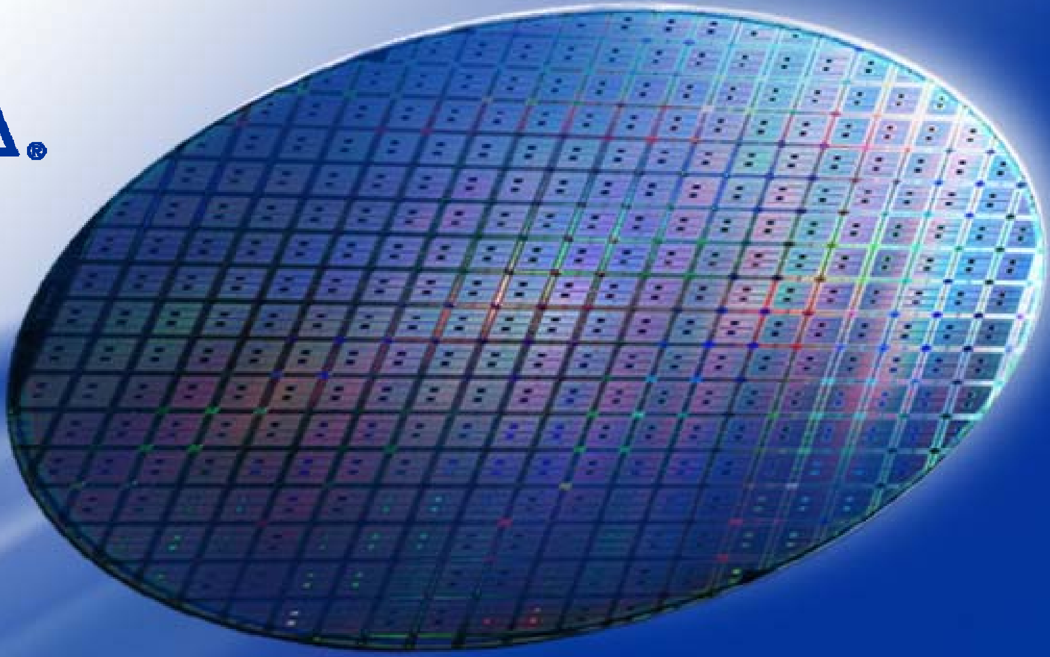


Rapid SDR Waveform Development in FPGAs Using DSP Builder

Joel A. Seely – Altera

Steven W. Cox – General Dynamics C4 Systems

Agenda

- Introduction
- Traditional Waveform Design
- Waveform Design Using Higher-Level Tools (DSP Builder)
- Example Design & Steps
- Efficiency Analysis
- Conclusion

