Automated Verification And Synthesis Of Cortex-M1 Enabled By IP-XACT

ARM Developers Conference 2007
Santa Clara
Introduction

- Introducing IP-XACT?
  - An XML Databook and Modular Design Generation Standard for Massive IP Reuse

- Introducing the ARM Cortex-M1 Processor
  - The First ARM Processor Designed Specifically For FPGA

- Rapidly Creating and Verifying Cortex-M1 Designs
  - Building Cortex-M1 Designs for FPGA using IP-XACT Databooks

- Synthesis Options
  - Precision Synthesis give Multiple FPGA vendor options
Introducing IP-XACT

XML Databooks and Generators
Why Is IP Reuse So Important?

- Designs are no longer ‘silicon-bound’.
  - The challenge is how to usefully utilize available silicon capacity.

- Designs and IP is growing in complexity by 5x every 3 years.
  - Designer productivity cannot match that growth rate without radical changes in design

- All indicators point to IP Reuse as the key design productivity driver.
  - Reuse known good designs completed by experts, rather than reinvention on every project.

![Graph showing IP Reuse and New Content over years]

---

**IP Reuse**
- Block Based Design
- Top Down Design

**New Content**
- Gates (k)
- Year

**Complete Design**
- New
- ReUse
- %
Introduction to IP-XACT

- IP-XACT is The SPIRIT Consortium Standard for documenting IP.
  - Enabling highly automated design creation and configuration.
  - Based on a technology donation from Mentor Graphics

- The SPIRIT Consortium is + 50 more reviewers

- Current SPIRIT Consortium Activities include
  - IP-XACT 1.4 - Extending IP-XACT to accommodate ESL
  - Verification - Enabling new Higher Level Modular Verification.
  - Debug - Enabling Embedded Software Tool Configurations.
Cortex-M1 is supplied with IP-XACT data
- IP-XACT enables the automation of design creation using that Databook information
  - Specialist design knowledge is embedded in IP-XACT generators
  - Create designs in all sorts of formats (HDL, SW, documentation)
  - Configure tools automatically based on design content

With the right tools, rapid prototyping capabilities enabled by FPGA technology can be matched by equally rapid IP-XACT-enabled design creation
- Through this session, we will be looking at design capabilities enabled by IP-XACT.
IP-XACT enables designs to be created from IP from multiple sources

- Language neutral
- Process neutral
- IP-XACT enabled tools will automatically integrate all aspects of your chosen IP directly into designs
An IP-XACT Design Environment
Drag and Drop Design in Minutes

IP described in XML Databook

Consistent HW and SW programmers view

Simple system diagrams represent complex designs
IP-XACT : How Generators Work

- IP Modules from different sources...
- … Processed by modular Generators ...
- Into Design Data, Automatically.

**IP-XACT Design Environment**

**HDL**
- SystemC
  - module
  - {nand2}

**Software**
- Simulator

**Design Docs**
- Parts List
  - 1 x UART
  - 2 x OCP Bridge
  - 1 x Interrupt

**Simulator**
- map WORK mylib
- timescale 1ns

**Design Team Data**
- UART  v1.304
  - John W - designed 20/Jun/06
- DSP  v3.145
  - Wally R - designed 15/May/05
- FlexRay v1.03
  - Serge L - designed 13/03/05

**Rom S-Records**
- S1130000285F245F2212226A0
- S11300100002000800084E423430
- S11323484729387ABF9834230
- S107003000144ED492
- S9030000FC

**SPIRIT XML**
- Mentor
  - UART
  - VPB
  - Serial
  - Verilog
  - Swdriver
  - verilog_source

**Generator**
- Mentor
- Arm
- Philips
- Synopsys
Exploiting IP-XACT Information For Design

HDL Generation
Bus Infrastructure Generation
Testbenches and Verification IP
Seamless HW/SW Coverification
Questa HDL Simulation
Embedded Software Testbenches and Verification IP
HTML Documentation
... and so much more!
Introducing Cortex-M1 Processor
ARM Cortex-M1 Applications

- FPGAs are an ideal implementation route for
  - Embedded systems shipping in volumes < 100k units/yr
  - Rapid development of time-to-market critical designs
  - Initial product runs, design entry, prototyping
  - Long-life applications – no risk of obsolescence

