OpenSoC Fabric

An On-Chip Network Generator

Farzad Fatollahi-Fard, Dave Donofrio,
George Michelogiannakis, John Shalf

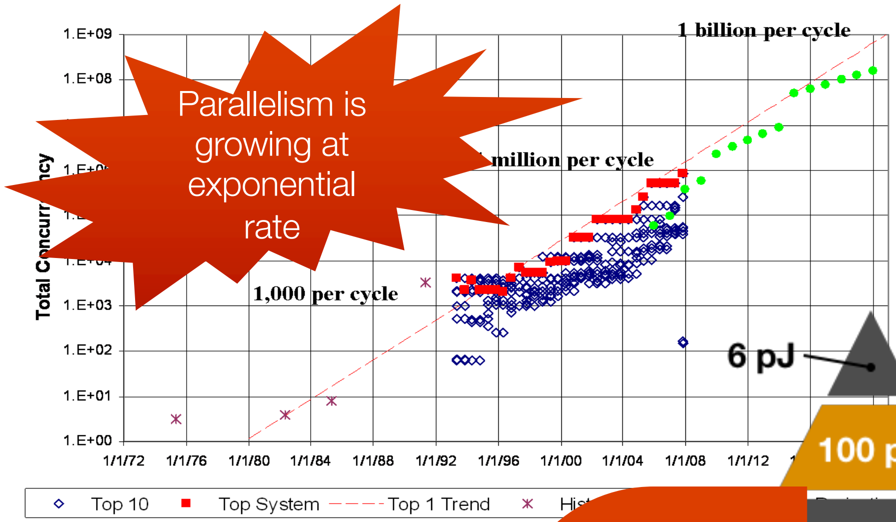
CoDEx

2016 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS-2016)
April17-19, 2016. Uppsala, Sweden.

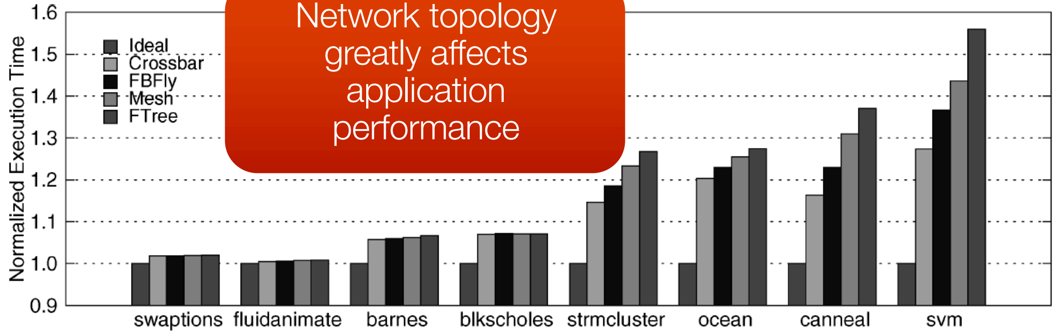
Motivation

Why Are We Doing This?

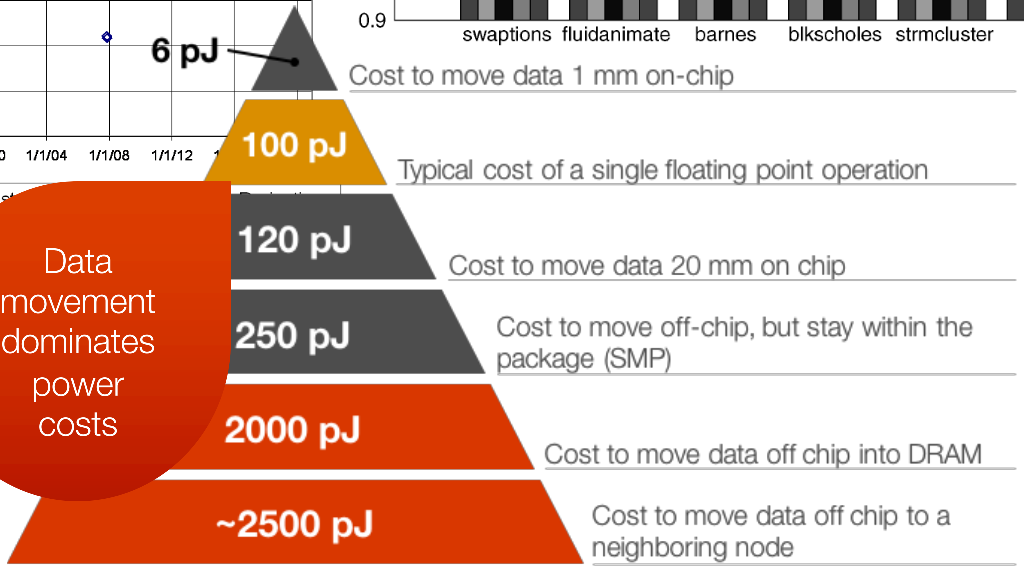
- **Want to build and model candidate future HPC chip multiprocessors**



Parallelism is growing at exponential rate



Network topology greatly affects application performance


6 pJ
Cost to move data 1 mm on-chip
100 pJ
Typical cost of a single floating point operation
120 pJ
Cost to move data 20 mm on chip
250 pJ
Cost to move off-chip, but stay within the package (SMP)
2000 pJ
Cost to move data off chip into DRAM
~2500 pJ
Cost to move data off chip to a neighboring node

Data movement dominates power costs

An analysis of on-chip interconnection networks for large-scale chip multiprocessors
ACM Transactions on computer architecture and code optimization (TACO), April 2010

