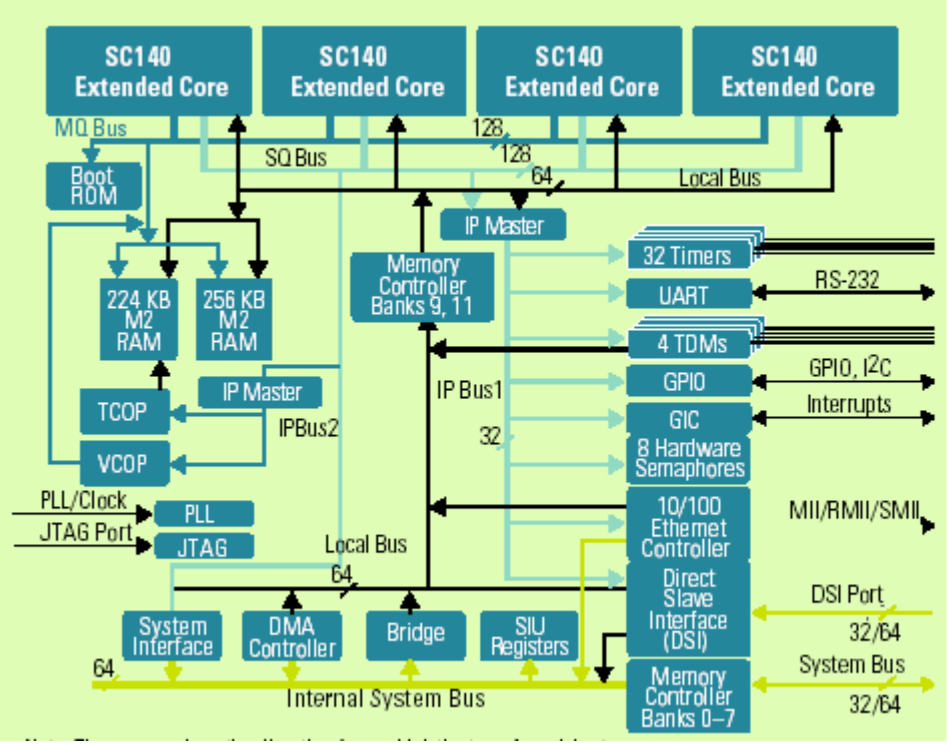
An Application-Specific Design Methodology for STbus Crossbar Generation

Srinivasan Murali, Giovanni De Micheli
Stanford University
{smurali, nanni}@stanford.edu

Introduction

- Systems On Chips have multiple components, cores
- Communication between cores rapidly increasing
- Wire scaling not on par with transistor scaling
- Communication architecture becomes major bottleneck
  - Scalability
  - Delay
  - Power and
  - Reliability



