Celoxica

Software-Compiled System Design:

A Methodology for Field Programmable System-on-Chip Design

EDP paper presentation

Jeff Jussel

April 15, 2003

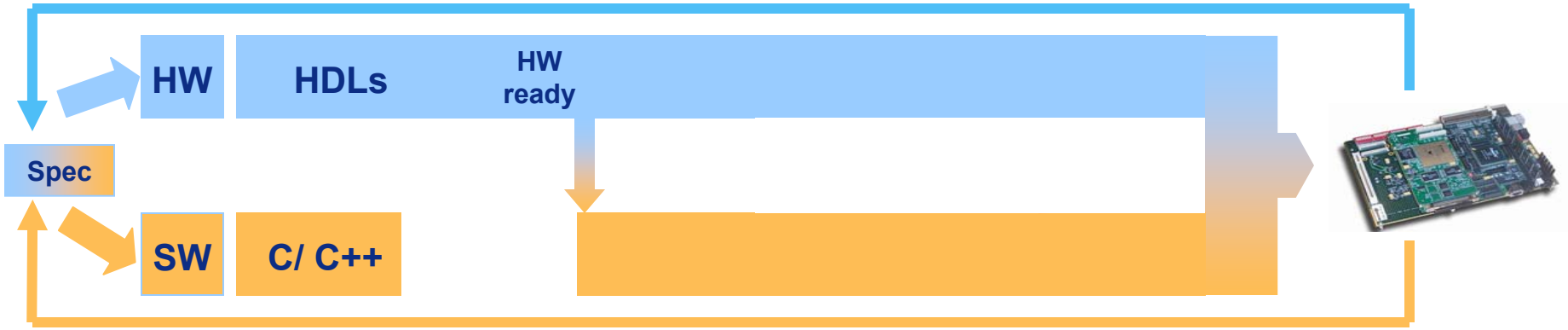
Methodology Target Devices

- Field Programmable System-on-Chip (FPSoC)
- Definition
  - Over 1 million usable gates in FPGA logic
  - Embedded processor cores
  - On-board peripherals and memory
- Usage
  - Rapid prototypes (verification stage and ASIC RTL sign-off)
  - Low NRE implementation
  - Reconfigurable systems
- Examples
  - Altera Excalibur
    - Apex FPGA (up to 1M gates)
    - NIOS (32/16 bit RISC soft core)
    - ARM922T (32-bit, AMBA)
  - Xilinx Virtex-II Pro
    - Virtex-II FPGA (M's gates)
    - MicroBlaze (32 bit RISC soft core)
    - IBM PowerPC 405 (32-bit, Coreconnect)

System Design Challenges

- FPSoC design (as S0C) presents system design challenges
  - Over half of FPGA designs include over 500K gates*
  - Requires convergence of HW/SW methodologies

- Current flow deficiencies
  - Poorly profiled partitions lead to sub-optimal performance
  - Incompatible flow, verification between HW/SW designs
  - Gap between specification and hardware RTL
  - Impact of specification changes

