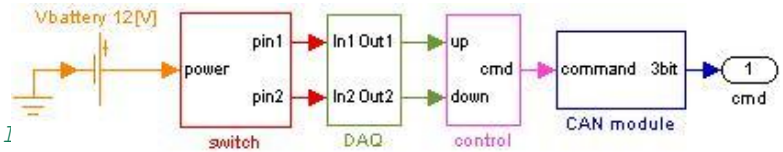
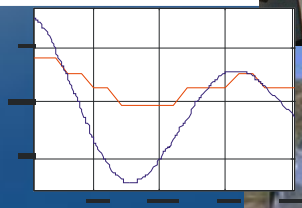
MATLAB® and Simulink® for Embedded System Design




```
591 /* l
592 rtb_either = power_w                    trol_b
593 || power_window_con_
594
595 /* Logic: '<S13>/al
596 * Inport: '<Root>/di
597 * Logic: '<S13>/ove
598 */
599 rtb_temp34 = power_w
600 && (!(rtb_either));
```



Pieter J. Mosterman
pieter.mosterman@mathworks.com

Senior Research Scientist
The MathWorks, Inc.

Introduction

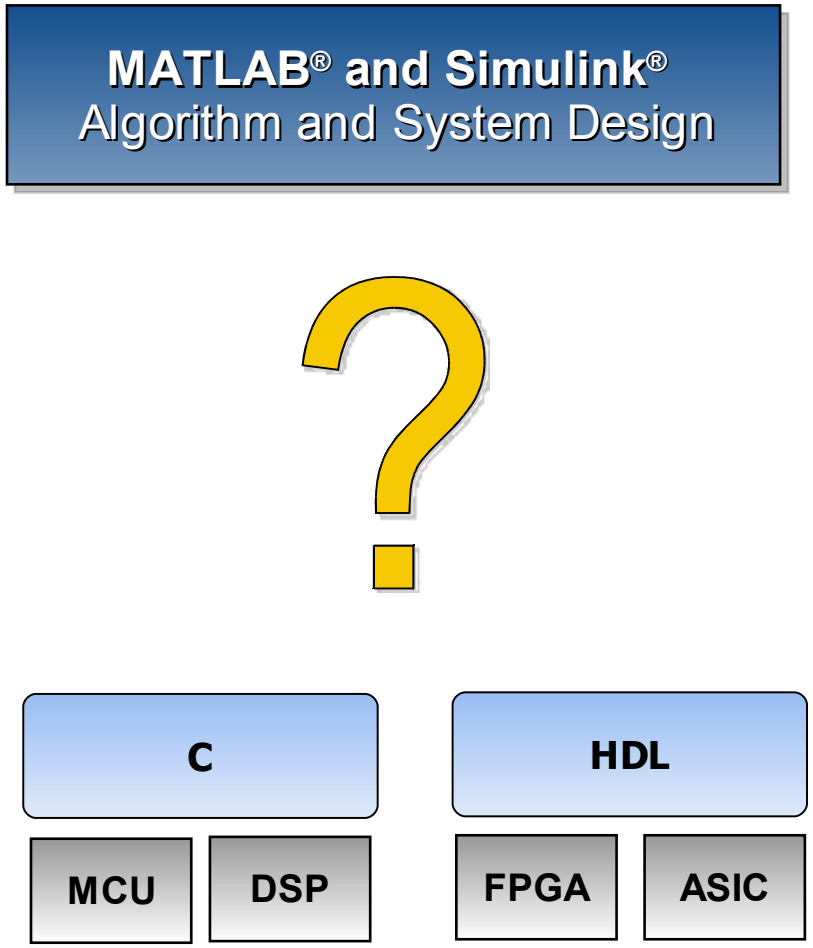
- Increasing complexity of embedded systems
- Complexity
  - Intricacy
  - Size
- Raising the Level of Abstraction
- Compilers to handle the complexity because of size

Agenda

- The System Design Challenge
- Software Design Flow
- Hardware Design Flow
- Summary

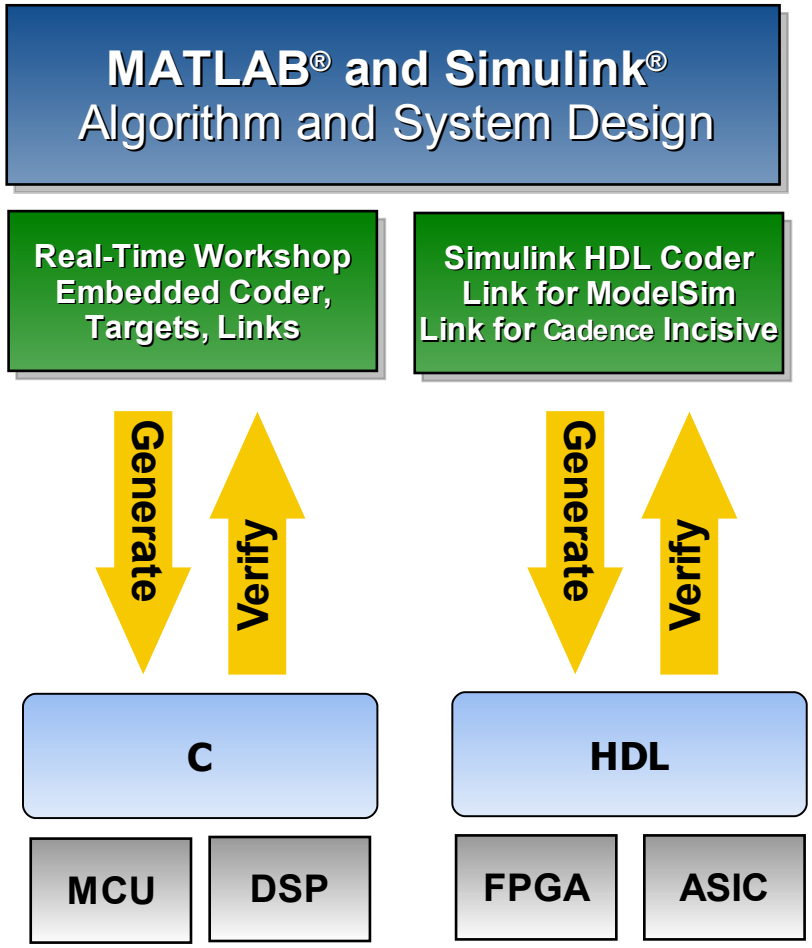
The System Design Challenge

- We design, simulate, and validate system models and algorithms in MATLAB® and/or Simulink®
- How can we implement and verify designs on DSPs and GPPs?
- How can we implement and verify designs on FPGAs and ASICs?



Integrated Design Flow for Embedded Software and Hardware

- Design, simulate, and validate system models and algorithms in MATLAB and Simulink
- Automatically generate C and HDL
- Verify hardware and software implementations against the system and algorithm models

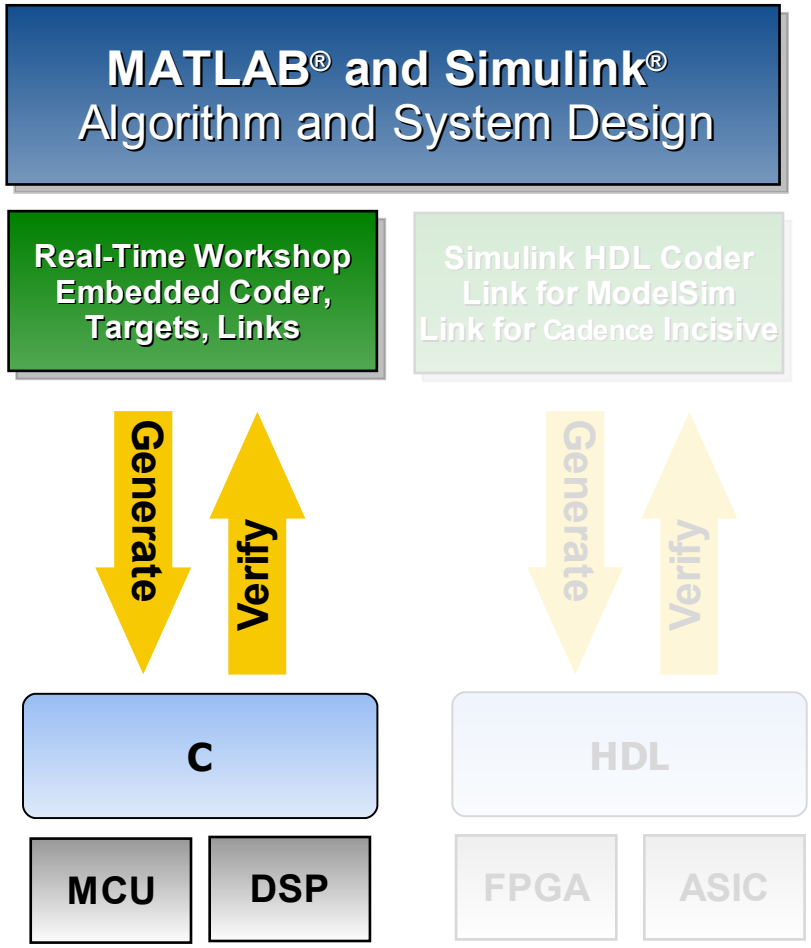


Agenda

- The System Design Challenge
- Software Design Flow
- Hardware Design Flow
- Summary

Integrated Design Flow for Embedded Software

- Implementation with automatic C code generation
- Implementation
  - Floating- and fixed-point code
  - Integration with downstream IDEs and tools
  - Links to verification
  - Device drivers
  - Optimization options