- Powerful risk reduction strategy
  - ARM processors proven in billions of ARM powered devices
  - Broad development tools and OS support
  - First class ARM support team
  - Coupled with low risk of FPGA deployment
ARM Cortex-M1 Processor

- High frequency, low area microcontroller processor for FPGAs
  - Between 70MHz – 170MHz (depending on FPGA device)
  - Occupies less than 15% area on the most popular FPGA device sizes
  - Cortex–M1 upwards compatible with Cortex family on ASIC/ASSP/MCU
  - Performance will continue to increase as FPGA technology advances

- Optimised for synthesis
  - Can target any FPGA device
ARM Cortex-M1

Freedom to Reduce Costs and Time-to-Market

- Rationalize across multiple projects on FPGA, ASIC/ASSP or MCU
  - Single RTL source for any FPGA device (Actel, Altera, Lattice, Xilinx…)
  - Reuse existing ARM tools and expertise
  - Consolidate software base and development tools
ARM Cortex-M1

Freedom to Reduce Costs Further in the Future

- Design entry to production with Cortex-M1 on FPGA
- Migrate easily to ARM Cortex-M3 on ASIC in the future
  - Benefit from efficient, low-power, ASIC-optimized Cortex-M3 design
  - Processor AMBA AHB bus interface allows system design reuse
  - Software compatibility allows code reuse
  - Cortex-M1, M3 supported across popular tools and OSs
ARM and Mentor cooperate and connect on a wide range of products and standards from front end to physical implementation

- EDGE Software Development Tools for ARM Processors
- Nucleus Real-Time Operating System for ARM Processors
- Seamless Co-Verification Models for ARM Processors
- Precision Synthesis optimized for ARM FPGA Designs
- 0-In Protocol Monitors for ARM AMBA Bus Verification
- TestKompress ATPG for ARM Processors
- Calibre

- The SPIRIT Consortium: Mentor and ARM are founding members
  - IP-XACT Databooks enabling fast IP-based design creation
ARM Cortex Processor Family

- Common architecture across the performance spectrum
  - Thumb®-2 blended 16/32-bit ISA
  - Performance and efficiency

- Three ARM Cortex Series
  - Applications (A series)
  - Real-time (R series)
  - Microcontroller (M series)

- ARM Cortex-M1 Processor
  - Smallest ARM processor
  - Upwards compatible with Cortex range
  - Designed for implementation in FPGA programmable logic
ARM Cortex-M1 Compatibility

- ARM Cortex-M1 implements a lightweight Thumb-2 profile
  - Blended 16-bit and 32-bit instruction set
  - Area savings through code density
- Can execute existing Thumb code
  - i.e., Thumb code from ARM7TDMI, ARM926EJ-S onwards
- Upwards compatible with Cortex processors
  - Migration to ASIC/ASSP & MCU at all performance points
ARM Cortex-M1 Availability

- ARM Cortex-M1 processor RTL license from ARM
  - Support for vendor-independent flows
  - Flexible, FPGA-agnostic implementation
  - Support for FPGA vendor tools (Actel Libero, Altera Quartus, Xilinx ISE.)

- FPGA development solution available from ARM
  - ARM Cortex-M1 Processor
  - ARM RealView Development Suite
  - ARM RealView Emulation Baseboard

- Available on Actel FPGA
  - Available to Actel customers at no license fee
  - See www.actel.com
ARM Cortex-M1 Processor Summary

- The first ARM processor designed for FPGAs
  - Compatibility and ecosystem across FPGA, ASIC/ASSP/MCU
  - Low-cost entry for designers wanting to create ARM SoC
  - Migration from MCU to ASIC/ASSP
- Actel lead partner and first licensee
- ARM will also license RTL directly to end users
  - Vendor-independent beta available to lead partners
  - Due end Q207
Rapidly Creating And Verifying Cortex-M1 Designs
How To Exploit Cortex-M1 In YOUR Design

- Cortex-M1 is designed for rapid design and deployment

- Explore the data, tools and processes that can ensure your Cortex-M1 based design can be
  - Created
  - Verified
  - Synthesized
  - Booting Software