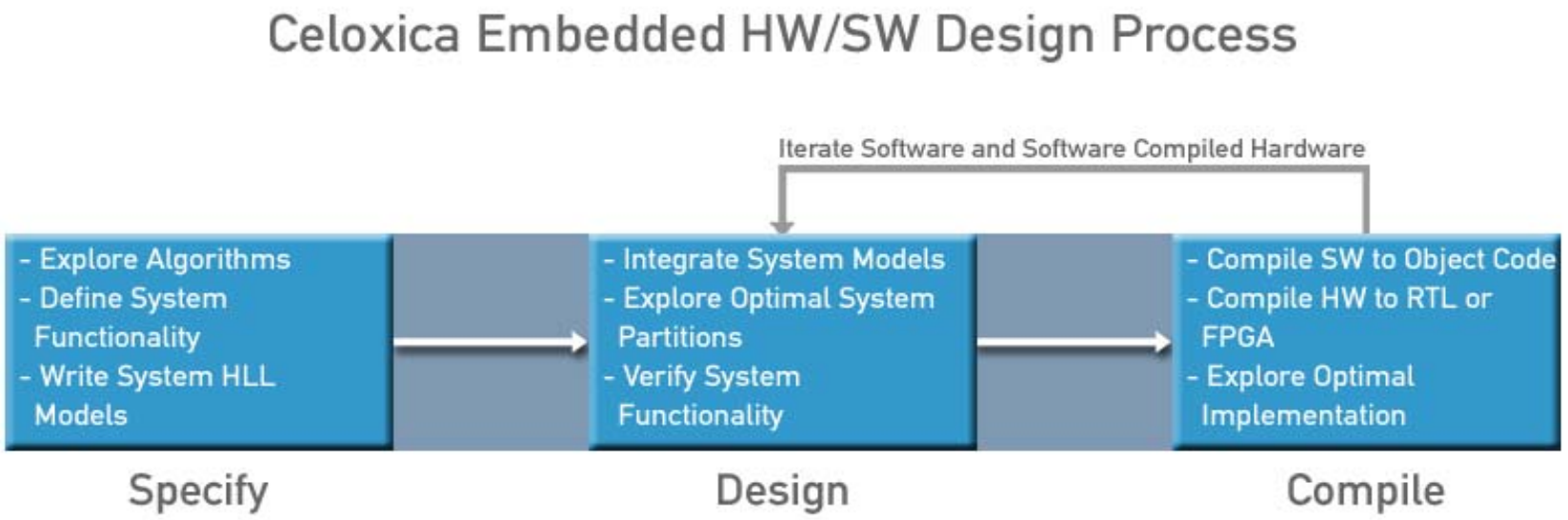


*Source "ASIC Design Times Spiral Out of Control", Gartner, April 2002

Software-Compiled System Design

Methodology is defined by:

- Use of higher-level languages (HLL) : eg C, C++, SystemC, Handel-C, SpecC
- Partitioning in common language before HW or SW implementation
- Verification driven from system specification
- Direct implementation path from HLL descriptions to HW and SW



Software-Compiled System Design Flow

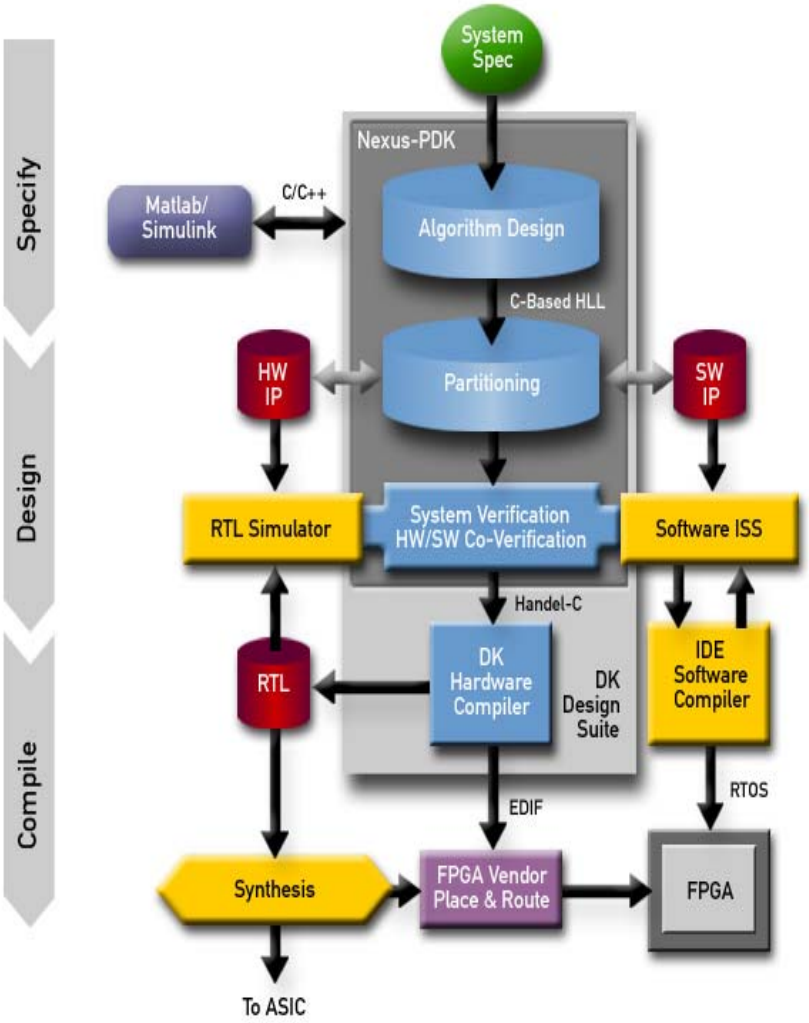
System Design Functions:

Co-design
- Provide rapid iteration of partitioning decisions throughout flow

Co-verification
- Drive continuous system verification from concept to hardware

C to RTL
- Generate human-readable VHDL and Verilog for ASIC RTL hand-off

C to FPGA (FPSoC)
- Enable direct implementation to device optimized programmable logic

