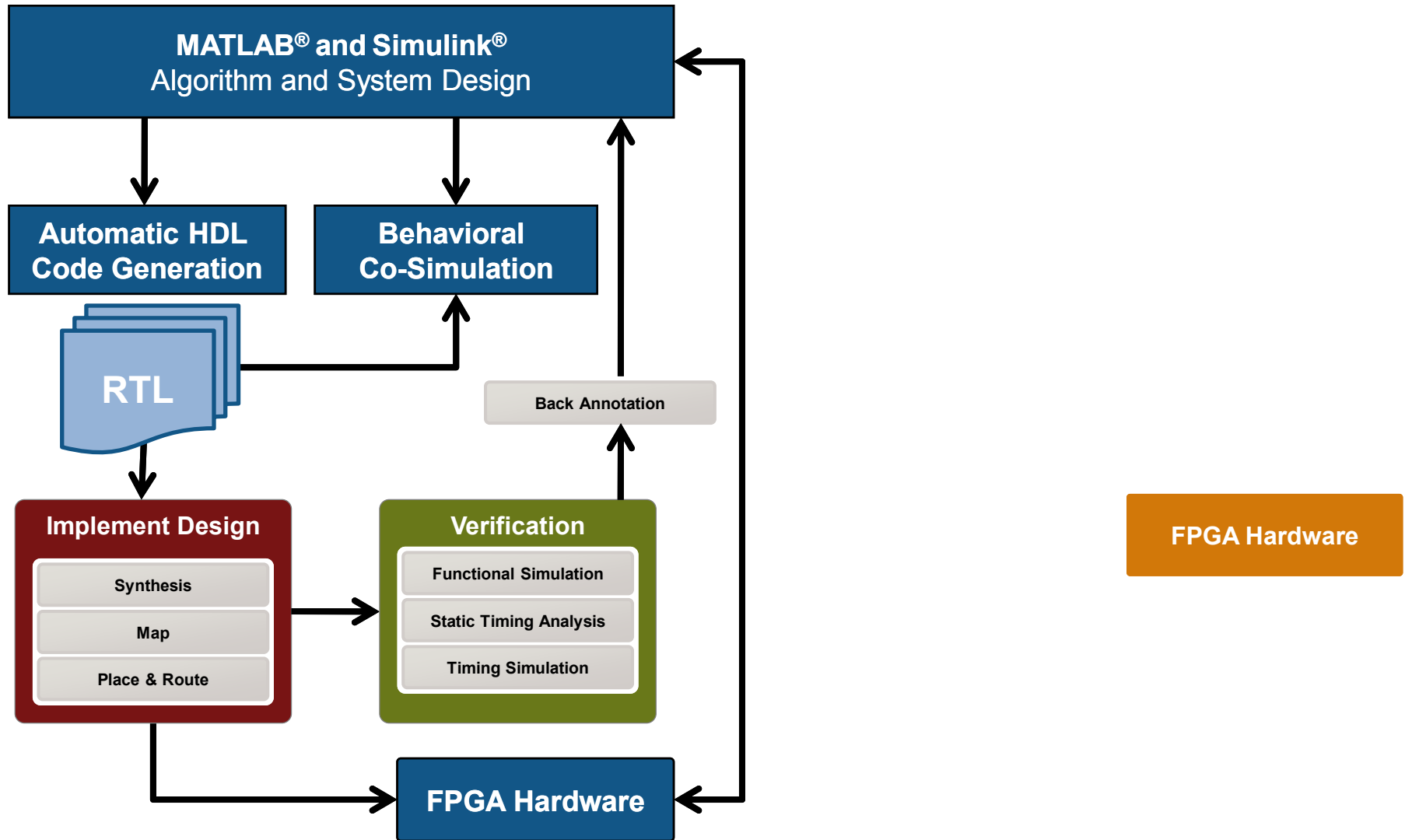
# Model Based Design for Implementation



# Model Based Design for Implementation



# Model Based Design for Implementation



# Faraday Accelerates SIP Development and Shrinks NAND Flash Controller ECC Engine Gate Count by 57%

## Challenge

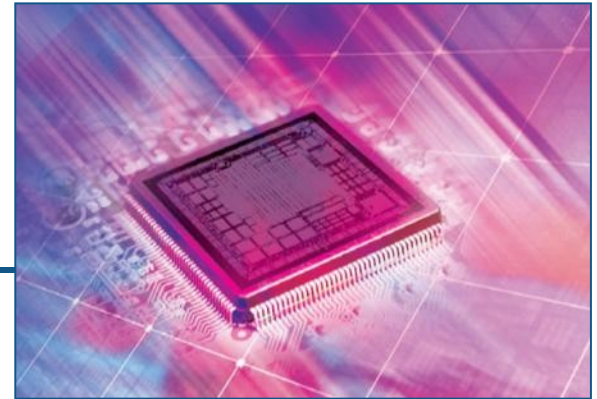
Accelerate the development of SoCs and ASICs