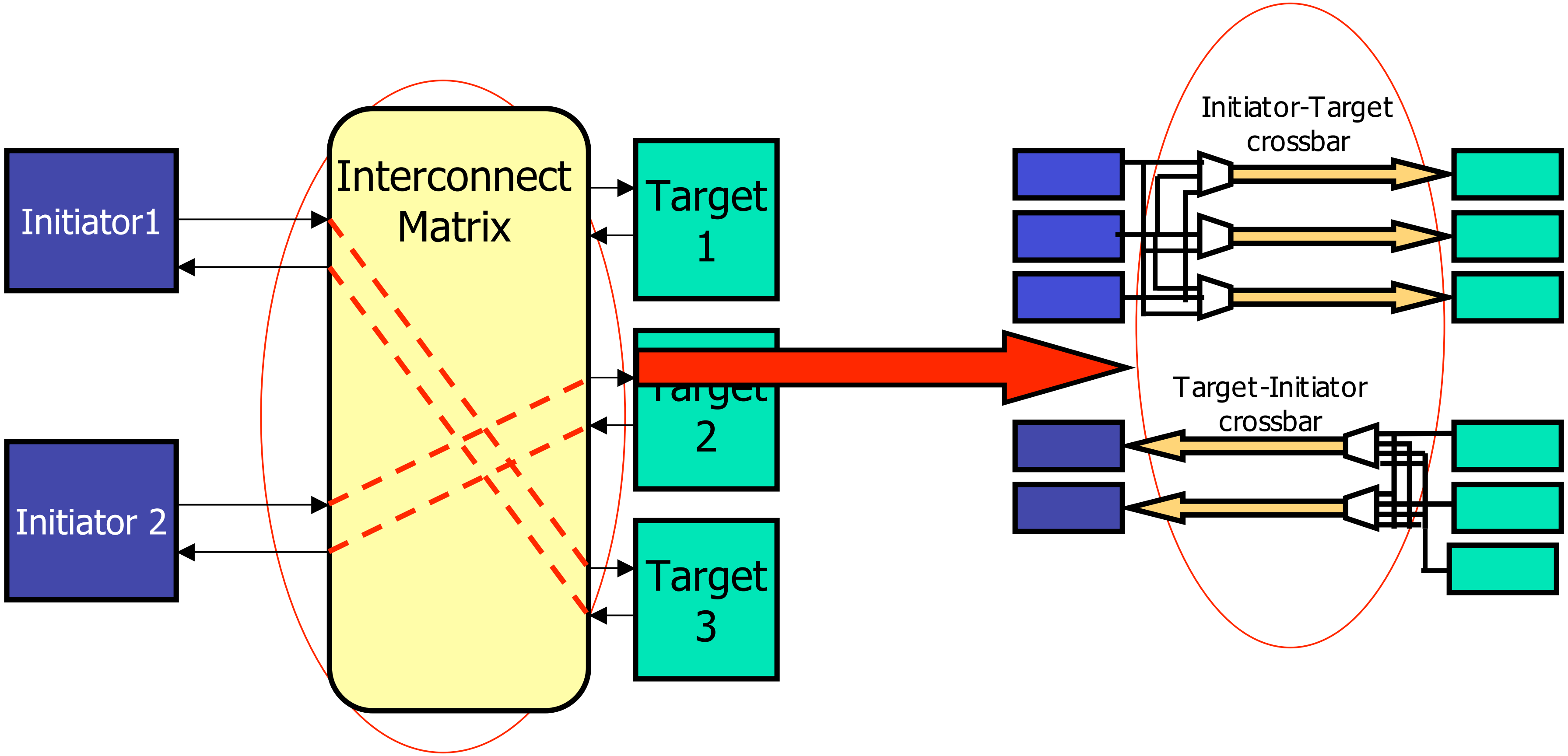
Motorola’s MSC8126 SoC platform (3G base stations)

Communication Architecture

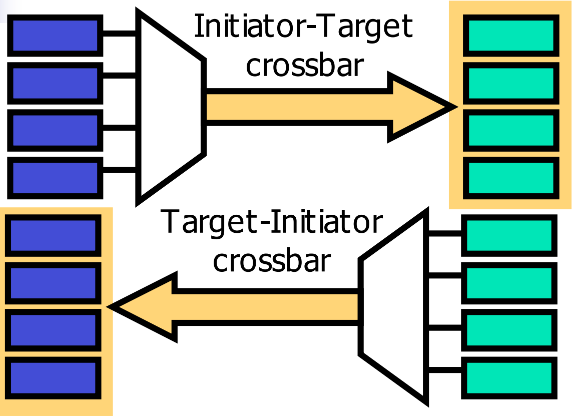
- Several standard bus architectures:
  - Large semiconductor firms (e.g. IBM Coreconnect, STMicro STbus)
  - Core vendors (e.g. ARM AMBA)
  - Interconnect IP vendors (e.g. SiliconBackplane)
- Evolution of communication architectures:
  - Single bus
  - Bridged buses
  - Crossbars (multiple parallel buses)
    - AMBA Multi-layer
    - STbus crossbar …
  - Networks on Chips