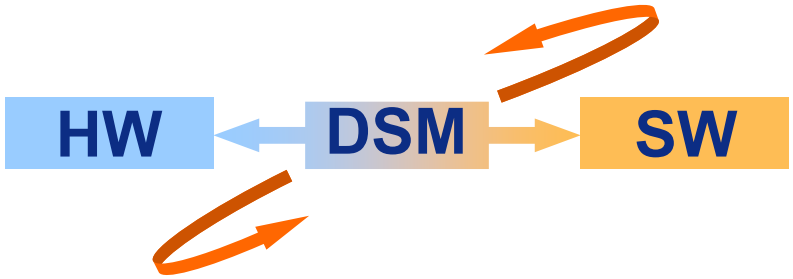


Celoxica

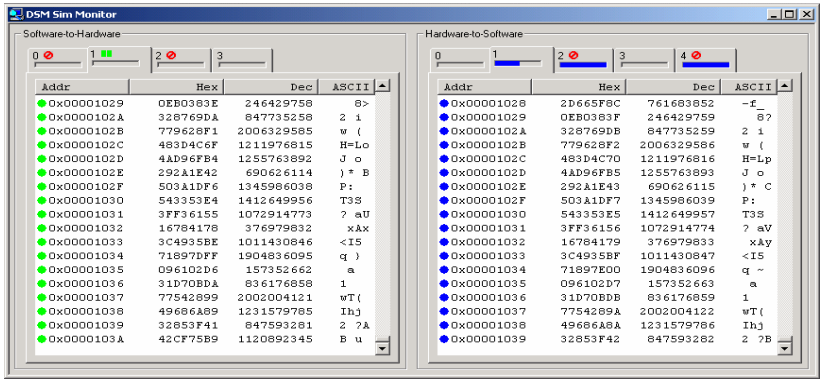
System Co-design

HW-SW Partition and Profiling

- Determine the optimal design partition at the system level
- Profile system hardware, software and interface performance
- Retarget code between hardware and software using DSM API
- Iterate solutions to explore the design space
- Deliver optimal Quality of Design (QoD)



DSM API assists code retargeting



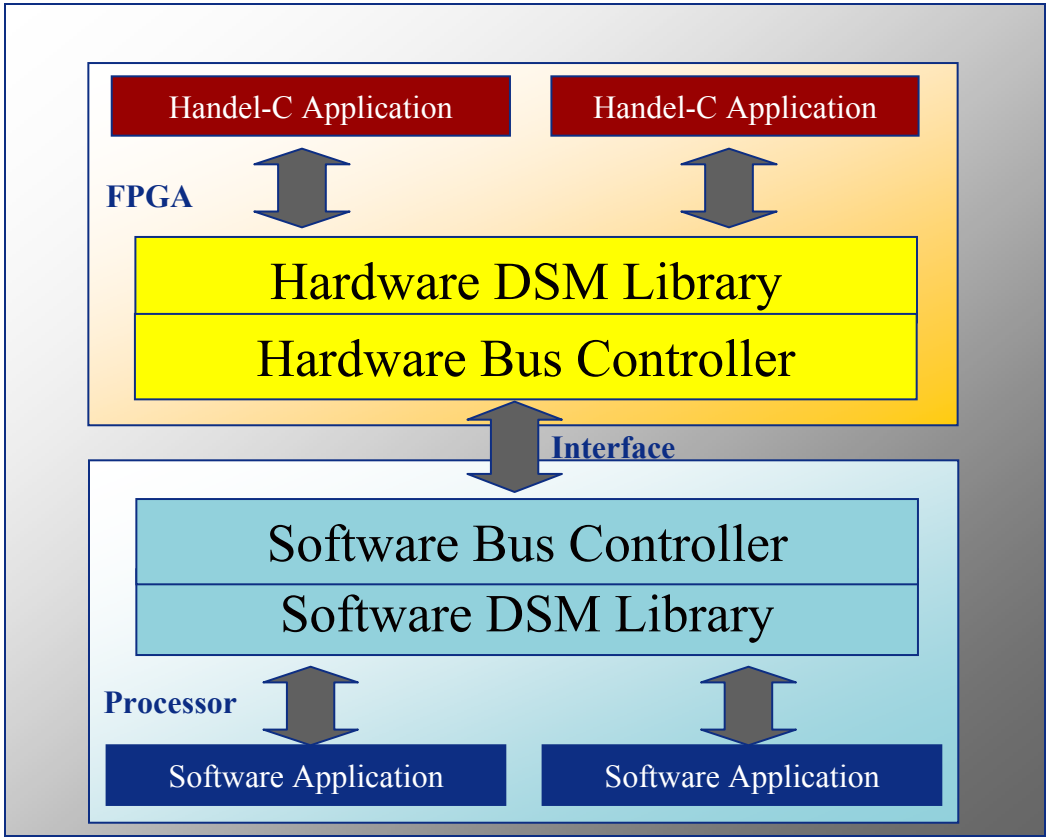
DSM monitor assists debug

Celoxica

DSM – Data Streaming Manager

- Supports the SCSD partitioning methodology
- DSM is BUS/ interconnect and OS independent
- An interface for transferring multiple independent streams of data between hw & sw
- Provides a functionally accurate simulation environment

