


---

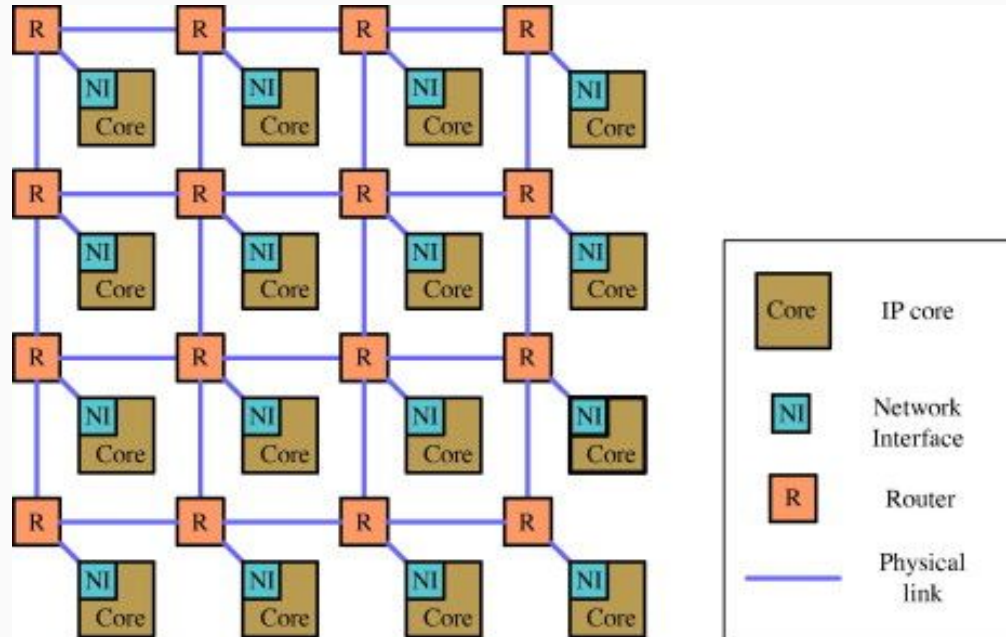
# PANE: Pluggable Asynchronous Network-on-Chip Simulator

---

Sarabjeet Singh • 24.04.2017  
Under guidance of,  
Prof. Joycee Mekie



# Network-on-Chip (NoC)



- System level NoC parameters such as packet latencies, throughput.
- Existing Synchronous NoC tools such as BookSim2<sup>[1]</sup>, Noxim.
- Heterogeneous Multi Processors (HMP), such as Nvidia Xavier, Parker and Tegra XI, Samsung Exynos 8895, where communication across clock domains is required. -> unidentical routers and links!

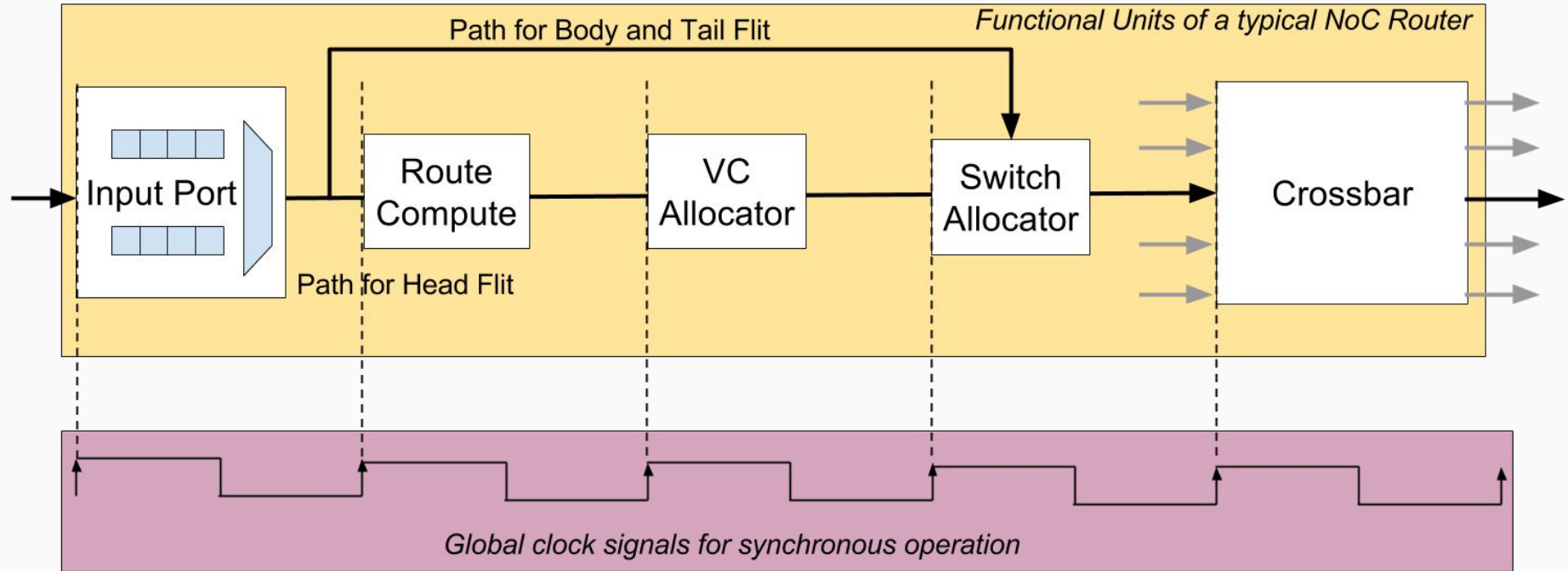
# Therefore, PANE!