Representative Application – Video

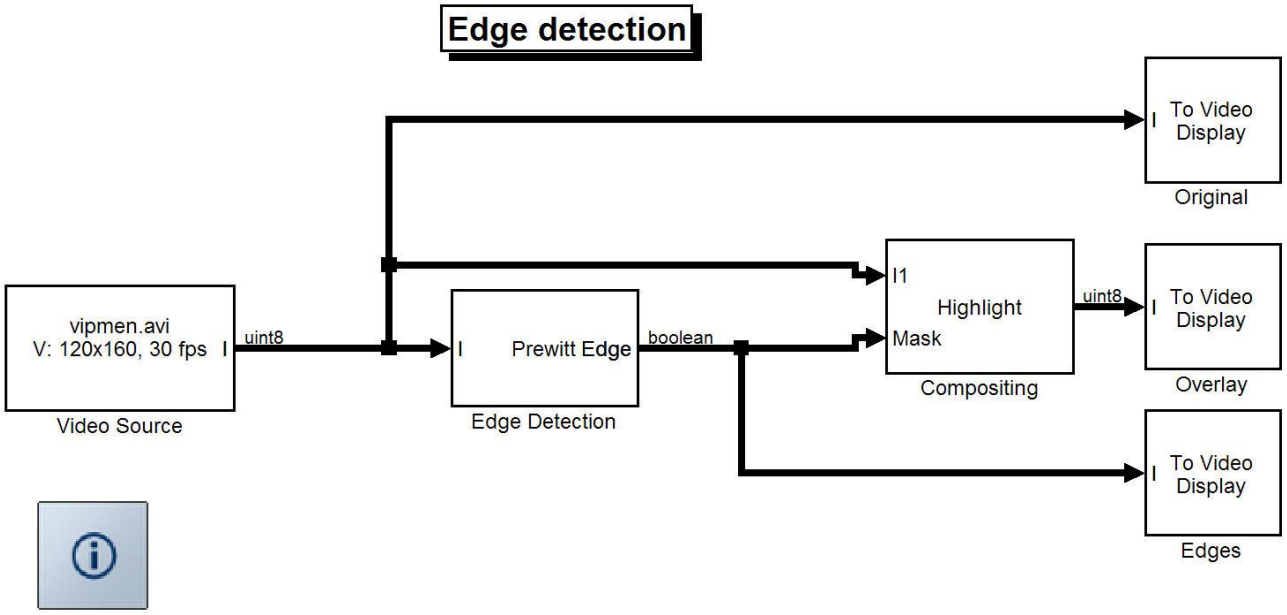
- Video system design and implementation
  - Encapsulates challenges: complexity, convergence, time-to-market
  - Sophisticated algorithms
  - Floating- and fixed-point issues
  - DSP or FPGA/ASIC implementations
- Design flows and steps shown directly applicable to other signal processing applications

Embedded Software Case Study: Video Edge Detection on a DSP

Live MATLAB Demo

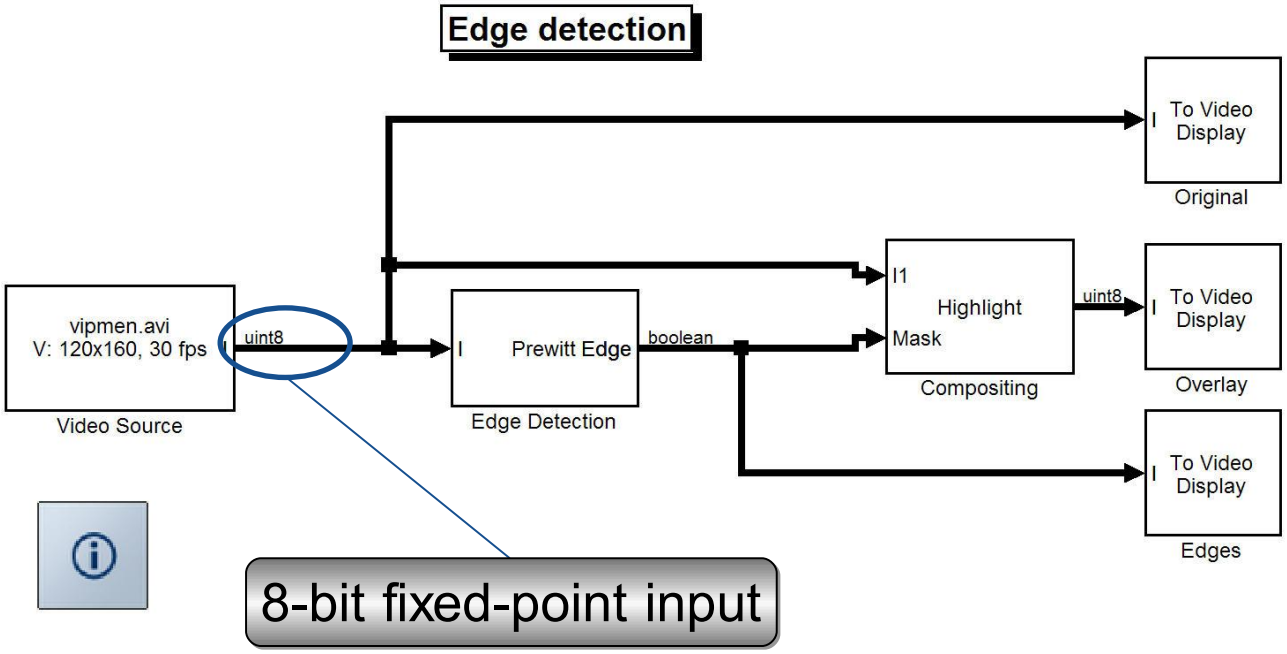
Example: Video Edge Detection

- Floating point video edge detection system based on Prewitt algorithm
- Compositing original image with detected edges
- Utilizes blocks from Video and Image Processing Blockset

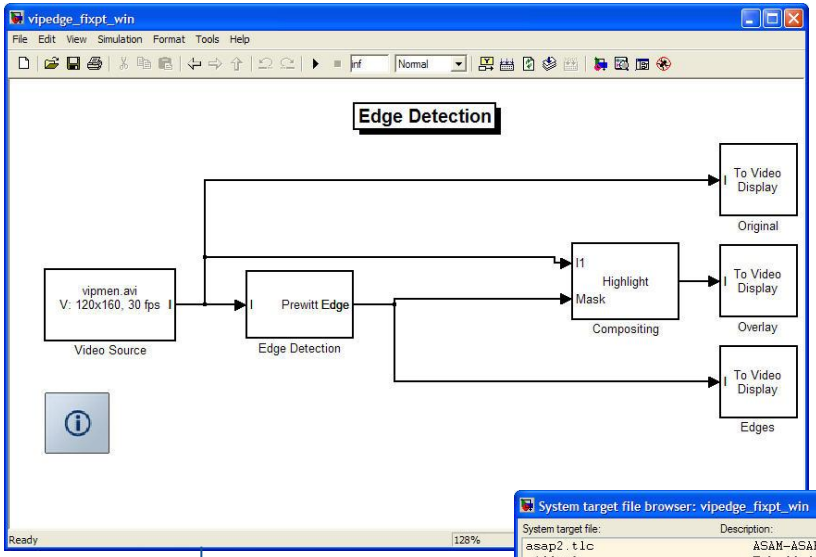


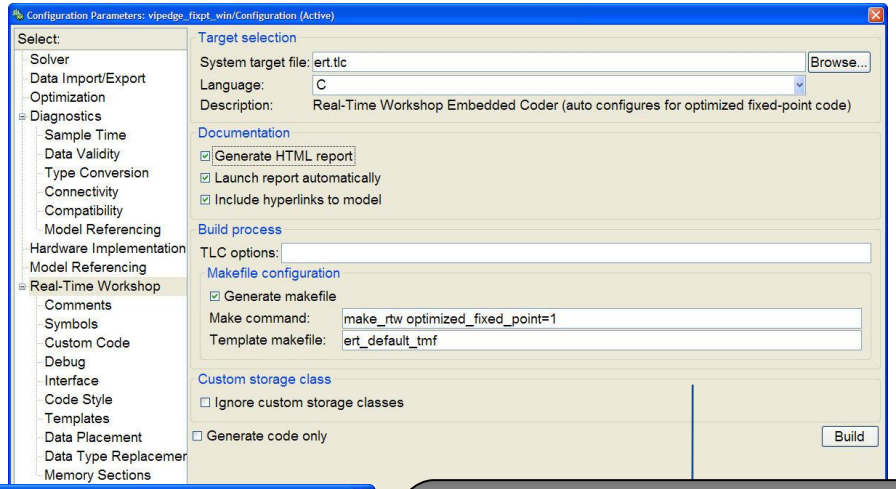
Converting to Fixed-Point

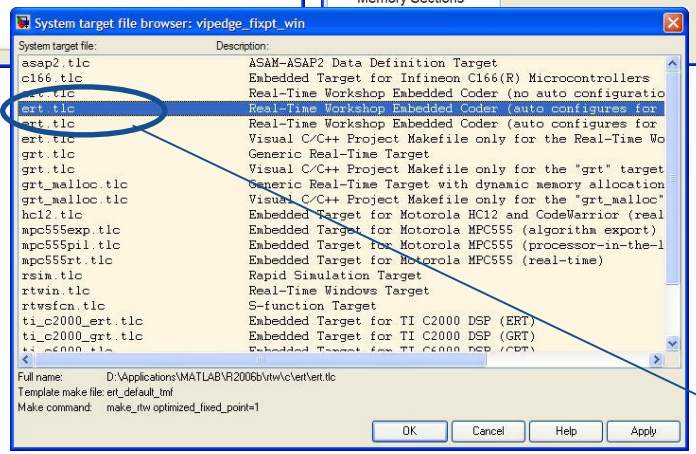
- Polymorphic blocks capable of floating- and fixed-point operation
- 8-bit input datatype – blocks inherit fixed-point data
- Simulink Accelerator provides fast fixed-point simulation