DSM is a portable codesign API

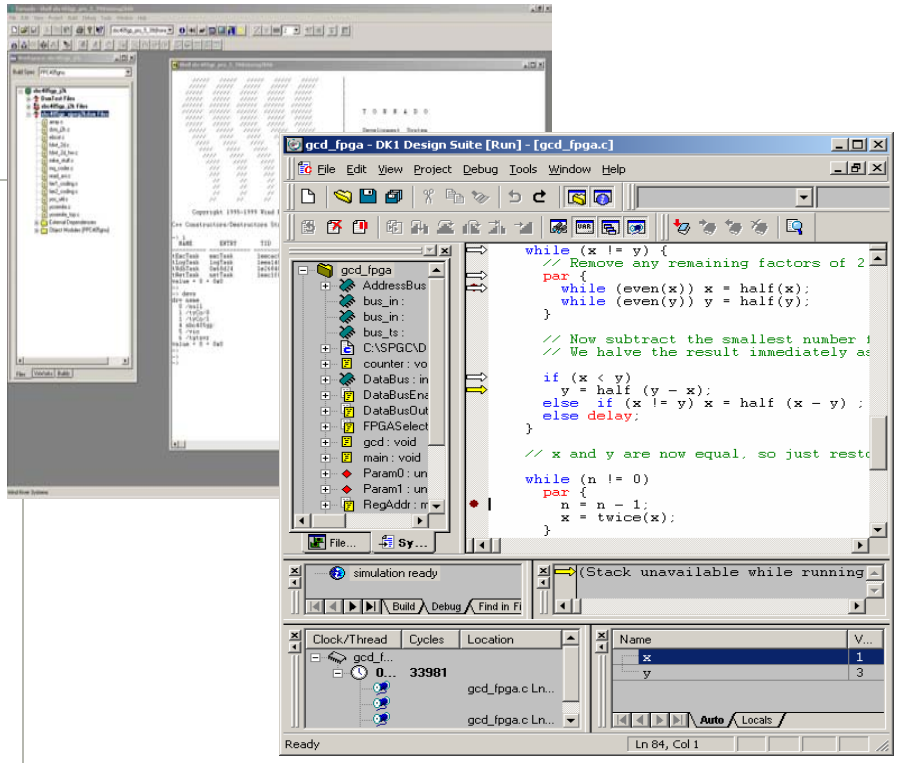


Celoxica

System Co-verification

System Verification Environment

- Drive verification flow from C-based system specifications
- Speed system simulations by modulating model abstraction levels
- Multiple language simulation support for C, C++, SystemC, SpecC, Handel-C, VHDL, Verilog
- Simulate with Matlab/Simulink models
- Reduces verification time, risk and cost



Co-simulation between ISS software environment, RTL simulators and Nexus-PDK

Celoxica

Direct FPSoC Implementation

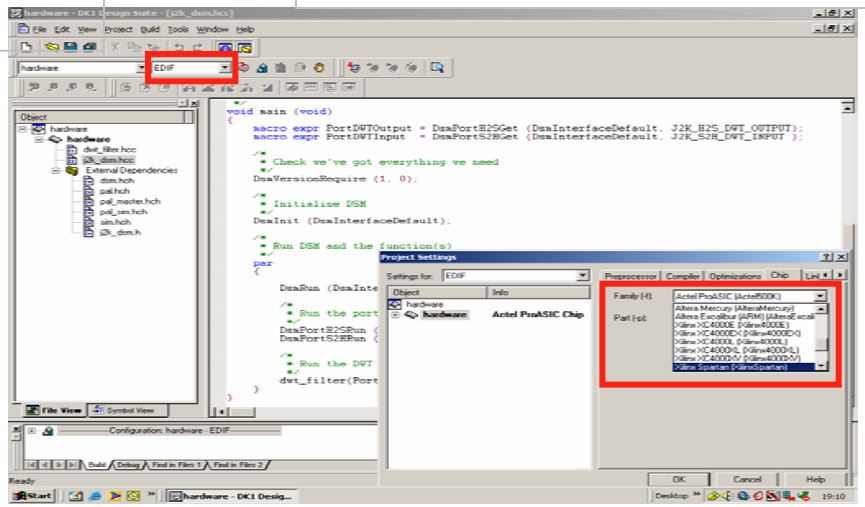
C to FPGA

- Directly compile C to optimized EDIF output for programmable logic
- Delivers efficient FPSoC hardware and proven QoR
- Fastest route to FPGA logic for cost-efficient rapid prototyping or device production