Pluggable Asynchronous Network-on-Chip  
Simulator

an event-driven asynchronous NoC simulator

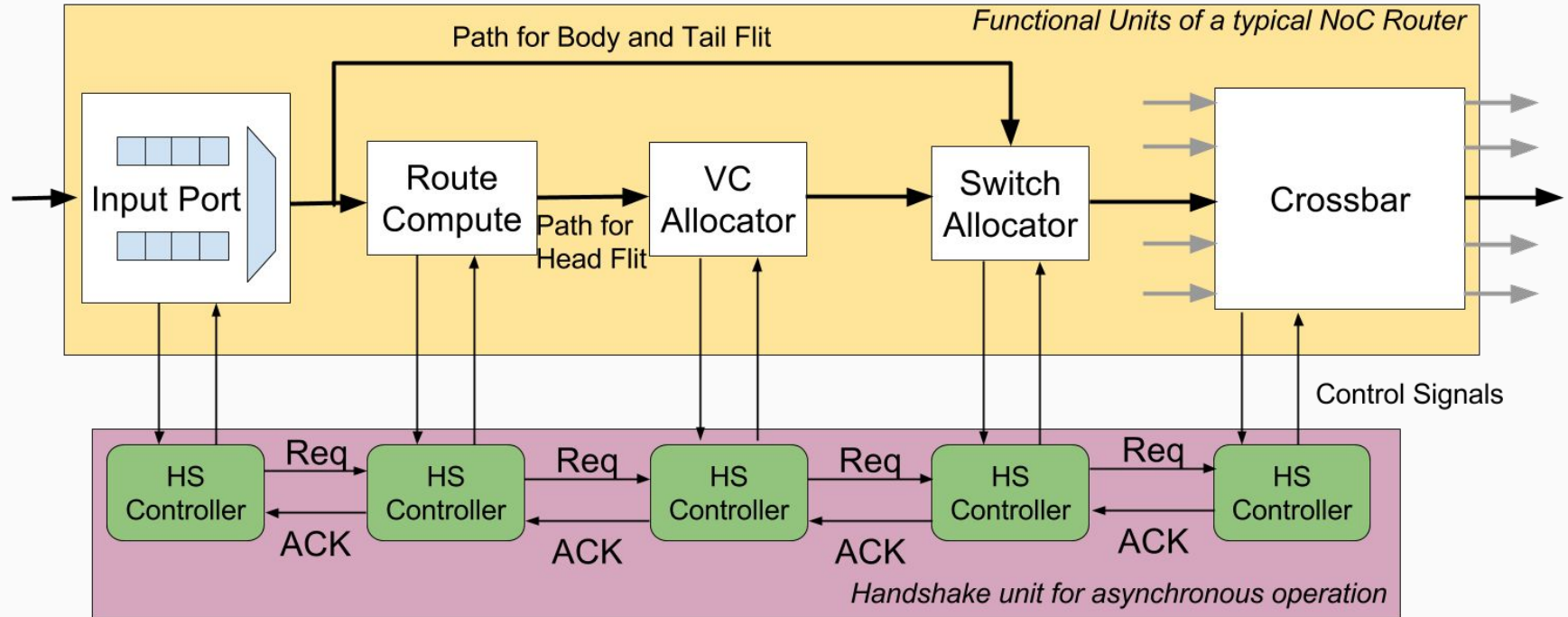
- UNIX compatible tool
- Model a variety of NoC designs:
  - ◆ Different topologies
  - ◆ Routing algorithms, allocation and arbitration mechanisms
  - ◆ Flit widths and buffering schemes
  - ◆ Real benchmark applications and synthetic traffic patterns.
- Complete system level analysis of synchronous, asynchronous and heterogeneous NoC architectures.

# Router Design: Synchronous Circuits



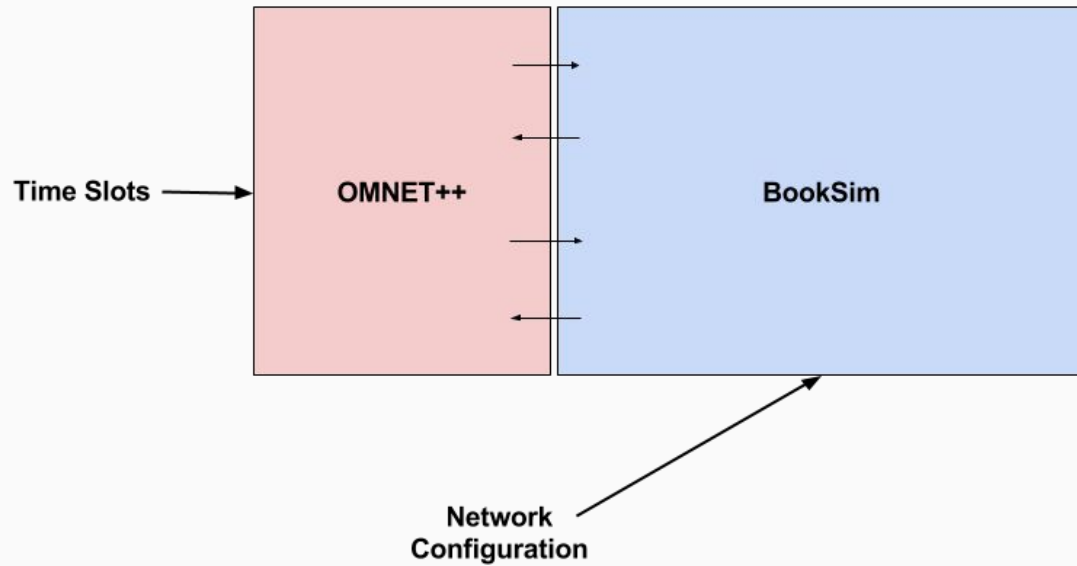
```
if( CLOCK == TRUE)
    SendNext();
```

# Router Design: Asynchronous Circuits



```
if( ( TimeSlot == TRUE ) && ( Ack == TRUE ) )  
    SendNext();
```

# PANE



# BookSim2<sup>[1]</sup>

A Cycle-Accurate Interconnection  
Network Simulator

- C++ based synchronous NoC Simulator
- BookSim2<sup>[1]</sup> gives a detailed description of the routers and the interconnect behavior for the NoC designers to test and validate their NoCs.
- However, BookSim2<sup>[1]</sup> is designed to simulate only homogeneous NoCs where all the routers operate at the same clock frequency and does not support asynchronous and heterogeneous NoC simulation.