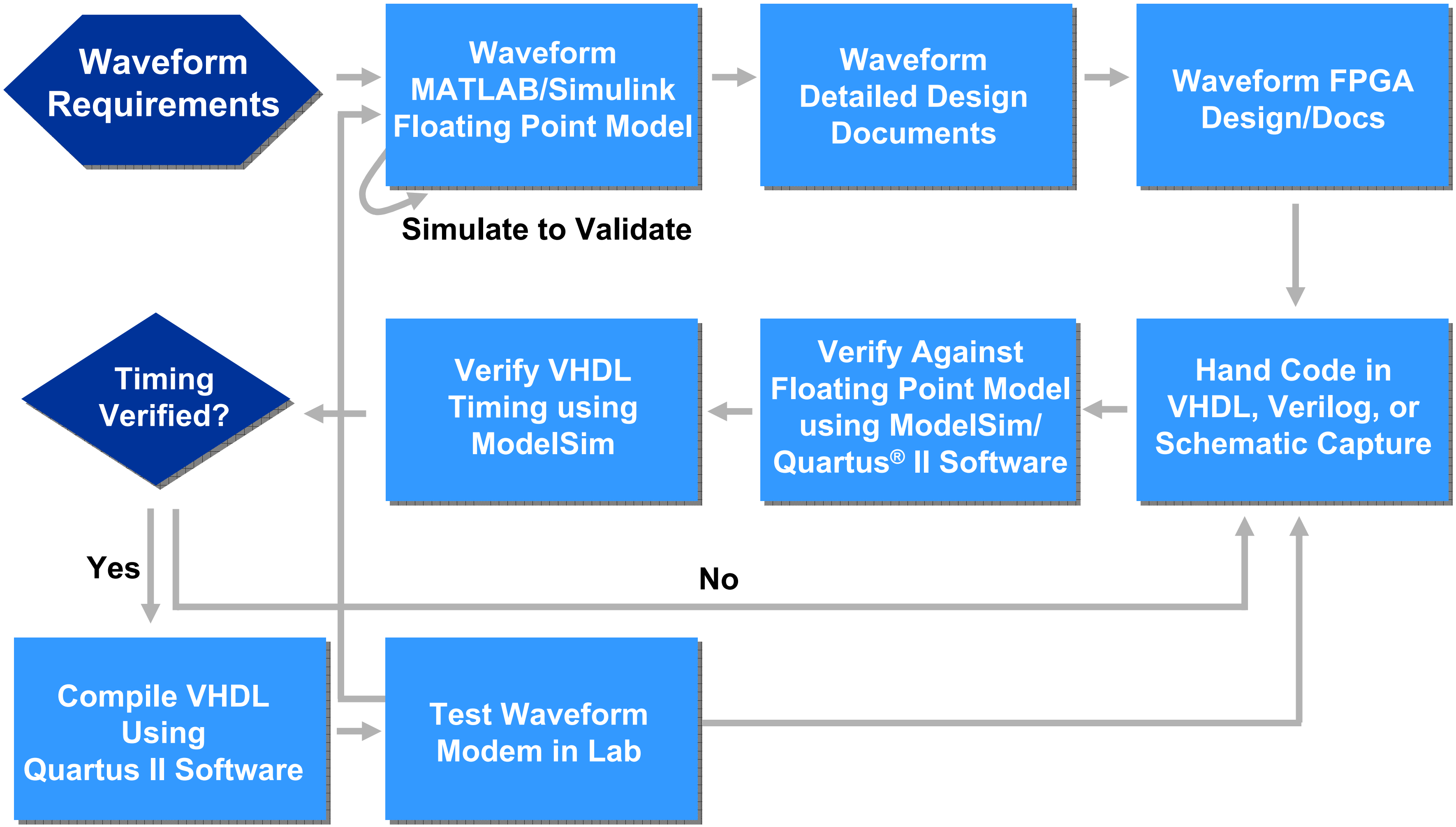
Introduction

- Software-Defined Radios Are Becoming Ubiquitous
- Three Major Programmable Components
  - GPP, DSP, FPGA
- FPGA is Taking on More “Heavy Lifting” Computationally
  - Lowering Burden of DSP & GPP
- As FPGA Designs Become More Complex, New Tools Are Required
- DSP Builder Is an Example of One of These Tools
  - The MathWorks’ Simulink Capabilities
  - Fixed Point Blockset
    - Interfaces to Third-Party Tools to Generate Synthesizable FPGA HDL
  - Allows for Design, Simulation & Verification Prior to Hardware Implementation

Traditional FPGA Waveform Design

- Start with System-Level Specifications & Simulations
- Use These to Hand-Code HDL
- Typically System Designer Had No Insight Into FPGA's Implementation Details
- Designer Needed to be an Expert in HDL—Not the sort of Expertise an Engineer Would Pick Up Overnight
- Manual HDL Coding is Inefficient as Waveforms Become More Complex
  - Tedious
  - Time-Consuming
  - Potential for Lots of Bugs
  - Increased Development Time & Cost

