

Lessons from the field  
IP/SoC integration techniques that work

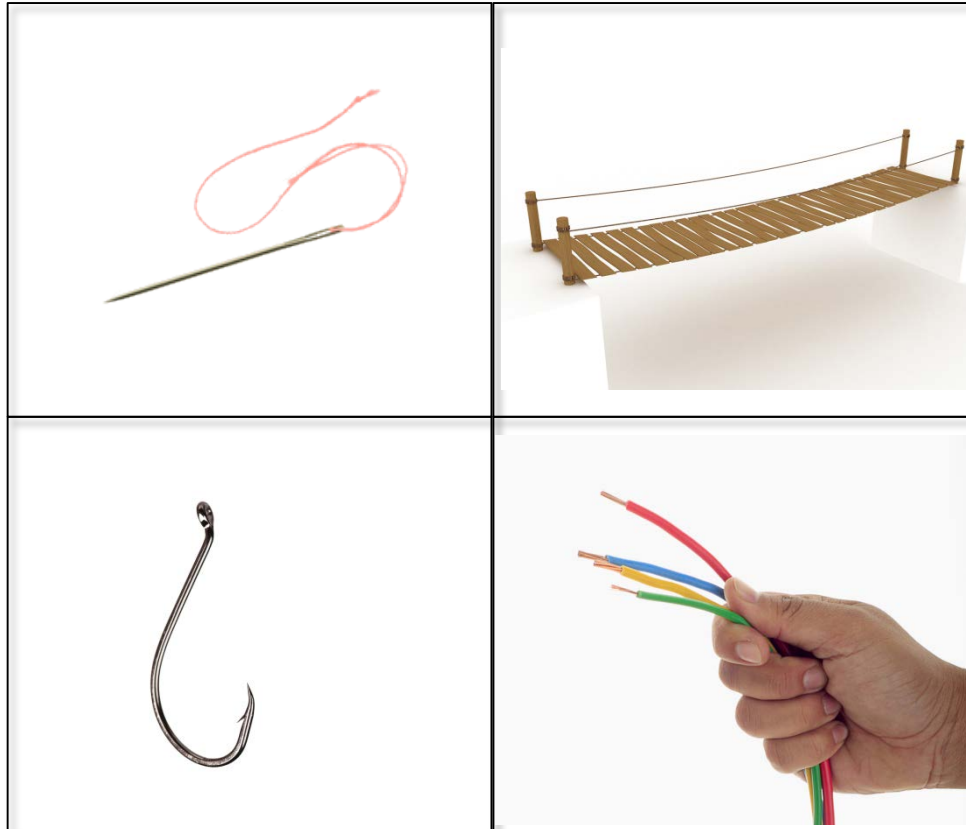
David Murray, CTO,  
Duolog Technologies



# Acknowledgements

- Sean Boylan, Duolog
- Sujatha Sriram, ARM

# 4 Pictures – 1 Word



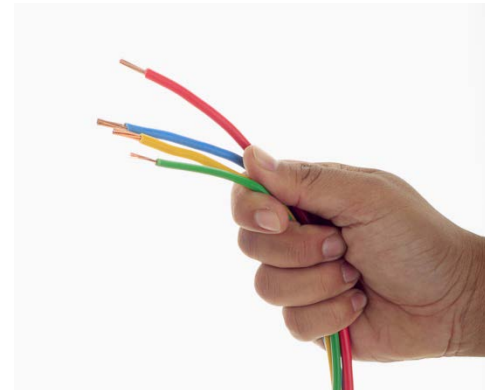
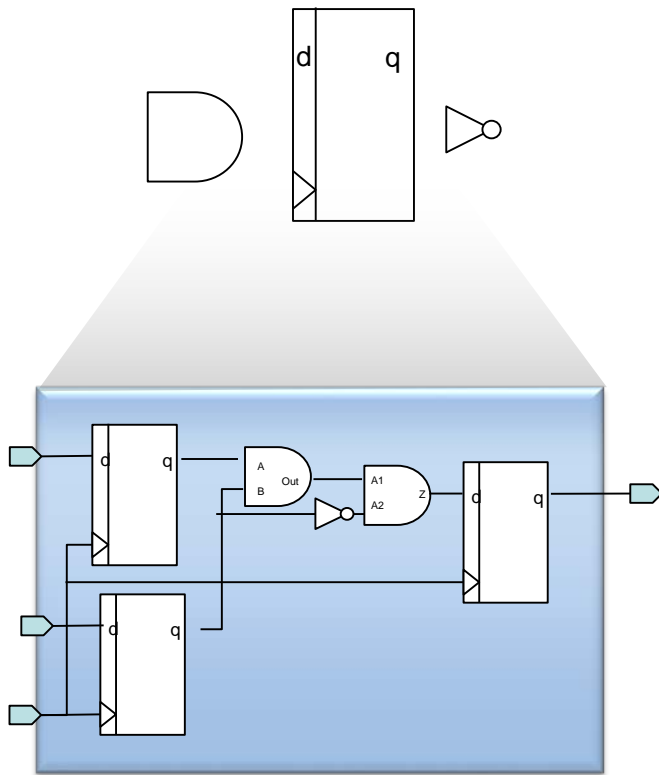
C O N N E C T



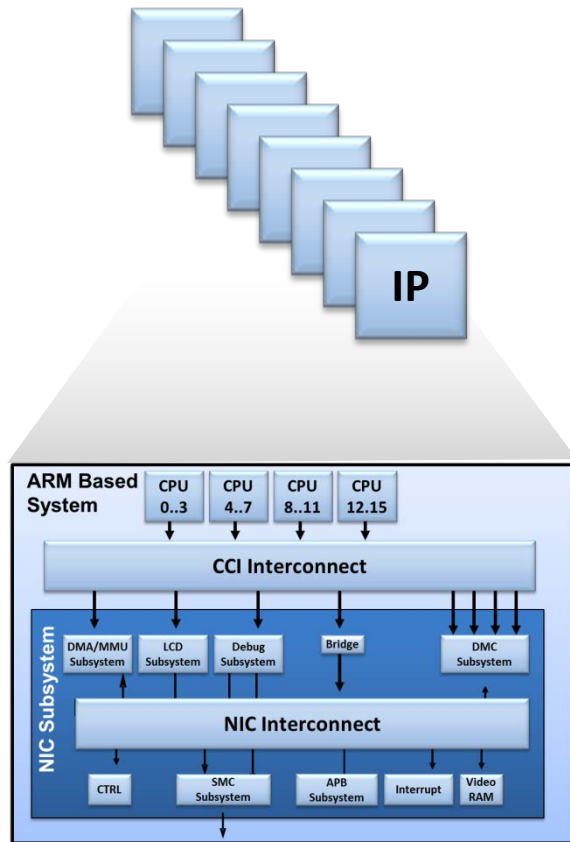
# Overview

- Challenges for IP-based SoC Integration
- Integration methodology (Rules-based)
- Case-Study/Results
- Future Directions