... in the shortest possible time, and with the highest level of confidence
Cortex-M1 is supplied with IP-XACT data

- IP-XACT is the XML Databook format from The SPIRIT Consortium ([www.spiritconsortium.org](http://www.spiritconsortium.org))

```xml
<spirit:vendor>arm.com</spirit:vendor>
<spirit:library>ip.cortex</spirit:library>
<spirit:name>cortex_m1</spirit:name>
<spirit:version>r0p0_00bet0</spirit:version>
<spirit:busInterfaces>
    <spirit:busInterface>
        <spirit:name>External</spirit:name>
        <spirit:busType spirit:version="r1p0" spirit:library="busdef.amba.amba3" spirit:name="ahblite" spirit:vendor="amba.com" />
        <spirit:master>
            <spirit:addressSpaceRef spirit:addressSpaceRef="AHBExternal">
                <spirit:baseAddress>0x00000000</spirit:baseAddress>
            </spirit:addressSpaceRef>
        </spirit:master>
    </spirit:busInterface>
    <spirit:connection>required</spirit:connection>
    <spirit:signalMap>
        <spirit:signalName>
            <spirit:componentSignalName>HCLK</spirit:componentSignalName>
            <spirit:busSignalName>HCLK</spirit:busSignalName>
        </spirit:signalName>
    </spirit:signalMap>
</spirit:busInterfaces>
```
Platform Express
Drag and Drop Design in Minutes

Simple system diagrams represent complex designs

Consistent HW and SW programmers view

IP described in XML Databook
3 Clicks To Create a Synthesizable AMBA AHB Bus
Seamless RTL Environment Generation

Parameterize Component Data
Generate Top HDL & DecoderNetlist
Compile HDL
Configure Boot code for Memory options
Create Instance Specific Diagnostic Modules
Compile C Modules
Assemble & Link With Boot Code
Generate Seamless Configuration
Generate Dynamic AddressHandling

Seamless RTL

Modelsim Library

SW Object File with Debug Info
CVE File
Modelsim/CVE Remap Scripts
Make A Design Change

- Parameterize Component Data
- Generate Top HDL & DecoderNetlist
- Compile HDL
- Configure Boot code for Memory options
- Create Instance Specific Diagnostic Modules
- Compile C Modules
- Assemble & Link With Boot Code
- Generate Seamless Configuration
- Generate Dynamic AddressHandling

Seamless RTL

- Modelsim Library
- SW Object File with Debug Info
- CVE File
- Modelsim/CVE Remap Scripts
Demo 1

- Create and Verify and ARM design from nothing!
IP Reuse Verification Strategies

- When working with IP that is not fully understood, a rigorous automatic verification is essential.

- The primary focus
  - Ensure the IP is working in the new design as the original designer intended.
  - Ensure the IP is correctly integrated into the design.
  - Ensure the IP is functioning as required in the design.
Assertions

- Assertions enable the IP designer to embed expert knowledge into the IP
  - Checking the IP is being operated with as the original designer intended.
  - Checks for the IP being used in unintended/untested modes.

- In Platform Express
  - Supported Assertions formats include PSL, OVL.
  - Assertions delivered with IP are documented as IP-XACT filesets.
  - Available assertions are automatically built into the verification environment as part of the design generation process.
Adding Information About Assertions

- Assertion information is documented in the IP-XACT component data information.
  - Files can have more than one type for embedded assertions.

- Generators look for, and pick up this information for use in the target verification environment.

```xml
<spirit:fileSet spirit:fileSetId="vhdlSrc_psl">
  <spirit:file>uart.vhd</spirit:file>
  <spirit:fileType>vhdl</spirit:fileType>
  <spirit:fileType>psl</spirit:fileType>
</spirit:fileSet>

-- psl property valid5Parity is always
  {ValidStart; ((not WLS0) and (not WLS1) and (PEN))} |=>
  {((SAMPLECNT = "0111") and (RxClkEnab = '1'))[->6];
   (DIN = (parityCheck (EPS, SP, RX_INT)))}}
  abort (not RESETN));

-- psl assert valid5Parity;
```
Interface Protocol Monitors

- Protocol Monitors
  - The IP may work standalone, but does it work when connected to other IP.
    - How closely does an interface conform to the standard?
    - Many interface types have high variability which can result in ‘uncertain’ design.
      - It depends to what these interfaces are connected.