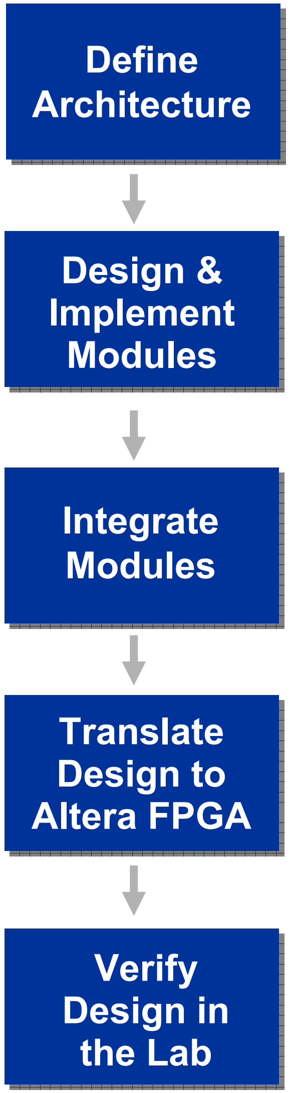
Traditional Waveform Design Flow


Waveform Requirements
Waveform MATLAB/Simulink Floating Point Model
Waveform Detailed Design Documents
Waveform FPGA Design/Docs
Simulate to Validate
Timing Verified?
Verify VHDL Timing using ModelSim
Verify Against Floating Point Model using ModelSim/ Quartus® II Software
Hand Code in VHDL, Verilog, or Schematic Capture
Yes
No
Compile VHDL Using Quartus II Software
Test Waveform Modem in Lab

DSP Builder – Higher-Level Design Tool

- Developed to Address Issues in Complex System Development
- New Design Flow Needed
  - Define Architecture
  - Implement / Design / Re-Use Modules
  - Integrate of Modules
  - Translate Design in FPGA
  - Verify FPGA Design in the Lab

Waveform Design Flow Using DSP Builder



Define Architecture

Design & Implement Modules

Integrate Modules

Translate Design to Altera FPGA

Verify Design in the Lab

Protocol Definition

- Start with Existing Floating Point Simulink Model

Sub Blocks

- Design in DSP Builder Blocks, Get Data From SImulink Model
- Timing/Detail Design Uses ModelSim
- Run DSP Builder in SImulink when Verifying Data

All Blocks

- Divide Simulation in Fast & Slow Clocks Rates if Possible
- Use Sims to Examine Boundary Conditions in Design & Timing Issues in ModelSim
- Use DSP Builder to Run Sims & Verify Data & for Initial Sizing & Synthesize

FPGA Design

- Remove Stimulus from Design for Synthesis
- Generate Quartus Symbol with DSP Builder Script, Insert Into Total Design & Compile
- Check Timing