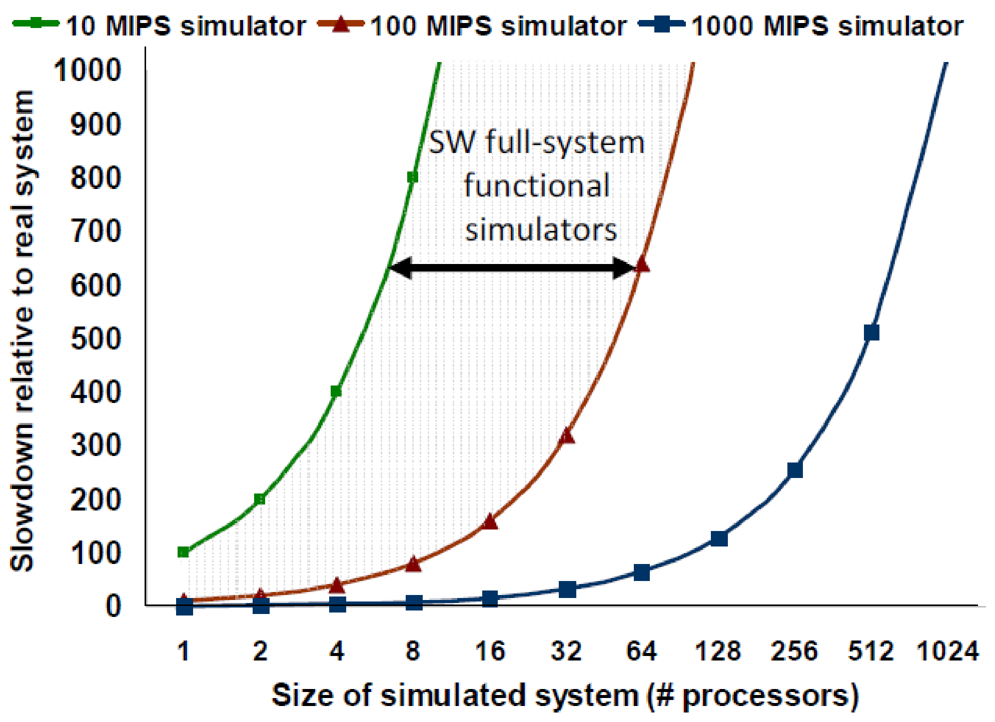
What Interconnect Provides the Performance? Is it Open Source?

What tools exist to answer these questions?

What tools exist for SoC research

What tools do we have to evaluate large, complex networks of cores?

- **Software models**
  - Fast to create, but plagued by long runtimes as system size increases
- **Hardware emulation**
  - Fast, accurate evaluate that scales with system size but suffers from long development time



A complexity-effective architecture for accelerating full-system multiprocessor simulations using FPGAs. FPGA 2008

Comparison of NoCs

Software Tools

	Language	Accuracy	Verification	Drawbacks
Booksim	C++	Cycle-Accurate	RTL	Long runtimes limit simulation size
Garnet	C++ (GEM5)	Event-Driven	Other Simulators	Not fast enough for larger simulations (1K+ cores)
NoCTweak	SystemC	Cycle-Accurate	RTL	Long runtimes limit simulation size
PhoenixSim	OMNeT++	Event-Driven	Other Simulators	For Photonics on-chip networks
Topaz	C++ (GEM5)	Cycle-Accurate	Other Simulators	Not fast enough for larger simulations (1K+ cores)

Comparison of NoCs

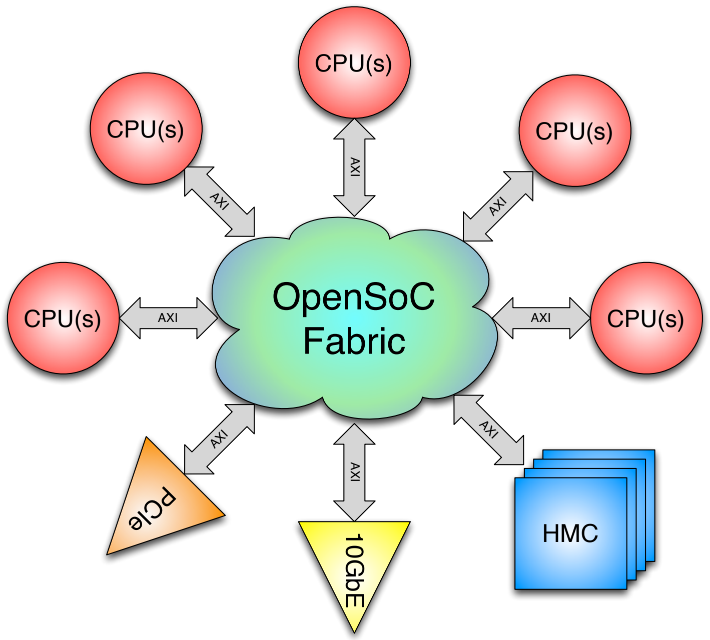
Hardware Tools

	Language	Features	Open Source?	Drawbacks
Stanford NoC Router	Verilog	Long list of Verilog parameters	Yes	-Hard to configure
CONNECT	Bluspec SystemVerilog	Completely customizable via website	Yes (noncommercial)	-Designed for FPGAs
ARM CoreLink	Pre-generated IP	Up to clusters of 48 cores	No	-Designed for ARM cores (not design space exploration) -For "small" designs -Cache Coherent
Arteris FlexNoC	Pre-generated IP	Tool optimized for VLSI design	No	-Full parameters unknown

