Advance your Design and Verification Flow Using IP-XACT

Edwin Dankert, Arm
Maximilian Albrecht, Texas Instruments
Vincent Thibaut, Magillem Design Services
Tutorial Outline

• Arm: IP-XACT Enabling the eco-system
• TI: IP-XACT based automation flow
• Magillem: IP-XACT usage in the SoC design domain and beyond
• Q&A
Arm IP-XACT
Enabling the eco-system

Edwin Dankert
Overview

• Why?
• IP Process
• Bus Definition Process
• Socrates
• Usage of M3 Design Start for IP-XACT 202? Validation
• Roadmap
• Feedback ...
Why?

• To provide an interoperable representation of an Arm IP or protocol in a machine readable language (Integration ready IP)

• For our IP partners to enable
  – easy integration of Arm IPs into their Systems
  – generation of netlists, documentation, header files
  – enable down-stream (verification) flows

• For our EDA partners to build their tools around
IP Creation Process: Guidelines

• Define Arm specific internal guidelines to ensure Consistency

• Limiting Capabilities
  – Makes it easier for EDA tools to understand ARM IP-XACT
  – Not using features like channels, mirrored interfaces, alternate-registers, isPresent
  – Resolving parameters during configuration

• Ensure the names of interfaces, registers, memory-maps are consistent for different Arm IP
  – To ease integration/understanding. (DBG_S vs DEBUG_Slave)
IP Creation Process : Quality

• Automated Validation
  – Arm specific rules
    • All Ports mapped into interfaces
    • Interface naming
  – RTL vs IP-XACT
    • Matching Ports and parameterisation

• Reviews
  – Non automated checks ...
  – All Resets and Clocks should be mapped into an interface
Bus Definition: Creation Process

• Guidelines - to ensure consistency and ease of use
  – (Re)use of pre-defined bus-defs (central repository)
  – Naming of bus-definitions and abstraction definitions
  – Extending of bus and abstraction definitions
  – Logical Port require qualifiers (requiresDriver for clock and reset) and direction
  – Require default values when optional

• Quality
  – Automated Validation (IP-XACT IEEE + Arm Compliance)
  – Reviews
    • Ensure the bus-def correspond to the protocol definition, ensure clocks and resets are defined for synchronous bus-defs ...
  – Release (AMBA Bundle, Socrates, IP Specific)
Extending bus definitions, many options can be chosen:

- Extend to enable updates (v1.0 -> v1.1)
- Extend to support different protocol versions (AXI AMBA3 -> AXI AMBA4)
- Extend to connect between members of the same protocol families

Arm has decided to only extend between members of the same protocol family:

- Note: Phantom Bridges for other conversions will be provided on request.
Bus Definition: arm:revision

• To enable updates to Arm Bus-Definitions
• VLNV update causes major issues for our partners
  – Requires updating of all IP used within a system to match the new bus-def
  – Or requires the extends mechanism to be used by the bus-definition with the updated VLNV (no longer allowing for other extends AXI5-Lite->AXI5)
• Instead arm:revision was introduced
  – Each compatible update is accompanied by an arm:revision increase
• Note: incompatible updates will still see a VLNV change, we are trying to avoid these as much as we can!
Bus Definition: arm.com vs amba.com

- **arm.com**: non-AMBA bus-definitions
  - Clock, Interrupt, DFT, MBIST, StaticCfg, IP (CoreSight/Crypto/Core) specific

- **amba.com** bus-defs use a very open license
  - use as part of own IP and to re-distribute (with EDA tools)

- **amba.com** bus-defs are released at intervals
  - ideally to coincide with any AMBA protocol spec releases

- Latest **amba.com** versions can be downloaded from:
  - [https://silver.arm.com/browse/AR500](https://silver.arm.com/browse/AR500)

- Also included with Socrates and with individual IP Bundles!
Socrates

• Configures Arm IP with consistent IP-XACT
• Ensures IP-XACT quality
• Ensures all IP-XACT is provided using IEEE 1685:2009
• Ensures that a consistent set of Bus-Definitions and Abstraction Definitions are shared between all configured IP
• Introduces Phantom bridges for protocol conversion
IP-XACT 2.1 Validation