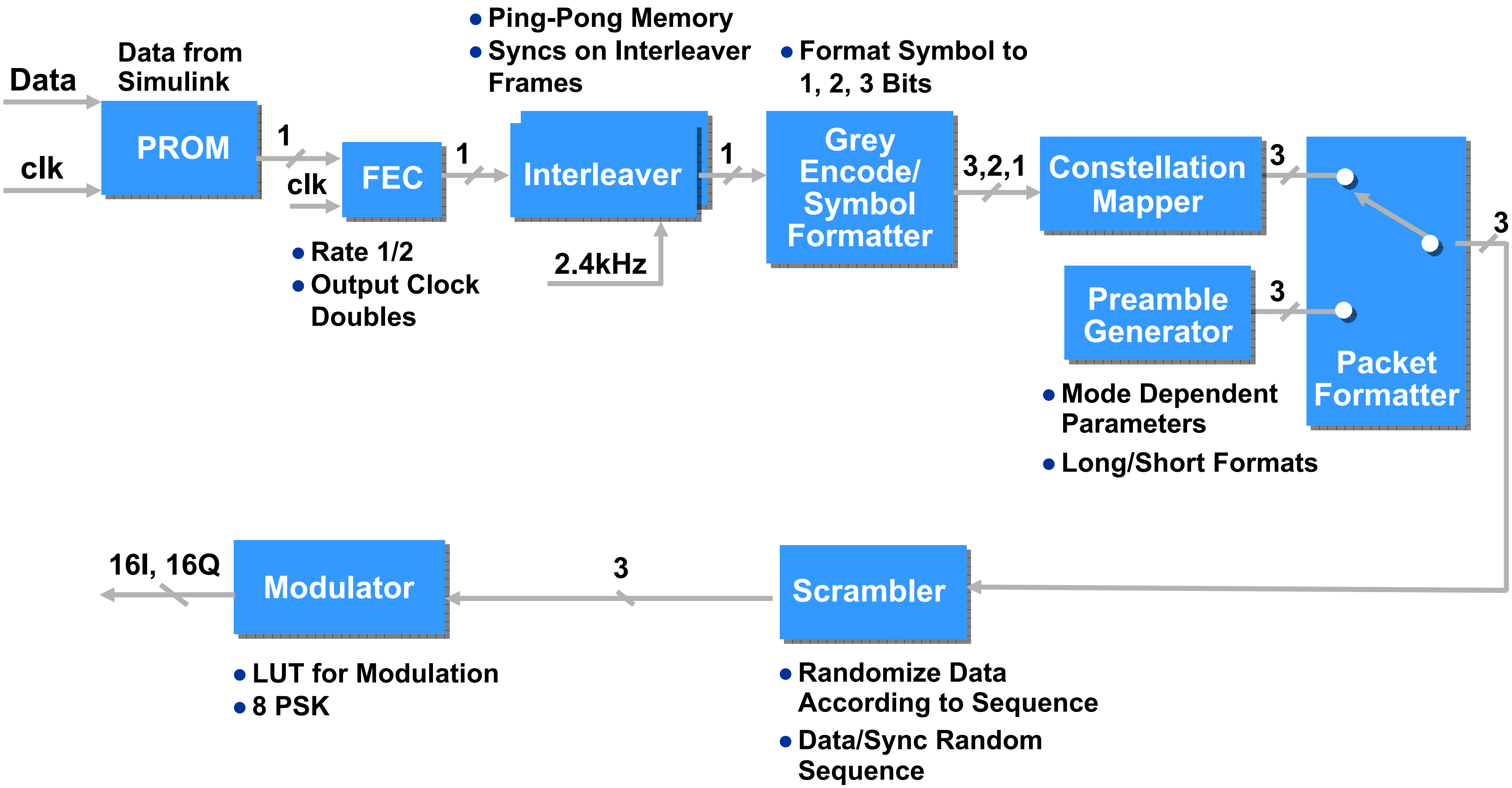
Lab Verification

- Check Data with Logic Analyzer
- Store Data from Logic Analyzer to file
- Analyze Final Data from Logic Analyzer in Simulink

Example Design – MIL-STD 110A

- Starting Point for the SDR Architecture
- Used the 1,200 Bits/Second Transmit Mode of the Specification
- Floating-Point Model Was Used For
  - Guideline & Comparison
  - Initial Sizing
  - Architecture Mapping Estimates

Example Design – MIL-STD 110A



- Ping-Pong Memory
- Syncs on Interleaver Frames

- Format Symbol to 1, 2, 3 Bits

- Rate 1/2
- Output Clock Doubles

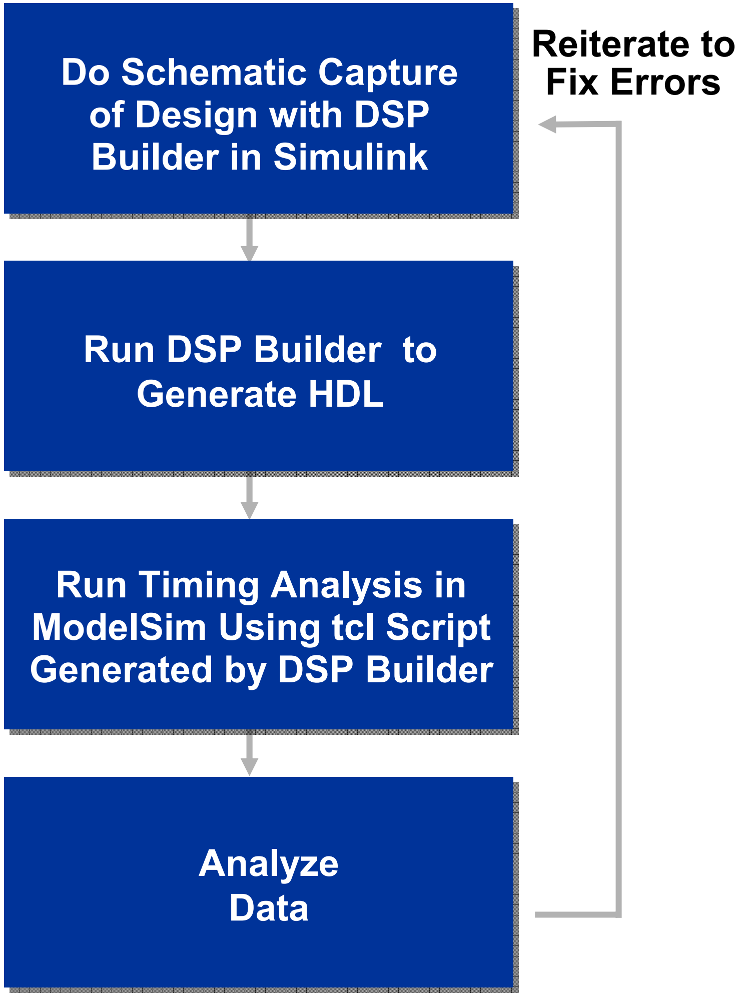
- Mode Dependent Parameters
- Long/Short Formats