OpenSoC Fabric

An Open-Source, Flexible, Parameterized, NoC Generator

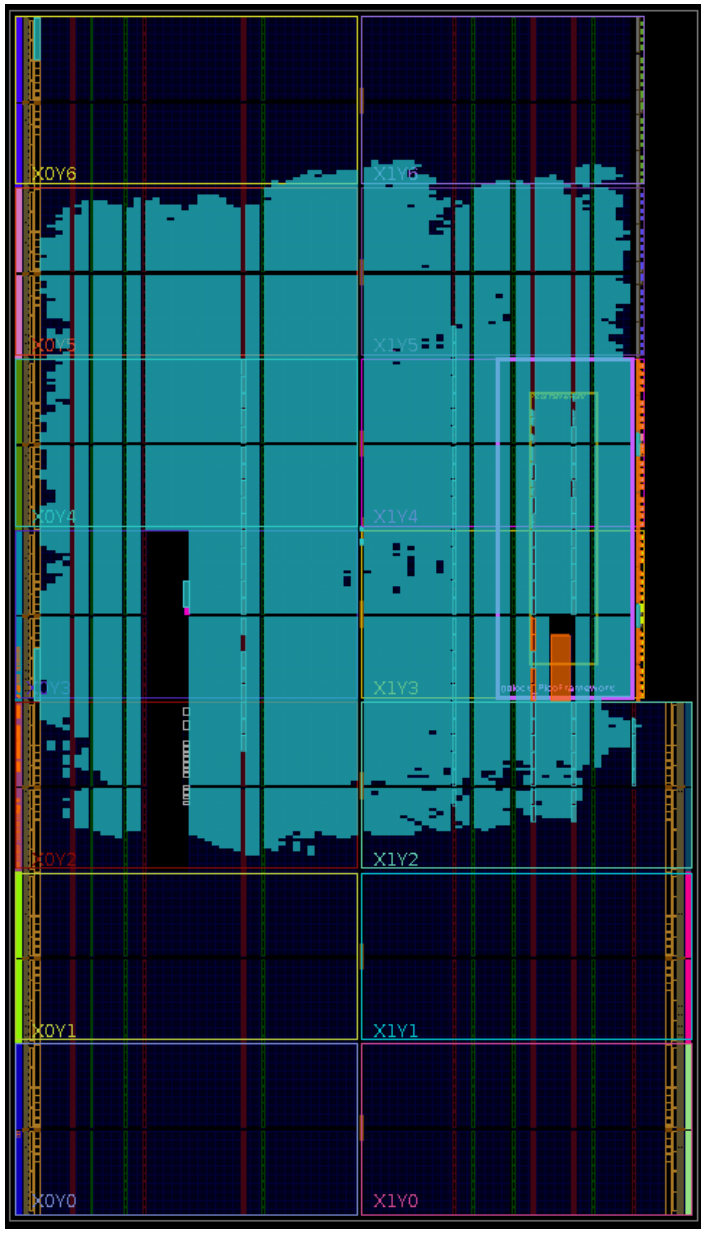
- **Part of the CoDEx tool suite**
- **Written in Chisel**
- **Dimensions, topology, VCs all configurable**
- **Fast functional C++ model for functional validation**
- **Verilog based description for FPGA or ASIC**
  - Synthesis path enables accurate power / energy modeling



Current Status

Version 1.1.2 Released

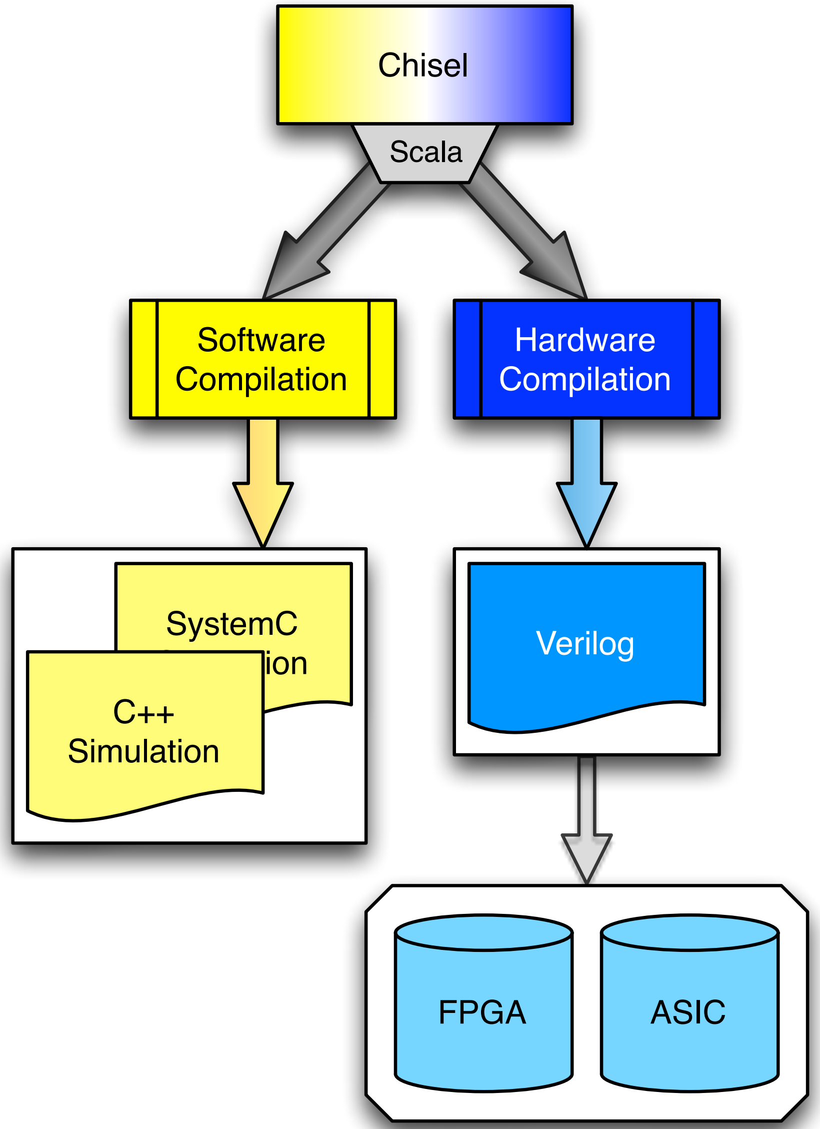
- **Multiple Topologies**
  - Mesh
  - Flattened Butterfly
- **Wormhole Flow Control**
- **Virtual Channels**
- **Run both through ASIC and FPGA tools**
- **Available for download**
  - www.opensocfabric.org