## Solution

Use MathWorks tools for Model-Based Design to speed up system-level simulations, improve system performance, and shorten time-to-market

## Results

- Simulations 200 times faster
- Throughput performance increased by 15%
- Gate count cut by 57%



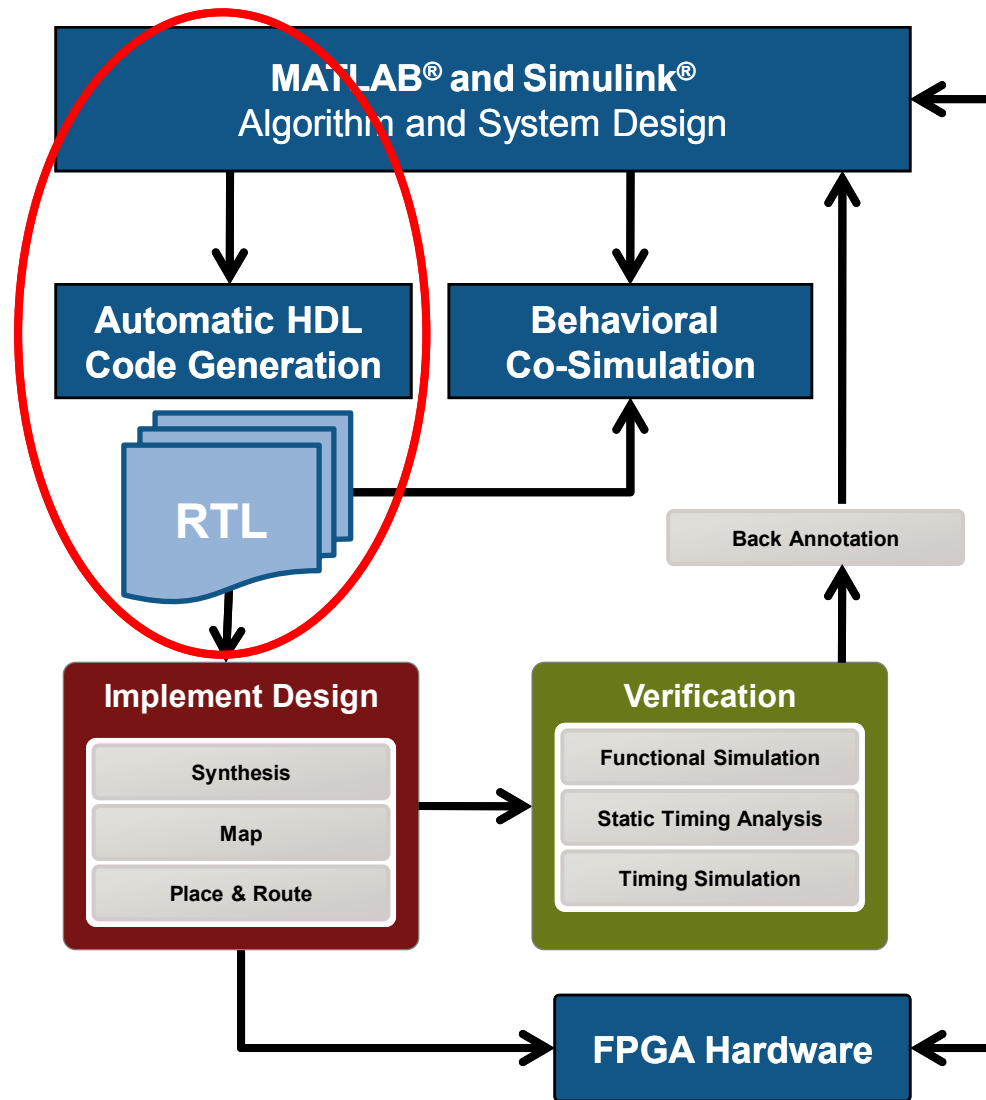
Faraday's silicon IP on an SoC.