Initialization

Start BookSim

Flit  
Generation

Core + NI  
Generate new  
FLITS

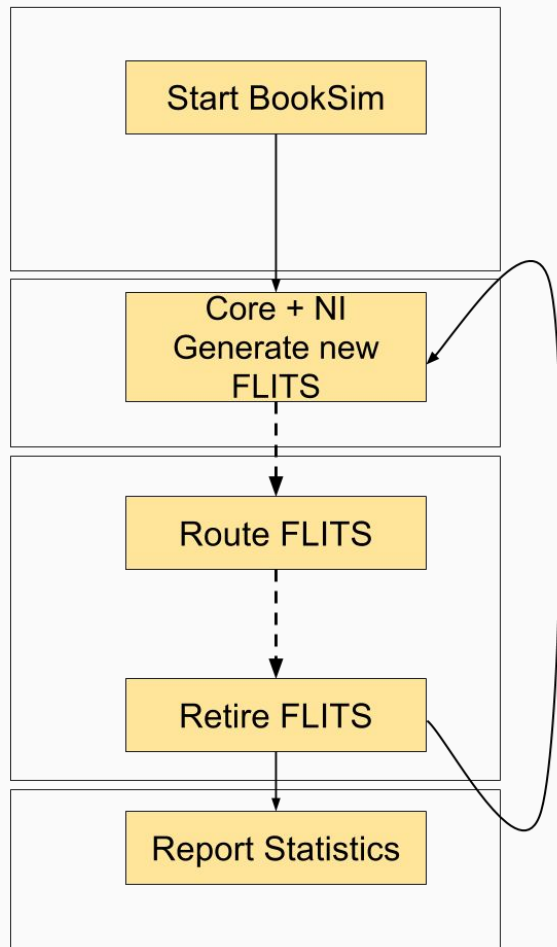
Flit Movement

Route FLITS

Retire FLITS

Finish

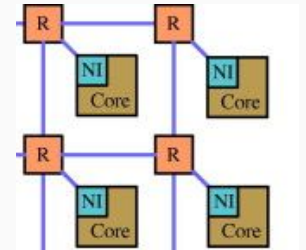
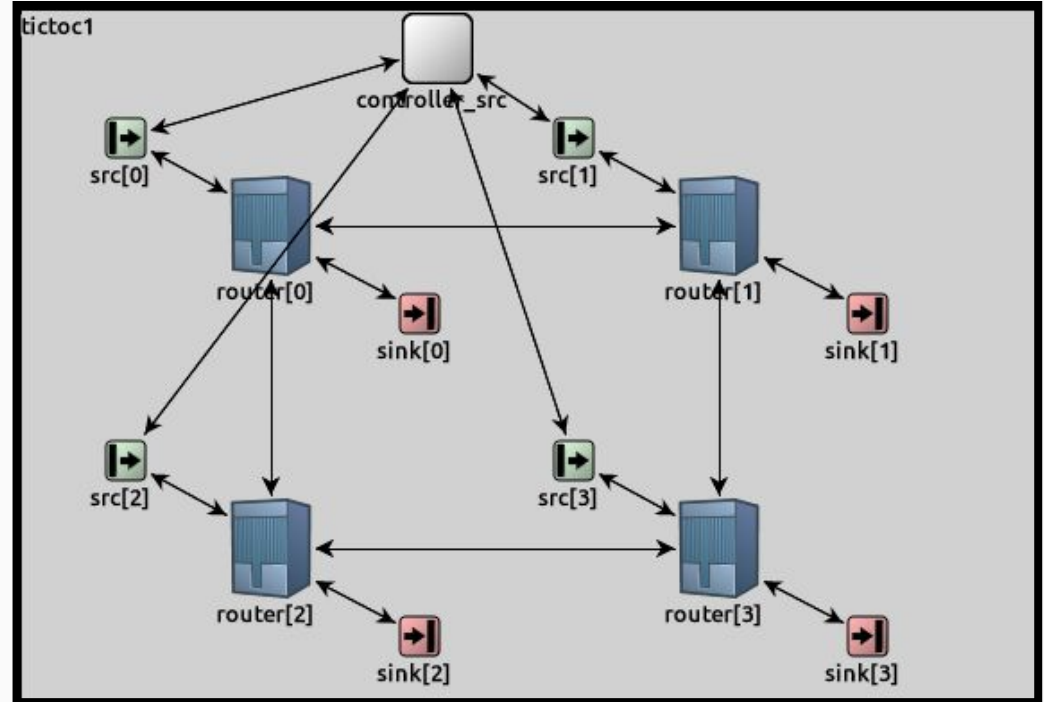
Report Statistics



# OMNeT++<sup>[2]</sup>

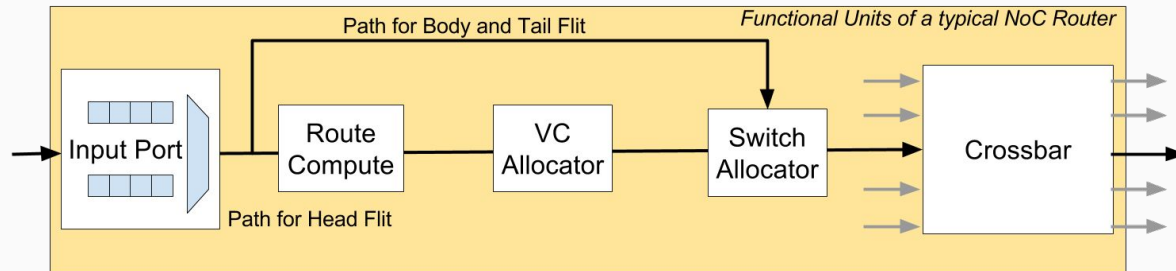
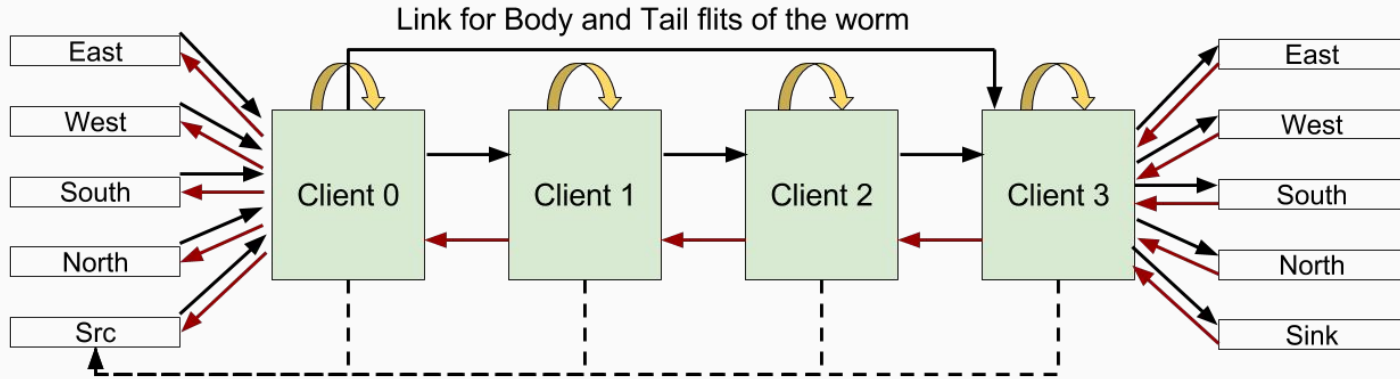
## Discrete Event Simulator

- C++ framework to design and simulate network designs.
- Supports event-driven simulation!
- Network topologies can be easily described in OMNeT++
- Additionally it provides network animation, which helps in network visualization, debugging and traceability.
- OMNeT++ is available as open-source, free for non-profit use and is used by a fairly large community of users for research and educational purposes.