- LUT for Modulation
- 8 PSK

- Randomize Data According to Sequence
- Data/Sync Random Sequence

Implementation/Simulation

- Used Altera DSP Builder & Simulink Toolbox Blocksets
- Uses Schematic Entry
- Ties to Third-Party Simulators (ModelSim) for Timing Verification & Control
- Automated Creation of
  - HDL
  - Stimulus for Sub-Blocks
  - Scripts to Load & Compile the Updated Design
- Streamlines the Iterative Process of Simulating a Design


Do Schematic Capture of Design with DSP Builder in Simulink

Run DSP Builder to Generate HDL

Run Timing Analysis in ModelSim Using tcl Script Generated by DSP Builder

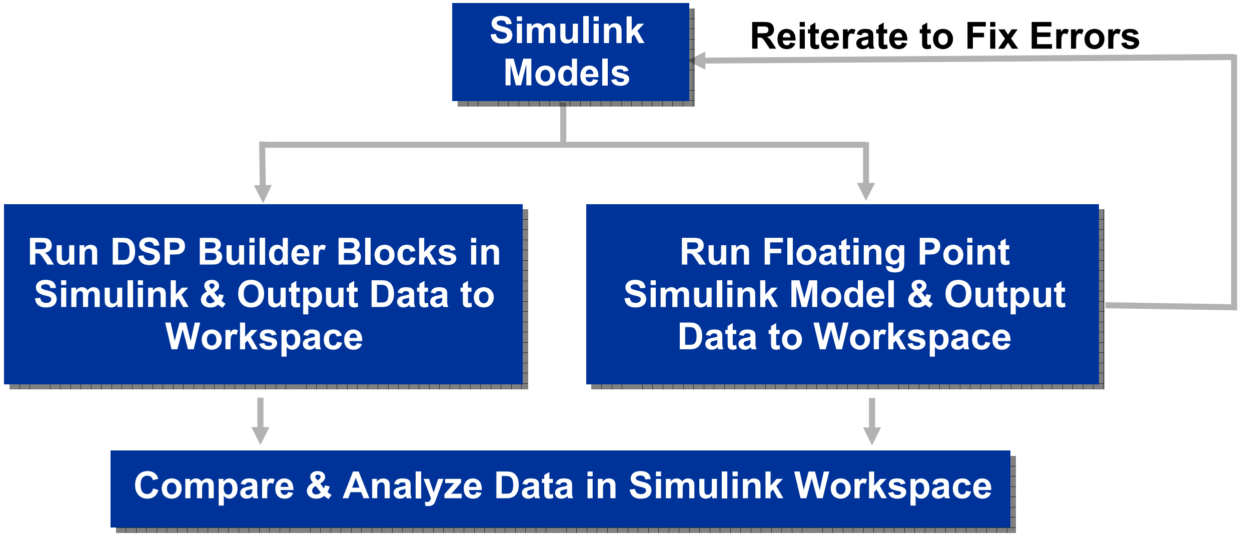
Analyze Data

Reiterate to Fix Errors

© 2004 Altera Corporation - Confidential

Implementation/Simulation

- Floating Point & DSP Builder Models are Compared & Verified
- DSP Builder Model & Floating-Point Models Are Run Separately in Simulink
- Commands in Simulink Manipulate the DSP Builder Fixed-Point Data to Compare to the Floating-Point Model
- To Correct Errors, Update the Models in Simulink & Rerun the Simulations to Verify


Simulink Models
Reiterate to Fix Errors
Run DSP Builder Blocks in Simulink & Output Data to Workspace
Run Floating Point Simulink Model & Output Data to Workspace
Compare & Analyze Data in Simulink Workspace

Integration

- Data Validation Done in Simulink
  - DSP Builder Models are C Code
  - C-Code Simulators Run Faster than HDL Interpretative Simulators
- After Validation Preliminary Sizing & Synthesis Estimates Made
  - Provides an Early Alert for Sizing & Timing Constraints
  - Allows for Fixing Problems in the Early Stages of the Design Cycle
- Optimizations for Simulation During the Integration Phase
  - Changed Input Data Clock to 0.66 MHz (Instead of 1,200 Hz) -> Faster Simulation
  - Decreased Bits/Frame to 1,440 to 120

Integration (Cont.)