- Allowing for IP-XACT 2.1 specific updates for validation purposes ...
- Donation of M3 Design Start IP-XACT descriptions
- IP-XACT for the IPs (including port and register descriptions)
- Enabling a description of the full sub-system
Roadmap

• Update to AMBA bus-defs (Nov 2019)
  – Including AMBA5/AMBA4 AXI RO/WO, CHI-C/CHI-D, ACP, P-Channel/Q-Channel

• IP-XACT 2014 support (2020)
  – Updates to the AMBA bus-defs (more than just a simple up-conversion, includes adding parameterisation and other 2014 features)
  – Updates to the Components to support this parameterisation
  – All IP supported (using Socrates)
Feedback

• We welcome any feedback around our delivered IP-XACT!
• Please feel free to let our support team know about any of your questions/requirements/requests!
• Otherwise, feel free to contact me directly: edwin.dankert@arm.com
IP-XACT based automation flow

Maximilian Albrecht
Robert Lessmeier, Mark Jung, Franz Mayr
Agenda

1. Higher SoC and workflow complexity
2. Modularity of SoC architecture
3. Flow requirements
4. IP description in IP-XACT
5. Peripheral database as a single golden source
6. IP Packaging & Mapping BusDefs
7. SoC Configuration
8. Generating SoC design data: Netlist and Power Intent
9. Documentation generation
10. Header file generation
11. Lessons learned
12. Summary slide
Higher SoC and workflow complexity

- Cross-functional distribution of deliverables
- Roles and responsibilities distributed across the globe
- Existing flows are now inefficient or dysfunctional
Modularity of SoC architecture

• Modular platform for a family of devices

• Dozens of high-quality re-usable IPs

• Highly configurable

• Flawless execution

• Rapid spins
Flow requirements

Fast spins and high quality

Automation requires standardization

Use proven industry standard IP-XACT (IEEE-1685)

Distribute development & automation across the organization

IP-XACT XMLs need abstraction and version control

© Accellera Systems Initiative 20
IP description in IP-XACT (1)

- Registers and bitfields organized in address blocks
- Bit properties
- Connectivity interface
- Address map
- Component parameters
IP description in IP-XACT (2)

- Registers and bitfields organized in address blocks
- Bit properties
- Connectivity interface
- Address map
- Component parameters
IP description in IP-XACT (3)

- Registers and bitfields organized in address blocks
- Bit properties
- Connectivity interface
- Address map
- Component parameters
Peripheral database as a single golden source

**IP Dashboard**
- Activity
- Tag
-Latest control
- QC Report
- Generator Status

**Quality Checks**

**IP Database**
- Standardized
- Rule-checked
- Machine-readable
- Consolidated
- Cross-functional
Metadata abstraction delivered with each IP release
Used by SoC integration and for IP documentation
Physical ports in the HDL are mapped to logical interface ports of a bus

- IP-XACT buses comprise a logical set of ports
- Bus interfaces of the same bus definition (VLNV) can be connected on SoC level
- Enables rule based connectivity
SoC configuration

- IP Instantiation and parameterization
- Pinout
- Interrupt assignments
SoC Netlist Creation

SoC Configuration
- SoC Config
- IP Config
- IP Database

SoC Design
- Connect Rules

SOC Creation
- SOC IP Creation
- Power Intent (isolation)

SoC IP
- UART
- SPI

Power Intent
- Isolation

IP Config
- IP-XACT

SOC Netlist (HDL)
Create Documentation

SoC Collateral Creation

Doc Creation

SoC-specific Datasheet
- Peripheral Memory Map
- Functional Block Diagram
- Pinout - Drawing & Tables
- Device Tables (IRQ, DMA, Event, IO)
- Spin Comparison Table

Device-Family TRM
- Component Register Descriptions
- Family Comparison Table
Header file generation

IDE support files and CMSIS-compliant software collaterals
Lessons learned

- Missing features in IP-XACT 2009 standard
- Potential of expanding standard by vendor extensions and parameters
- Early involvement of cross-functional stakeholders
- Single golden source
- Workflow based on version control
Summary

- Innovative approach based on automation on all fronts
- Scalable solution for fast spins
- Potential re-use of workflows and generators by other teams
IP-XACT usage in the SoC design domain and beyond

Vincent Thibaut
Coping with exploding complexity

Intellectual Properties

IP reuse methodology defines components boundaries to become the abstraction level for today’s design flow task such as SOC Assembly.