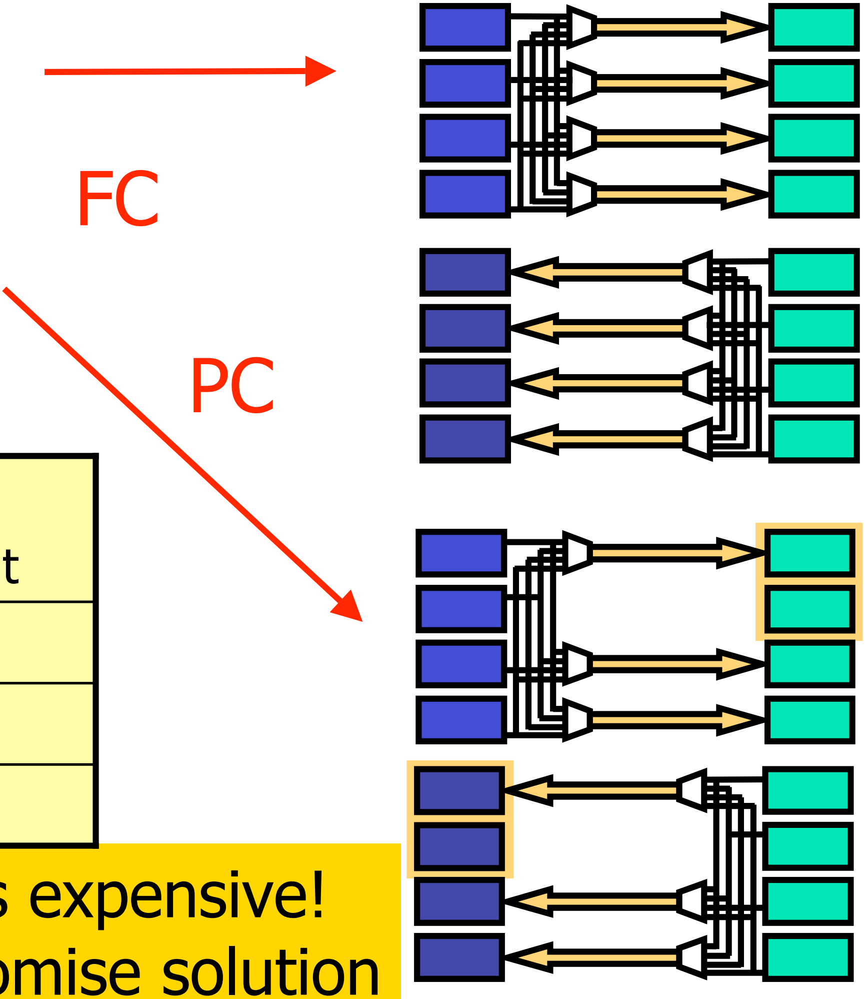
Crossbar Architecture

- Low-latency, high bandwidth infrastructure



Crossbar & Partial CB cost





Type	Avg. Lat (cycles)	Max. Lat (cycles)	Bus Count
BUS	35.1	51	2
FC	6	9	21
PC	9.9	20	6

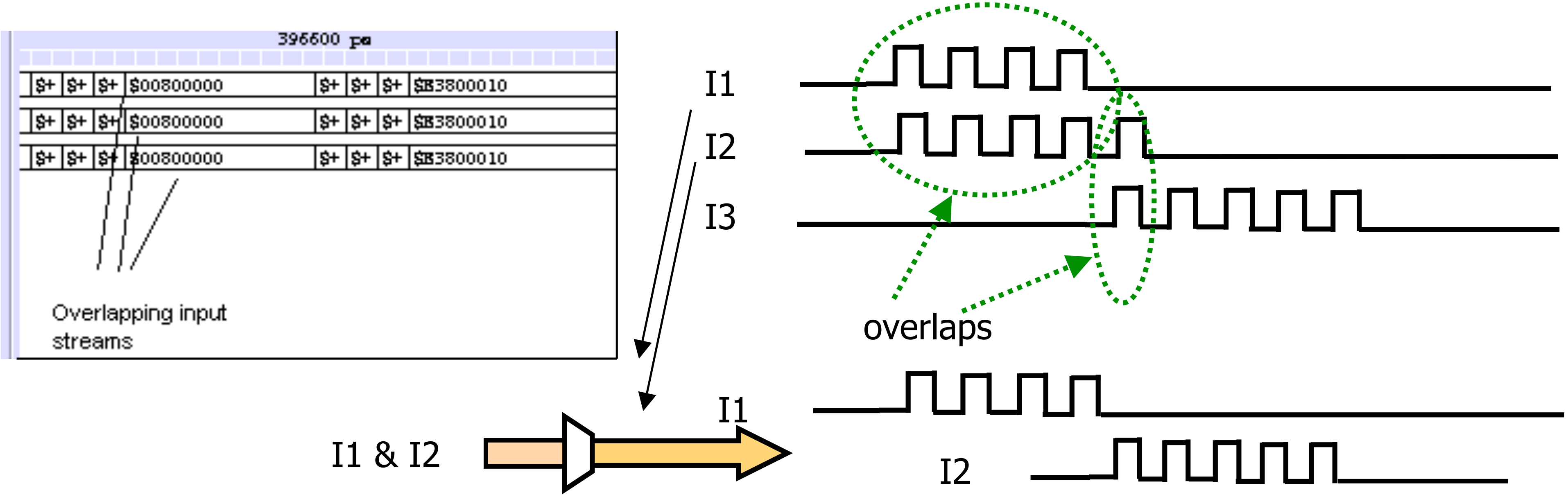
Key issue: Full crossbar is expensive!
Partial crossbar is a compromise solution

Motivation

- Full STbus crossbar:
  - lot of wires & gates
  - e.g. Area_cell_4x4/Area_cell_bus ~2
- Optimum Partial crossbar:
  - Latency close to Full crossbar
  - Fewer components, area, power
- How to design best partial crossbar for applications ?
- General design methodology
  - Fine-tuned to particular architecture (in this work: STbus)