Chisel: A New Hardware DSL

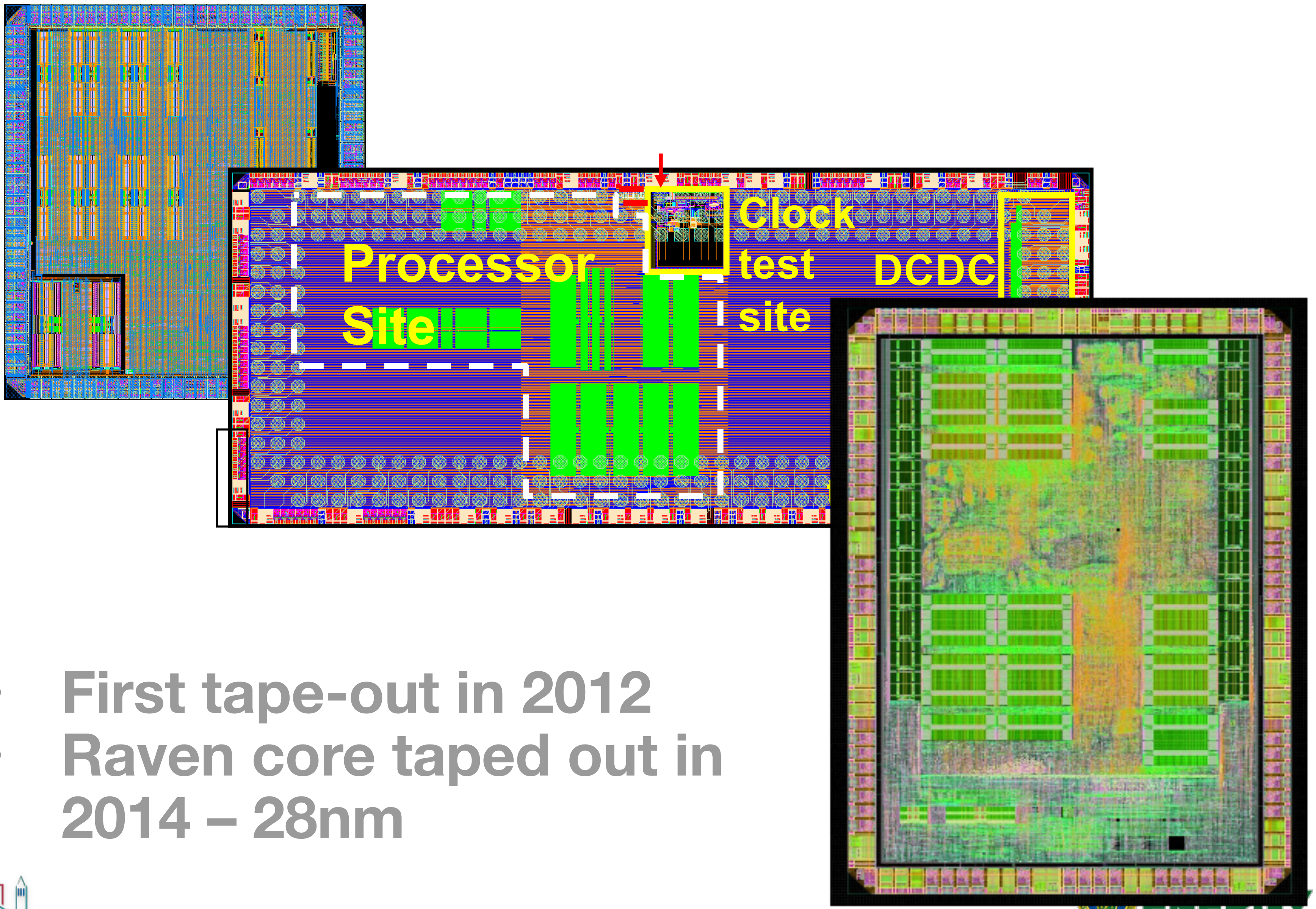
Using Scala to construct Verilog and C++ descriptions

- **Chisel provides both software and hardware models from the same codebase**
- **Object-oriented hardware development**
  - Allows definition of structs and other high-level constructs
- **Powerful libraries and components ready to use**
- **Working processors fabricated using chisel**



Recent Chisel Designs

Chisel code successfully boots Linux



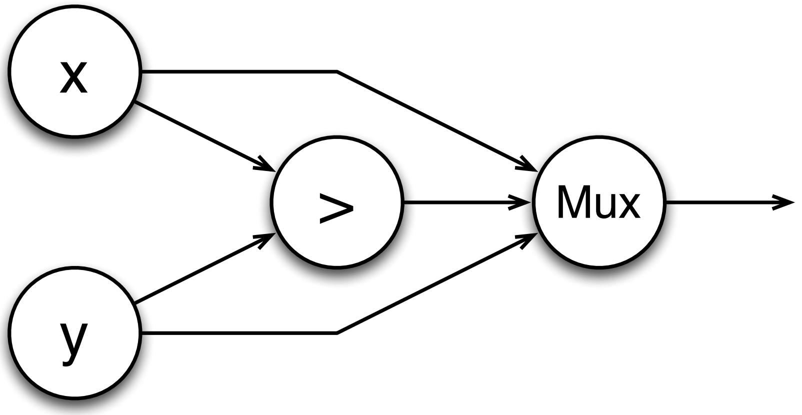
- First tape-out in 2012
- Raven core taped out in 2014 – 28nm