C to RTL

- Generation of Verilog and VHDL RTL descriptions
- Provides human-readable output for RTL hand-off to ASIC design flows
- Optimized for existing RTL flows to provide direct path from C specifications

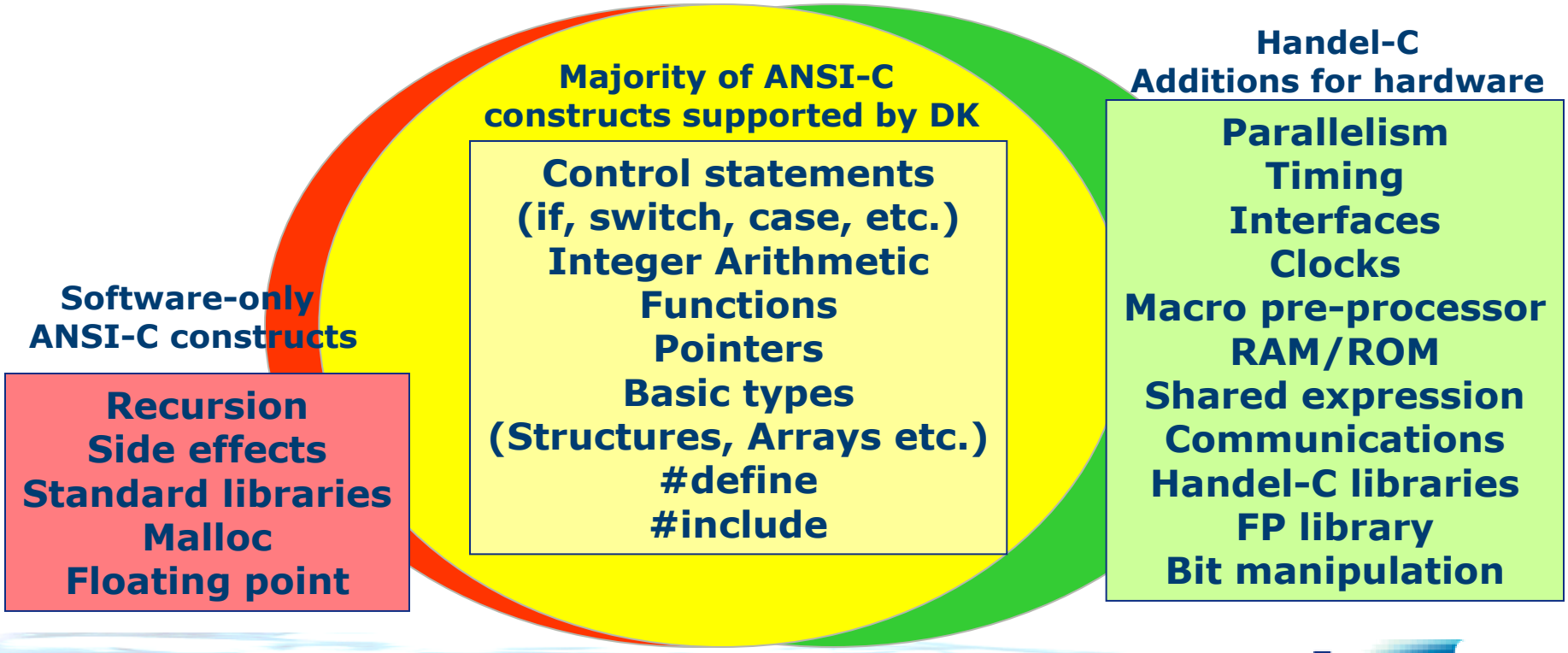
Direct System Hardware Implementation from Handel-C