# Block Diagram of OMNeT++ Router

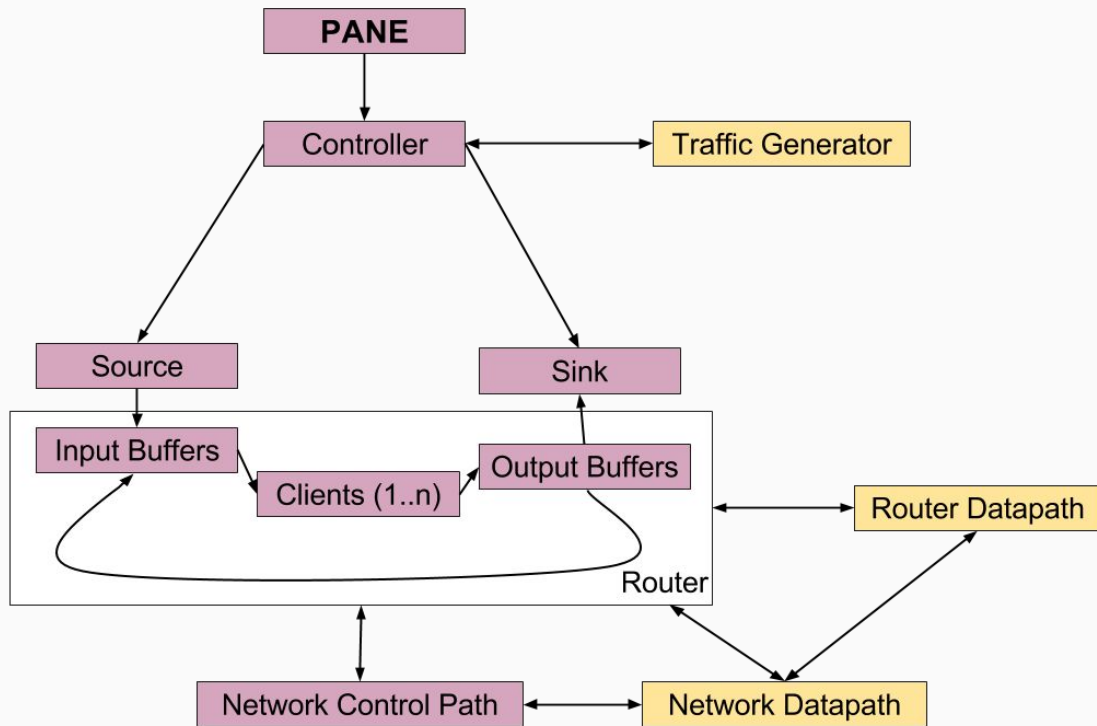


# MY WORK

Building an OMNeT++<sup>[2]</sup> plug-in to extend BookSim2<sup>[1]</sup> for heterogeneous NoCs

- Building an OMNeT++ model of an NoC
- Establishing communication between BookSim and OMNeT++<sup>[2]</sup>

# Connection graph of modules in PANE

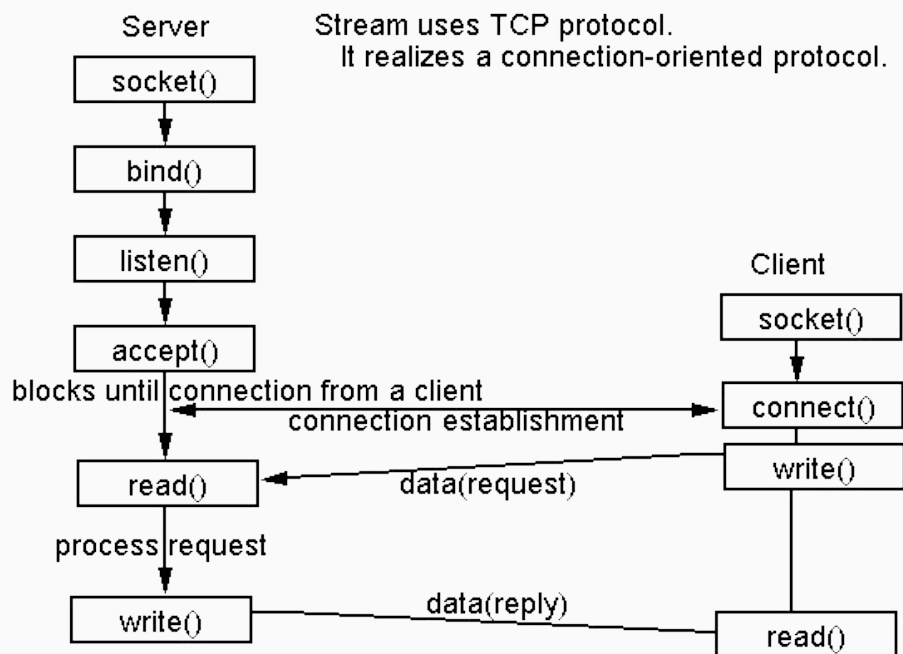


# IPC - sockets<sup>[3]</sup>

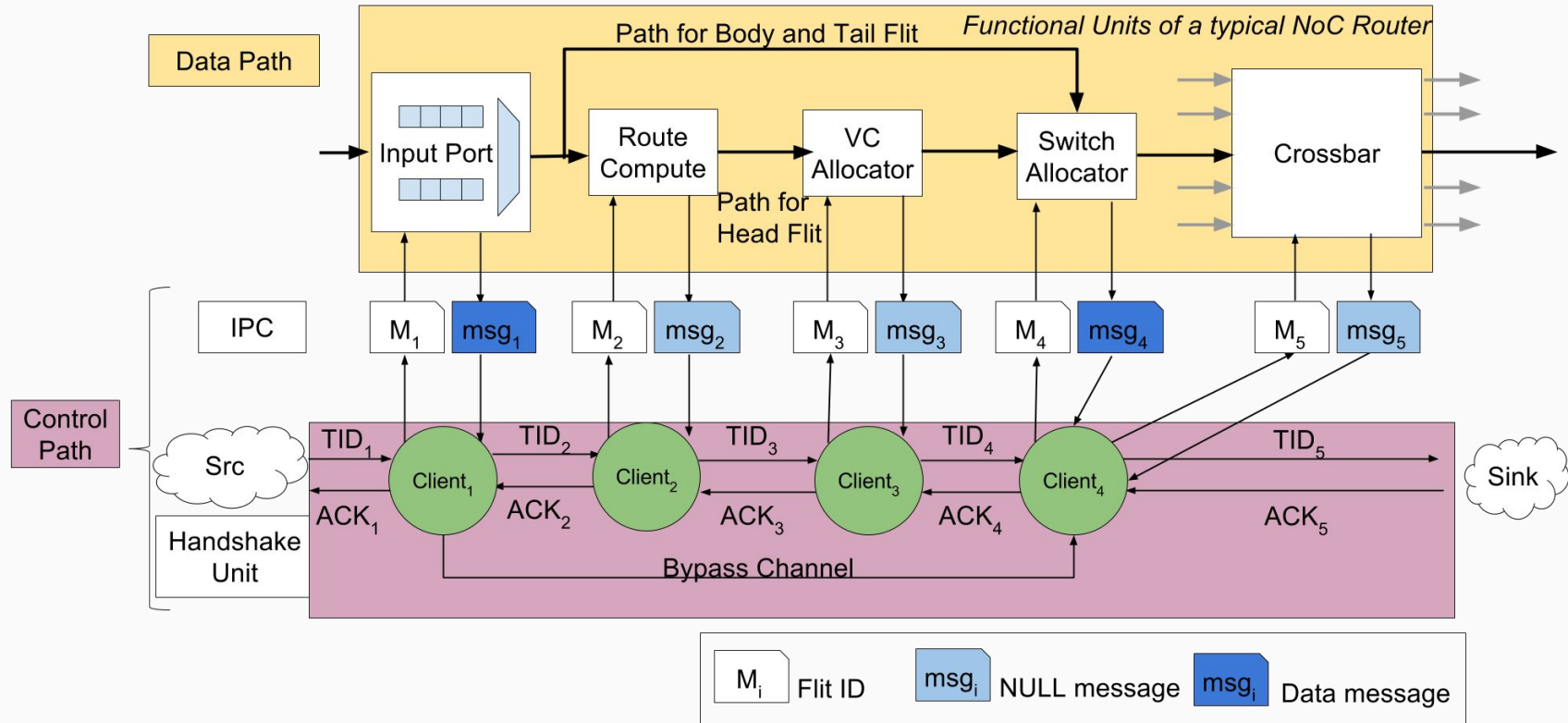
Sockets provide point-to-point, two-way communication between two processes.