**“The Simulink environment is ideal for system-level architecture exploration. The simulations are 200 times faster than they were in our previous workflow — and Simulink models can be easily converted to C as well as to HDL code, which enables high scalability and reusability.”**

**Ken Chen**  
Faraday

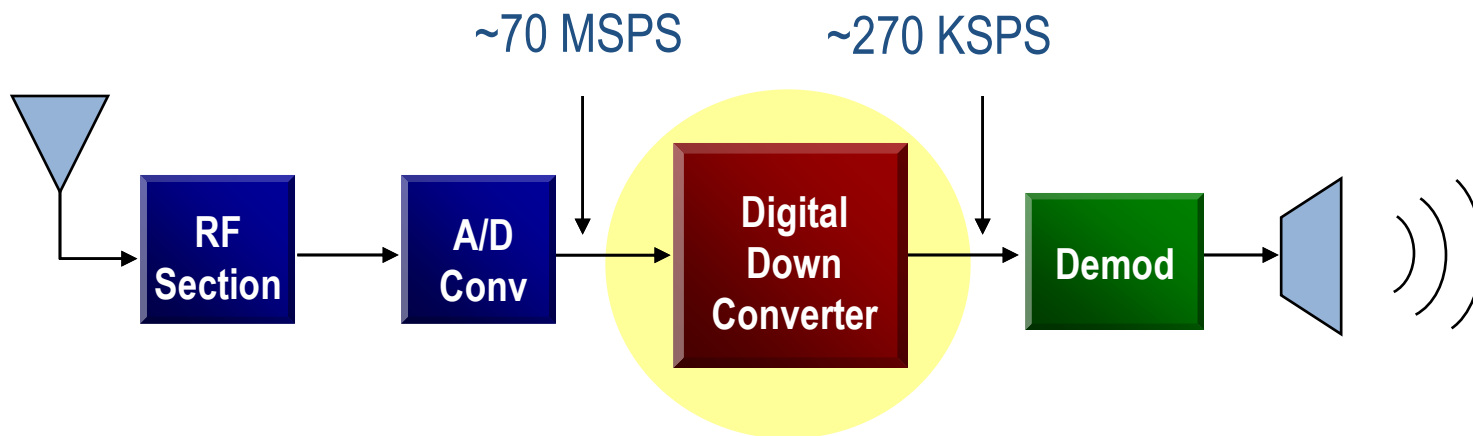


# From Algorithm to Synthesizable RTL



# Digital Down Converter

- DDC accepts
  - A high sample-rate passband signal (may be 50 to 100 Msps)
- DDC produces
  - A low sample-rate baseband signal ready for demodulation

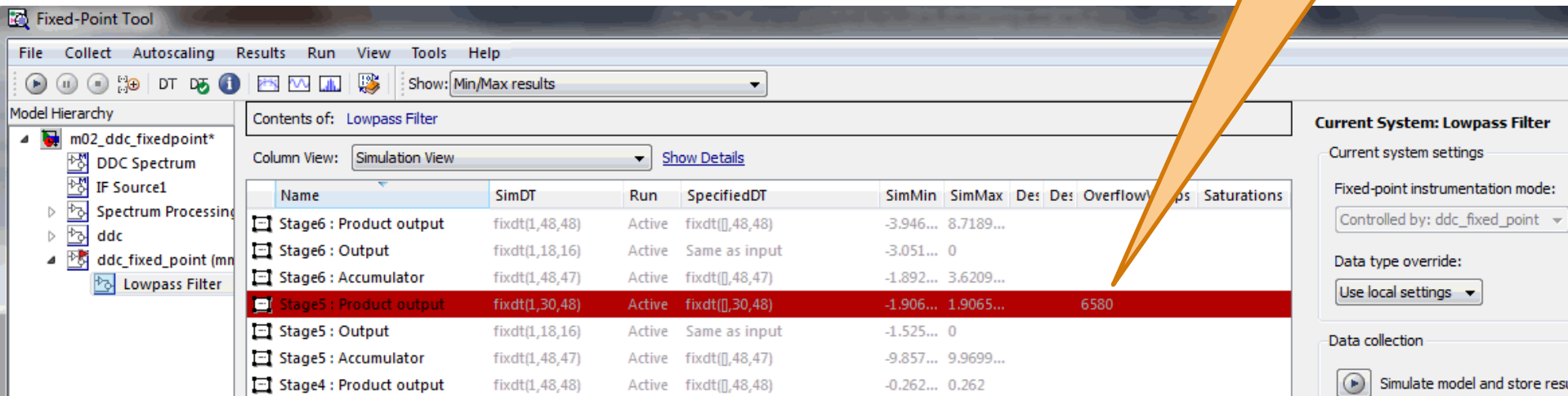


# Fixed Point Analysis

## Digital Down Converter

- Convert floating point to fixed point models
  - Automatic tracking of signal range (also intermediate quantities)
  - Fraction lengths recommendation
- Bit-true models in the same environment
  - Quantify the impact of fixed point quantization

**Find and fix issues  
with fixed point  
easily**



The screenshot shows the Fixed-Point Tool interface with the 'Lowpass Filter' model selected. The 'Simulation View' column shows the results of the simulation. The table below is a representation of the data shown in the tool.

