Hardware-Software Co-Design and Prototyping on SoC FPGAs

Puneet Kumar
Prateek Sikka
Application Engineering Team
Agenda

- Integrated Hardware / Software Top down Workflow for SoC FPGA’s, highlighting:
  - Model Based Design Workflow for SoC FPGA’s
  - Automatic Code Generation:
    - HDL code generation for the FPGA fabric and C-Code generation for the ARM MCU
  - Automatic Interface Logic Generation:
    - Generation of the interface logic and software between the FPGA and ARM.
  - Integrated Verification:
    - Integrated HDL Verification using HDL Co-simulation and FPGA-in-Loop

- Next Steps, Q&A
Agenda

- Integrated Hardware / Software Top down Workflow for SoC FPGA’s, highlighting:
  - Model Based Design Workflow for SoC FPGA’s
  - Automatic Code Generation:
    - HDL code generation for the FPGA fabric and C-Code generation for the ARM MCU
  - Automatic Interface Logic Generation:
    - Generation of the interface logic and software between the FPGA and ARM.
  - Integrated Verification:
    - Integrated HDL Verification using HDL Co-simulation and FPGA-in-Loop

- Next Steps, Q&A
Edge Detection Demo – Behavioral Model
You May Have Some Questions?

- How can we:
  - Implement designs on SoC FPGA’s?
  - Partition the HW and SW?
  - Generate the Interface Logic?
Design Challenges for Soc FPGA’s

- **FPGA Designers** not familiar with programming processors
- **DSP/Processor programmers** not familiar with FPGAs
- What should run on the FPGA vs. what should run on the ARM?
- No established rules for hooking up the interface between FPGA and ARM processor
Model-Based Design
Why Model-Based Design?

Requirements Development
Simulation
Code Generation
Continuous Verification
Model-Based Design:

From Concept to Production

- Automate regression testing
- Detect design errors
- Support certification and standards
- Generate efficient code
- Explore and optimize implementation tradeoffs

- Model multi-domain systems
- Explore and optimize system behavior in floating point and fixed point
- Collaborate across teams and continents

- C, C++
- VHDL, Verilog
- ARM
- FPGA

INTEGRATION

TEST & VERIFICATION

IMPLEMENTATION

DESIGN

REQUIREMENTS

RESEARCH
SoC FPGA Design Flow

User defines partitioning

MathWorks automates code and interface-model generation

MathWorks automates the build and download through the FPGA tools

RESEARCH

REQUIREMENTS

DESIGN

Top-Level System Model

Software Model

Hardware Model

IMPLEMENTATION

Embedded Coder®

HDL Coder™

ARM

FPGA

SoC FPGA Template

Xilinx/Altera Embedded System Integration

Real-Time Parameter Tuning and Verification
Model-Based Design for SoC FPGA
Solution: C and HDL Code Generation

- Design, execute, and verify algorithms in MATLAB
- Automatically generate C or HDL code
- Deploy generated code on hardware
Code Generation Products for VHDL/Verilog

- **HDL Coder™**: Automatically generate VHDL or Verilog from MATLAB code and Simulink Model.

- **MATLAB® Coder™**: Automatically generate C and C++ from MATLAB code.

- **Fixed-Point Designer™**: Provides fixed-point data types and arithmetic.
Code Generation Products for C/C++

- **Embedded Coder™**
  Automatically generate C and C++ optimized for embedded systems

- **Simulink® Coder™**
  Automatically generate C and C++ from Simulink models and Stateflow charts

- **MATLAB® Coder™**
  Automatically generate C and C++ from MATLAB code
Edge Detection Demo – Behavioral Model
Workflow for Video Image Processing

- Concept Development
- Algorithm Development
- Prototyping
- Architecture design
- Prototyping
- Chip design

Frame based
- Image/Video Engineer

Pixel based
- HW Engineer
Vision HDL Toolbox

*Design and prototype video image processing systems*

- Modeling hardware behavior of the algorithms
  - Pixel-based functions and blocks
  - Conversion between frames and pixels
  - Standard and custom frame sizes

- Prototyping algorithms on hardware
  - Efficient and readable HDL code
  - FPGA-in-the-loop testing and acceleration
Agenda

- Integrated Hardware / Software Top down Workflow for SoC FPGA’s, highlighting:
  - Model Based Design Workflow for SoC FPGA’s
  - Automatic Code Generation:
    - HDL code generation for the FPGA fabric and C-Code generation for the ARM MCU
  - Automatic Interface Logic Generation:
    - Generation of the interface logic and software between the FPGA and ARM.
  - Integrated Verification:
    - Integrated HDL Verification using HDL Co-simulation and FPGA-in-Loop

- Next Steps, Q&A
HW-SW Co-Design: It’s all about the **Workflow**

**MATLAB**

<table>
<thead>
<tr>
<th>Prepare model for IP core generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure Interface Logic</td>
</tr>
<tr>
<td>RTL Code Generation for IP Core</td>
</tr>
<tr>
<td>Generate Software/Hardware Model</td>
</tr>
<tr>
<td>Synthesis/ Bit File Generation</td>
</tr>
<tr>
<td>Deployment</td>
</tr>
</tbody>
</table>

**SIMULINK**

**HW-SW Co-Design Workflow**

- **C/C++**
- **VHDL or Verilog**

**HW-SW Co-design**
Model-Based Design flow using MATLAB/Simulink
from Algorithm to FPGA Implementation
SoC FPGA Model-Based Design Workflow
SoC FPGA Model-Based Design Workflow