- In Platform Express
  - Supported Protocol monitor formats include 0-In, PSL, OVL.
  - Protocol monitors are documented as IP-XACT IP with monitor interfaces.
Powerful Functional Debug and Analysis

Adding Protocol Monitors
Powerful Functional Debug and Analysis
Analyzing The Results
Modular Testbenches

- When IP is added to a design, testbenches used to verify that design must be updated in synchronization.

- In Platform Express
  - Testbench content can be automatically added when an IP is added to the design.
  - Testbenches can be built manually use IP-XACT documented Verification IP.

- Two key technology steps to look out for:
  - The Advanced Verification Methodology is helping modularize and sequence testbench components
  - IP-XACT 1.4 will add support for AVM interface types
Mentor Verification IP

- Multi-abstraction VIP that ties RTL and TLM together
  - Built using Advanced Verification Methodologies (AVM) principles

- Portable across the environment and languages
  - Simulation, formal verification, acceleration and emulation

- Benefits
  - Reduces testbench development and refinement time
  - Improves overall verification efficacy with smarter verification cycles
  - Self-contained verification components for maximal reuse
Advanced Verification Methodology

- Verification ReUse across Abstraction and Projects
  - SystemVerilog & SystemC implementations
  - Mixed language support
  - ABV, CDV, C-R testing & scenario generation

- ‘Componentized’ verification offers many new IP ReUse automation possibilities.
Software Based Verification

- For Processor-based designs
  - Software testing can be an efficient way of building system tests
  - Booting a processor system, and testing that all peripherals can respond to the processor is a key milestone in any SoC design.

- In Platform Express
  - Diagnostics SW modules packaged with IP are documented as IP-XACT filesets.
  - Generators bind the IP SW modules with boot code to create an executable program.
  - The SW can be targeted for different verification environments - including live targets.
Let’s Get Some Software Running!

- Some basic software applications can be directly generated from the IP-XACT design
  - Boot the processor and get some diagnostic responses

- For application software development, a fully-fledged OS enables fully optimized exploitation of the processor’s capabilities
  - Build your applications upon an industrially-proven foundation
IP-XACT: Getting That First Program to Boot

```c
extern int a92SDiag_1(void);
extern void printToPort(int, char *);
int pxDiagEnd(void);
int TestMain()
{
    int errorCode = 0;
    pxSetIrr(0);
    pxRestoreDefaultIrr();
    pxDisableInterrts();
    printToPort(0, "a92S Diagnosis ...");
    errorCode = a92SDiag_1();
    switch (errorCode)
    {
    }
}
```

Boot Code

- Interrupt Handler
- UART Tests
- CAN Tests
- FlexRay Tests
- C Main Intent

Compile

```
cc
ld
```

test.x
Porting the OS

- Boot Code
- Interrupt Handler
- UART Tests
- CAN Tests
- FlexRay Tests
- C Main
- Intent

cc
ld
test.x

OS & Device Drivers
Nucleus OS and Cortex-M1

- Cortex-M1 is designed for efficient implementation within an FPGA and has software-specific configurability.

- Nucleus OS takes advantage of the optional hardware:
  - System timers
  - Interrupt controller
  - Banked exception registers for speedy context switching

- Nucleus was the 1st OS to fully support all the major FPGA vendors processors, and is proven over a wide range of applications.
Nucleus OS

- Nucleus is the world’s most widely deployed OS
  - Mobile/consumer/medical/industrial application focus
  - Highly scalable
  - With a wealth of middleware for all embedded applications
    - 1st commercial OS to be ported to ARM in 1993
    - 1st commercial OS to be ported to Cortex-M3
    - Is on more ARM devices than any other commercial RTOS

- #1 in cell phones – 400 to 500 million last year
- #1 in DVD players
- #1 in Set top boxes
- Supports ARM7, ARM9, ARM10, ARM11 and Cortex families
Nucleus Is Part of a Complete Embedded Software Solution