Name	SimDT	Run	SpecifiedDT	SimMin	SimMax	Des	Des	Overflow	ps	Saturations
Stage6 : Product output	fixdt(1,48,48)	Active	fixdt([],48,48)	-3.946...	8.7189...					
Stage6 : Output	fixdt(1,18,16)	Active	Same as input	-3.051...	0					
Stage6 : Accumulator	fixdt(1,48,47)	Active	fixdt([],48,47)	-1.892...	3.6209...					
Stage5 : Product output	fixdt(1,30,48)	Active	fixdt([],30,48)	-1.906...	1.9065...			6580		
Stage5 : Output	fixdt(1,18,16)	Active	Same as input	-1.525...	0					
Stage5 : Accumulator	fixdt(1,48,47)	Active	fixdt([],48,47)	-9.857...	9.9699...					
Stage4 : Product output	fixdt(1,48,48)	Active	fixdt([],48,48)	-0.262...	0.262					

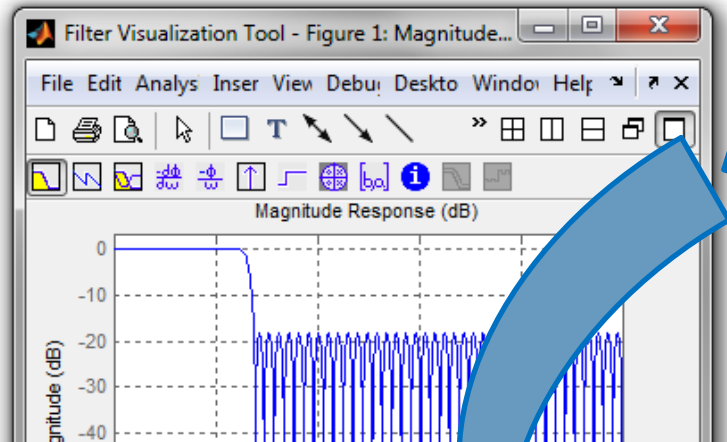
On the right side of the interface, the 'Current System: Lowpass Filter' settings are visible, including 'Fixed-point instrumentation mode' set to 'Controlled by: ddc\_fixed\_point' and 'Data type override' set to 'Use local settings'.

# Automatic HDL Code Generation

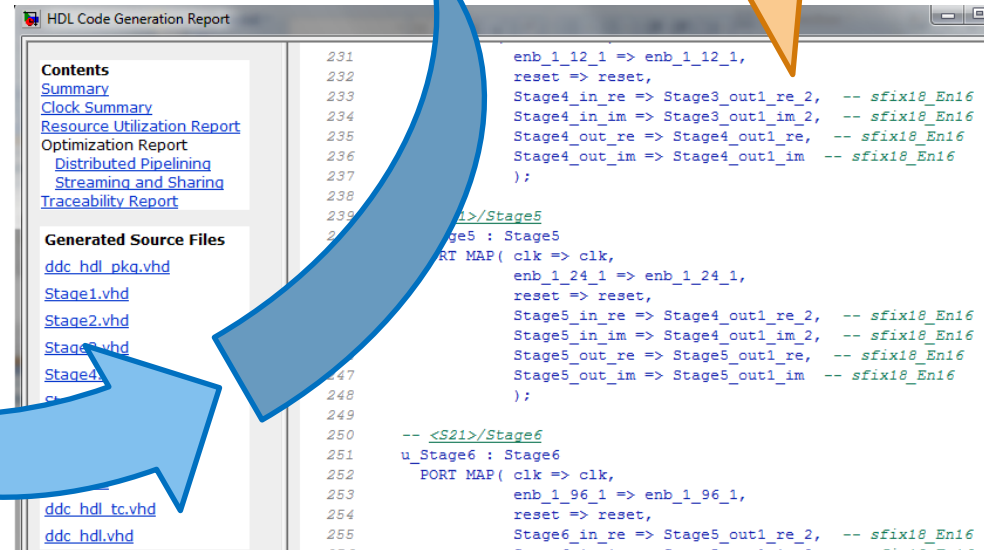
## Digital Down Converter



**Automatically generate bit true, cycle accurate HDL code from Simulink, MATLAB and Stateflow**



**Full traceability between model and HDL code**



HDL Code Generation Report

**Contents**

- [Summary](#)
- [Clock Summary](#)
- [Resource Utilization Report](#)
- [Optimization Report](#)
- [Distributed Pipelining](#)
- [Streaming and Sharing](#)
- [Traceability Report](#)