# IP Integration

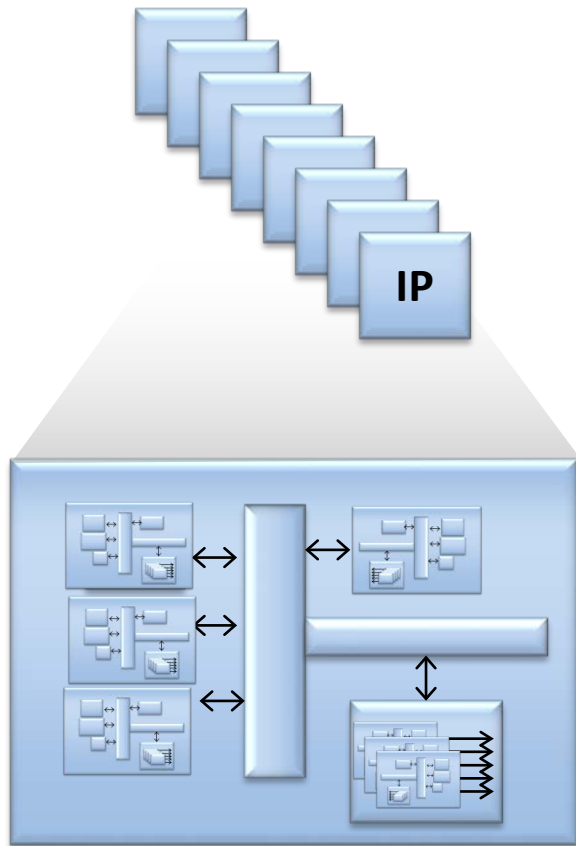


# IP-Based SoC Integration



- ARM Based system
  - 4 Processor clusters
  - 3 bus Interconnect systems (CCI/NICx2)
  - 4 sub-systems, (DMC, Peripheral, LCD, Interconnect)
  - 35 independent IP's to be integrated
  - The AMBA® protocols within the system included APB™, AHB™, AHB-Lite™, AXI™, ATP™, LPI™, AXI4™, APB4™, ACE™ and ACE-Lite™

# Challenge - Complexity



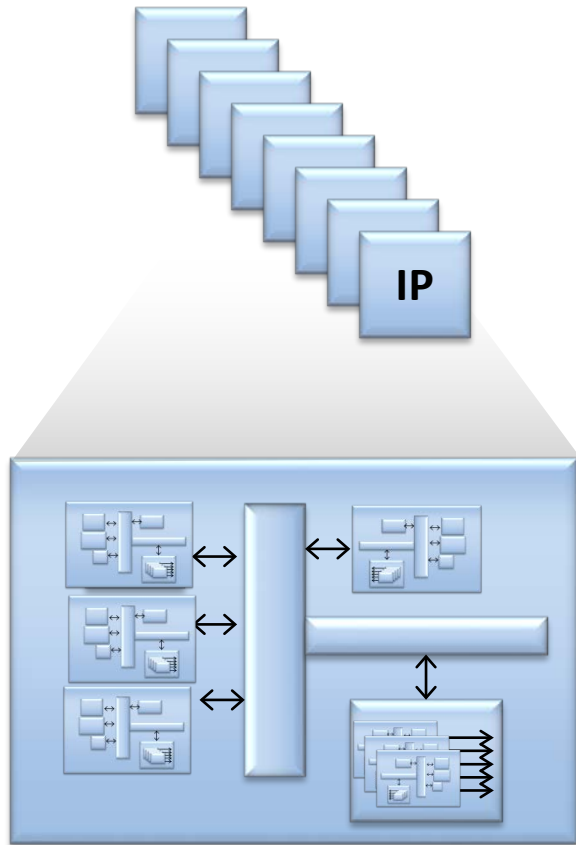
- Complexity
- Configurability
- Standardization
- Quality
- Collaboration