UNIX domain sockets allow IPC among two different processes running on the same machine.

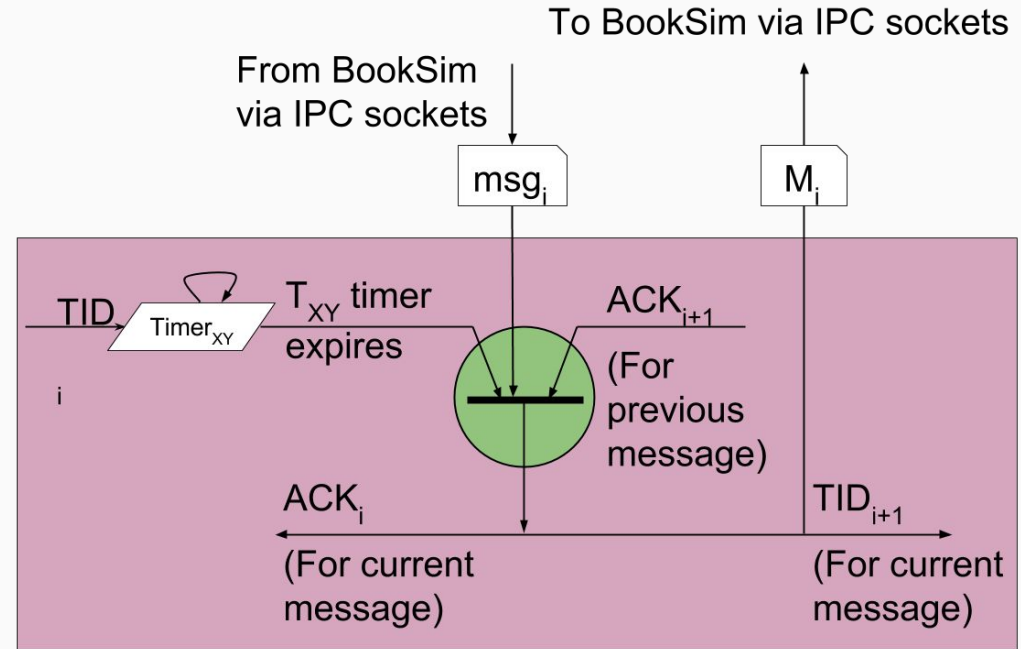
No ancestor process required



# Schematic representation of asynchronous router operation in PANE



# Schematic of a PANE Client



## Control Path Progress Window

```

Speed:
Mess:
** Event #5344768 t=40054 Elapsed: 44.028s (0m 44s)
Speed: ev/sec=109513 simsec/sec=777.111 ev/simsec=140.923
Messages: created: 4205820 present: 32133 in FES: 85
** Event #5563136 t=41730 Elapsed: 46.031s (0m 46s)
Speed: ev/sec=109075 simsec/sec=837.163 ev/simsec=130.291
Messages: created: 4376606 present: 33483 in FES: 89
** Event #5818112 t=43615 Elapsed: 48.032s (0m 48s)
Speed: ev/sec=127488 simsec/sec=942.5 ev/simsec=135.266
Messages: created: 4576514 present: 35037 in FES: 107
** Event #6029056 t=45232 Elapsed: 50.033s (0m 50s)
Speed: ev/sec=105419 simsec/sec=808.096 ev/simsec=130.454
Messages: created: 4741296 present: 36328 in FES: 101
** Event #6239232 t=46809 Elapsed: 52.034s (0m 52s)
Speed: ev/sec=105088 simsec/sec=788.5 ev/simsec=133.276
Messages: created: 4905206 present: 37609 in FES: 86
** Event #6456064 t=48336 Elapsed: 54.034s (0m 54s)
Speed: ev/sec=108416 simsec/sec=763.5 ev/simsec=141.999
Messages: created: 5077469 present: 38897 in FES: 114
** Event #6673920 t=49978 Elapsed: 56.035s (0m 56s)
Speed: ev/sec=108928 simsec/sec=821 ev/simsec=132.677
Messages: created: 5249666 present: 40181 in FES: 111
** Event #6892544 t=51616 Elapsed: 58.036s (0m 58s)
Speed: ev/sec=109257 simsec/sec=818.591 ev/simsec=133.47
Messages: created: 5421212 present: 41513 in FES: 115
** Event #7152896 t=53479 Elapsed: 60.037s (1m 00s)
Speed: ev/sec=130176 simsec/sec=931.5 ev/simsec=139.749
Messages: created: 5626531 present: 43049 in FES: 96
** Event #7361792 t=55083 Elapsed: 62.040s (1m 02s)
Speed: ev/sec=104344 simsec/sec=801.199 ev/simsec=130.234
Messages: created: 5791498 present: 44258 in FES: 91
** Event #7566592 t=56641 Elapsed: 64.042s (1m 04s)
Speed: ev/sec=102298 simsec/sec=778.222 ev/simsec=131.451
Messages: created: 5951197 present: 45496 in FES: 92
** Event #7779840 t=58286 Elapsed: 66.043s (1m 06s)
Speed: ev/sec=106571 simsec/sec=822.089 ev/simsec=129.634
Messages: created: 6118099 present: 46813 in FES: 83
** Event #8066816 t=60367 Elapsed: 68.044s (1m 08s)
Speed: ev/sec=143488 simsec/sec=1040.5 ev/simsec=137.903
Messages: created: 6355568 present: 48373 in FES: 335
Simulation End!
End Time = 61001s
FINISH:
Number of packets generated = 48037
FINISH:
Number of packets returned = 48037