Application Traffic Analysis

- Example traffic trace from 3 initiators



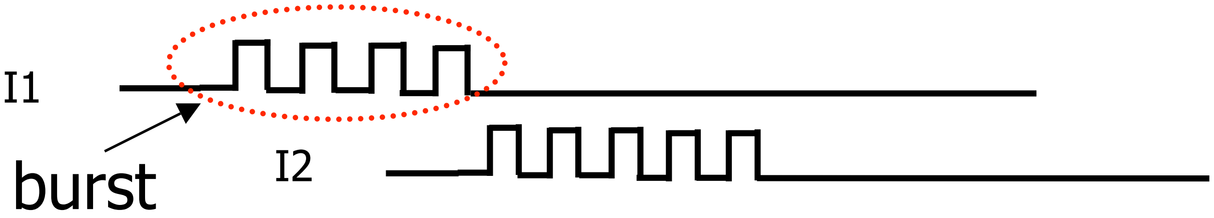
```
396600 ps
$+ $+ $+ $00800000    $+ $+ $+ $B3800010
$+ $+ $+ $00800000    $+ $+ $+ $B3800010
$+ $+ $+ $00800000    $+ $+ $+ $B3800010
```

Overlapping input streams

- overlap increases average and peak latency
- local variations in traffic rates

Crossbar Design Constraints

- Match the application characteristics
- Minimize average & peak packet latency
  - Support the bandwidth requirements of communication
  - Consider local variations in traffic rates as well



- Consider criticality of streams (partial QoS support)
- Objective: Minimum components /power consumption

Previous Work

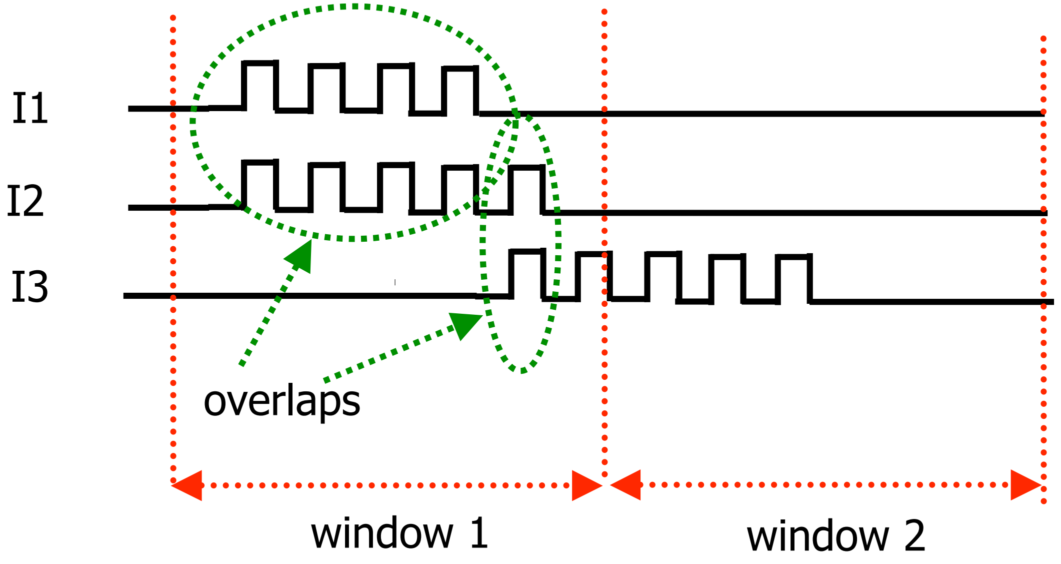
- Bus and Networks on Chip synthesis
  - Average bandwidth analysis
    - Pinto et al. (DAC ‘02, ICCD ‘03)
    - Hu et al. (ASPDAC ‘03, DATE ‘03)
    - Our earlier works (DATE ‘04, DAC ‘04)
  - Peak bandwidth based
    - Ho et al. (HPCA ‘03)
  - Statistical traffic generators
    - Bolotin, et al. (JSA ‘04)
  - Regulating traffic injection
    - Lahiri et al. (TCAD ’04), our earlier work (ASPDAC ‘05)

Previous Work

- Bus mapping & protocol design (Lahiri et al. (TCAD ‘04))
- Automatic bus and network generation
  - T. Yen et al. (ICCAD ‘95)
  - Gasteier et al. (ACM TODAES ‘99)
  - K. Ryu et al. (DATE ‘03)
  - Xpipescompiler (DATE ‘04)

