A Case Study in Incremental Prototyping with Reconfigurable Hardware: DSRC Software Defined-Radio

Jared Dulmage jaredd@ee.ucla.edu
Minko Tsai minko@ee.ucla.edu
Michael Fitz fitz@ee.ucla.edu
Babak Daneshrad babak@ee.ucla.edu
UCLA DSRC Testbed

- Developed to provide a platform for DSRC-oriented research and testing
  - Algorithm design, implementation, and test in the high mobility wireless environment
  - Conformance testing of 3rd party DSRC devices
  - Physical layer interface for network and/or application layer testing
  - Channel sounding and signal measurement

- Hardware
  - Low-cost: COTS and/or low-cost custom subsystems
  - Portable: largest component is 19” box; further size reductions possible
  - Reprogrammable: FPGA is primary processing device

- Software
  - Modular: loosely coupled subsystems with intuitive interfaces
  - Highly parameterized: flexible simulation setups
  - High performance: efficient frame-based implementation, optional hardware-in-the-loop acceleration
Hardware Equipment

**Host PC**
- FPGA Baseband Processor, ADC/DAC
- Coax cable
- RF front-end

- Xilinx FPGA board by Nallatech
- Custom built IF to 2.4GHz converter by E-Monitoring
- Custom built 2.4GHz to 5.9GHz converter by Tomany Consulting
Design Software/Hardware Partitioning

- Design may be arbitrarily partitioned between hardware and software

**Software**
- Programmable
- Advanced debug
- Access PC resources
- Non-real-time
- Single sample interface
- Limited performance

**Hardware**
- Relatively static
- Limited debug
- Access hardware resources
- Real-time capable
- High performance
- Block data interface

**Synchronous interface**
- Fine-grained control simplifies debugging
- Straightforward implementation

**Asynchronous interface**
- Real-time operation for access to external hardware (e.g. ADC/DAC)

Design may be arbitrarily partitioned between hardware and software:
- Programmable
- Advanced debug
- Access PC resources
- Non-real-time
- Single sample interface
- Limited performance

- Relatively static
- Limited debug
- Access hardware resources
- Real-time capable
- High performance
- Block data interface

**Synchronous interface**
- Fine-grained control simplifies debugging
- Straightforward implementation

**Asynchronous interface**
- Real-time operation for access to external hardware (e.g. ADC/DAC)
Hardware/Software Partitioning

- FPGA co-processor
  - Hardware co-simulation block inserted into data path
  - Accelerated simulation and test

- PC co-processor
  - Software processing inserted within feedback loop to hardware co-simulation block
  - Virtual hardware testbench
Software Design

- Floating-point model
  - Block-diagram based graphical programming
  - Fully-integrated into testbench
  - Channel estimation, residual frequency offset tracking, soft-output demapper, deinterleaving, viterbi decoder

Floating-point simplified LLR soft-output demapper
Hardware Design

- Fixed-point simulation
- FPGA Synthesis direct from block diagram
- Synchronous and asynchronous co-simulation interfaces between hardware/software designs

Fixed-point simplified LLR soft-output demapper
Common Testbench

- Floating-point model
  - IEEE 802.11a/p standards conformant transmitter
  - Wideband, time variant, Rician fading channel model
  - DAC/ADC hardware emulation, RF impairments (time/frequency/phase offsets)
  - Visualizations and statistics
  - Graphical control
Virtual Vector Signal Analyzer

• PC co-processor architecture
  – FPGA as acquisition card
  – visualizations in software

1Vpp, 10 Hz sine wave

1Vpp, 25 MHz sine wave; 1024-pt FFT, 16 count spectral average; approximately 60 dB SFDR
Conclusions

• Integrated hardware/software design keys
  – **Subsystem interface definition**
  – Hardware design developed around floating-point data path

• Various points of partitioning allow a single model/testbench to accomplish several goals
  – Design and debug subsystem algorithms and hardware designs with a unified testbench
  – Test integrated hardware design in virtual environment for maximum control and monitoring capability
  – Use UI integrated with testbench to perform bench and field tests
  – Rapid design iterations possible once baseline system completed including integrating floating-point algorithms into hardware data path
## Development Stages