Chisel Overview

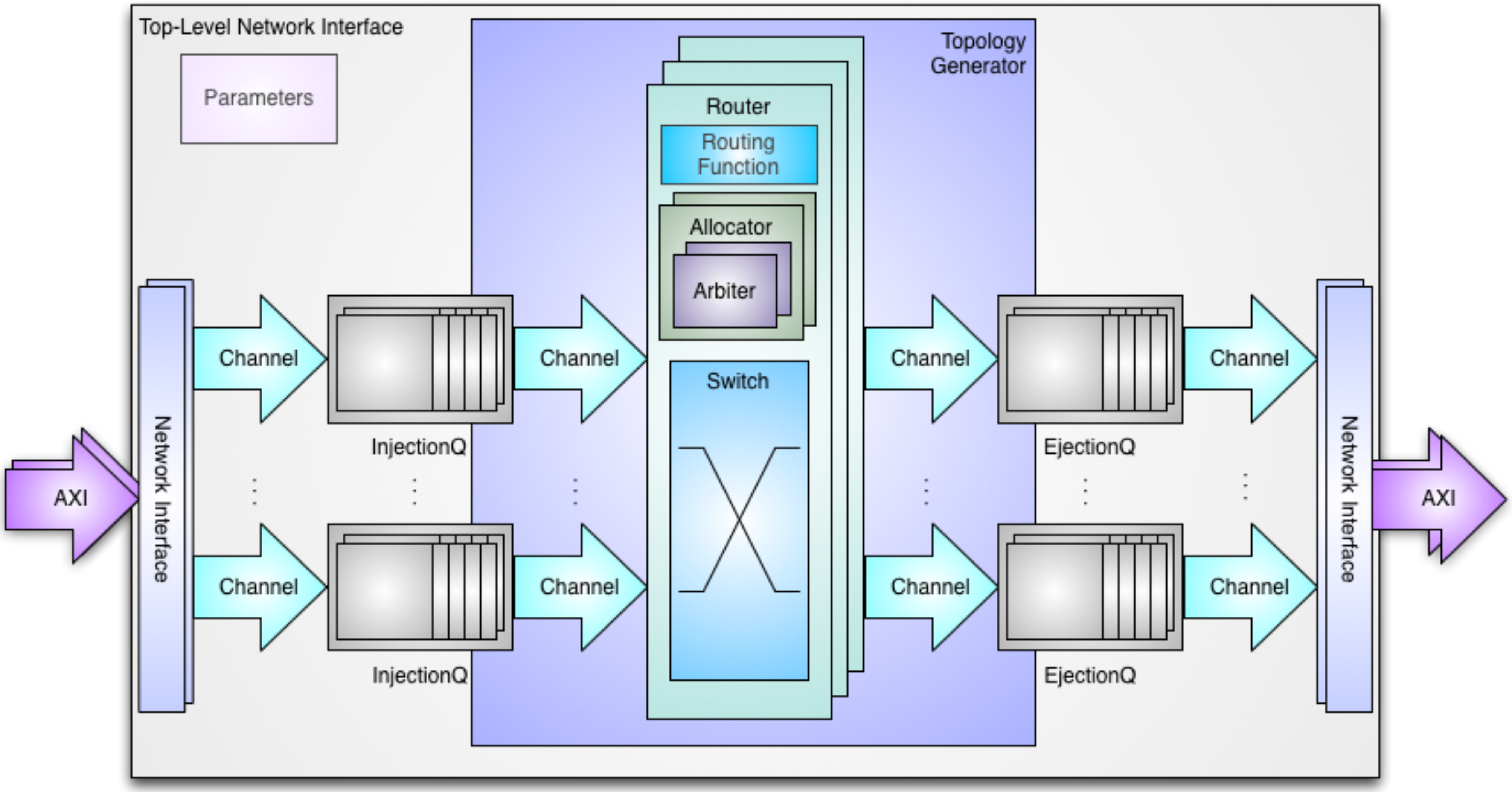
How does Chisel work?

- **Not “Scala to Gates”**
- **Describe hardware functionality**
- **Chisel creates graph representation**
  - Flattened
- **Each node translated to Verilog or C++**