Automatic Code Generation for Implementation on GPPs and DSPs





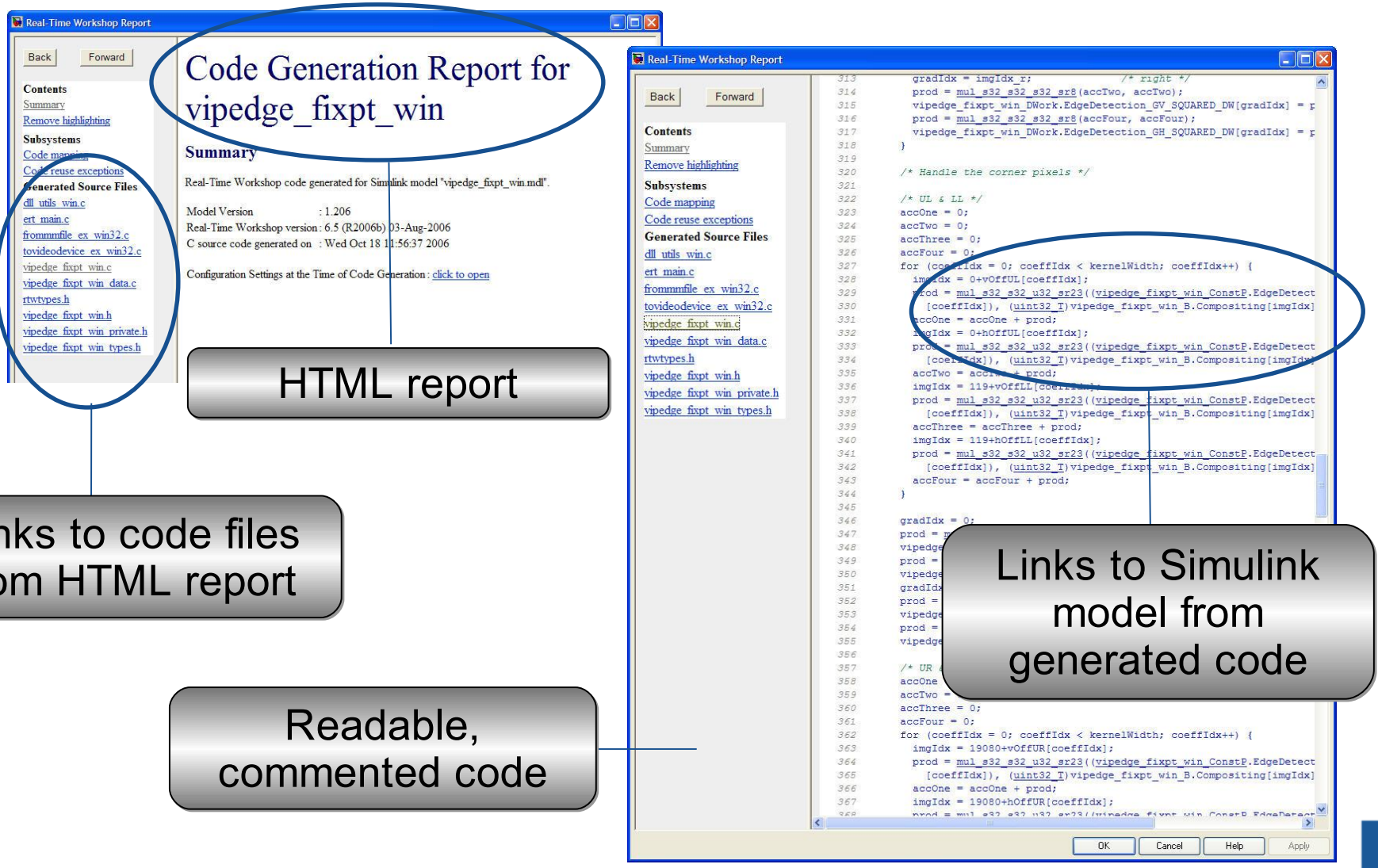


Fixed-point model

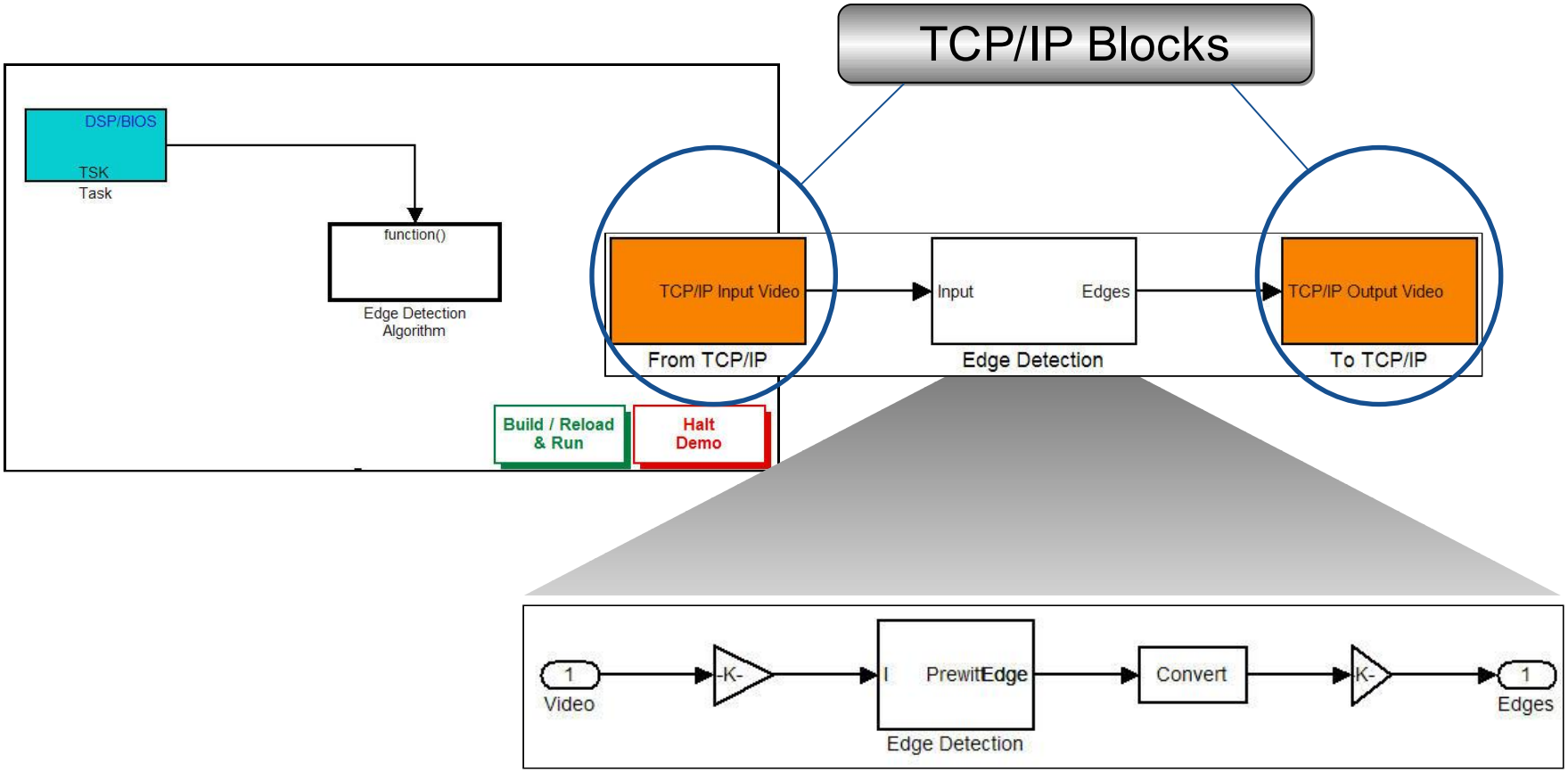
Code generation options and preferences

Select target or flavor of generated code

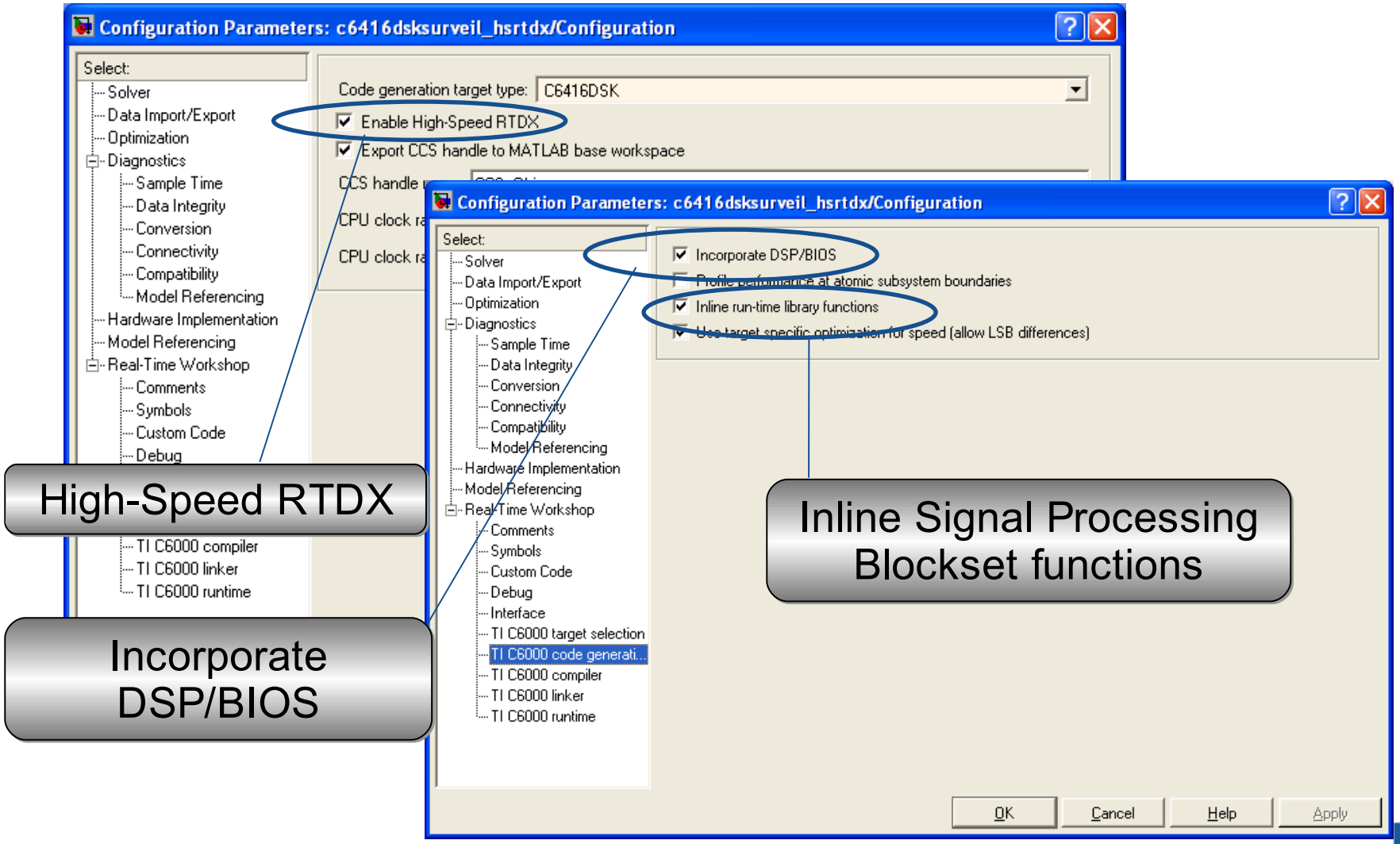
Code Generation Report



Video Edge Detection Embedded Software System on TI 6000™

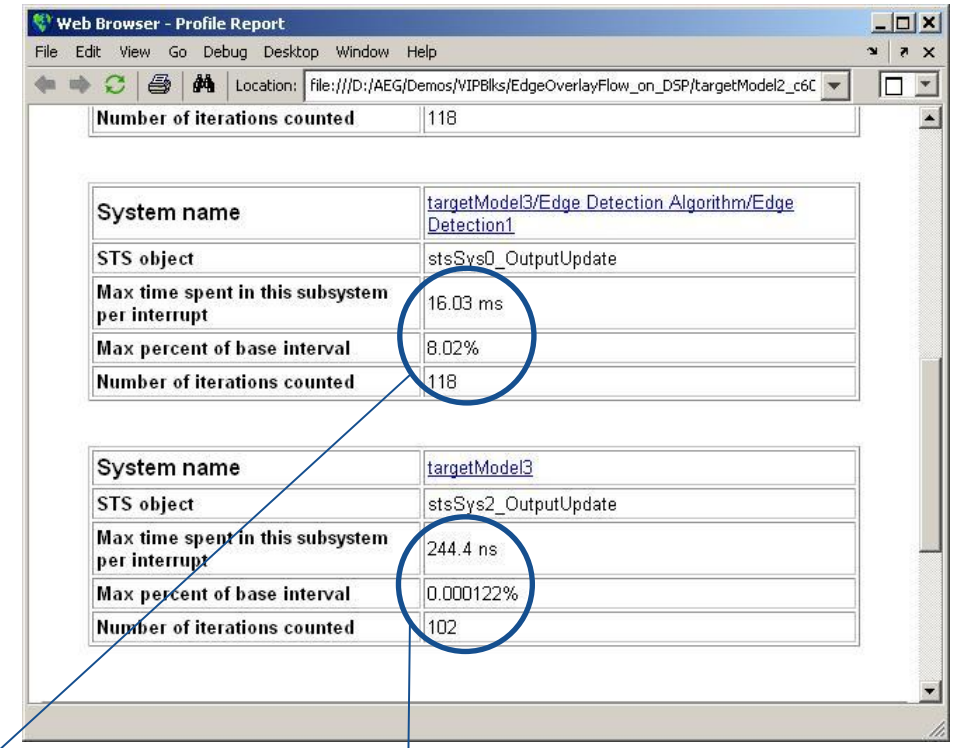


Target Code Generation Options



Code Execution on Target and Profiling

- Build and execute
  - Auto-generate C and ASM
  - Integrate RTOS and scheduler
  - Create full CCS project