# Integration Solution

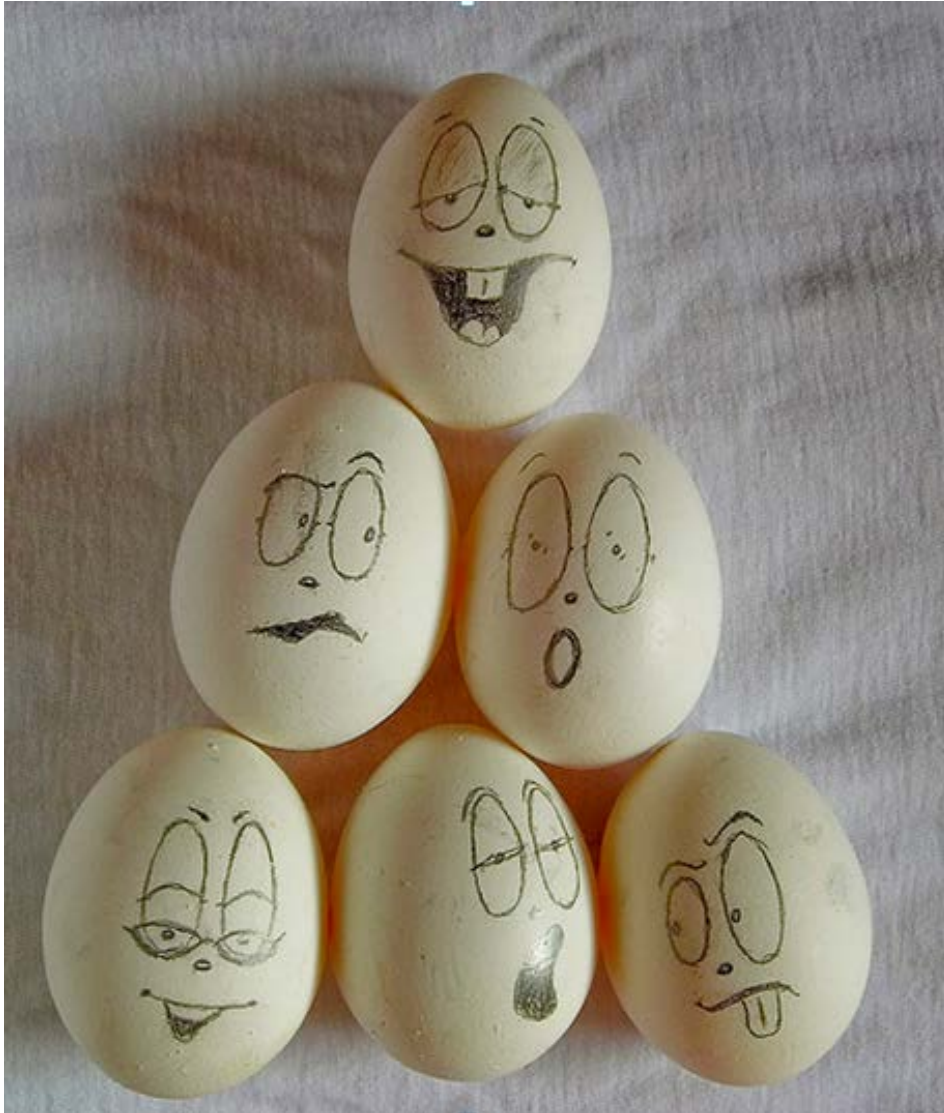


# IP-Based SoC Integration Solution



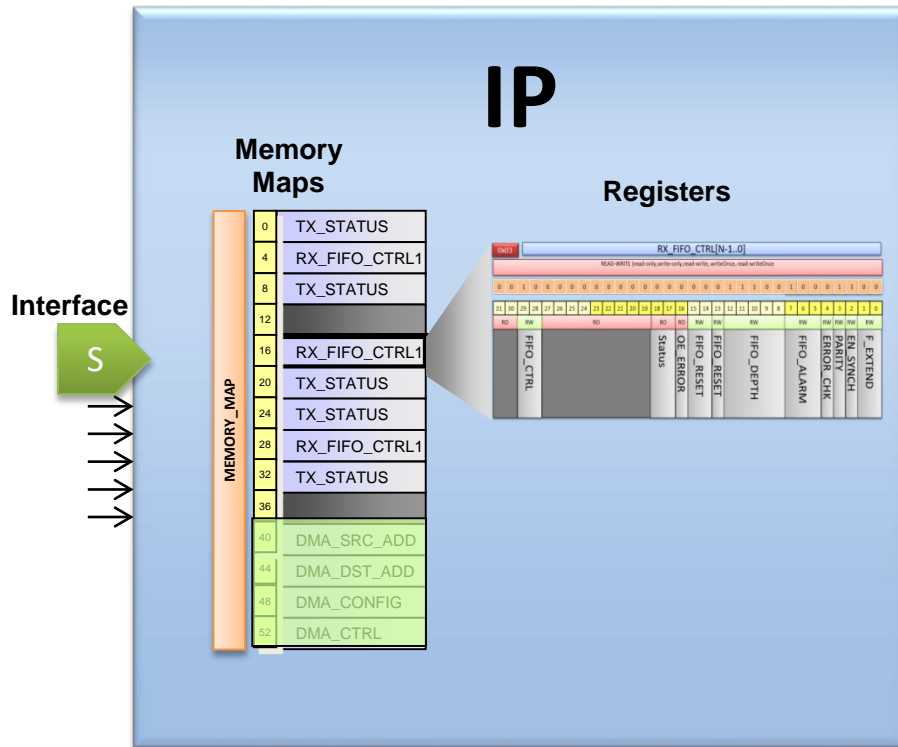
- IP Interface standardization
- Efficient SoC Assembly/Connectivity
  - Standard Connectivity
  - Hierarchy Management
  - Configurability
  - Reusability
  - Interoperability
  - Usability!

# Managing Hierarchy



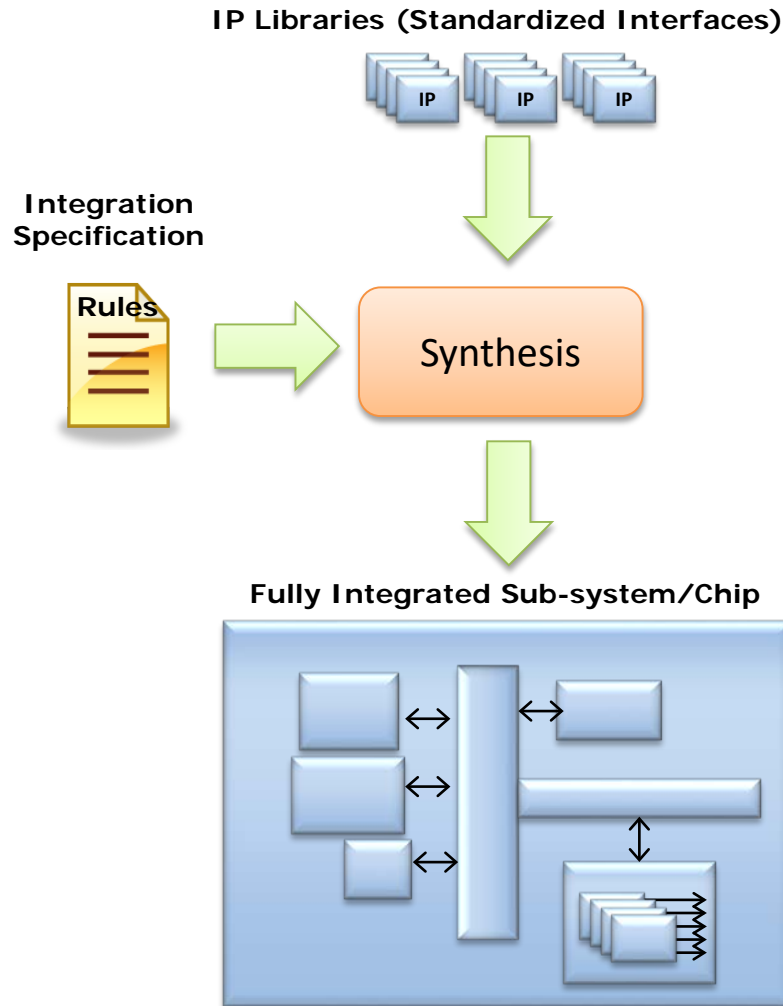
- Hierarchy
  - Managing Boundaries
  - Collaboration
  - Fluidity
  - Refactoring

# IP Interface Standardization



- Executable IP Interface Specification
  - HW Interface
    - Bus Interfaces
    - Ports (Mapping to bus definition signals)
  - SW Interface
    - Memory Maps
    - Registers
    - Bitfields
- Consumption
  - IP Automation Flows
  - Integration Flows
- Industry Standardization
  - IP-XACT
  - Common Bus definitions