Celoxica

Handel-C extensions to ANSI-C

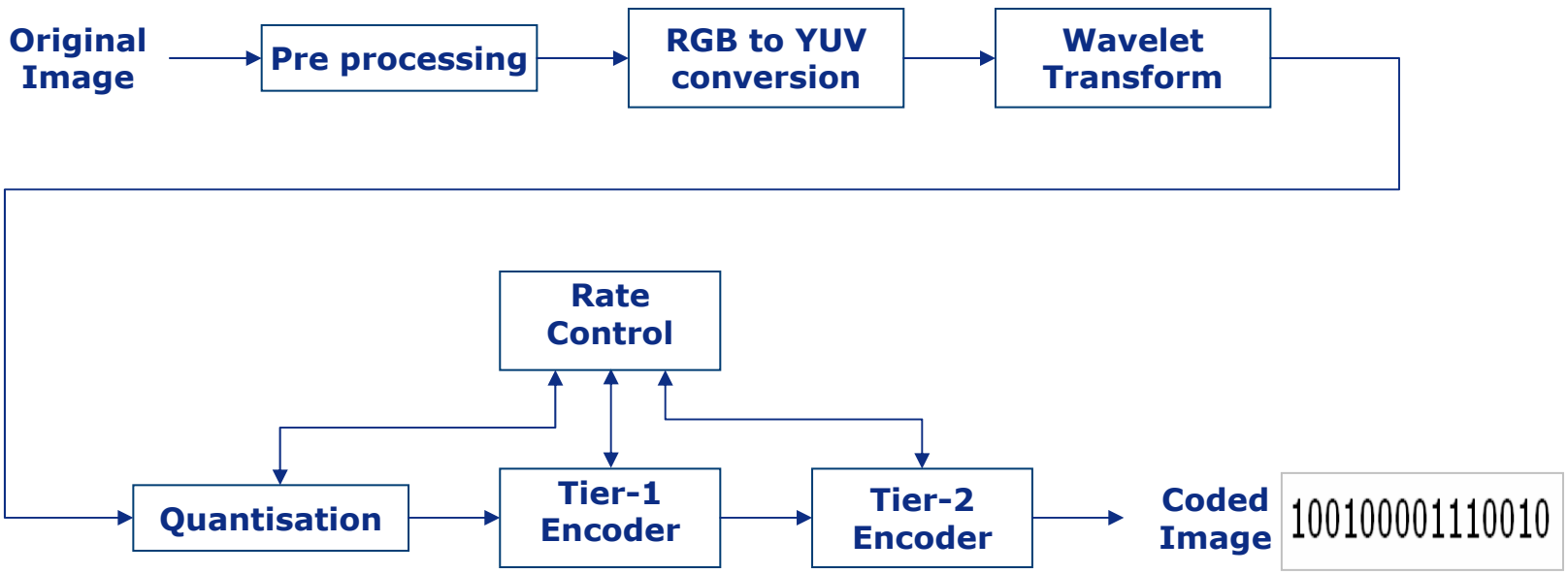
- **Handel-C adds constructs to ANSI-C to enable DK to directly implement hardware**
  - fully synthesizable HW programming language based on ANSI-C
  - Implements C algorithm direct to optimized FPGA or outputs RTL from C



Celoxica

SCSD Design Case Study: JPEG2000

- Xilinx project benchmark for Platform FPGA (FPSoC)
  - Start with C description of JPEG2000 algorithm
  - Use Software-Compiled System Design methodology
  - Partition and Implement JPEG2000 Design
  - Compare results against original VHDL design performance



Top level block diagram for JPEG2000 operation

Celoxica

JPEG2000 Case Study: Flow Steps

Phase 1: Profile and Verify

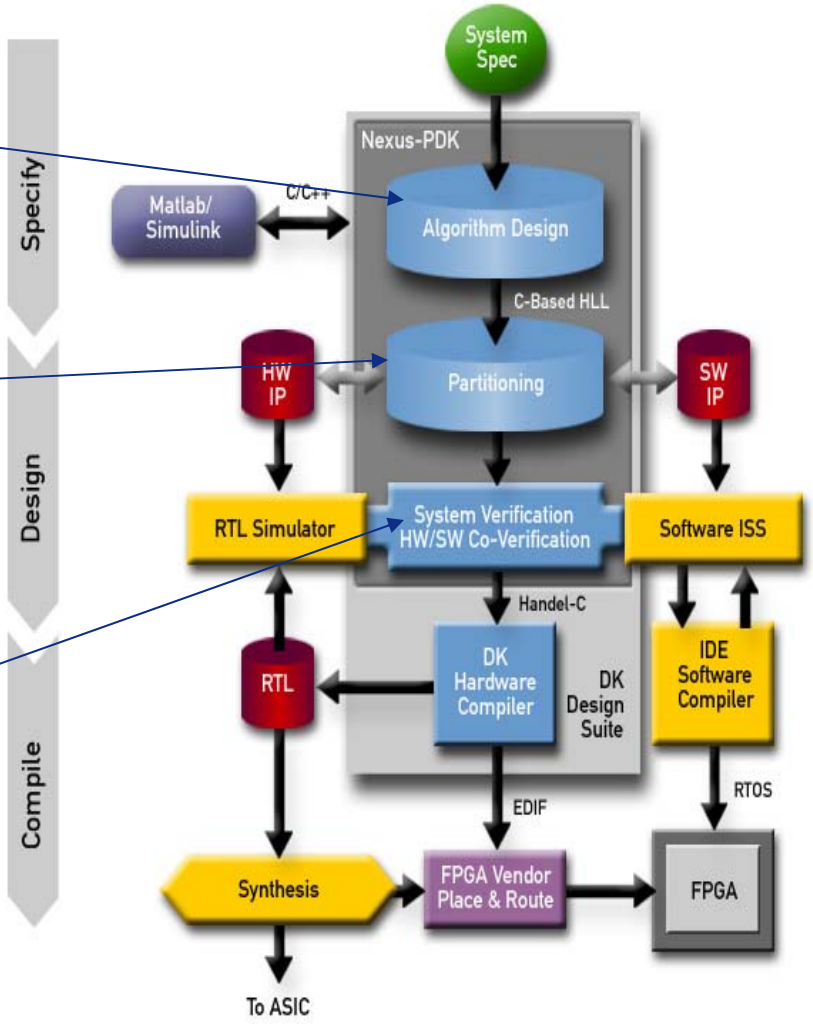
- Simulate C algorithm on PowerPC 405 processor ISS, identify bottlenecks
- Establish verification plan