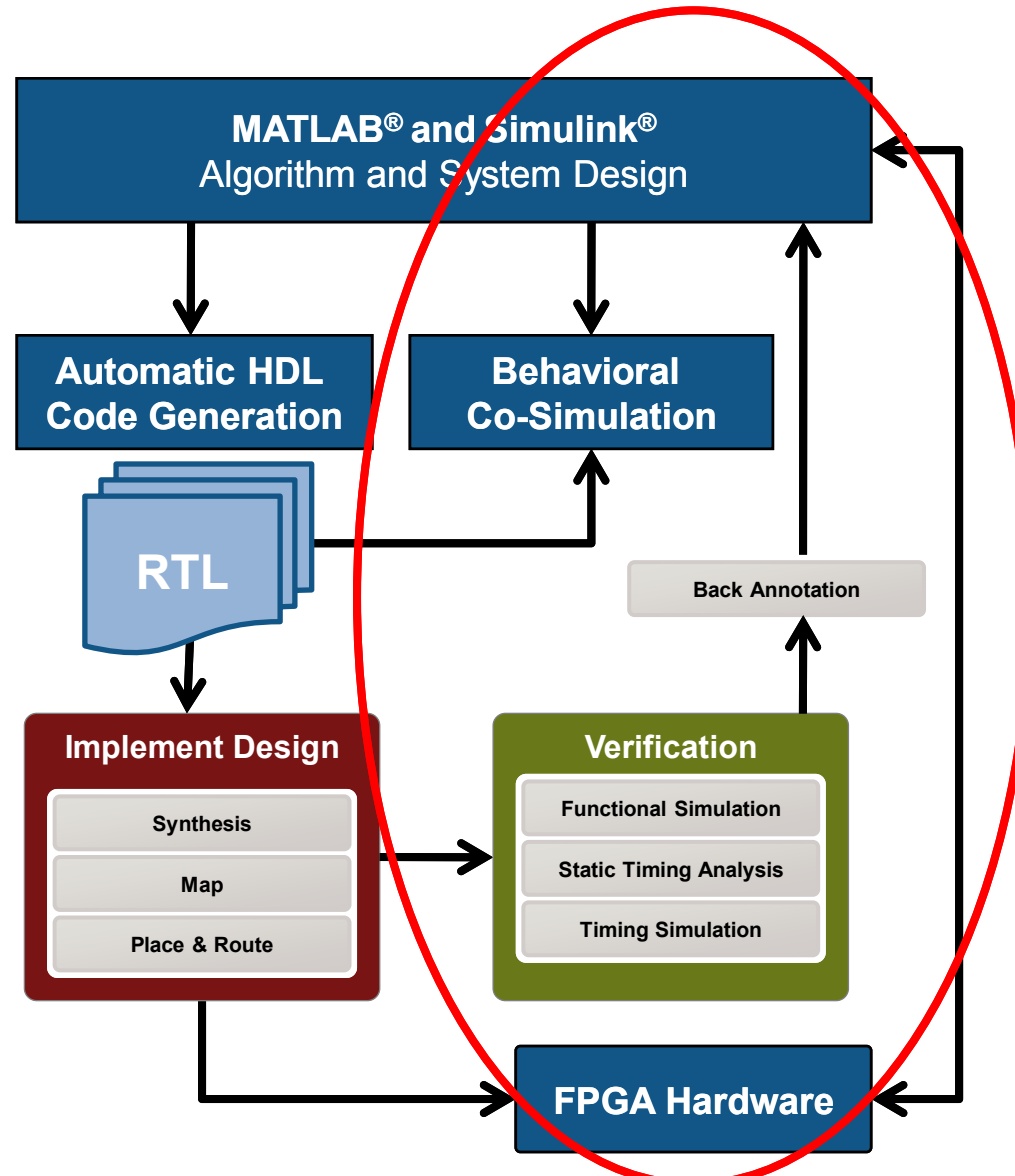
**Generated Source Files**

- [ddc\\_hdl\\_pkg.vhd](#)
- [Stage1.vhd](#)
- [Stage2.vhd](#)
- [Stage3.vhd](#)
- [Stage4.vhd](#)
- [Stage5.vhd](#)
- [Stage6.vhd](#)
- [u\\_ddc.vhd](#)
- [ddc\\_hdl\\_tc.vhd](#)
- [ddc\\_hdl.vhd](#)

```

231 enb_1_12_1 => enb_1_12_1,
232 reset => reset,
233 Stage4_in_re => Stage3_out1_re_2, -- sfix18_En16
234 Stage4_in_im => Stage3_out1_im_2, -- sfix18_En16
235 Stage4_out_re => Stage4_out1_re, -- sfix18_En16
236 Stage4_out_im => Stage4_out1_im -- sfix18_En16
237 );
238
239 -->/Stage5
240 Stage5 : Stage5
241 PORT MAP( clk => clk,
242 enb_1_24_1 => enb_1_24_1,
243 reset => reset,
244 Stage5_in_re => Stage4_out1_re_2, -- sfix18_En16
245 Stage5_in_im => Stage4_out1_im_2, -- sfix18_En16