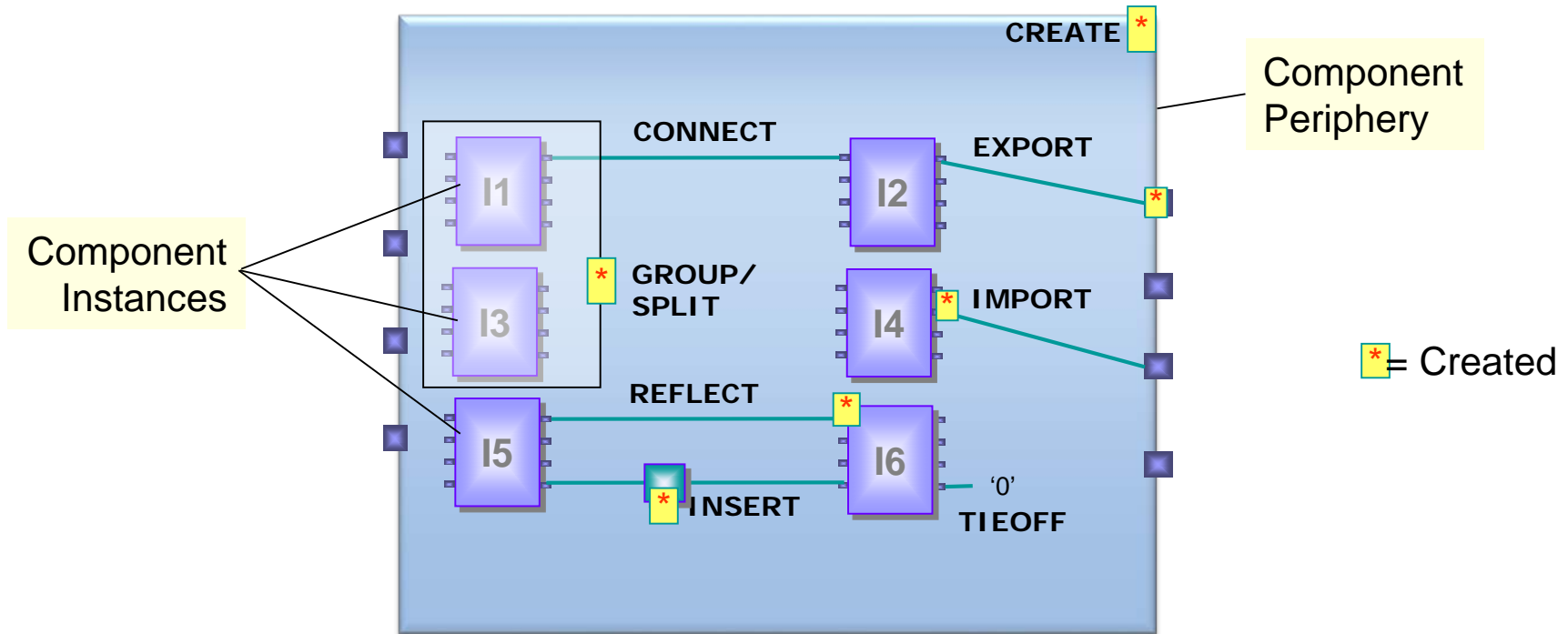
# Rules-Based Integration Methodology



- Select & configure integration-ready IP from libraries
- Rules-based system integration
  - Text-based Integration instructions
- Define & refine system hierarchy

# Integration Instructions

- Powerful Integration Instructions operate on IP Interface metadata



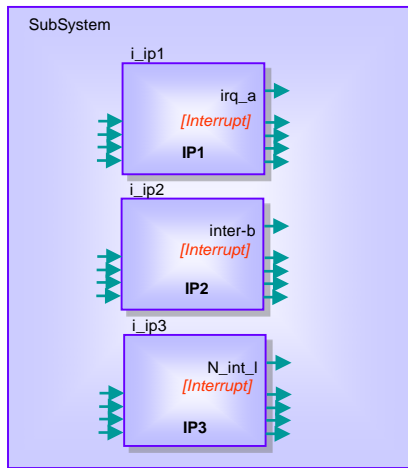
# Example

In the sample SubSystem, bring all instance ports of type **[Interrupt]** up to the next level of hierarchy and prefix the interface ports with the name of the source instance

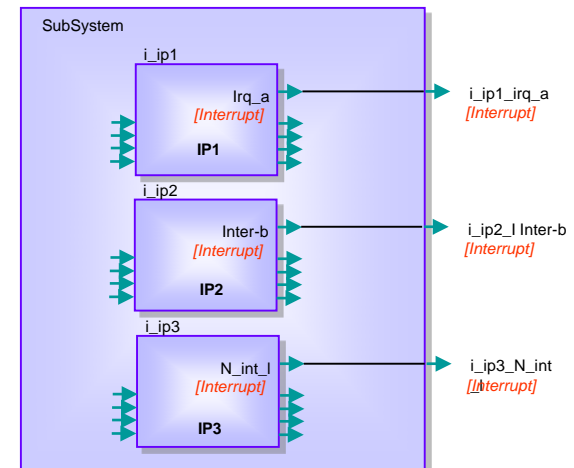
Integration  
Specification



```
# Export ports of type Interrupt
rule("Export Interrupts") {
    export instances.ports(definition "Interrupt"), :port_name => "${instance}_${port}"
}
```



Synthesize



# Connection Example

ACE Interface has 60+ signals

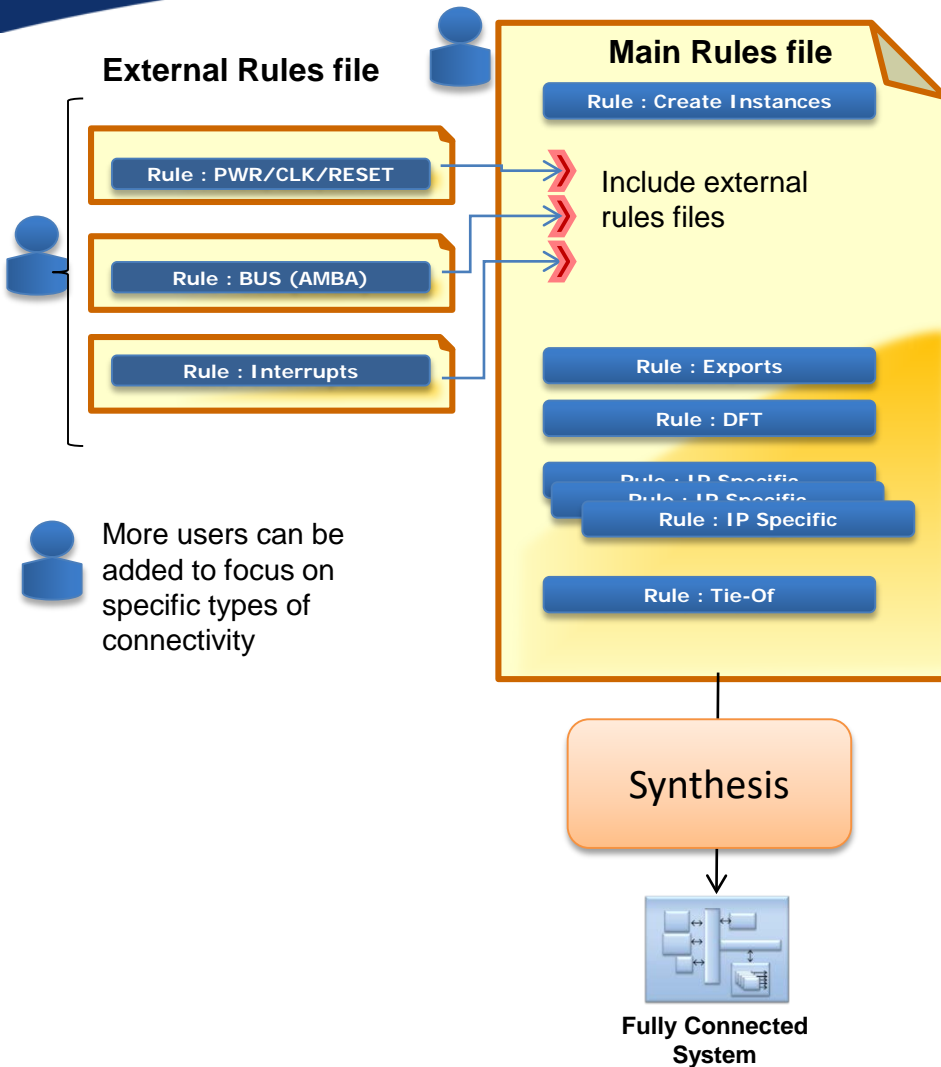
ACE
ACADDR
ACLK
ACLKEN
ACPROT
ACREADY
ACSNOOP
ACVALID
ARADDR
ARBAR
ARBURST
ARCACHE
ARDOMAIN
ARRESETn
ARID
ARLEN
ARLOCK
ARPROT
ARQOS
ARREADY
ARREGION
ARSIZE
ARSNOOP
ARUSER
ARVALID
AWADDR
AWBAR
AWBURST
AWCACHE
AWDOMAIN
AWID
AWLEN
AWLOCK
AWPROT
AWQOS
AWREADY
AWREGION
AWSIZE
AWSNOOP
AWUSER
AWVALID
BID
BREADY
BRESP
RUSER
RVALID
CDATA
CDLAST
CDREADY
CDVALID
CRREADY
CRRESP
CRVALID
RACK
RDATA
RID
RLAST
RREADY
RRESP
RUSER
RVALID
WACK
WDATA
WLAST
WREADY
WSTRB
WUSER
WVALID