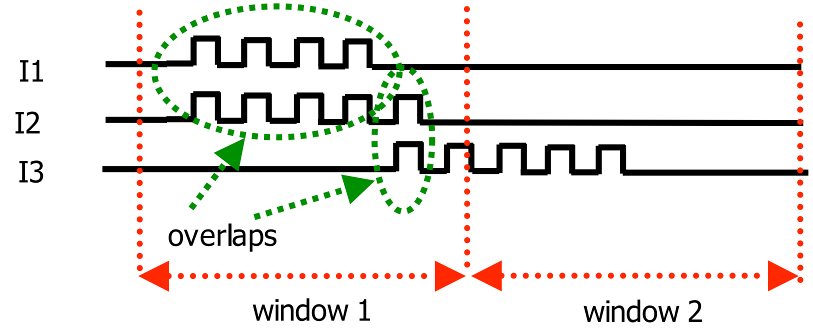
Crossbar Design Approach

- Functional traffic of application for design
- Simulation time window for analysis:
  - Split to Fixed sized windows



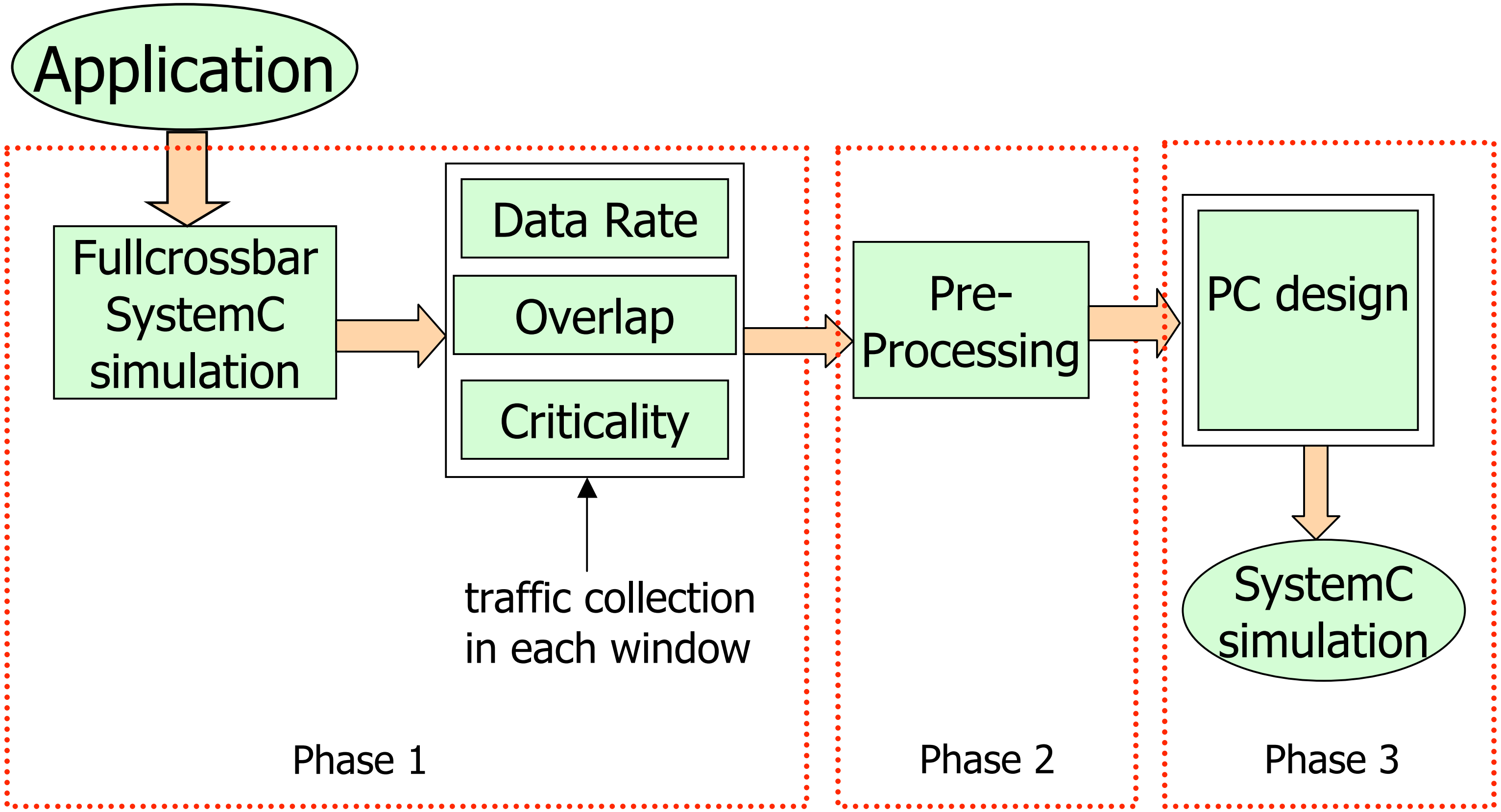
Crossbar Design Approach

- In each simulation window
  - Satisfy bandwidth requirements
  - Minimize overlaps among streams
  - Consider criticality of streams