  - Invoke compiler, linker, and download code
  - Run on target
- Profile code performance



System profiling includes entire DSP application code

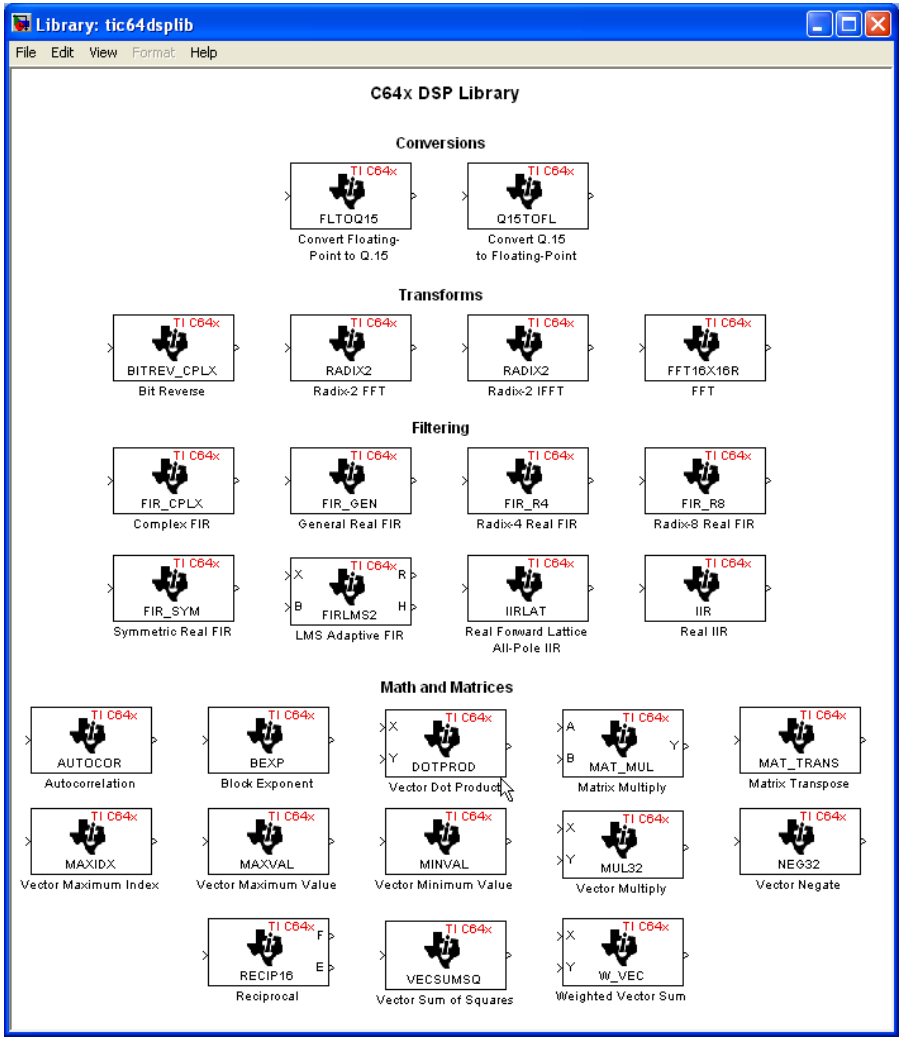
16.03 ms
8.02%
118

244.4 ns
0.000122%
102

Subsystem profiling

Code Optimization Options

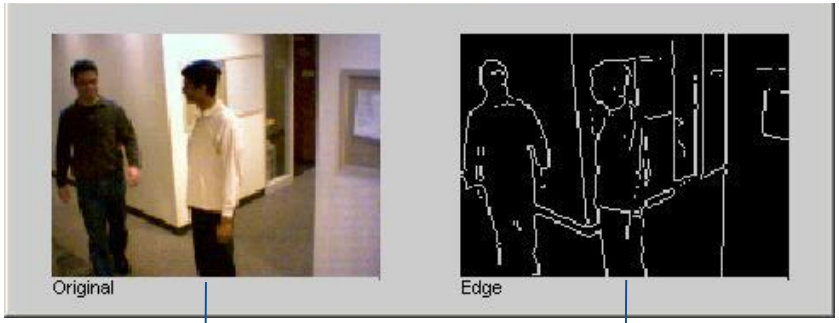
- Utilize target-specific blocks
  - C-callable assembler libraries
  - Simulate bit-true in Simulink
  - Generate calls to hand-optimized assembler libraries
  - Highly optimized implementation of core functionality
  - C62x and C64x fixed-point DSPs
- Manual optimization by user