- ACADDR
- ACLK
- ACLKEN
- ACPROT
- ACREADY
- ACSNOOP
- ACVALID
- ARADDR
- ..

Connect instances("uP1").interface("M1"),  
instances("uP1").interface("S1")

ACE
ACADDR
ACLK
ACLKEN
ACPROT
ACREADY
ACSNOOP
ACVALID
ARADDR
ARBAR
ARBURST
ARCACHE
ARDOMAIN
ARRESETn
ARID
ARLEN
ARLOCK
ARPROT
ARQOS
ARREADY
ARREGION
ARSIZE
ARSNOOP
ARUSER
ARVALID
AWADDR
AWBAR
AWBURST
AWCACHE
AWDOMAIN
AWID
AWLEN
AWLOCK
AWPROT
AWQOS
AWREADY
AWREGION
AWSIZE
AWSNOOP
AWUSER
AWVALID
BID
BREADY
BRESP
RUSER
RVALID
CDATA
CDLAST
CDREADY
CDVALID
CRREADY
CRRESP
CRVALID
RACK
RDATA
RID
RLAST
RREADY
RRESP
RUSER
RVALID
WACK
WDATA
WLAST
WREADY
WSTRB
WUSER
WVALID

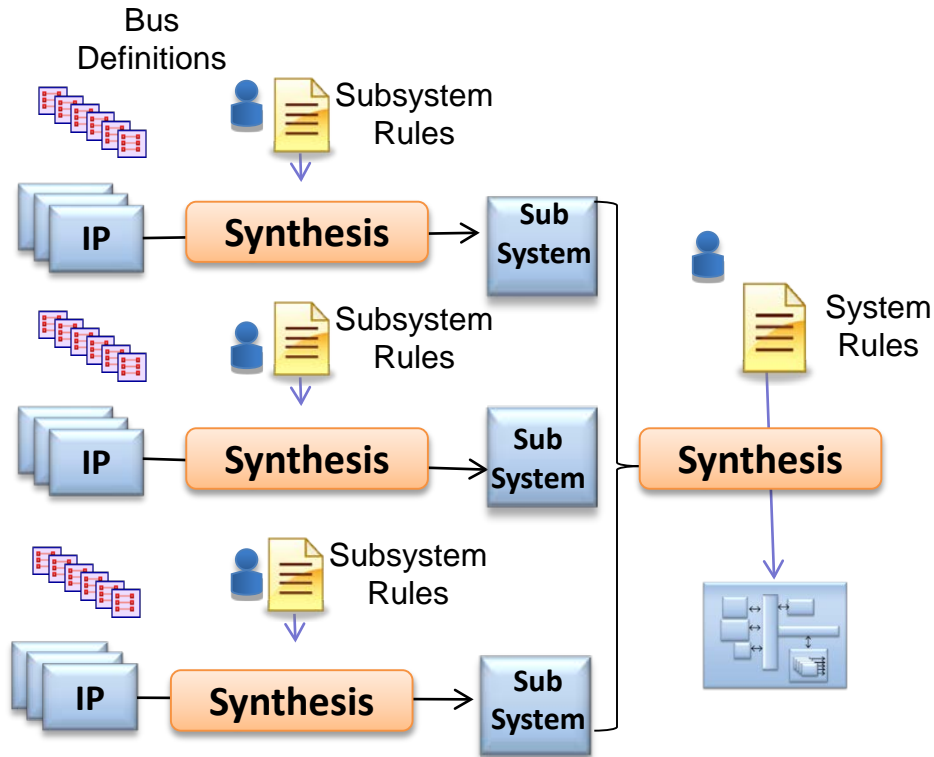
# Concurrent Integration



- Connectivity Ownership
- Merging or Include
- Autogenerated IP interfaces



# Full System Creation



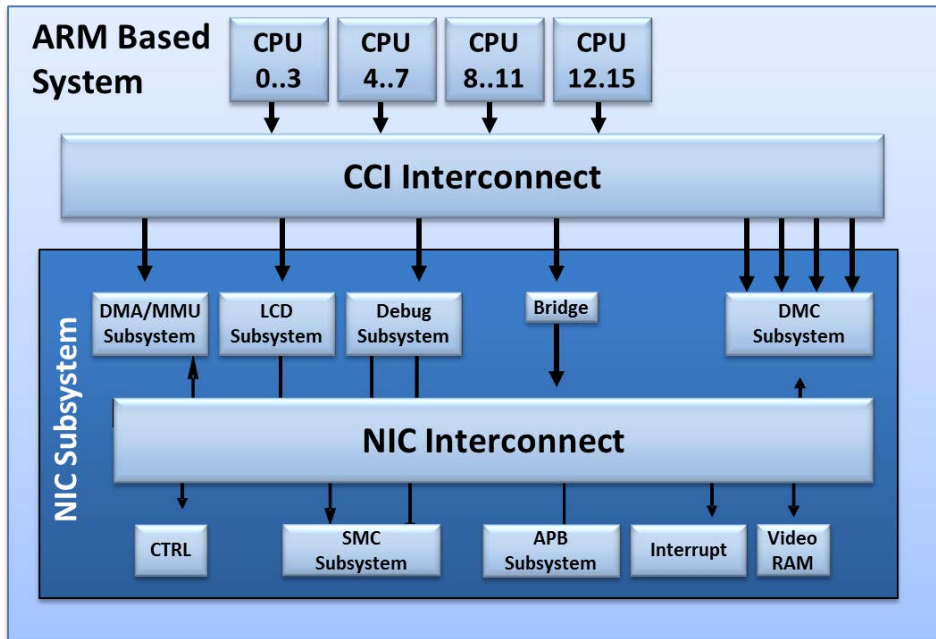
## • Features

- Synthesis of hierarchical system connectivity
- Netlist generation
- Rapid Integration timeframes
- Integrated with other flows
  - IO, Bus fabric, power, verification