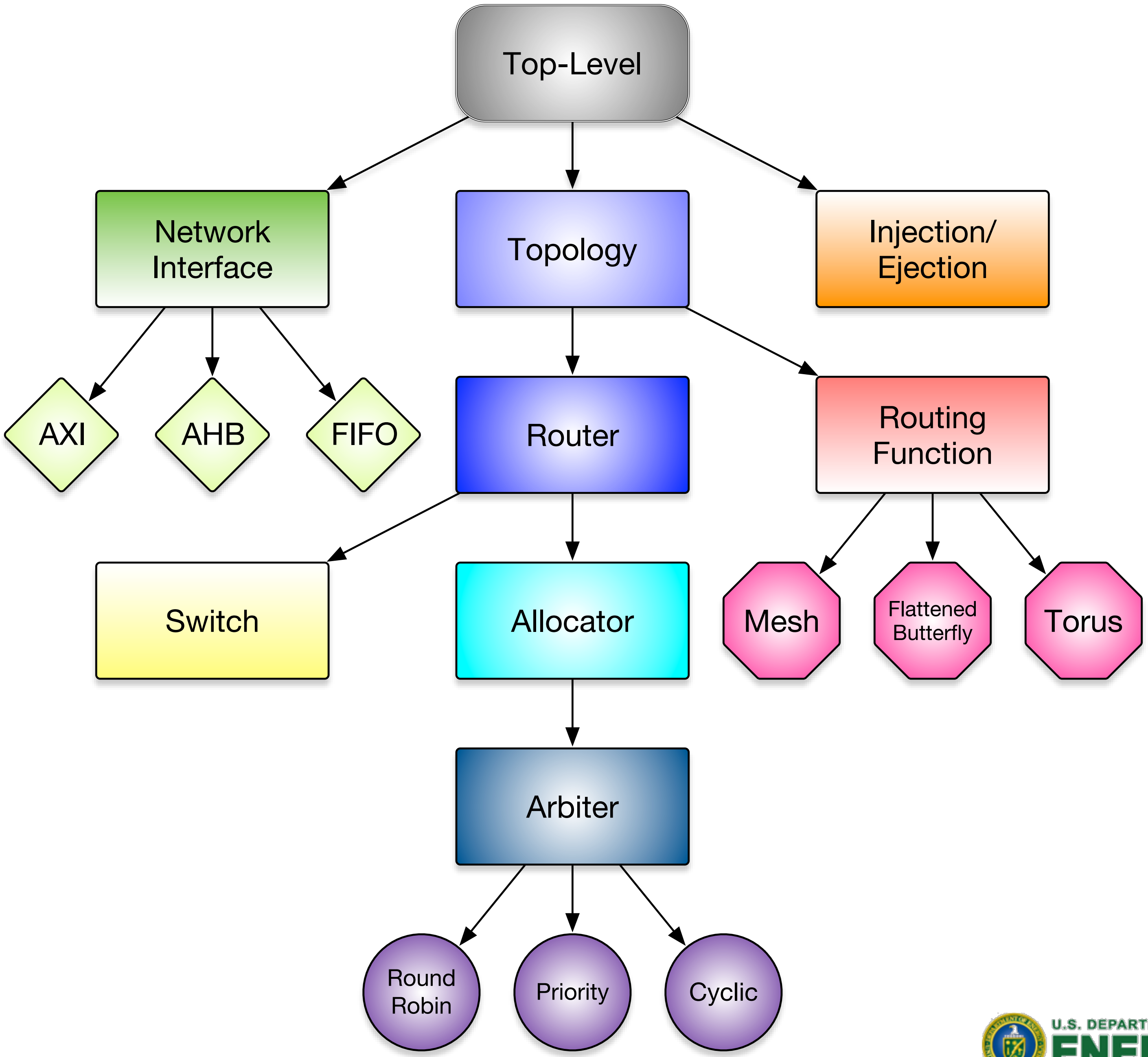
```
Mux(x > y, x, y)
```



OpenSoC – Top Level Diagram



OpenSoC – Functional Hierarchy



Configuring

Parameters

- **OpenSoC configured at run time through *Parameters* class**
  - Declared at top level, sub modules can add / change parameters tree
- **Not limited to just numerical values**
  - Leverage Scala to pass functions to parameterize module creation
    - Example: Routing Function constructor passed as parameter to router

Configuring

Parameters

- **All OpenSoC Modules take a Parameters class as a constructor argument**
- **Setting parameters:**
  - parms.child("MySwitch", Map(("numInPorts"->Soft(8)),
    ("numOutPorts"->Soft(3)))
- **Getting a parameter:**
  - val numInPorts = parms.get[Int]("numInPorts")

