DSP Tutorial II

Real-life Implementation
Collider-Accelerator Complex

Kevin Smith
Tom Hayes
Freddy Severino
Kevin Mernick
Denny Nembhard
Geetha Narayan
Outline

- High-level Algorithm flow
- Simulink with XILINX System Generator
- Real life example – concept to proof
- Demo with HW-SW Co-Simulation
- Designs in development
Survey of System Designers

What do you use for high-level algorithm development?

- Mostly MATLAB
- MATLAB & Simulink
- Mostly C/C++
- None – just hand-code RTL or use IP
<table>
<thead>
<tr>
<th>Option</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mostly MATLAB</td>
<td>38.2%</td>
</tr>
<tr>
<td>MATLAB &amp; Simulink</td>
<td>29.1%</td>
</tr>
<tr>
<td>Mostly C/C++</td>
<td>12.7%</td>
</tr>
<tr>
<td>None – just hand-code RTL or use IP</td>
<td>20.0%</td>
</tr>
</tbody>
</table>
Objective

Demonstrate the integrated flow from system design to implementation of real-time DSP applications on FPGAs

- Use MATLAB / Simulink to validate a simple DSP algorithm
- Implement algorithm with XILINX System Generator
- Verifying the DSP system using Simulink and HDL simulator
- Preparing design for Co-Simulation on SP605 (Spartan-6) Board
- Performing Hardware/Software Co-Simulation for the DSP system

Developers with little FPGA design experience can quickly create FPGA implementations of DSP algorithms in a fraction of traditional RTL development times.
Outline

- High-level Algorithm flow
- **Simulink with XILINX System Generator**
- Real life example – concept to proof
- Demo with HW-SW Co-Simulation
- Designs in development
System Generator Design Flow
System Generator Features

- **DSP modeling**
  Xilinx blockset contains about 100 IPs with functions such as
  - signal processing (e.g., FIR filters, FFTs)
  - error correction (e.g., Viterbi decoder, Reed-Solomon encoder/decoder)
  - arithmetic, memories (e.g., FIFO, RAM, ROM), and digital logic

- **Bit and cycle accurate floating and fixed-point implementation**

- **Automatic code generation of VHDL or Verilog from Simulink**
  - Integrate legacy RTL

- **Hardware co-simulation**
  Validate working hardware and accelerate simulations using
  - Ethernet (10/100/Gigabit)
  - JTAG communication between a hardware platform and Simulink

- **Hardware / software co-design of embedded systems**
  Build and debug DSP co-processors for the Xilinx MicroBlaze™ soft processor core
Design Flow Strategies

- **Algorithm Exploration**
  - useful for algorithm exploration and model analysis
  - for architecture exploration such as HW/SW partitioning
  - estimate the cost and performance of an implementation in hardware

- **Implementing Part of a Larger Design**
  - ideal to implement data paths and control
  - less suited for sophisticated external interfaces
  - design can be exported as an application specific IP to be integrated into a system

- **Implementing a Complete Design**
  - ‘Generate’ button instructs System Generator to translate the design
  - HDL that implements the design
  - HDL testbench that transforms results from Simulink simulations to be used in a logic simulator
  - scripts that guide downstream tools, such as XST for synthesis
Outline

- High-level Algorithm flow
- Simulink with XILINX System Generator
- **Real life example – concept to proof**
- Demo with HW-SW Co-Simulation
- Designs in development
Coherent Phase Detector for AGS Booster

Determine beam bunch phase with respect to net RF voltage per turn in a Heterodyne System

LO = $f_{RFsys} + 10.7$ MHz

Phase detector is a main building block in loop control applications
Digital phase detecting method making use of 90° phase shift property of Hilbert transform.

- Heterodyne signals are sampled by A/Ds
- An analytic signal is generated from a real signal by using the Hilbert transform
- An FIR filter is used for the approximation of the Hilbert transform
- Signals are down-converted to DC by mixing
- After LPF and decimation phase difference is computed
Received signals:

\[ s_1(t) = a_1(t) \cos(\omega t + \Phi_1) \]
\[ s_2(t) = a_2(t) \cos(\omega t + \Phi_2) \]

Hilbert Transform is equivalent to a 90° phase shift linear filter

\[ H[s_1(t)] = a_1(t) \sin(\omega t + \Phi_1) \]
\[ H[s_2(t)] = a_2(t) \sin(\omega t + \Phi_2) \]

\[ Q = s_1(t) \cdot H[s_2(t)] - H[s_1(t)] \cdot s_2(t) \]
\[ = a_1(t) \cdot a_2(t) \cdot \sin(\Phi_2 - \Phi_1) \]

\[ I = s_1(t) \cdot s_2(t) + H[s_1(t)] \cdot H[s_2(t)] \]
\[ = a_1(t) \cdot a_2(t) \cdot \cos(\Phi_2 - \Phi_1) \]