Case-Study  
ARM IP-Based SoC Integration

# ARM IP- Based System



- ARM Based system
  - 4 Processor clusters
  - 3 bus Interconnect systems (CCI/NICx2)
  - 4 sub-systems, (DMC, Peripheral, LCD, Interconnect)
- 12,000+ lines of verilog code describing connectivity
  - 6-7 Weeks to develop normally
  - Many connectivity errors

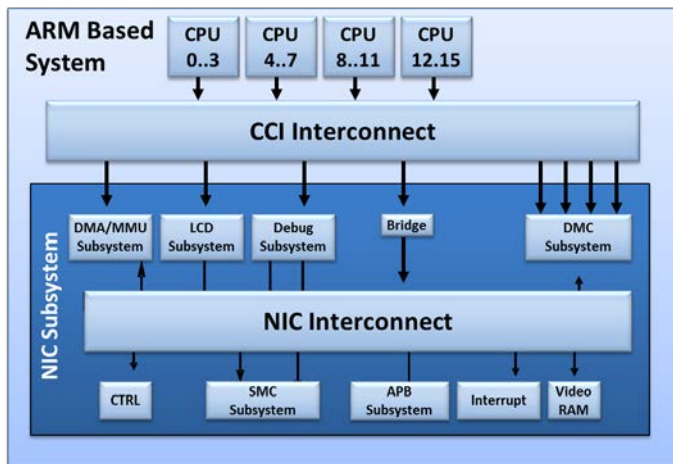


# IP Interface Standardization

- 35 IP leaf components
- Most IP had IP-XACT descriptions
- Additional IP packaging
  - The packaging of each IP, with creation of relevant bus interfaces took 1-2 hours per IP.
- Utilized 109 IP-XACT bus definitions

# IP Integration

- 3 engineers worked within 3 levels of hierarchy



Integration Focus	Sub Task	Designer 1	Designer 2	Designer 3
NIC Subsystem	DMC Subsystem	x		
	SMC Subsystem		x	
	LCD Subsystem			x
	NIC Subsystem	x		
Top-Level	Instance Creation	x		
	AMBA Connectivity	x		
	Export Connectivity		x	
	Interrupt Connectivity		x	
	Clock/Reset	x		
	Power Connectivity	x		
	CPU Cluster Connectivity		x	

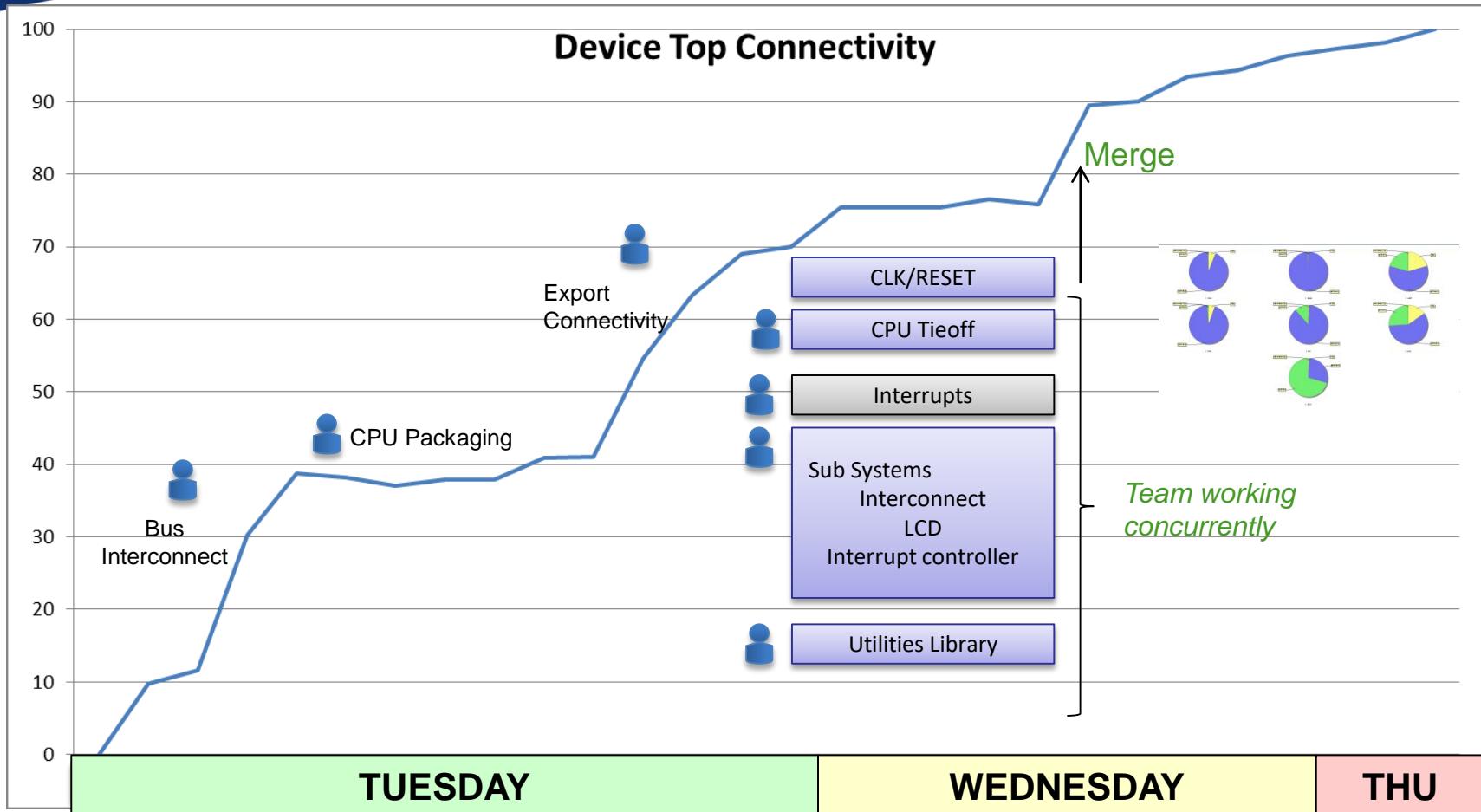
# Results

Integration Task	Duration (Days)
DMC Subsystem	1
SMC Subsystem	1
LCD Subsystem	1
NIC Subsystem	2
Top-level	2