Development Tools

<table>
<thead>
<tr>
<th>EDGE</th>
<th>IDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compiler</td>
<td>Debugger</td>
</tr>
<tr>
<td>Simulation &amp; Testing</td>
<td>Profiler</td>
</tr>
<tr>
<td>JTAG</td>
<td></td>
</tr>
</tbody>
</table>

Application Platform

Inflexion Platform

- Multimedia
- Camera
- Telephony
- VoIP
- Customer Application
- Customer Application

Inflexion Platform UI

Operating System

Nucleus OS

- Networking
- GUI
- File System
- USB
- Bus Support
- Security

Kernel
Demo 2

- Add Verification IP
Time To Synthesize
Precision FPGA Synthesis

- Precision Synthesis is the tool of choice to target Cortex-M1 designs at the FPGA technology of **YOUR** choice

- High performance, multi-vendor, physically aware synthesis
  - Excellent quality of results using advanced optimization techniques
  - Excellent incremental debug and analysis environment identifies and fixes problems early in the design process
  - Excellent ease-of-use

- Language neutrality supports any combination of VHDL, Verilog, SystemVerilog and EDIF usage
  - A key capability when using many different sources of IP
The Design Works, So Let’s Synthesize!

- Invoke the Precision Synthesis generator
- Select the target FPGA technology
- Choose to work in batch or interactive mode
Integration to All Major FPGA Vendor P&R Tools

- Select the optimum FPGA technology for your design
  - Locking into one vendor too early in the design process can be an expensive design choice
Powerful Functional Debug and Analysis Cross Probing

- Analyze the actual design
  - Powerful cross-probing capabilities
    - Gate-level domain
    - RTL domain
  - Critical paths views
    - Fragmented path view
    - Gate-level technology view
Powerful Functional Debug and Analysis
Accurate Timing Analysis and Debug

- Powerful integrated timing analysis
  - Incremental timing
    - Fast optimization feedback
  - Perform “what-if” analysis
    - Without re-synthesizing
  - Easily identify design issues
    - Critical instances
    - Long interconnects

Precision earned FPGA Journal users’ choice award for Best Design Analysis capabilities
From FPGA to ASIC Prototyping

- For ASIC prototyping, your synthesis tool needs:
  - SDC timing constraint support
  - Precision Synthesis supports industry-standard Synopsys® Design Constraint (SDC) format
  - Use Precision Synthesis-generated SDC file for ASIC design after completing FPGA implementation

- Gated clock conversion functionality
  - Automatic translation to clock-enabled logic takes advantage of dedicated clock circuitry

- TCL command shell capability
  - Fully supported SDC command set, including complex multi-cycle and false path definitions
  - Interactive design query and traversal commands
  - Easy automation through scripting

```
# Clock Constraints
create_clock clk -period 10

# Input Constraints
set_input_delay 2 data_en -clock clk
set_input_delay 3 data_in* -clock clk

# Output Constraints
set_output_delay 3 data_stb -clock clk
set_output_delay 8 data_out* -clock clk
```
Demo 3

- Add Cortex M1 and Synthesize
Summary
For More Information

- **Processors**: www.arm.com
  - Cortex M1: arm.com/fpga

- **Design Tools**: www.mentor.com
  - Platform Express: mentor.com/platform_ex
  - Scalable Verification: mentor.com/products/fv/
  - Precision FGPA Synthesis: mentor.com/synthesis

- **Operating Systems**: www.mentor.com
  - Nucleus RTOS: mentor.com/embedded

- **IP-XACT**: www.spiritconsortium.org
Summary

- ARM Cortex-M1: optimized embedded FPGA Design
  - Optimized for all major FPGA technologies
  - Leveraging industry-standard processor architectures and tools

- Mentor Graphics makes it easy to use Cortex-M1 in your embedded FPGA designs
  - Leveraging IP-XACT information to rapidly create and verify designs
  - Targeting those designs for a wide range of FPGA technologies
  - Providing extensive FPGA-proven OS support for application development

- Mentor Graphics and ARM
  - Creating your complete embedded FPGA processor design applications has never been easier