246 Stage5_out_re => Stage5_out1_re, -- sfix18_En16
247 Stage5_out_im => Stage5_out1_im -- sfix18_En16
248 );
249
250 -->/Stage6
251 u_Stage6 : Stage6
252 PORT MAP( clk => clk,
253 enb_1_96_1 => enb_1_96_1,
254 reset => reset,
255 Stage6_in_re => Stage5_out1_re_2, -- sfix18_En16
256 Stage6_in_im => Stage5_out1_im_2, -- sfix18_En16
257 );
  
```

# Integrated HDL Verification



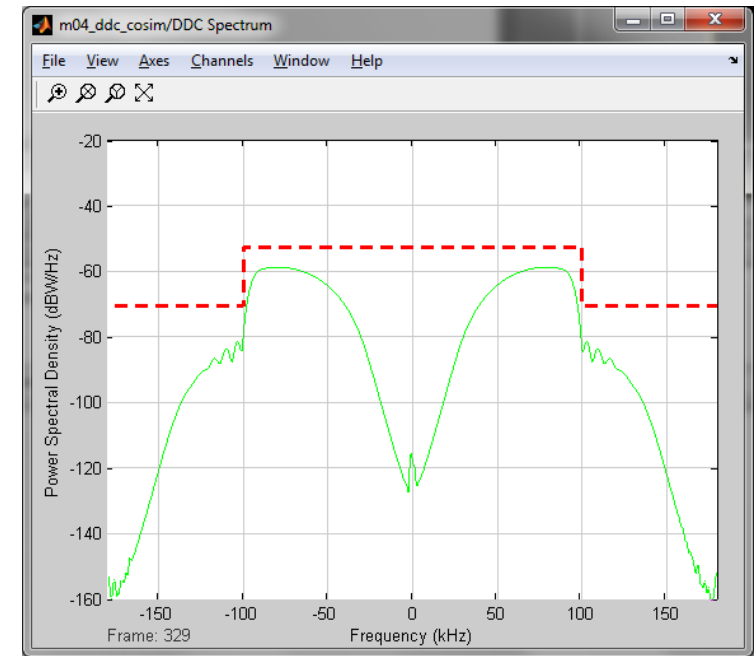
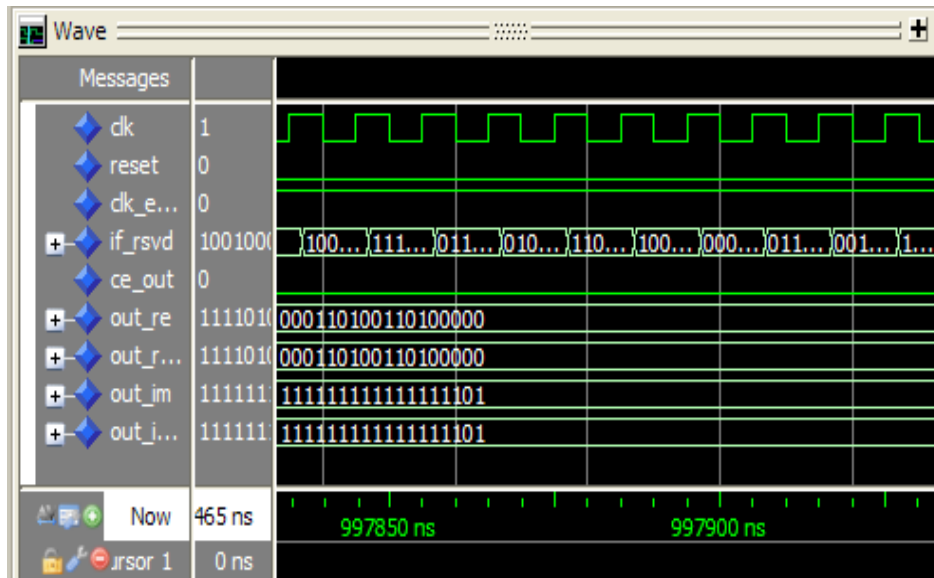
# Verification Challenges:

## HDL Verification

- Design the Test Bench twice
  - 10 – to – 1 ratio of Test bench LOC – to – Design LOC
- Many stimulus files from MATLAB
- These are ideal references which require pre- and post-processing
- How to analyze results?