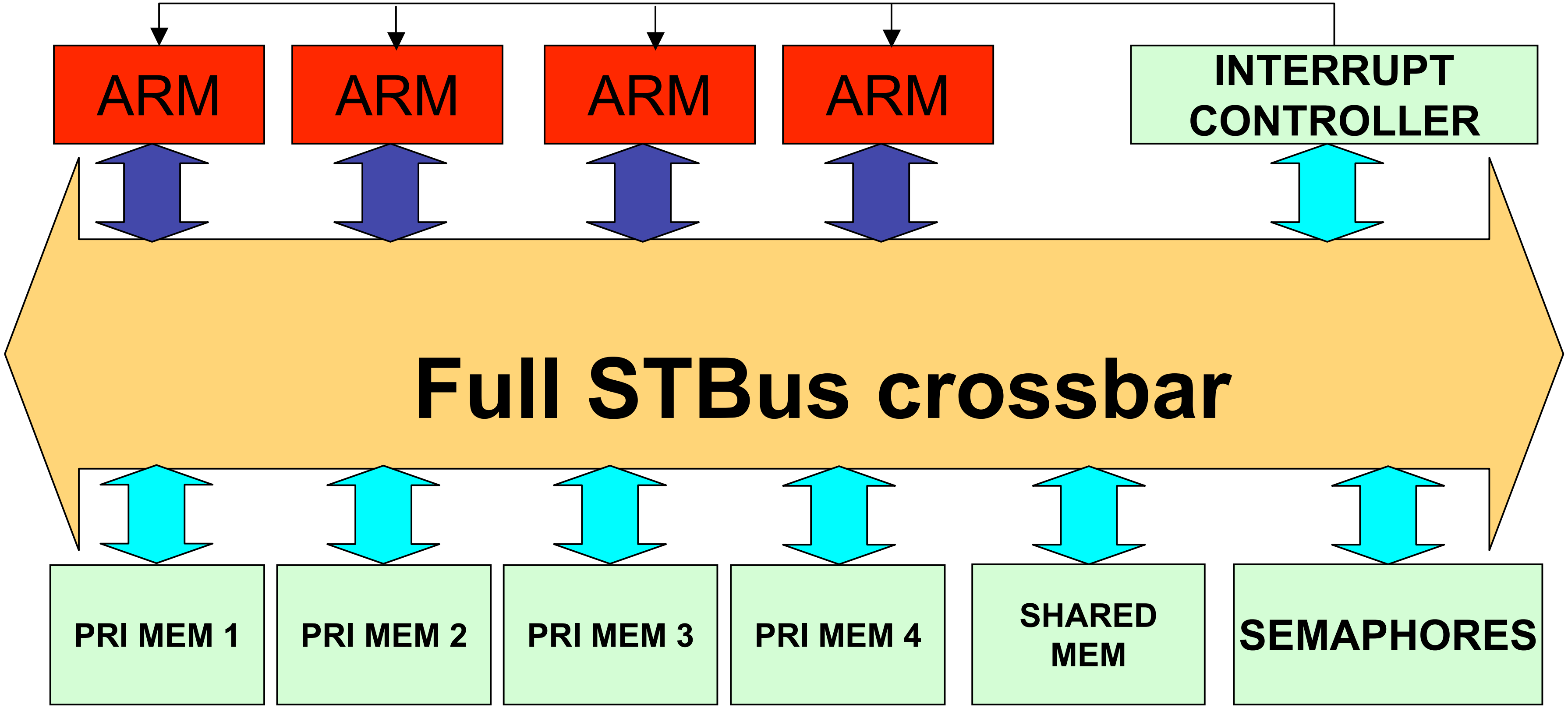
- Merge channels with non-overlapping traffic
- Time windows tighten worst-case
- Methodology spans an entire design space spectrum
  - Average and peak bandwidth based analysis are the two extreme points
  - Design point varied by varying window size