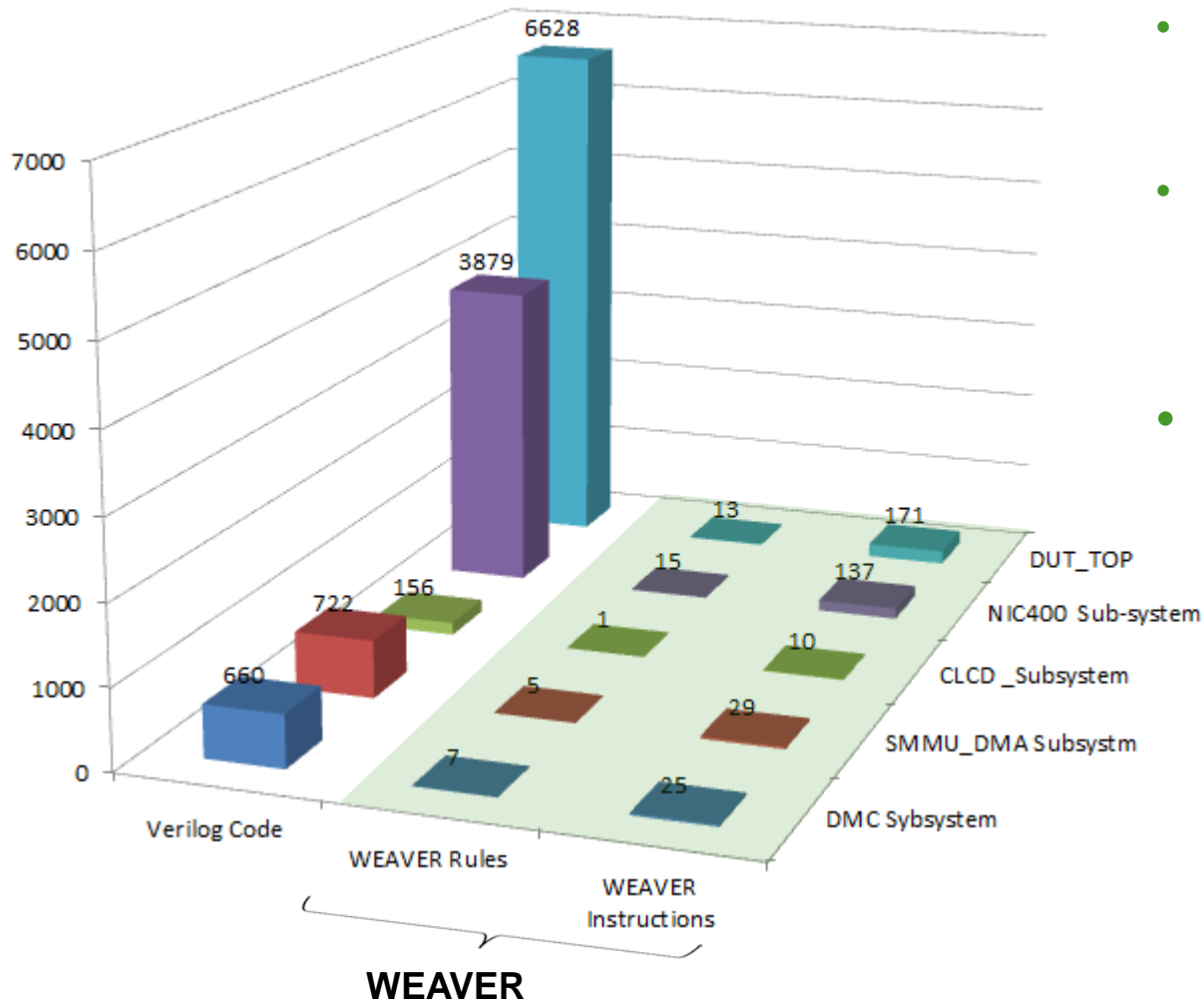
- The integration of 3 sub-systems, 1 major sub-system and a top-level integration was completed within 4 schedule days.
- This activity could have taken 35+ working days in the past.

**An 8x schedule improvement over previous methods**

# Metrics - % Connectivity/Day



# Metrics



- The 3 levels of hierarchy were put together using 41 rules, with 372 instructions.
- These were synthesized and netlisted to 12,045 lines of Verilog code in total

## Rules

- Higher level of abstraction
- High level of reuse
- Easier to maintain
- Higher quality

The average ratio of instructions to lines of Verilog code was 31:1





# Considerations

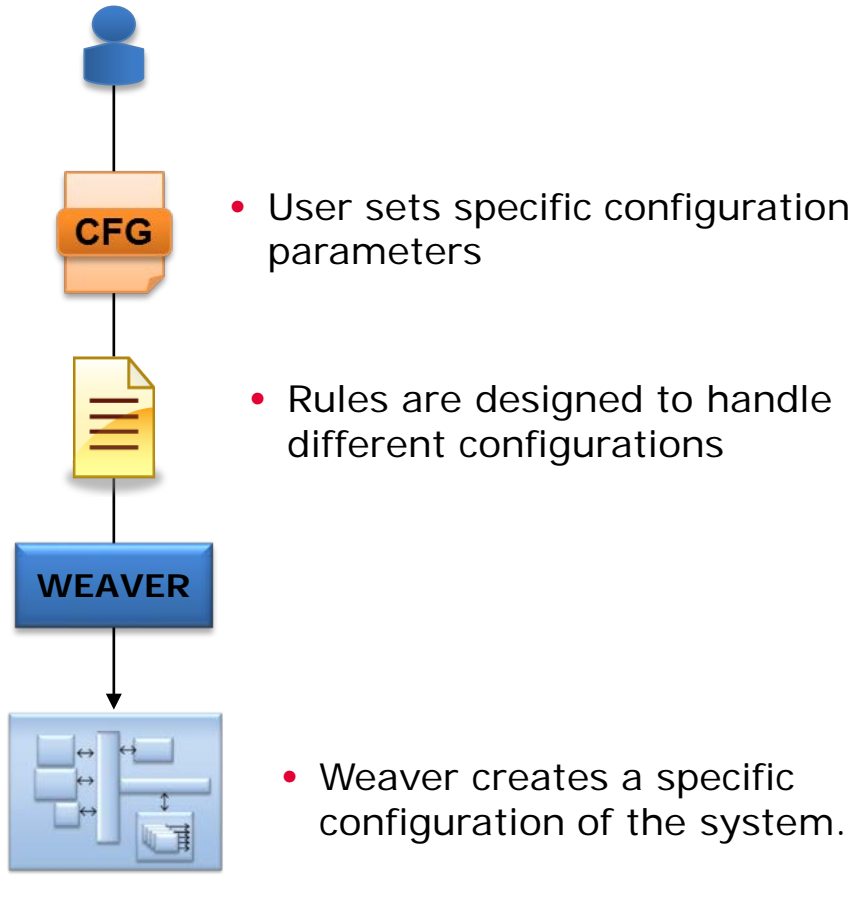
- Some members of the team were not familiar with the target architecture and needed to understand the connectivity by walking through the legacy Verilog code
- Some members were not familiar with Weaver and rules-based approach and needed a quick ramp up
- The rules layout had to be developed throughout this task
- Some packaging had to be done within the integration activity
- Rules optimizations were performed within the integration. This included the creation of macros for commonly repeated tasks.