# Verification Challenges:

## HDL Verification



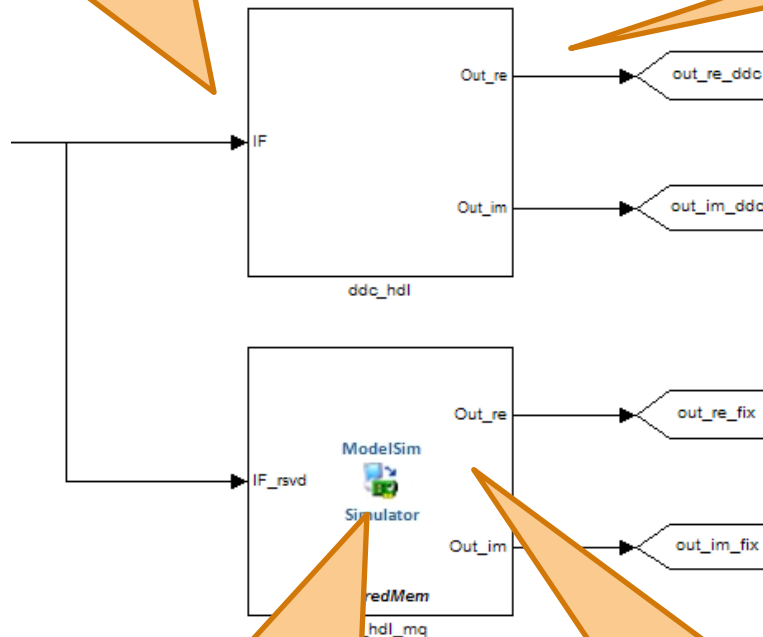
## Demo: Re-Use System Level Test Bench

# Co-Simulation with HDL simulators

## Digital Down Converter

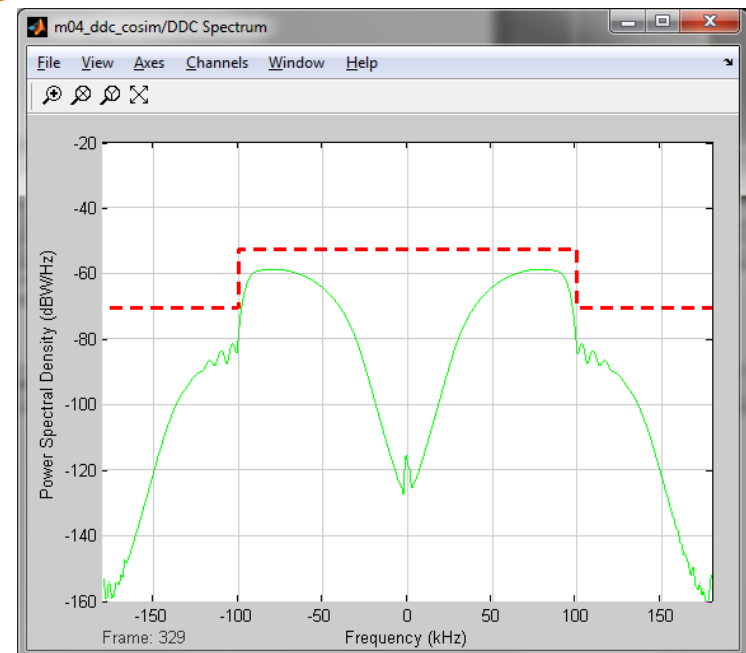
Re-use system level test bench

Flexible test bench creation in Simulink



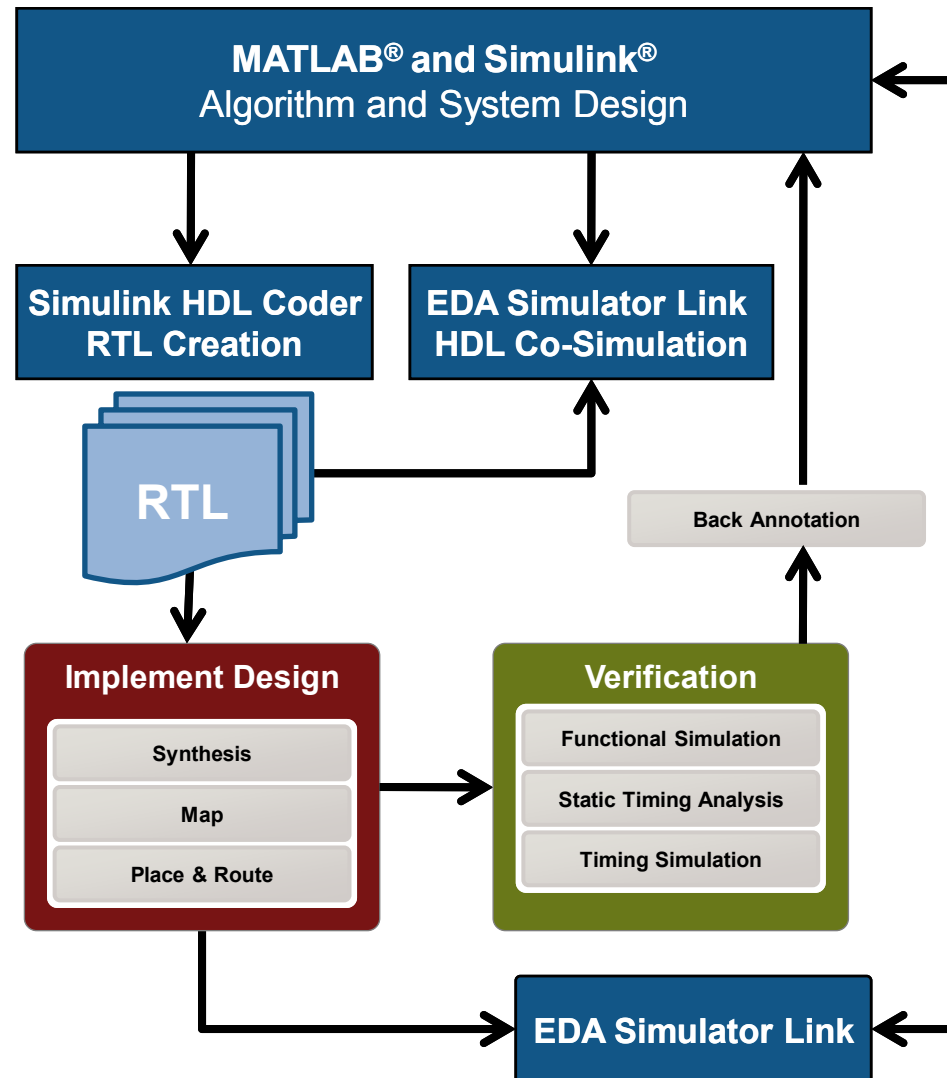
Direct simulation link to HDL Simulators

Automatically generated co-simulation models and Wizards for legacy HDL code