Developing

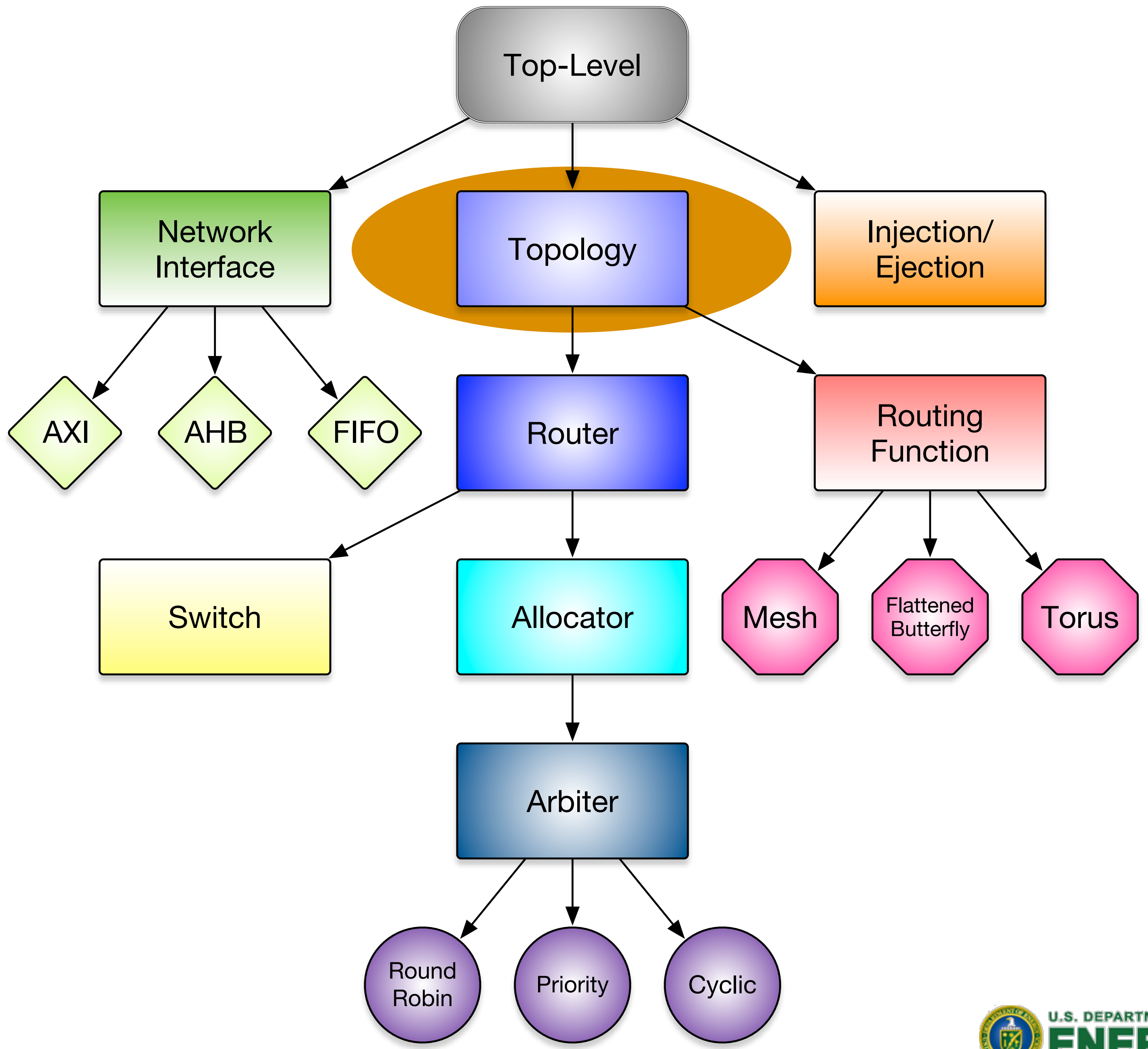
Incredibly Fast Development Time

- **Modules have a standard interface that you inherit**
- **Development of modules is very quick**
  - Flattened Butterfly took 2 hours of development

```
abstract class VCRouter(parms: Parameters)
    extends Module(parms) {
  val numInChannels = parms.get[Int]
    ("numInChannels")
  val numOutChannels = parms.get[Int]
    ("numOutChannels")
  val nunVCs = parms.get[Int]("numVCs")
  val io = new Bundle {
    val inChannels = Vec.fill(numInChannels)
      { new ChannelVC(parms) }
    val outChannels = Vec.fill(numOutChannels)
      { new ChannelVC(parms).flip() }
  }
}

class SimpleVCRouter(parms: Parameters)
    extends VCRouter(parms) {
  // Implementation
}
```