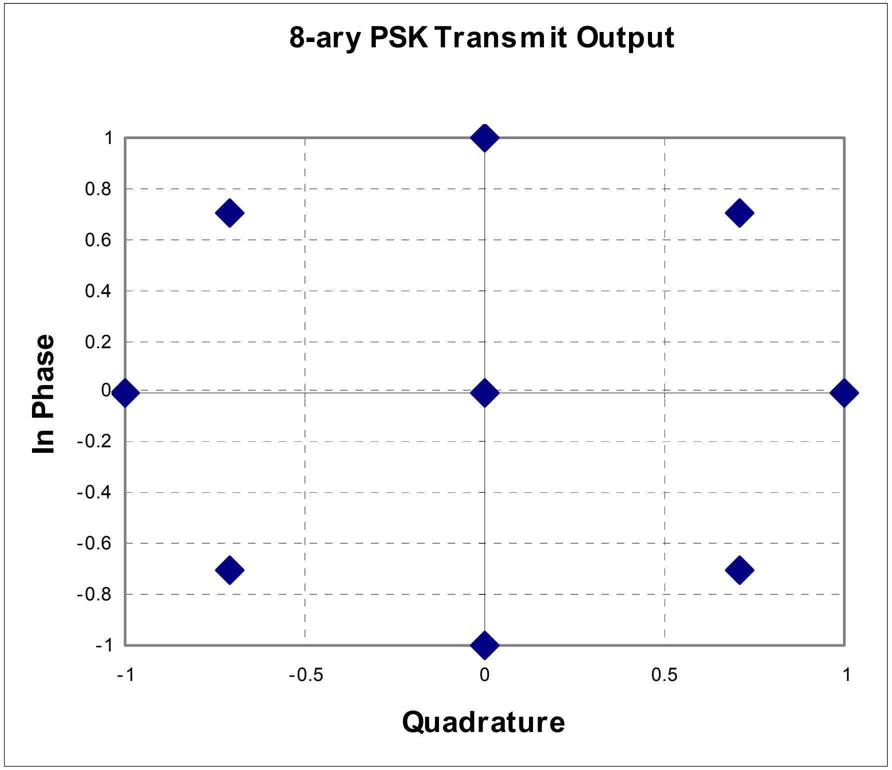
- Separated Slow & Fast Clock Dependencies
  - Slow Clock Dependences Run From Beginning of Waveform Chain to Input of Scrambler
  - Fast Clock Dependencies Run From Scrambler to Output of Modulator
  - Only Valid Data at Output of Data Formatter Was Captured to Workspace
  - Fast Dependencies Were Run As Separate Simulation With Only Valid Data Output From Formatter
- This Integration/Simulation Methodology Significantly Reduced Simulation Time For High-Speed Portion of Circuit & Allowed Efficient Design Validation

Synthesis

- First Replace Simulink Stimulus with Input Pins
- DSP Builder Generates Quartus II Script for Loading DSP Builder Design & Create Symbol
- Anticipate Test Points Needed for Debugging New Design
  - If Additional Test Points Needed, Must Update the DSP Builder Model
- Synthesis & Compilation Done in Quartus II Software
  - Other Synthesis Tools Also Available
- Timing Results Analyzed
- Raw Binary File Created