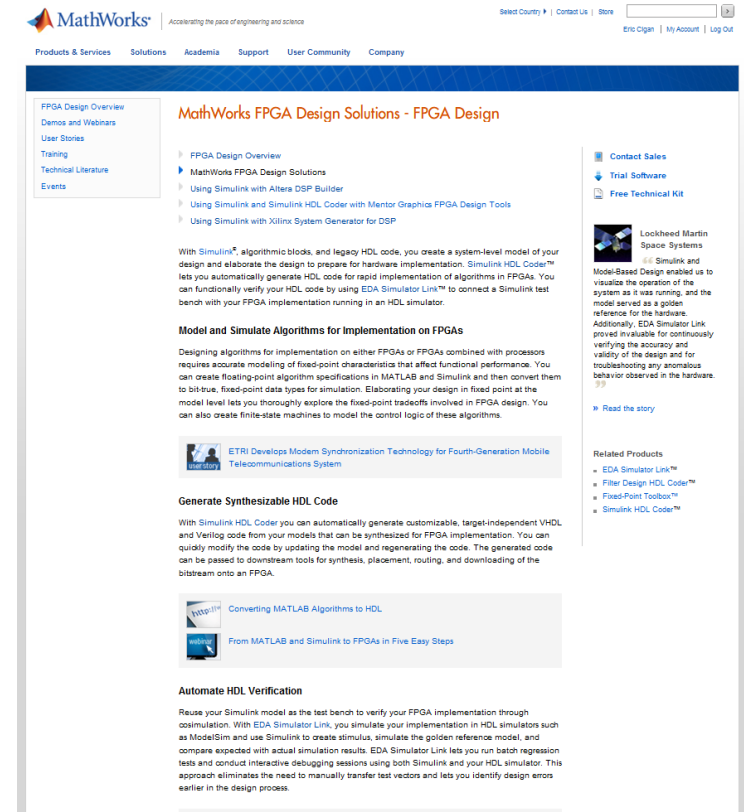


# From Algorithm to FPGA Prototyping and Verification



# Next Steps ...

1. Visit [www.mathworks.com/fpga](http://www.mathworks.com/fpga) for more information
2. Watch our FPGA webinars: [mathworks.com/company/events/webinars](http://mathworks.com/company/events/webinars)
3. Contact your local sales reps for a **trial** of MathWorks HDL code generation and verification products



## Questions?