<table>
<thead>
<tr>
<th>Method</th>
<th>Theoretical accuracy</th>
<th>Programmability</th>
<th>Design Effort</th>
<th>Execution Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulink floating-point</td>
<td>High</td>
<td>High</td>
<td>Low</td>
<td>Moderate</td>
</tr>
<tr>
<td>Simulink fixed-point</td>
<td>Moderate</td>
<td>High</td>
<td>Low</td>
<td>Moderate</td>
</tr>
<tr>
<td>Sysgen SW sim</td>
<td>Low</td>
<td>Moderate</td>
<td>Moderate</td>
<td>Low</td>
</tr>
<tr>
<td>Sysgen SS HW Cosim</td>
<td>Low</td>
<td>Low</td>
<td>Moderate</td>
<td>Moderate</td>
</tr>
<tr>
<td>Sysgen FR HW Cosim</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>High*</td>
</tr>
</tbody>
</table>

* in theory
### Development Stages

<table>
<thead>
<tr>
<th>Simulation Method</th>
<th>Accuracy</th>
<th>Programmability</th>
<th>Design Effort</th>
<th>Simulation Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulink floating-point sim</td>
<td>Most accurate</td>
<td>Most programmable</td>
<td>Least design effort</td>
<td>Moderate sim performance</td>
</tr>
<tr>
<td>Simulink fixed-point sim</td>
<td>Least accurate</td>
<td>Moderately programmable</td>
<td>Moderate design effort</td>
<td>Low sim performance</td>
</tr>
<tr>
<td>System Generator Simulation</td>
<td>Least accurate</td>
<td>Least programmable</td>
<td>Most design effort</td>
<td>High sim performance</td>
</tr>
<tr>
<td>Sysgen sync hw cosim</td>
<td>Least accurate</td>
<td>Least programmable</td>
<td>Most design effort</td>
<td>High sim performance</td>
</tr>
<tr>
<td>Sysgen async hw cosim</td>
<td>Least accurate</td>
<td>Least programmable</td>
<td>Most design effort</td>
<td>High sim performance</td>
</tr>
</tbody>
</table>
FPGA Co-processor

- FPGA acts as co-processor to host PC
  - Hardware co-simulation block inserted into data path
- Accelerated simulation and test
  - subsystem processing offloaded to hardware
  - Isolate hardware subsystem to aid debugging effort
Development Stages: System

- Subsystems may be developed concurrently.
- Integration may proceed in parallel until final integration.

Diagram:

Subsystem 1 --> Integrated 1
Subsystem 2 --> Integrated 1
Subsystem 3 --> Integrated 2
Subsystem n --> Integrated System

...
PC Co-processor

- PC acts as co-processor to FPGA hardware design
  - Break out data path from hardware co-simulation block and feed back after software processing
- Virtual environment simulations
  - Test hardware implementations using a simulated testbench
- Software implementation of complex subsystem
  - Augment hardware design with software-based subsystem where hardware implementation is cost prohibitive
PC monitor and control

- PC acts as monitor and control for FPGA design
  - FPGA designed with monitor and control hooks
- Measurements and visualizations
  - Software tools and testbenches to analyze hardware performance
- Adaptive and/or interactive control
  - Software to provide UI or automatic control and status monitoring
Development Stages

- **Floating-point simulation**
  - Debug algorithms, finalize data interfaces
  - Determine performance reference

- **Fixed-point simulation**
  - Determine bit widths and precision
  - Quantify performance degradation

- **Synchronous hardware co-simulation**
  - Ensure timing closure and correct synthesis
  - Determine resource usage

- **Asynchronous hardware co-simulation**
  - Integrate with block data interface
  - Run real-time/accelerated tests

- **Deployed prototype**
  - Test standalone access to FPGA resources without Simulink/Sysgen co-simulation
  - Stress tests in real-world environment