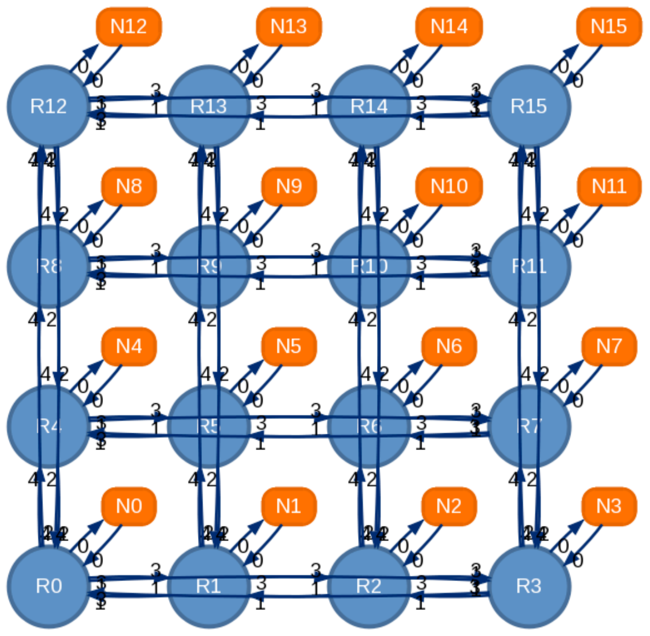
OpenSoC – Functional Hierarchy



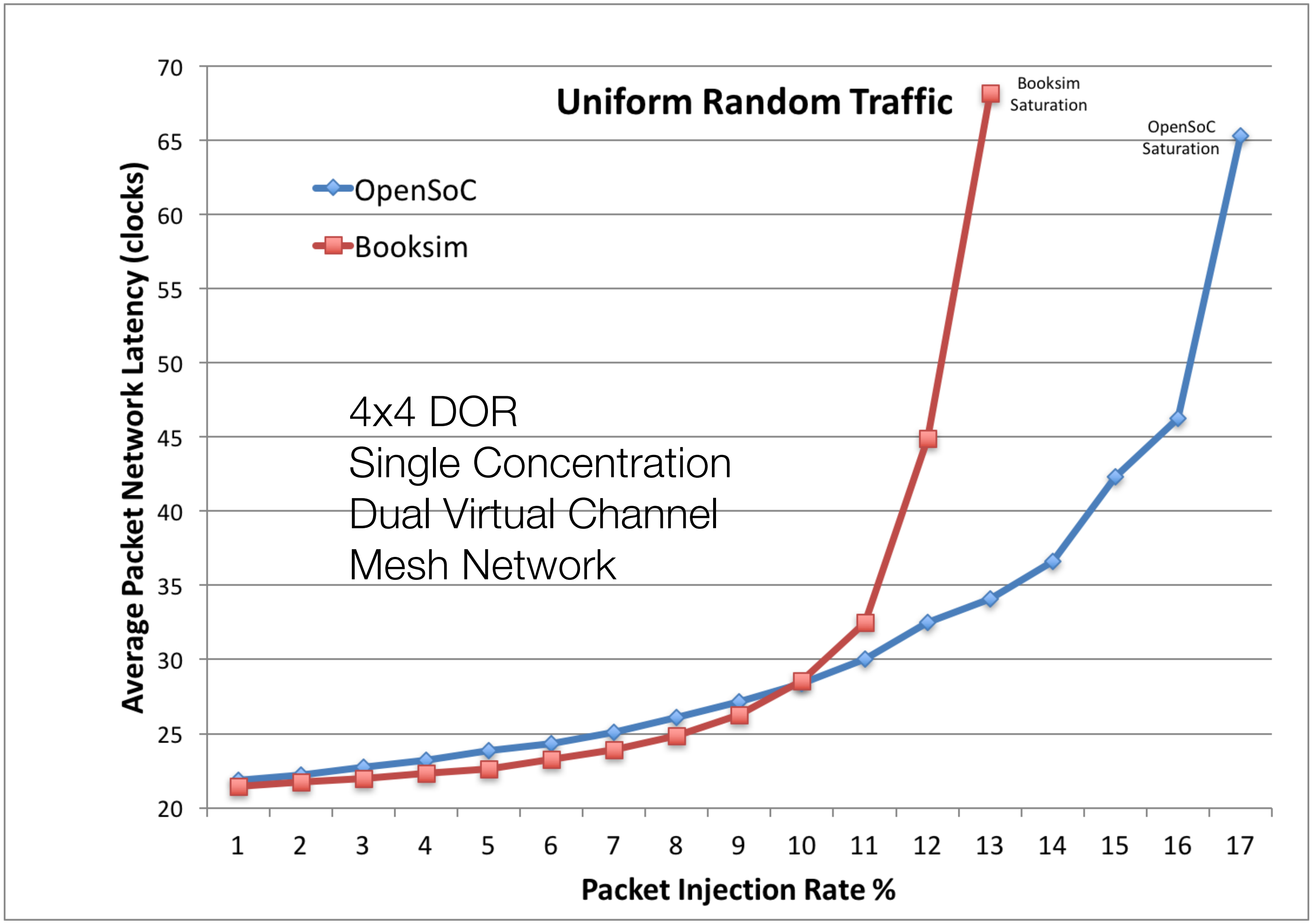
OpenSoC – Top Level Modules

Topology

- Stiches routers together
- Assigns routers individual ID
- Assigns Routing Function to routers
- Passes down Arbitration scheme
- Connections Injection and Ejection Queues for network endpoints



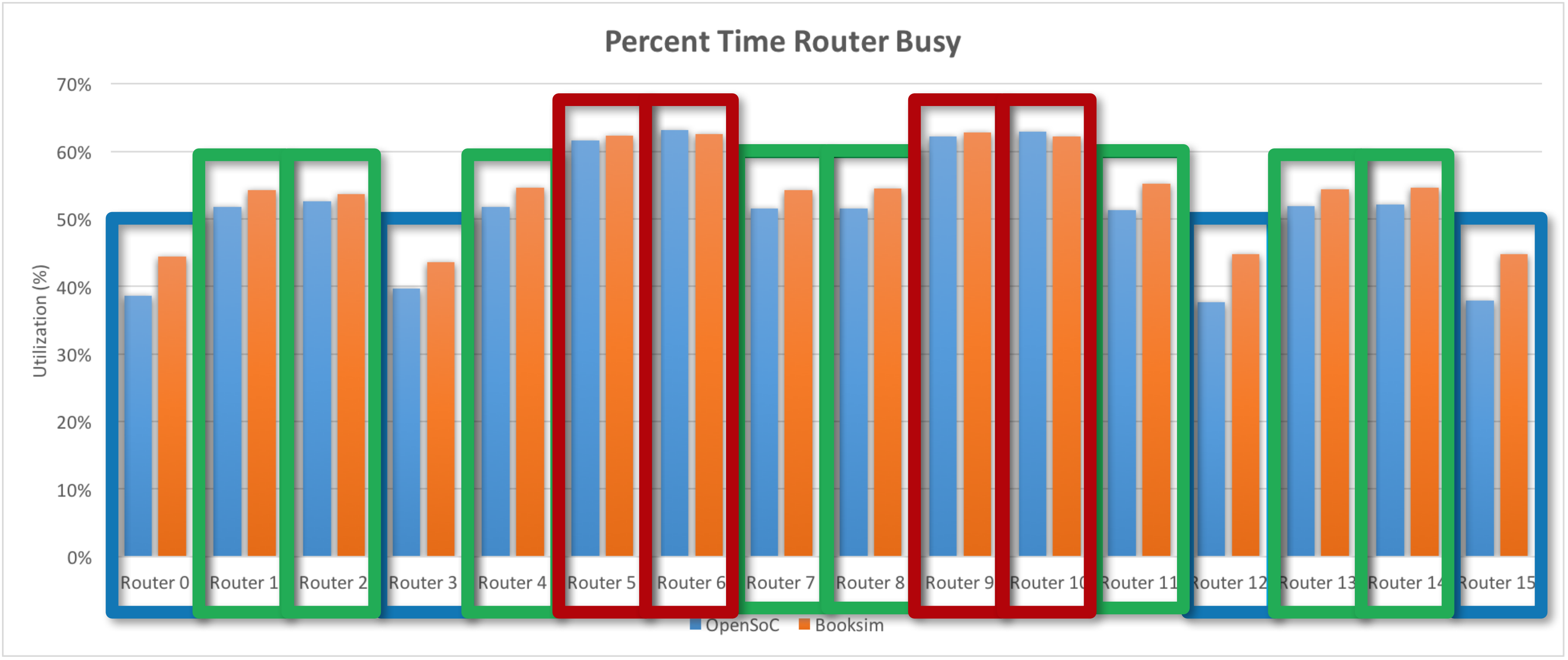
Results – Traffic Patterns



Results – Average Latency

Compared to Booksim	**OpenSoC Fabric (Software)**	**OpenSoC Fabric (Hardware)**
Uniform	+1.86%	+8.37%
Tornado	+0.84%	+0.42%
Transpose	+7.37%	+8.29%
Neighbor	+0.84%	+6.28%
Bit Reverse	+1.85%	+10.6%

Results – Latency and Utilization



Nearest Neighbor Traffic Pattern

Results – Application Traces

Compared to Booksim	**OpenSoC**
AMR Avg latency	-2.42%
MiniDFT Avg latency	-28.3%
AMG Avg latency	+16.3%
AMR Execution time	-2.19%
MiniDFT Execution time	-5.25%
AMG Execution time	+130.8%

Future additions

Towards a full set of features

- Upgrade OpenSoC Fabric to use Chisel 3
- A collection of topologies and routing functions
- Standardized interfaces at the endpoints
- Power modeling in the C++ model

Conclusion

- **This is an open-source community-driven infrastructure**
  - We are counting on your contributions

Acknowledgements

- UCB Chisel
- US Dept of Energy
- Laboratory for Physical Sciences
- Ke Wen
- Columbia LRL
- John Bachan

More Information

http://opensocfabric.org