# Connect\_AMBA function

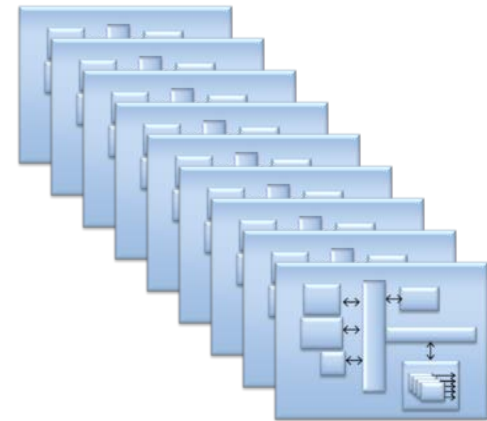
ACE	ACE-Lite	AXI4	AXI3
ACADDR			
ACLK	ACLK	ACLK	ACLK
ACLKEN	ACLKEN	ACLKEN	ACLKEN
ACPROT	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
ACREADY	Tied to 1/unconnected	Tied to 1/unconnected	Tied to 1/unconnected
ACSNNOOP	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
ACVALID	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
ARADDR	ARADDR	ARADDR	ARADDR
ARBAR	ARBAR	Tied to 0/unconnected	Tied to 0/unconnected
ARBURST	ARBURST	ARBURST	ARBURST
ARCACHE	ARCACHE	ARCACHE	ARCACHE
ARDOMAIN	ARDOMAIN	Tied to 0/unconnected	Tied to 0/unconnected
ARESETn	ARESETn	ARESETn	ARESETn
ARID	ARID	ARID	ARID
ARLEN	ARLEN	ARLEN	ARLEN
ARLOCK	ARLOCK	ARLOCK	ARLOCK
ARPROT	ARPROT	ARPROT	ARPROT
ARQOS	ARQOS	Tied to 0/unconnected	Tied to 0/unconnected
ARREADY	ARREADY	ARREADY	ARREADY
ARREGION	ARREGION	ARREGION	Tied to 0/unconnected
ARSIZE	ARSIZE	ARSIZE	ARSIZE
ARSNNOOP	ARSNNOOP	Tied to 0/unconnected	Tied to 0/unconnected
ARUSER	ARUSER	ARUSER	ARUSER
ARVALID	ARVALID	ARVALID	ARVALID
AWADDR	AWADDR	AWADDR	AWADDR
AWBAR	AWBAR	Tied to 0/unconnected	Tied to 0/unconnected
AWBURST	AWBURST	AWBURST	AWBURST
AWCACHE	AWCACHE	AWCACHE	AWCACHE
AWDOMAIN	AWDOMAIN	Tied to 0/unconnected	Tied to 0/unconnected
AWID	AWID	AWID	AWID
AWLEN	AWLEN	AWLEN	AWLEN
AWLOCK	AWLOCK	AWLOCK	AWLOCK
AWPROT	AWPROT	AWPROT	AWPROT
AWQOS	AWQOS	Tied to 0/unconnected	Tied to 0/unconnected
AWREADY	AWREADY	AWREADY	AWREADY
AWREGION	AWREGION	AWREGION	Tied to 0/unconnected
AWSIZE	AWSIZE	AWSIZE	AWSIZE
AWSNNOOP	AWSNNOOP	Tied to 0/unconnected	Tied to 0/unconnected
AWUSER	AWUSER	AWUSER	AWUSER
AWVALID	AWVALID	AWVALID	AWVALID
BID	BID	BID	BID
BREADY	BREADY	BREADY	BREADY
BRESP	BRESP	BRESP	BRESP
BUSER	BUSER	BUSER	BUSER
BVALID	BVALID	BVALID	BVALID
CDDATA	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
CDLAST	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
CDREADY	Tied to 1/unconnected	Tied to 1/unconnected	Tied to 1/unconnected
CDVALID	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
CRREADY	Tied to 1/unconnected	Tied to 1/unconnected	Tied to 1/unconnected
CRRESP	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
CRVALID	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
RACK	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
RDATA	RDATA	RDATA	RDATA
RID	RID	RID	RID
RLAST	RLAST	RLAST	RLAST
RREADY	RREADY	RREADY	RREADY
RRESP	RRESP	RRESP	RRESP
RUSER	RUSER	RUSER	RUSER
RVALID	RVALID	RVALID	RVALID
WACK	Tied to 0/unconnected	Tied to 0/unconnected	Tied to 0/unconnected
WDATA	WDATA	WDATA	WDATA
WID	WID	WID	WID
WLAST	WLAST	WLAST	WLAST
WREADY	WREADY	WREADY	WREADY
WSTRB	WSTRB	WSTRB	WSTRB
WUSER	WUSER	WUSER	WUSER
WVALID	WVALID	WVALID	WVALID

- Raising the level of user abstraction for AMBA connectivity
- ACE can connect to ACELite, AXI4, AXI3
- Different tieoffs needed

# System Configurability



Any one of thousands of possible configurations can be generated in less than 7 minutes





# 30x Schedule Improvement??

- Build 1 of 1000s of systems in 7 minutes
- Interface standardization – Interrupt controller upgrade in 15 minutes



# Benefits

- Schedule:
  - It is estimated that once the methodology is deployed, it is reasonable to expect a 10x-15x improvement in schedule for new projects and 20x-30x for derivative projects.
- Quality
  - This netlist was right-first-time as it passed a previous regression test. IP standardization with formal rules eliminates a lot of tedious errors.
- Productivity
  - Massive improvements in productivity (10x) is expected with this methodology



# Future Considerations

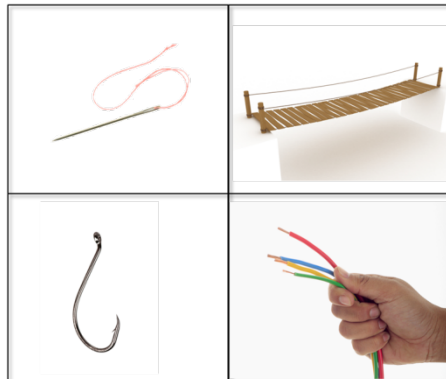
- Interface Standardization
- Chip-in-a-day
- Verification Enablement



# Conclusion

- A rules-base methodology with powerful integration instructions and selection mechanism can reduce the overall SoC integration task by a factor of 15x-30x
- IP Standardization is a key enabler for this flow and IP-XACT provides an interoperable method of formalizing IP interfaces
- The rules themselves are specified using a very small instruction set (DSL) that can be instantly used by anyone familiar with the integration domain
- Highly configurable systems can be rendered very quickly with this methodology

# Conclusion - 4 Pictures – 1 Word



C O N N E C T



W E A V E

- *To Make ..*
- *To Interlace ..*
- *To construct ..*





**Questions??**



# Thank You

- Please visit us in Booth 1002