Lab Verification

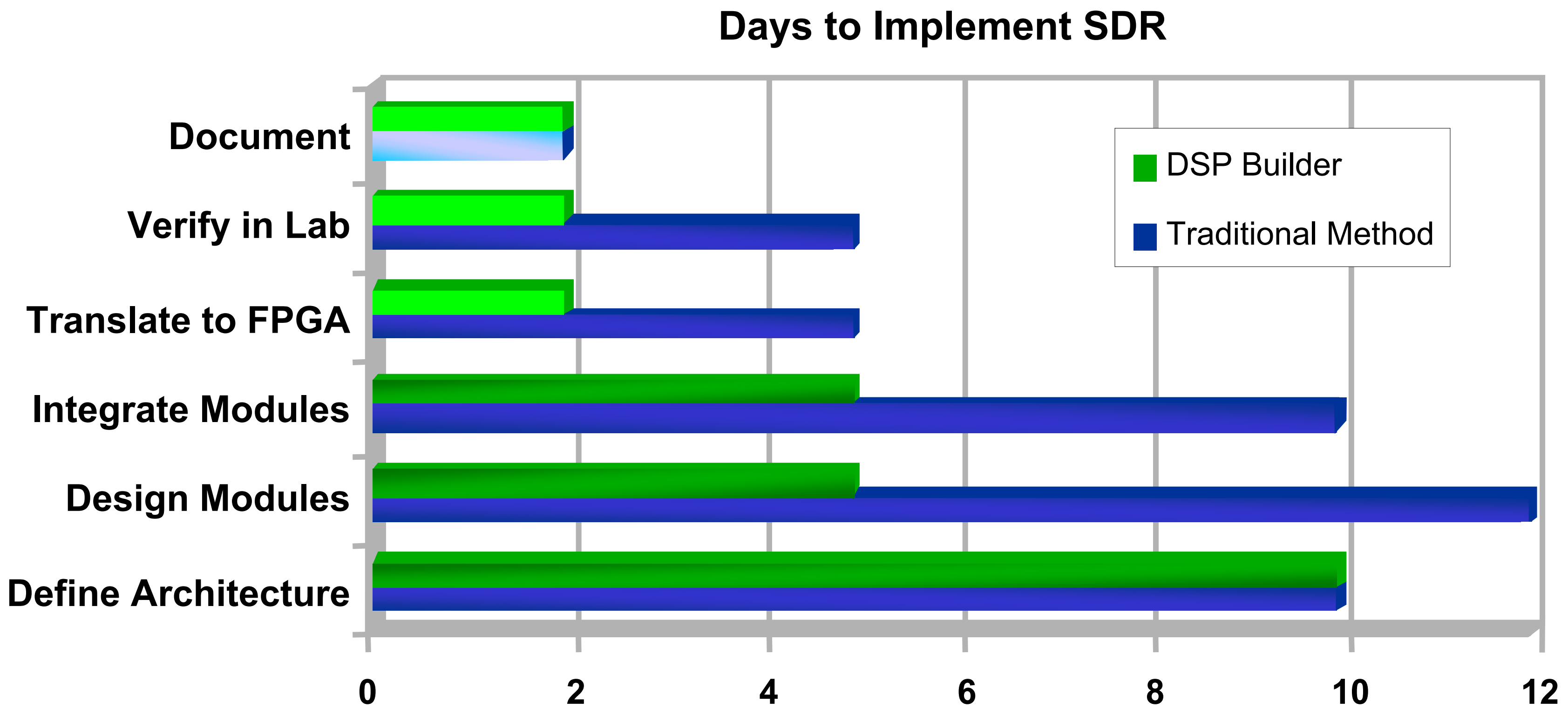
- Loaded RBF Onto FPGA on Software Radio
  - Altera EP20K1000 Device
- Each Sub-Block Was Checked With Logic Analyzer
- Design Yielded an 8psk Constellation
  - See Graph
- Logic Analyzer Captured the Final Data (I & Q)
  - Used to Verify Against the Data in the Floating Point Simulink Environment
  - Only Took a Few Days



Efficiency Analysis

- Major Improvements in Several Design Areas
  - Design
  - Integration
  - Translation (HDL Coding)
  - Verification
- Nearly 50% Improvement
  - 26 Days vs. 49 Days Through Traditional Method

Efficiency Analysis (Cont.)



Conclusion

- DSP Builder is Powerful Tool for Rapidly Developing SDR Waveforms on FPGAs
- DSP Builder Flow Allows You to Allocate Time in an Appropriate Manner for Developing Waveforms
- DSP Builder/Simulink/ModelSim Flow Allows You to Rapidly Identify Problems & Troubleshoot
  - Reduces Risk, Time & Resources in the Lab
- DSP Builder Tool Allows the Hardware to be Abstracted to a Higher Level
  - FPGA & System Waveform Developers Can Operate in a Common Environment