```

## Data Path Progress Window

```

Measured:
Class 0:
Remaining flits: (0 flits)
Measured flits: (0 flits)
Class 0:
Remaining flits: (0 flits)
Measured flits: (0 flits)
Class 0:
Remaining flits: (0 flits)
Measured flits: (0 flits)
Class 0:
Remaining flits: (0 flits)
Measured flits: (0 flits)
Time taken is 61000 cycles
===== Traffic class 0 =====

Total number of flits generated = 48037, changed lanes = 47457
Overall minimum packet latency = 29 (1 samples)
Overall average packet latency = 46.4438 (1 samples)
Overall maximum packet latency = 144 (1 samples)
Overall minimum network latency = 30 (1 samples)
Overall average network latency = 47.4438 (1 samples)
Overall maximum network latency = 145 (1 samples)
Overall minimum flit latency = 30 (1 samples)
Overall average flit latency = 47.4453 (1 samples)
Overall maximum flit latency = 145 (1 samples)
Overall minimum fragmentation = 0 (1 samples)
Overall average fragmentation = 0 (1 samples)
Overall maximum fragmentation = 0 (1 samples)
Overall minimum injected packet rate = 0.0482 (1 samples)
Overall average injected packet rate = 0.0501896 (1 samples)
Overall maximum injected packet rate = 0.0528667 (1 samples)
Overall minimum accepted packet rate = 0.0482333 (1 samples)
Overall average accepted packet rate = 0.0501562 (1 samples)
Overall maximum accepted packet rate = 0.0528333 (1 samples)
Overall minimum injected flit rate = 0.0482 (1 samples)
Overall average injected flit rate = 0.0501896 (1 samples)
Overall maximum injected flit rate = 0.0528667 (1 samples)
Overall minimum accepted flit rate = 0.0482333 (1 samples)
Overall average accepted flit rate = 0.0501562 (1 samples)
Overall maximum accepted flit rate = 0.0528333 (1 samples)
Overall average injected packet size = 1 (1 samples)
Overall average accepted packet size = 1 (1 samples)
Overall average hops = 4.02055 (1 samples)
Overall throughput of the network (flits/cycle) = 1.60123
Overall throughput of the network (packets/cycle) = 1.60123
Total run time 69.2234

```

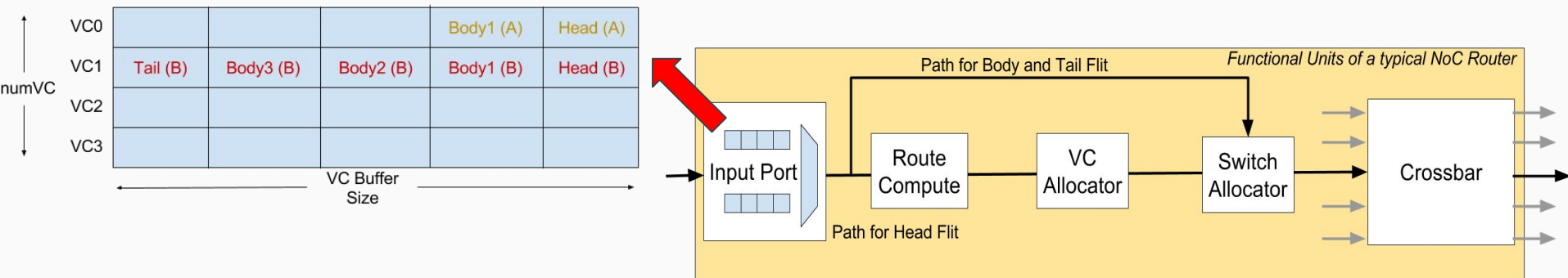
# Fixing Memory Leaks

- Memory consumption of 4GB and increasing to a saturation point of 50MB!
- Exhaustive search through Valgrind
- Bug: Self-msgs were relocated to new chunk of memory before being deleted



# Optimization: Number of sockets

- PANE Version 1 had  $8 \times 8 \times 5 \times 5 = 1600$  open sockets! (for a  $8 \times 8$  Mesh network)
- Different Buffer Organization



- Initially, no of sockets =  $8 \times 8 \times 5 \times 5 \times 4 = 6400$ !! (for a  $8 \times 8$  Mesh network, with 4 Virtual Channels)

# Optimization: Number of sockets

- Initially, no of sockets =  $8 \times 8 \times 5 \times 5 \times 4 = 6400!!$  (*for a 8X8 Mesh network, with 4 Virtual Channels*)
- 1 Data over 1 socket per cycle -----> Multiple Data over 1 socket per cycle.

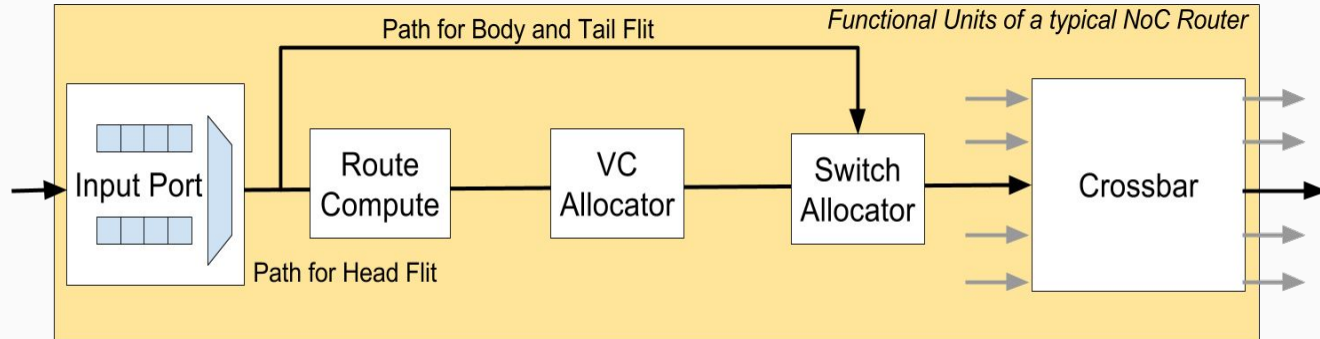
Problem: UNIX Domain Sockets map according to activity and not number of transfers done.

Solution: Concatenation of '\*' after multiple data and making VC# as an attribute of the msg.

- Reduced to  $8 \times 8 \times 5 \times 4 = 1280$  sockets! (*for a 8X8 Mesh network, with any number of Virtual Channels*)

# Optimization: Simulation Time

- Extension to multiple flit size
- Routing only the head of the packet.