Design Verification and Visualization:
MATLAB as software test bench

```
40
41  % Initialize motion threshold
42  global newThresh
43  lastThresh  = 0;
44  newThresh   = 1.5e5;
45
46  % Connect to CCS
47  CCS_Obj = connectToCCS(modelName);
48  saved_visibility = CCS_Obj.isvisible;
49  CCS_Obj.visible(1);
50
51  % Load application
52  loadApp(modelName, CCS_Obj);
53
54  % Run application
55  fprintf('Running application: %s\n', modelName);
56  CCS_Obj.run;
57
58  pause(3);
59
60  % Connect to the target
61  userPrompt = sprintf('Please enter the IP address or the host name of the %s board: ',
62  hostName = input(userPrompt, 's');
63  port = 49000;
64  fprintf('Connecting to TCP/IP server at: "%s:%d"\n', hostName, port)
65  connfd = tcpip(hostName, port);
66  set(connfd, 'OutputBufferSize', 64000);
67  set(connfd, 'InputBufferSize',  64000);
68  try
69      fopen(connfd);
70  catch
71      fprintf('Cannot established TCP/IP connection to "%s:%d".\n', hostName, port);
72      fprintf('Terminating demo.\n');
73      return;
74  end
75  set(connfd, 'ByteOrder', 'littleEndian');
76  fprintf('Established TCP/IP connection to "%s:%d"\n', hostName, port);
```

Visualize and debug embedded software with MATLAB

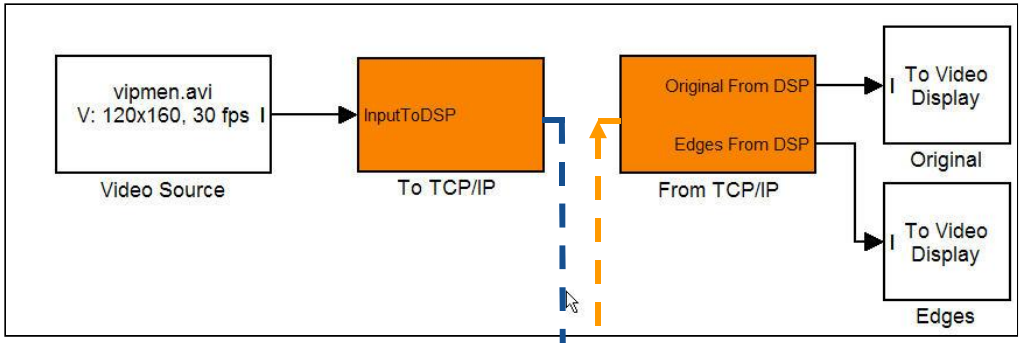


Input video

Captured video

MATLAB script (test bench)

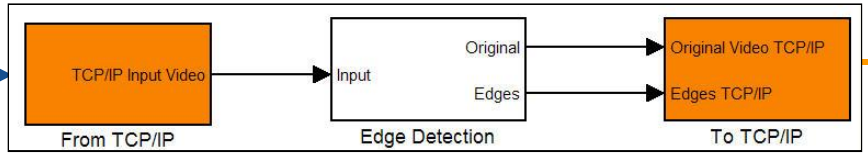
Design Verification and Visualization:
Simulink as software test bench



Processor and hardware-in-the-loop testing, simulation, visualization, and verification of embedded software with Simulink

Simulink test bench

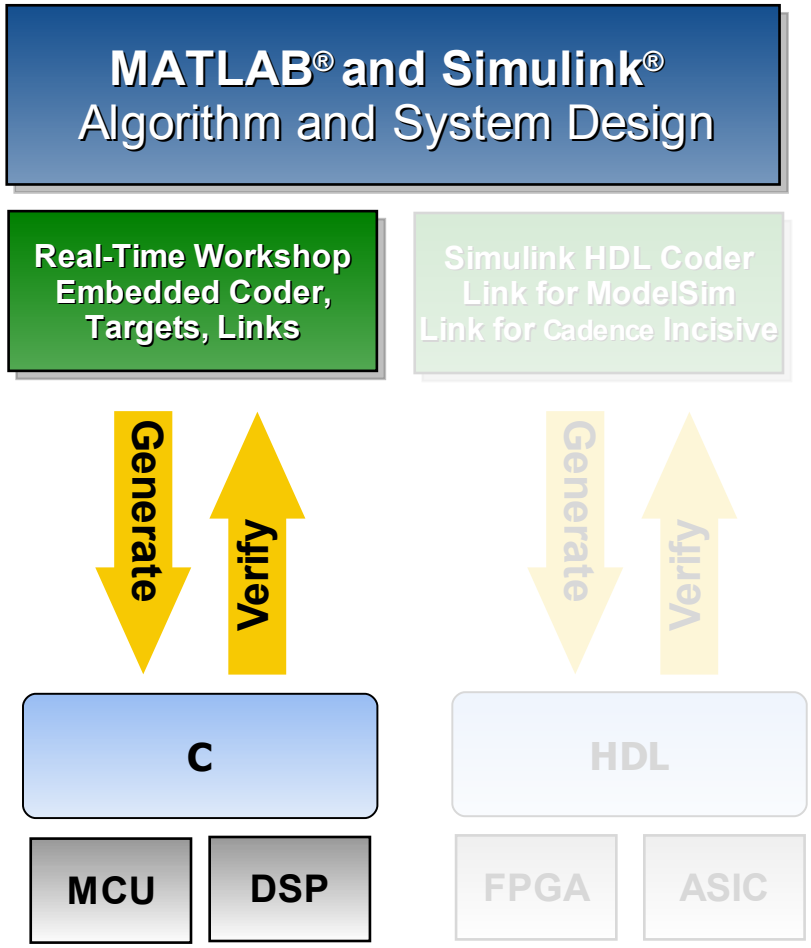
Device or design under test (DUT)



Simulink system design embedded on DSP

Review: Integrated Design Flow for Embedded Software

- Drive system development with an executable specification
- Quickly create complete working code base
- Use code profiles to identify and optimize bottlenecks
- Verify code with Links to IDEs and processors

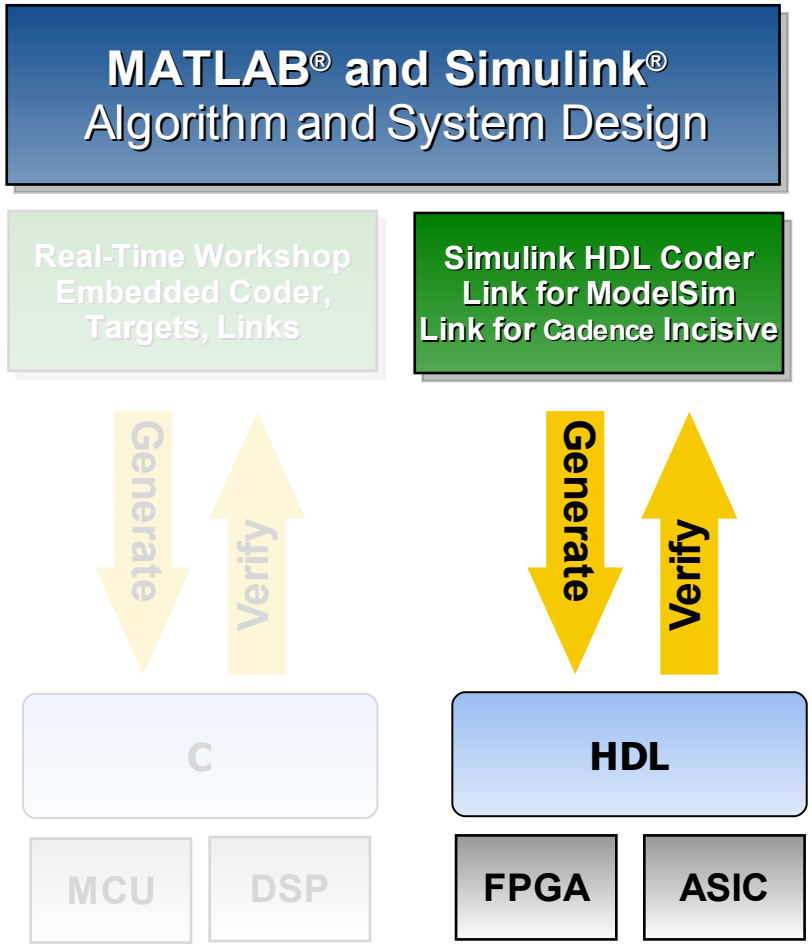


Agenda

- The System Design Challenge
- Software Design Flow
- Hardware Design Flow
- Summary

Integrated Design Flow for Hardware (FPGA and ASIC)