- RTL
- Gate level
- Transistor level
IP-XACT based front end design need a tool

For editing:
XML is not meant to be edited directly
Inside a single IP-XACT file multiple dependency and coherence need to be resolve

For Checking:
Good quality IP-XACT is a must for maximum benefit

For platforms elaboration:
Platforms are composed of multiple XML files and need to be support multiple configuration which need to be resolved

For generation:
It is critical to be able to generate multiple outputs from an IP-XACT description
Basic Flow for Different Users and Use Cases

End Users and Use Cases
- System architect: architecture design
- IC integrator: IC integration
- Verification engineer: test (bench) generation
- Embedded software engineer: HAL and drivers
- Technical writer: data sheet and user manual

Independent views from IP Providers

Coherent views in IC eco-system

"Electronic data sheets"
- Bus interfaces
- Ports
- Registers
- Parameters
- Views
- Files
- Interconnect
- Instances

Design Flow Automation

Design Repository

IEEE 1685

TCL/CLI/GUI

TGI

DOC

SW

ESL

RTL

V&V

IEEE 1685

© Accellera Systems Initiative 2019

Accellera Systems Initiative
NXP Engaging with Magillem for IP-XACT Tool Support

- SystemC-TLM IP integration flow since 2006
- Software register header and memory map flow since 2007
- RTL IP integration flow since 2009
- Hardware compilation and elaboration flow since 2010
- Mixed RTL/TLM, software-driven, test bench generation since 2011
- TLM register generation since 2012
- Verilog-AMS IP integration flow since 2013
- DITA-based documentation generation since 2014
- CMSIS-SVD generation since 2015
- UVM register model generation since 2015
- SystemC-AMS IP integration flow since 2016
- RTL register generation since 2017
General Design Flow Overview

Integrate – Assemble
Update
(Instantiate, Configure, Connect, Hierarchy, Restructuring, Partition, Incremental design)

Package

Check

Generate

Platform derivatives

Protocols
(Amba, OCP, custom)

TCL, Python, Ruby, Java

Elaborated Design DB

© Accellera Systems Initiative
Benefits of the solution:

- Preserving IP assets
- Reducing time to market
- Improving quality
- Saving money

“A catalyst of R&D methodology transformation”

Significant ROI: Overall Time to Market gain = 25% on first design! 50% on derivatives!
Building more automation on top of design

- Design flows
  - ECO
  - BIST Insertion
  - Daisy Chains
  - Power Intent

- Integration Flows
  - Parallel integration
  - Derivative design
Design Data Intelligence:
All reference information at hand

• Products, SoC and IP referenced in a single environment
• Crystal Bulb is a lightweight client application (web browser)
• All teams access up-to-date information in a glimpse
IP-XACT enable connection around the pure design activity

- Verification
  - UVM register verification
  - Testbench generation
- Documentation
  - Datasheet generation
- Traceability
Electronics system development

- Electronics system engineering follows a standard V-cycle integrated with the upper system layer.
- Critical electronic systems require a strong traceability for certification purpose.
- Electronic systems integration is based on the successive integration of larger blocks from IPs, to SoCs, to Boards and finally as part of the full system: system makers now design their SoC!
Ready for a complete System Design Capture

An electronic platform development process in a:

• Multi Site

• Multi Format

• Collaborative work context

requires a backbone of data exchange top down and bottom up
Vertical integration for HW design

Concept

HLR

System

Simulation

C++

SysC

Validation

Traceability

Documentation

Concept

HLR

System

Simulation

C++

SysC

Validation

Traceability

Documentation

Concept

HLR

System

Simulation

C++

SysC

Validation

Traceability

Documentation
Traceability and Documentation through the cycle

- **Technical document publication**
  - Integrate with technical formats
  - SoC Requirements and Documentation are available
  - Provide a full documentation publication solution

- **Linking Infrastructure**
  - Integrate with technical formats
  - Provides a multi domain synchronized referential
  - Enable real traceability by probing and monitoring actual design data
Traceability

- Model Interlinking
- Static and Dynamic views
- Full integration with Magillem Products
- Integrate with 3rd party tools
Questions