MATLAB® and Simulink®
Algorithm and System Design

Simulink Model
Configure Interface Logic

- Prepare model for IP core generation
- Configure Interface Logic
- RTL Code Generation for IP Core
- Generate Software/Hardware Model
- Synthesis/ Bit File Generation
- Deployment
RTL Code Generation for IP Core

1. Prepare model for IP core generation
2. Configure Interface Logic
3. RTL Code Generation for IP Core
4. Generate Software/Hardware Model
5. Synthesis/ Bit File Generation
6. Deployment
Full Bidirectional traceability
SoC FPGA Model-Based Design Workflow

MATLAB® and Simulink® Algorithm and System Design

HDL IP Core Generation

Programmable Logic IP Core

Algorithm from MATLAB/Simulink

AXI Lite Accessible Registers

AxI4-Stream Video In

AxI4-Stream Video Out

External Ports

HDL IP Core Generation
SoC FPGA Model-Based Design Workflow

MATLAB® and Simulink® Algorithm and System Design

HDL IP Core Generation

EDK Integration

FPGA Bitstream

Zynq Platform
Generate Software/Hardware model

- Prepare model for IP core generation
- Configure Interface Logic
- RTL Code Generation for IP Core
- Generate Software/Hardware Model
- Synthesis/ Bit File Generation
- Deployment
SoC FPGA Model-Based Design Workflow

MATLAB® and Simulink® Algorithm and System Design

HDL IP Core Generation

EDK Integration

SW Interface Model Generation

FPGA Bitstream

SW Build

Zynq Platform

HW

SW

SW Interface Model Generation

SW I/O Driver Blocks

SW Interface Model
SoC FPGA Model-Based Design Workflow

- Real-time Parameter Tuning and Verification
  - External Mode
  - Processor-in-the-loop
Fast Prototyping and Iteration

Fast prototyping, iteration, and live probing/tuning directly on SoC FPGA hardware.
Zynq HW/SW Co-design Workflow Summary

1. HW Design
   - SW Interface Model

2. IP Core Generation
   - FPGA IP Core
   - Algorithm from MATLAB and Simulink
   - AXI Lite Accessible registers
   - External Ports

3. Generate SW Interface Model
   - Processor
   - AXI4-Lite Bus
   - AXI Lite Accessible registers
   - Algorithm from MATLAB and Simulink
   - External Ports

4. SW Build
   - FPGA Bitstream

Simulink Model
Agenda

- Integrated Hardware / Software Top down Workflow for SoC FPGA’s, highlighting:
  - Model Based Design Workflow for SoC FPGA’s
  - Automatic Code Generation:
    - HDL code generation for the FPGA fabric and C-Code generation for the ARM MCU
  - Automatic Interface Logic Generation:
    - Generation of the interface logic and software between the FPGA and ARM.
  - Integrated Verification:
    - Integrated HDL Verification using HDL Co-simulation and FPGA-in-Loop

- Next Steps, Q&A
## Enroll in Upcoming Training Courses

<table>
<thead>
<tr>
<th>No</th>
<th>Start Date</th>
<th>Course Name</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>02\textsuperscript{nd} - 03\textsuperscript{rd} May</td>
<td>Signal Processing with MATLAB</td>
<td>Bangalore</td>
</tr>
<tr>
<td>02</td>
<td>04\textsuperscript{th} - 05\textsuperscript{th} May</td>
<td>Image Processing with MATLAB</td>
<td>Bangalore</td>
</tr>
<tr>
<td>03</td>
<td>06\textsuperscript{th} May</td>
<td>Computer Vision with MATLAB</td>
<td>Bangalore</td>
</tr>
<tr>
<td>04</td>
<td>16\textsuperscript{th} - 17\textsuperscript{th} June</td>
<td>Machine Learning with MATLAB</td>
<td>Bangalore</td>
</tr>
<tr>
<td>05</td>
<td>11\textsuperscript{th} - 12\textsuperscript{th} July</td>
<td>Generating HDL Code from Simulink</td>
<td>Bangalore</td>
</tr>
<tr>
<td>06</td>
<td>13\textsuperscript{th} - 14\textsuperscript{th} July</td>
<td>Programming Xilinx Zynq SoCs with MATLAB and Simulink</td>
<td>Bangalore</td>
</tr>
</tbody>
</table>

[Link to www.mathworks.in/training]
MathWorks India – Services and Offerings

- Technical Evangelist
  - Product Training: [www.mathworks.in/training](http://www.mathworks.in/training)
- Application Engineering
- Technical Support local for India: [www.mathworks.in/myservicerequests](http://www.mathworks.in/myservicerequests)
- Customer Service for non-technical questions: [info@mathworks.in](mailto:info@mathworks.in)
- Knowledge and marketing resources: [www.mathworks.in](http://www.mathworks.in), webinars, seminars, conferences
Contact MathWorks India

URL: http://www.mathworks.in

E-mail: info@mathworks.in

Technical Support: www.mathworks.in/myservicerequests

Tel: +91-80-6632 6000

Fax: +91-80-6632 6010

MathWorks India Private Limited
9th Floor, ‘B’ Wing, Etamin Block
Prestige Technology Park II
Marathahalli – Sarjapur Ring Road
Bangalore – 560103, Karnataka India

Thank You for Attending
Talk to Us – We are Happy to Support You