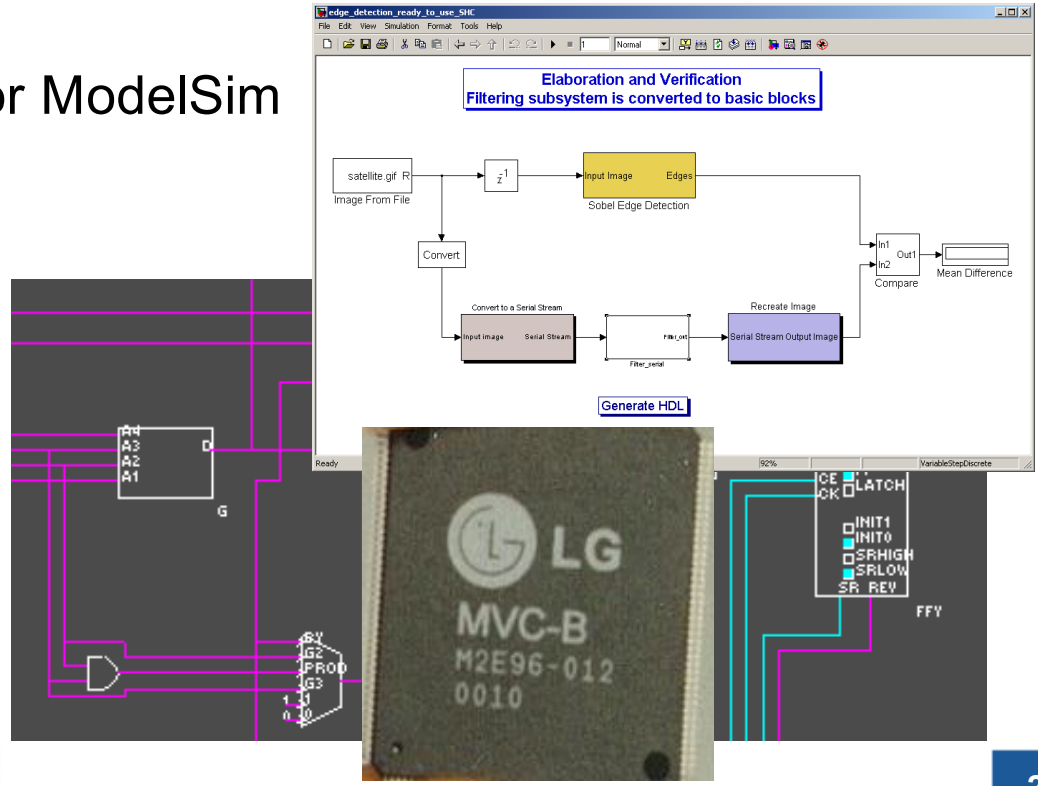
- Design elaboration
- Implementation
  - HDL code generation (VHDL and Verilog)
  - test bench generation
  - Links to verification
  - Integration with synthesis tools



What We Will Show In This Case Study

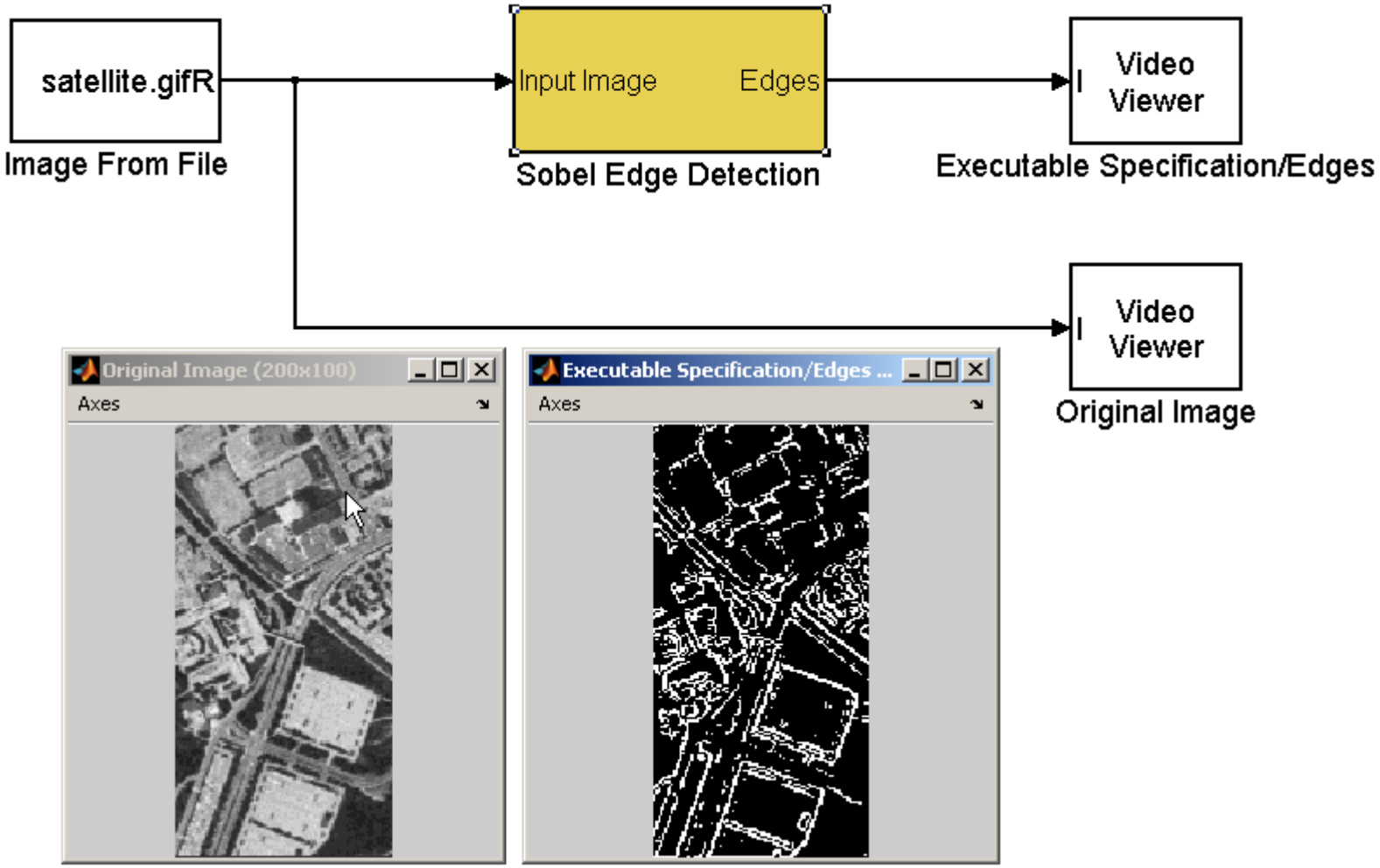
- Behavioral modeling and simulation
- Fixed-point modeling and simulation
- Design elaboration
- HDL generation
- Co-simulation using Link for ModelSim