Design Flow For PC Design



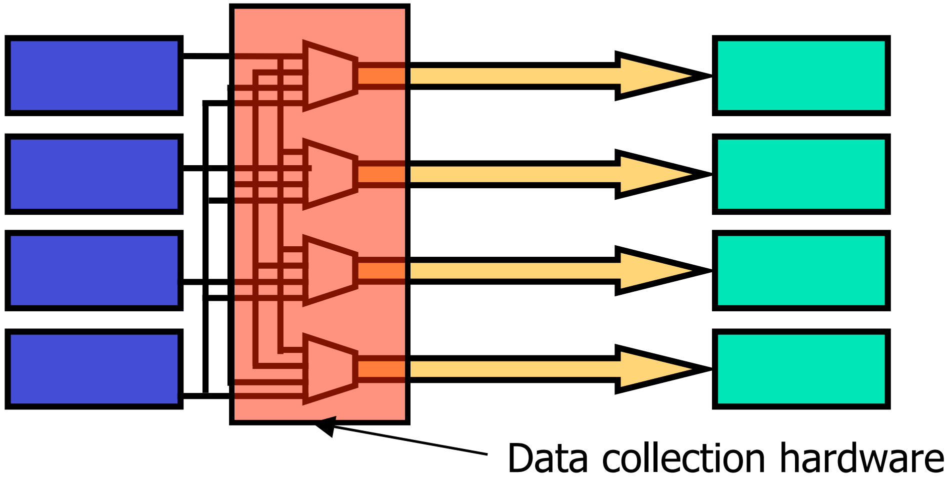
Phase 1: Initial Simulation

MPARM Simulation Environment



Phase 1

- Full crossbar results in perfect communication
- Data collection hardware added to arbiters

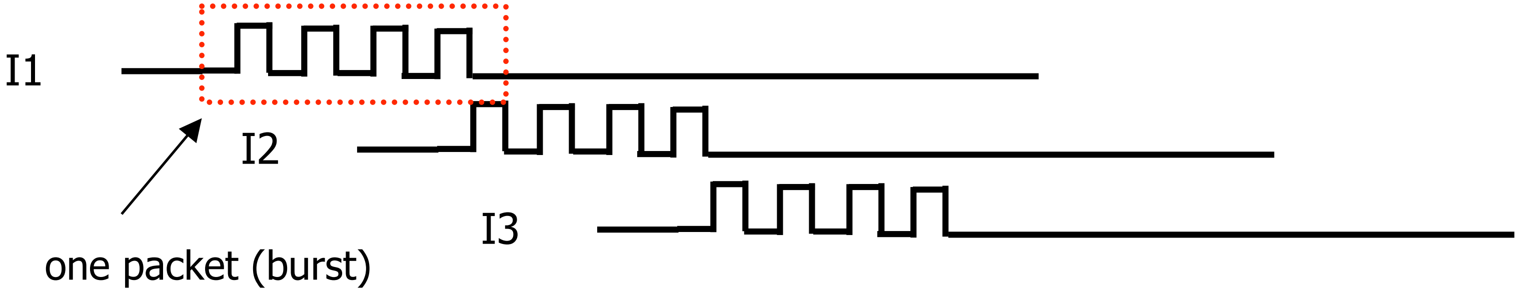


- Traffic collection on each window
  - Data rate for each core
  - Overlap among streams
  - Criticality of streams

Phase 2: Pre-processing

Identify

- cores that should be on different buses
  - Cores with large overlap (above threshold)
  - Cores with overlapping critical streams
- Maximum number of cores on bus
  - To bound maximum latency



Phase 3: Crossbar Design

- Start with a single bus
- Check for feasible solution
  - Satisfy window bandwidth constraints
  - Place forbidden core pairs on different buses
  - Fewer than maximum number of cores on each bus
- Repeat step 2, incrementing the number of buses by 1
- Optimal Binding
  - Minimize overlap on each bus
  - Satisfying the above constraints