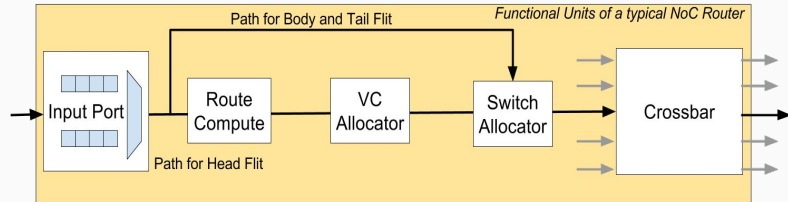
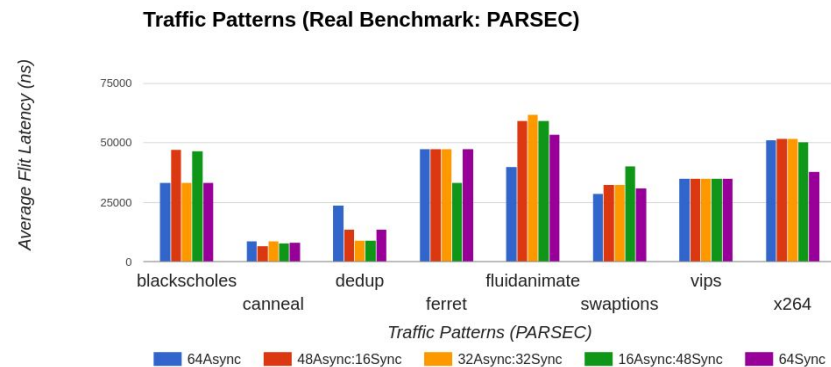
# Scripts

For changing configurations, running and collecting statistics of:

- Traffic Pattern: Synthetic and Real
- Routing Function
- Packet Size
- Injection Rate
- Network Size: Sockets
- Connection(ned) file: Topology and network size

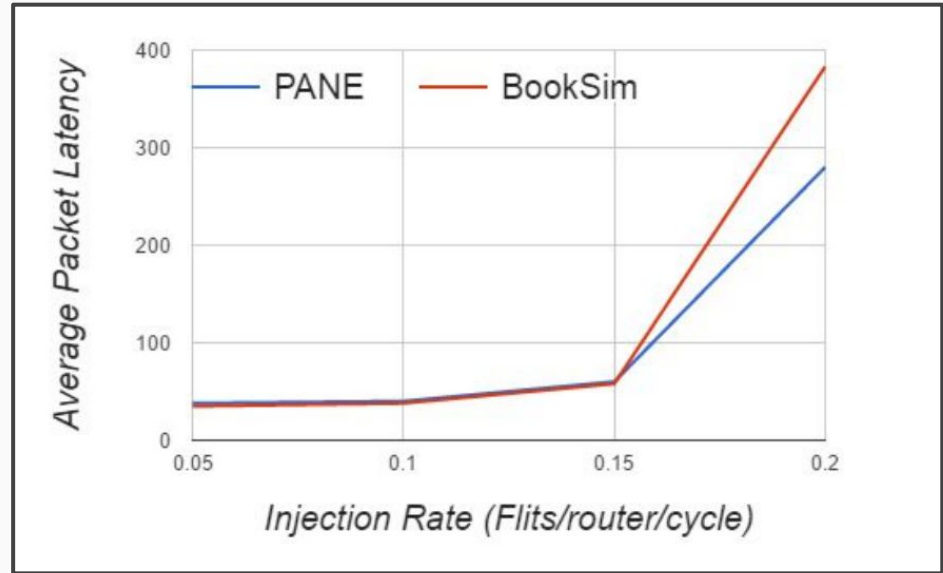
# Miscellaneous

- Addition of Configurable Heterogeneous Network
- BookSim Synchronous Behaviour Assumptions - Flush all data each cycle!
- PANE runs even if there are no packets in the network.
- Header files for all socket and miscellaneous functions



# Validating PANE With Synchronous NoC Simulator

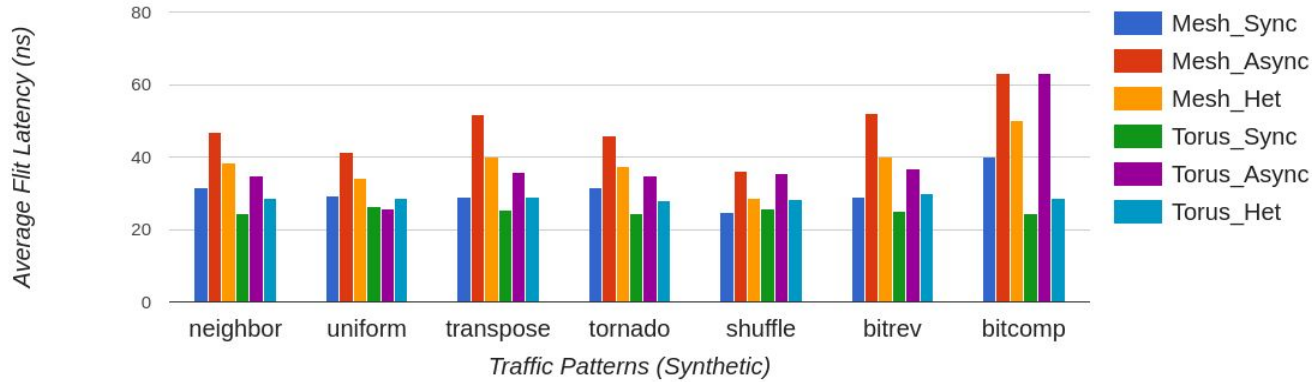
BookSim2<sup>[1]</sup> can be used as a reliable reference for validation. The PANE implementation, for the validation experiment, is such that each PANE client takes unit time to complete its action. In other words, PANE is configured to simulate a synchronous NoC. Since this is the same as simulating a NoC in BookSim2<sup>[1]</sup>, the results obtained after both the simulations should match.