Hardware Case Study:
Video Edge Detection
on an FPGA

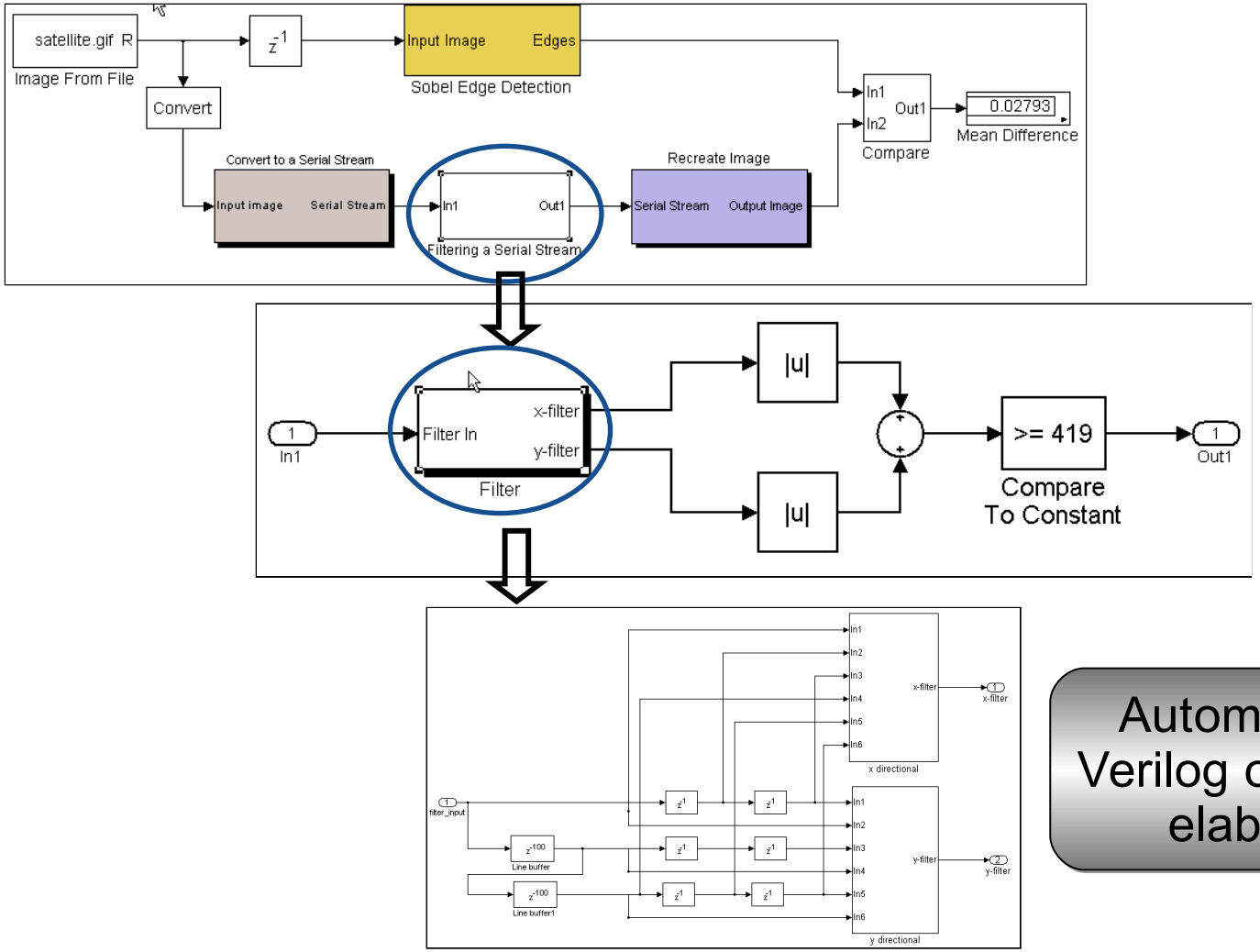


Live
MATLAB
Demo

Executable Specification

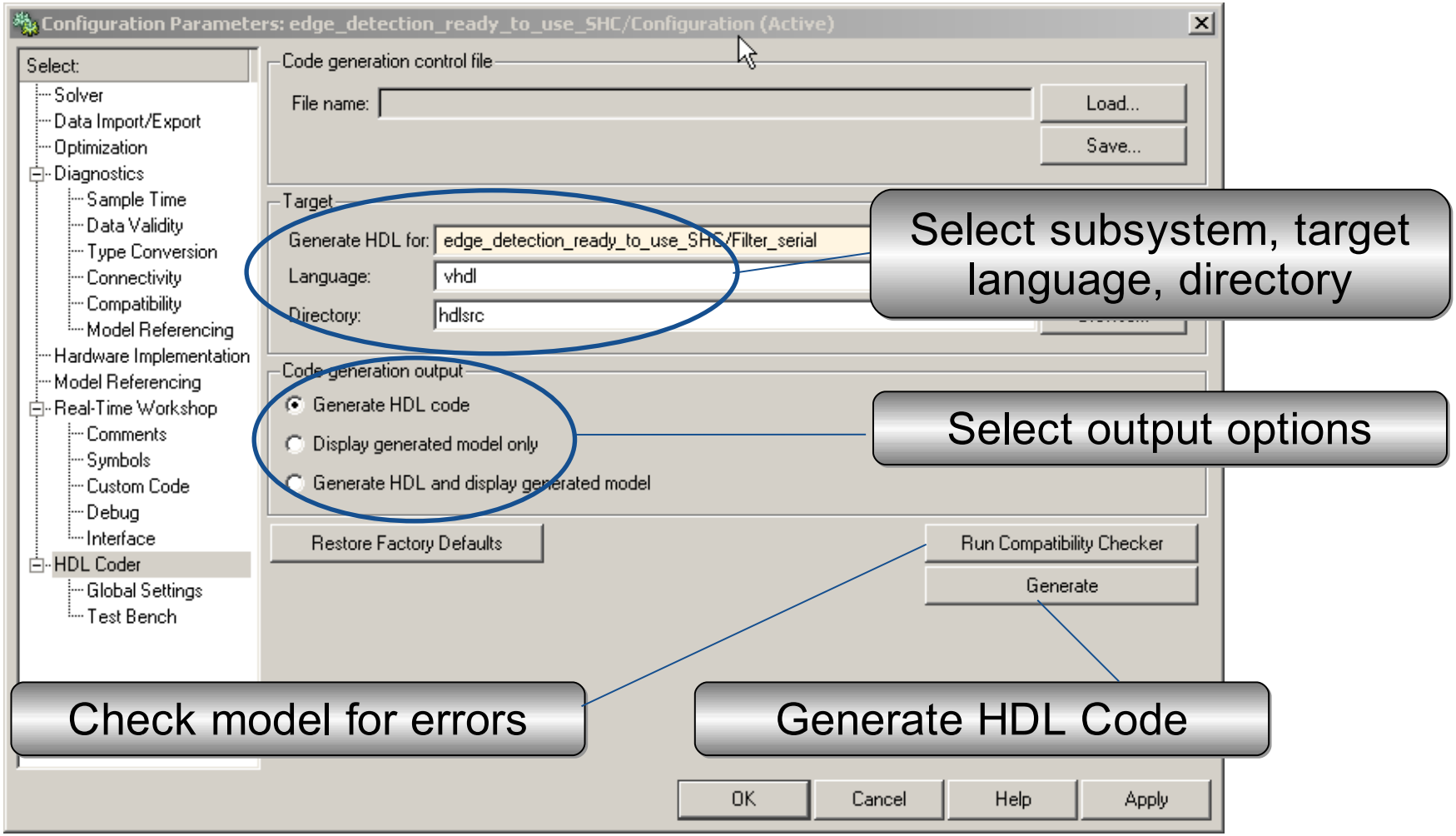


Design Elaboration

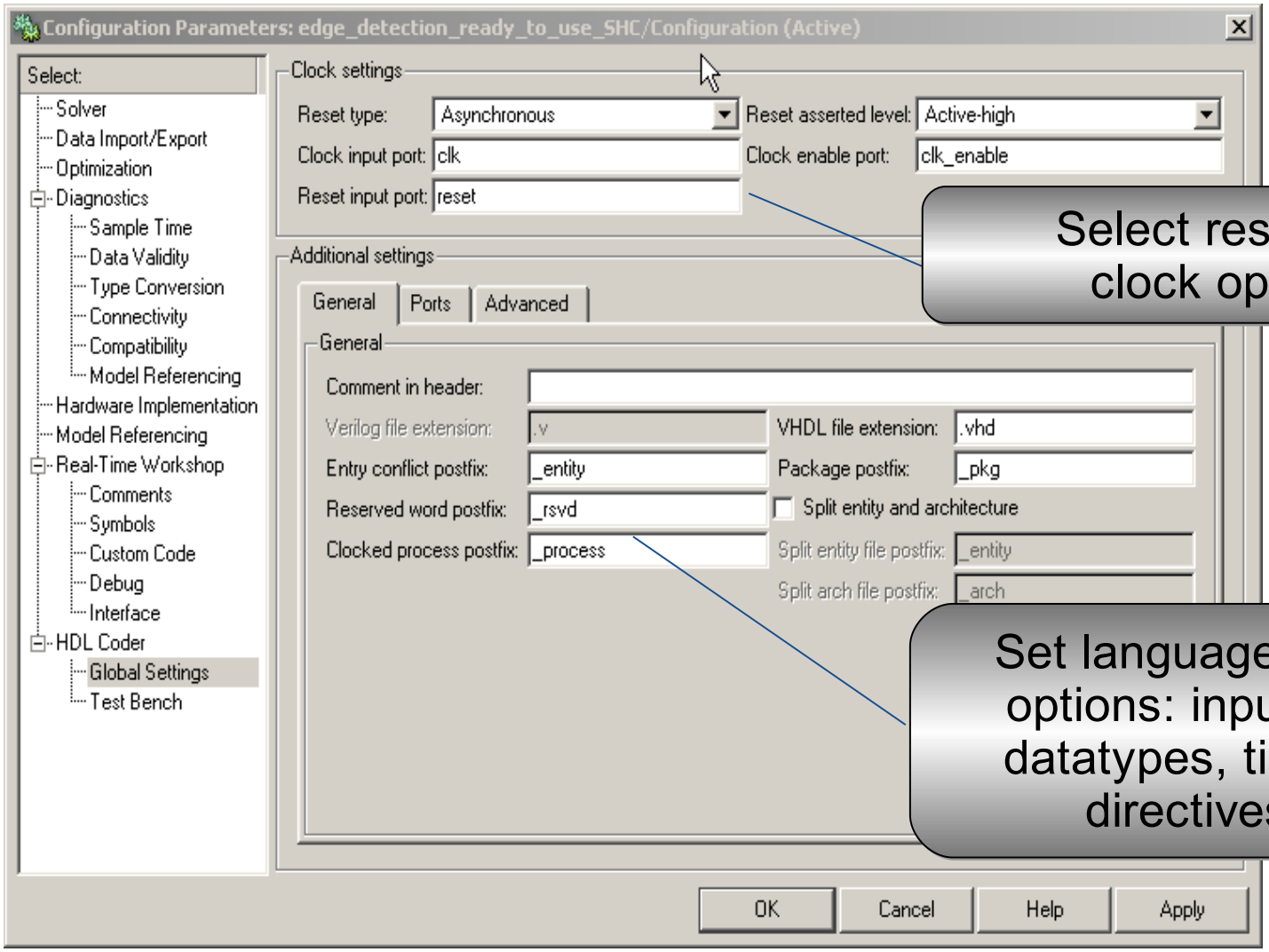


Automatically generate Verilog or VHDL code from elaborated models

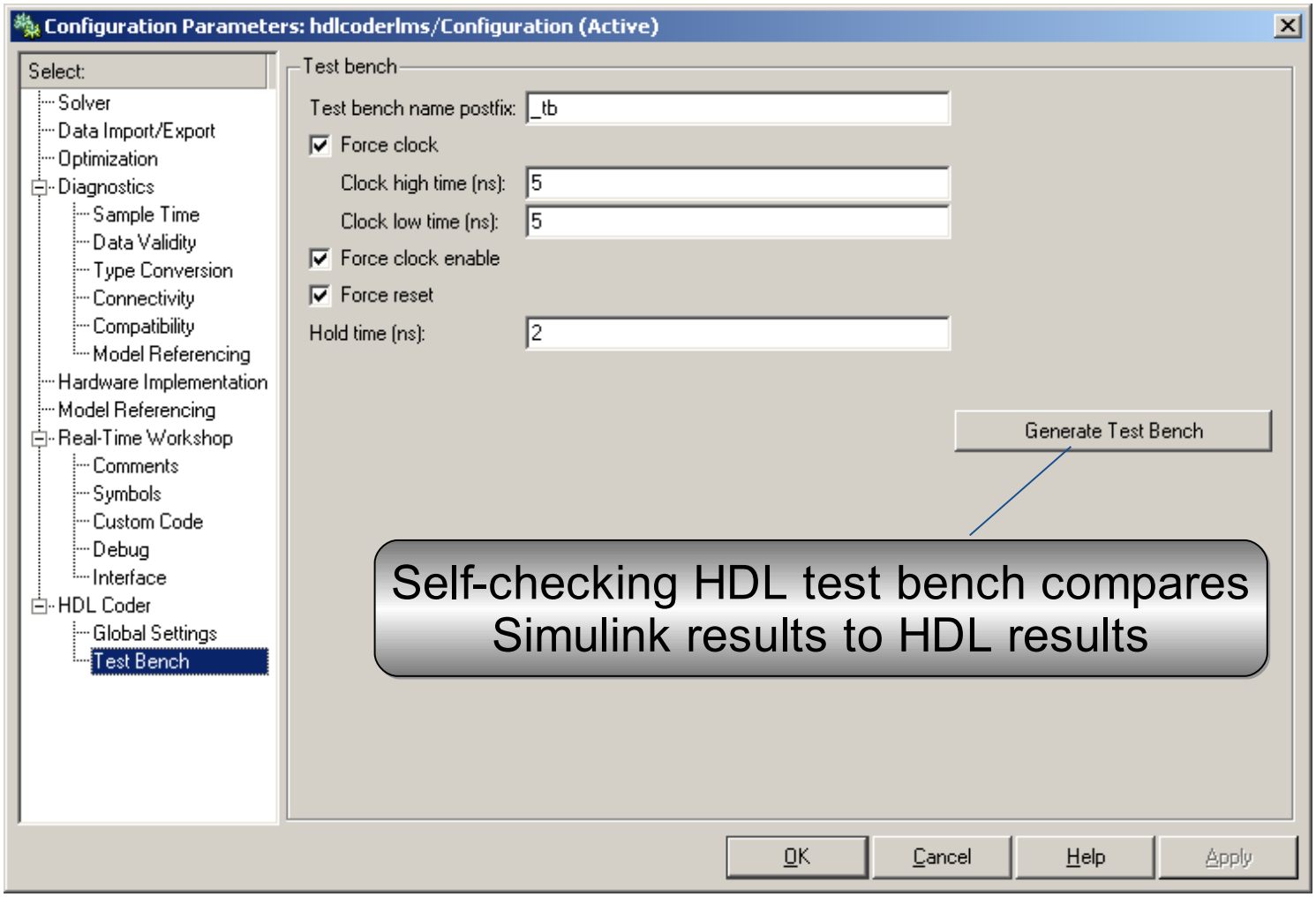
HDL Code Generation Using GUI



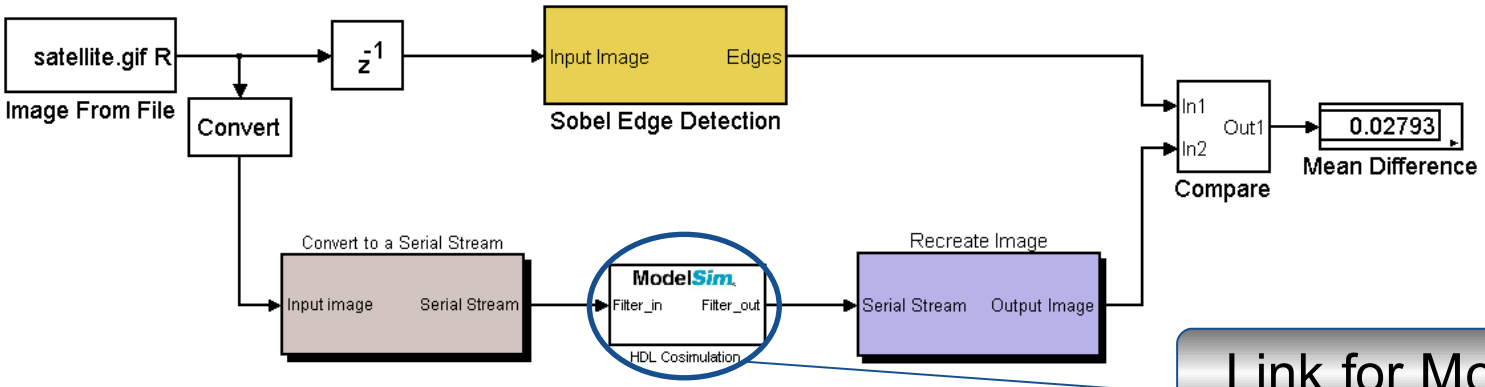
More Code Generation Options



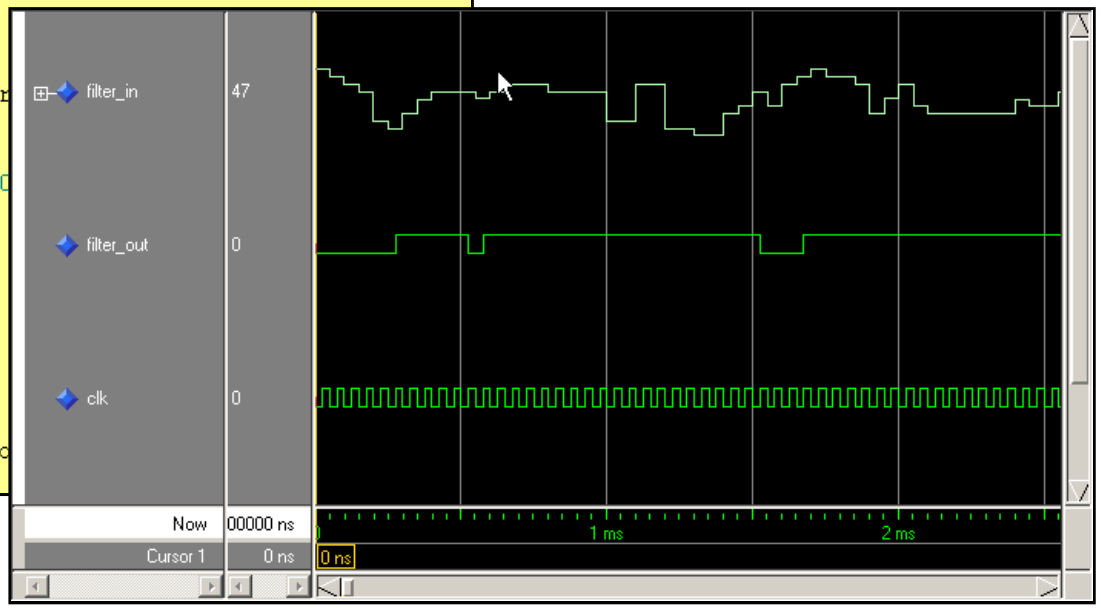
Automatically Generate Test Bench



Co-simulate Generated HDL



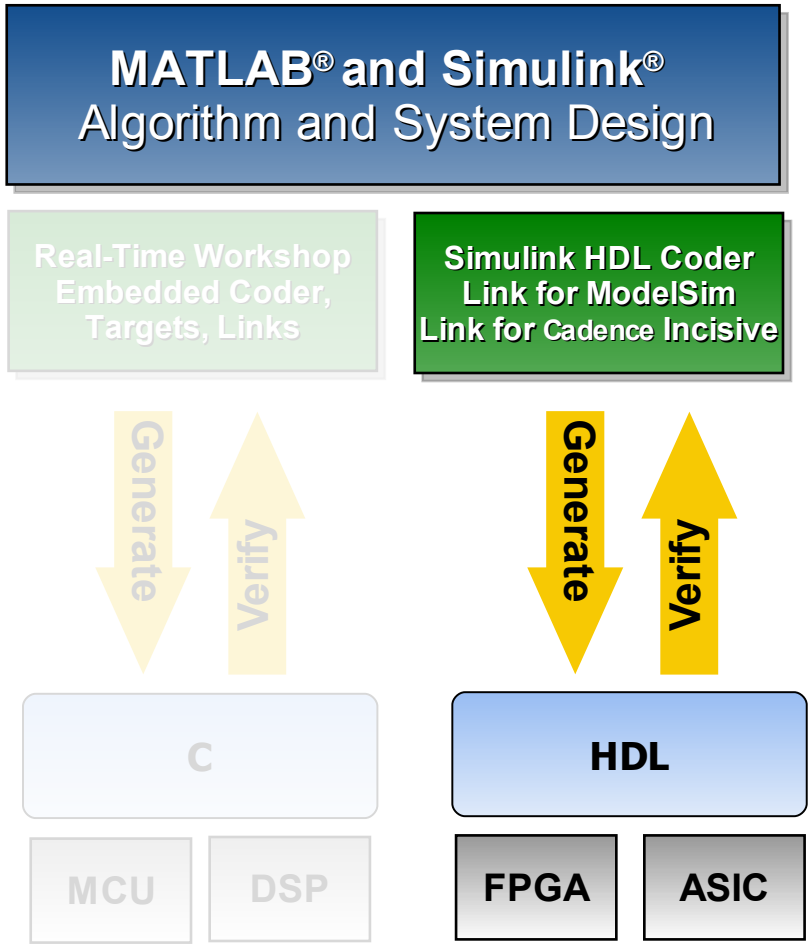

```
117           y_filter => y_directional_out1  -- ufix11
118           );
119       s <= signed(filter_input);
120
121       Delay2_process : PROCESS (clk, r
122       BEGIN
123         IF reset = '1' THEN
124           Delay2_out1 <= (OTHERS => '0
125         ELSIF clk'event AND clk = '1'
126           IF enb = '1' THEN
127             Delay2_out1 <= s;
128           END IF;
129         END IF;
130       END PROCESS Delay2_process;
131
132
133       s_1 <= std_logic_vector(Delay2_c
134
```



HDL code executing on ModelSim simulator

Review:
Integrated Design Flow for Hardware

- Drive system development with an executable specification
- Quickly create complete working HDL code base and test benches
- Verify code with Links to RTL simulators and synthesis tools



Agenda

- The System Design Challenge
- Software Design Flow
- Hardware Design Flow
- Summary

Summary

- Accelerate development using Model-Based Design
  - Generate
    - Real-Time Workshop
    - Simulink HDL Coder
  - Verify
    - Link for Cadence Incisive
- Design and verify software and hardware from MATLAB and Simulink