Phase Difference:

\[ \Delta \Phi = (\Phi_2 - \Phi_1) = \tan^{-1}(Q/I) \]
AGS Phase Detector Simulink Model

\[ H[s_1(t)] \]

\[ H[s_2(t)] \]

s_1(t)

s_2(t)

Q

I

ΔΦ

Brookhaven Science Associates

LLRF 2013
Simulink Toolbox & Test Bench
Xilinx library of bit and cycle-true models

Basic Elements

- System Generator
- BitBasher
- Concat
- Counter
- Expression
- Inverter
- Mux
- Reinterpret
- Serial to Parallel
- Time Division Multiplexer
- Addressable Shift Register
- Black Box
- Constant
- Delay
- Gateway In
- LFSR
- Parallel to Serial
- Relational
- Slice
- Time Division Demultiplexer
- Assert
- Clock Enable Probe
- Convert
- Down Sample
- Gateway Out
- Logical
- Register
- Reset Generator
- Time Division Demultiplexer

DSP Blocks

- CIC Compiler 2.0
- CIC Compiler 3.0
- CORDIC 4.0
- Complex Multiplier 3.1
- Complex Multiplier 5.0
- DDS Compiler 5.0
- DDS Compiler 4.0
- DSP48
- DSP48 Macro
- DSP48 macro 2.0
- DSP48E
- Divisor Generator 4.0
- Divisor Generator 3.0
- Fast Fourier Transform 7.1
- Fast Fourier Transform 8.0
- FIR Compiler 5.0
- FDATool
- LFSR
- Opmode
AGS PD System Generator Model

Configure for generation of
- Netlist or bitstream
- Hardware in the loop
- Export as hardware peripheral for an embedded processor

Gateway In block
- Double precision input data is quantized into Fixed Point representation

Gateway Out block
- Convert the fixed point representation into Simulink floating point

Hilbert filter
CIC filter
CORDIC
Video on Firmware Generation

(2 min)
Outline

- High-level Algorithm flow
- Simulink with XILINX System Generator
- Real life example – concept to proof
- **Demo with HW-SW Co-Simulation (HW in the Loop)**
- Designs in development
HW / SW Co-Simulation

HW running on SP605

HW / SW interfaces handled automatically by System Generator
Outline

- High-level Algorithm flow
- Simulink with XILINX System Generator
- Real life example – concept to proof
- Demo with HW-SW Co-Simulation
- Designs in development
Bunch-by-Bunch Phase Detector IP

- Fast feedback bunch phase measurement

Ref: Kevin Smith - Overview of LLRF Developments at the BNL Collider-Accelerator Complex
Bunch-by-Bunch Phase Detector System
Building Blocks of DSP Datapath

• NCO Subsystem
  • LUT based
  • Tunable phase offset
  • Generate orthogonal reference signals for down conversion

• CIC (cascaded integrator comb) Filters
  • Ideal for large rate changes and narrow band low pass filtering
  • Implementation efficient - no multipliers, only addition and subtraction are needed.

• CORDIC
  • Many functional configurations:
    • Vector rotation (polar to rectangular),
    • Vector translation (rectangular to polar),
    • Sin and Cos
    • Sinh and Cosh
    • Atan and Atanh
    • Square Root
CIC Filter Specifications

- Narrow band low pass filter of consecutive bunches to determine phase and magnitude
- Bunch train at 9.386 MHz
- Support variable decimation rate R
- Processing clock 100 MHz
- A/D sampling rate 100 Msps
- FPGA resources limited
- Parameterizable Bit Growth
- Resolve inherent memory issues when time multiplexing bunches

Implementation

- Single stage CIC sufficient for phase and magnitude detection
- Throughput meets requirement
- Decimation rate set before comb stage
- FIFOs to handle accumulator results from turn to turn
- CIC resource utilization (Virtex5)
  - Slice Reg = 200
  - BRAM = 1
  - DSP48E = 2
Custom 1-Stage CIC filter
HDL Co-Simulation

- Legacy (or new) HDL code can be imported into Simulink “black box”
- HDL is co-simulated transparently using industry-standard simulators like ModelSim or Active-HDL
- Single HDL simulator for multiple black boxes
- The time scale in HDL Simulator matches that in Simulink
Configuring a HDL Black-Box to import RTL

Create Block Description file (MATLAB M-Files)

Active-HDL co-simulation block

HDL Black-Box simulated in Active-HDL
Future Development
RF BPM Processor
Software Tools and Versions:

- MATLAB R2011a
  - Simulink
  - DSP System Toolbox
  - Signal Processing Design
  - Fixed-point Design
  - Fixed-point Blocks
- ISE Design Suite 13.2 System Edition
- Active-HDL 9.2

Demo Platform:

- SPARTAN-6 FPGA SP605 Evaluation Kit