Phase 2: Partition and Verify

- Use DSM to explore design space, move blocks to HW
- Analyze HW/SW communication interface
- Fine tune partition performance

Phase 3: Design and Verify

- Tune design for performance
- Combine SW calls
- Add HW parallelism, refine HC code
- Co-verify HW/SW (with WRS ISS)



Celoxica

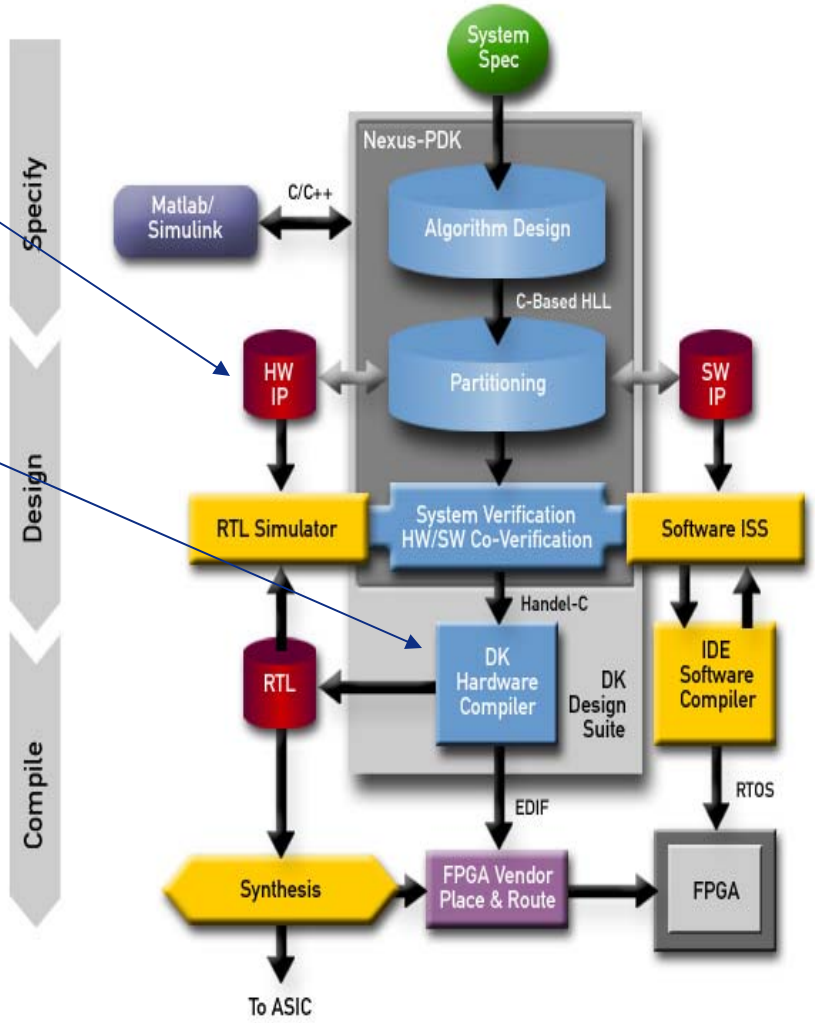
JPEG2000 Case Study: Flow Steps (cont)

Phase 3b: Specification Change

- Import VHDL IP for 2d DWT algorithm
- Treat as black box, called from HC
- Perform RTL/HC/ISS co-simulation