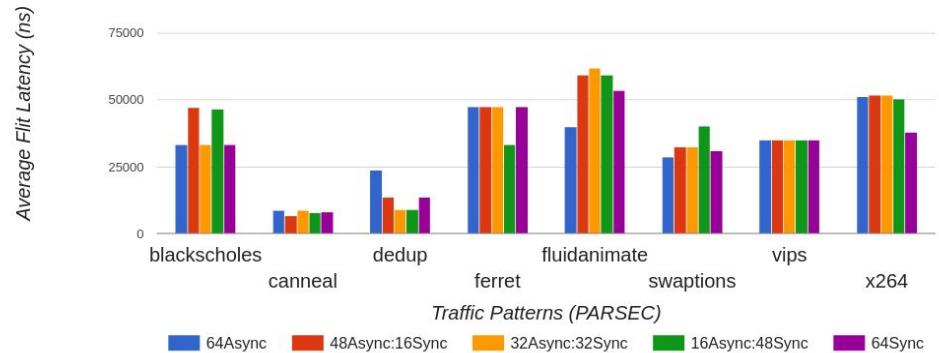
# PLOTS

# Traffic Pattern: Synthetic and PARSEC

## Traffic Patterns (Synthetic)

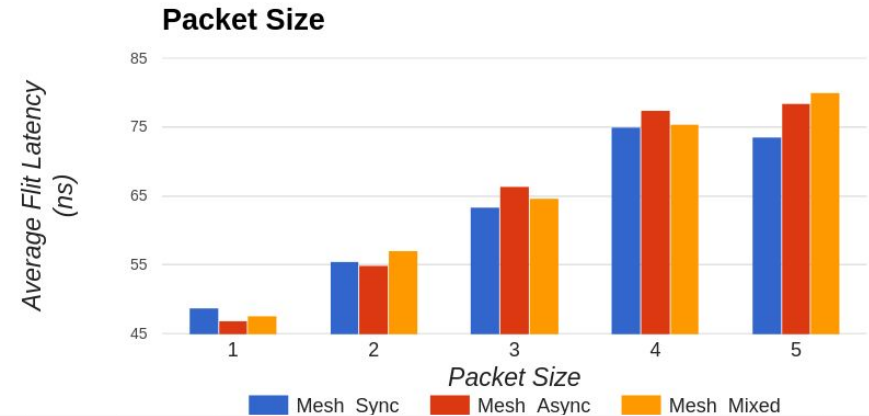
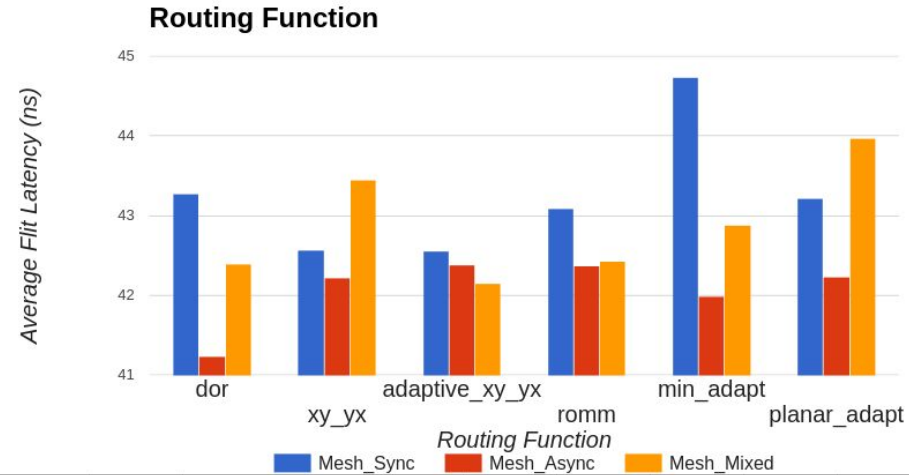


## Traffic Patterns (Real Benchmark: PARSEC)

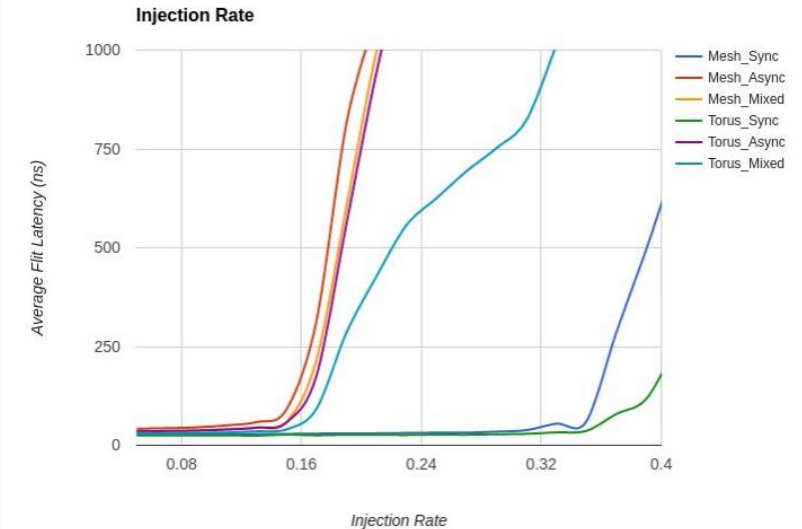
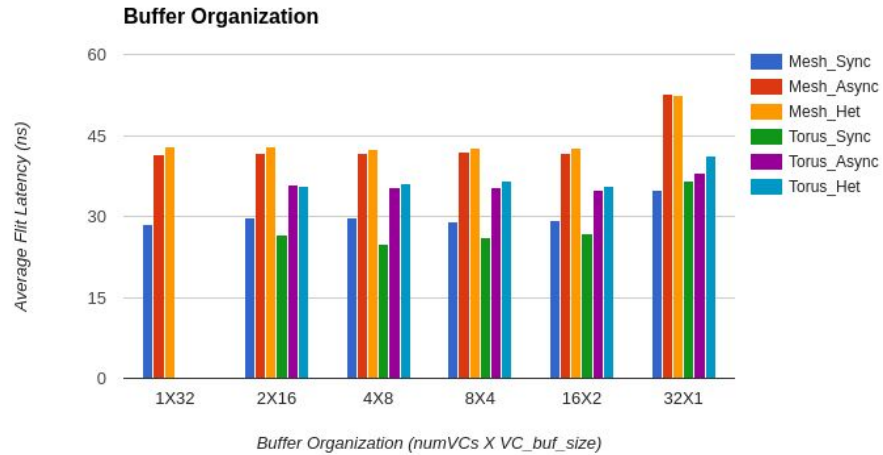




# Routing Function & Packet Size



# Buffer Organization & Injection Rate



# Next steps

## Progress Area 1

Reduce the number of Sockets by concatenation of multiple data over single socket!

## Progress Area 2

Reduce the Simulation Time.

## Progress Area 3

Develop a GUI and automation scripts for configuring and collecting statistics.

DEMO

# Thank You!

## References

1. Nan Jiang, Daniel U. Becker, George Michelogiannakis, James D. Balfour, Brian Towles, David E. Shaw, John Kim, and William J. Dally. 2013. A Detailed and Flexible Cycle-Accurate Network-on-Chip Simulator. In ISPASS.
2. Andras Varga and Rudolf Hornig. 2008. An Overview of the OMNeT++ Simulation Environment. In Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops (Simutools '08).
3. Stevens, W. Richard, Bill Fenner, and Andrew M. Rudoff. UNIX network programming. Vol. 1. Addison-Wesley Professional, 2004.