Phase 4: Implement and Verify

- Directly compile HC blocks to EDIF
- Import VHDL IP hard core as EDIF
- Compile SW to PPC under VxWorks
- Xilinx P&R tools for Virtex-II Pro target platform
- Initial platform: Wind River SBC405GP and Proteus FPGA daughter card
- Platform retargeted without app code changes using PAL API
- Final platform: Virtex-II Pro ML300 evaluation platform



Celoxica

JPEG2000 Case Study results

Celoxica 1st pass		2nd pass	Final	HDL	Observations
Slices	*646*	*546*	*758**	*800*	*Comparable*
Device utilization	*6%*	*5%*	*7%*	*7%*	
*Speed (MHz)**	*110*	*130*	*151*	*128*	*HC faster*
Lines of code	*386*	*395*	*395*	*435*	
Design time (days)	*6*	*7 (6+1)*	*7 (6+1)*	*20**	*HC quicker*
					Expert vs Novice

* *Lena used as testbench throughout, input bit width12, max 1K image widt*

* Includes IP Block Insertion

* Doesn't include partitioning spec. development

- Rapid Handel-C (HC) implementation by an engineer with no prior knowledge of JPEG2000. Primary design focus was **area efficiency**.
- Common language base made easy porting to hardware of the DWT source & DSM allowed partition, co verification & data to be easily moved between HW & SW
- Optimizations included using signals instead of registers, maximum use of dual ported memory & reduction in routing logic by syntax duplication in Handel-C. Place & Route tools configured to optimize the implementation for area efficiency
- **Final implementation** integrated existing HDL IP block into the design flow for maximum design re-use value (black boxing)

Celoxica

Celoxica System-Design Summary

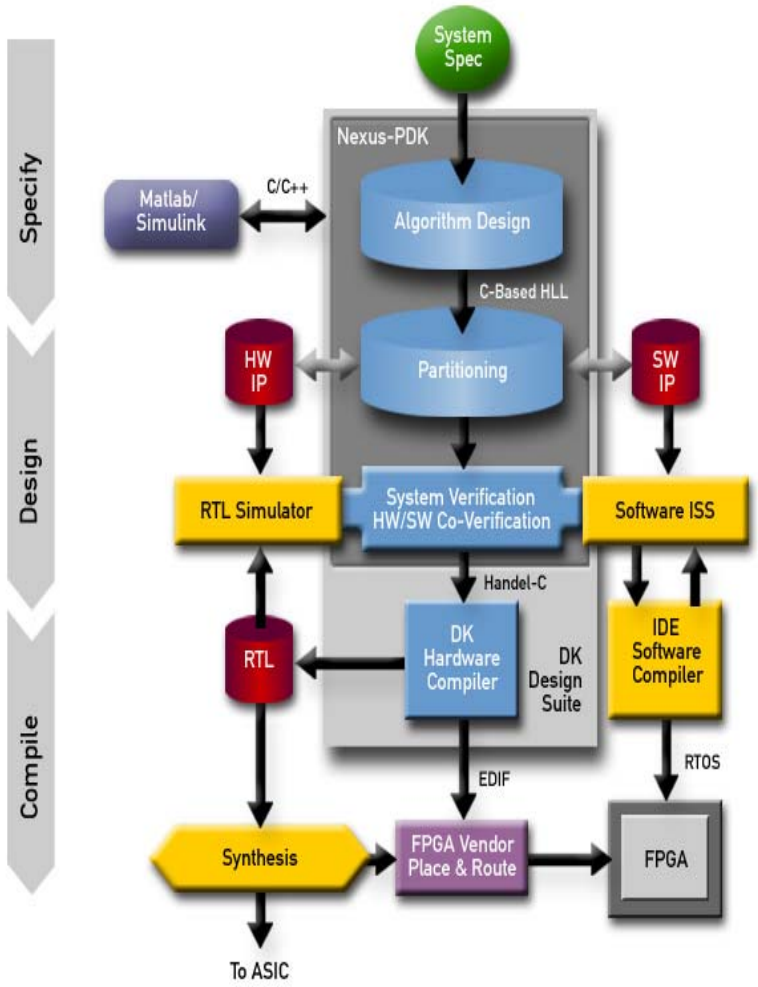
Software-Compiled System Design:

System Design Flow Evolution

- Design flow provides design environment for design of FPSoC devices
  - HLL usage
  - Iterative partitioning
  - System-driven verification
  - Direct compilation to FPSoC hardware

Proven Results

- Xilinx JPEG2000 is one example of Quality of Design and Design Productivity achieved using this system methodology and tools



Celoxica