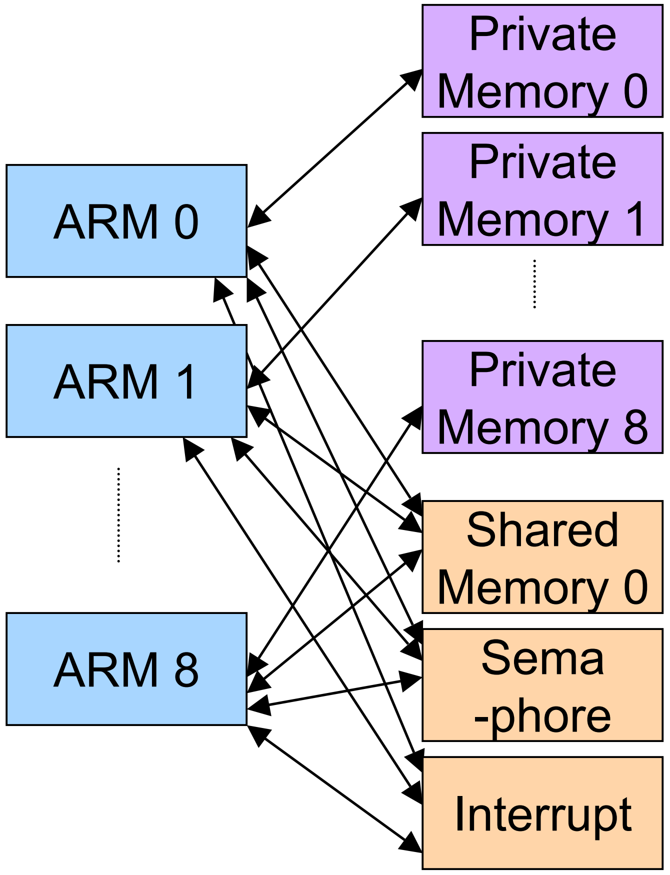
Phase 3: Crossbar Design

- Feasibility check & optimal bindings modeled as small Integer Linear Programs (ILPs).
    - Size of ILPs small (maximum cores is 32 in STbus)
    - ILPs solved using CPLEX package
    - Less than few hours for all simulations (1 Ghz SUN workstation)
- Simulate resulting crossbar in MPARM

Simulation Results

Analysis of PC Design

Matrix Multiplication Benchmark (21 cores)



- Traffic to shared targets smaller
- ARM – Private Memory flows have substantial overlap

Analysis of PC Design

- Designed PC: 3 buses (initiator-target)
- Each bus: 3 private and 1 shared target
- Targets with highly overlapping streams on different buses
- Result: Acceptable performance (latency)
- 3.5× reduction in the number of buses used

Experiments on Benchmarks

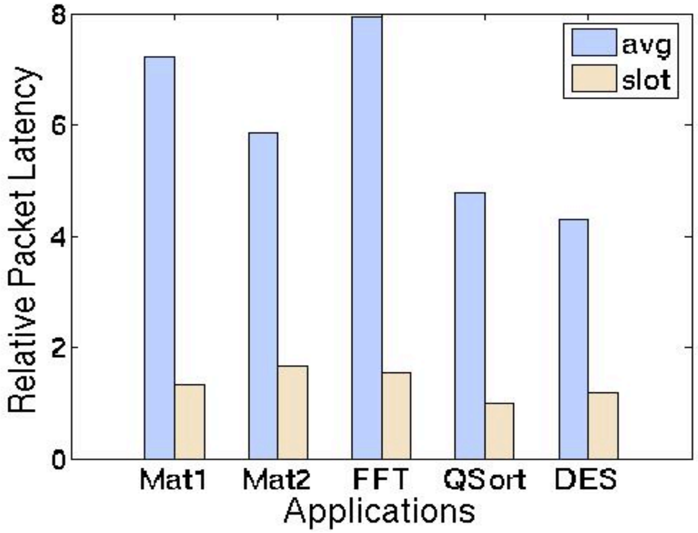
Component savings compared to Full Crossbar

App.	FC bus count	PC bus count	Ratio
Mat1	25	8	3.13
Mat2	21	6	3.5
FFT	29	15	1.93
Qsort	15	6	2.5
DES	19	6	3.12

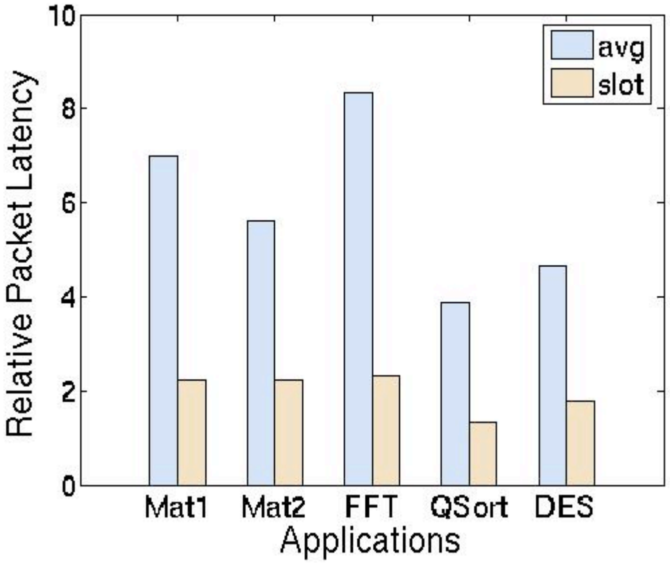
- Avg. & Peak latencies within few cycles of Full Crossbar

Use of Simulation Windows

Relative Average Latency

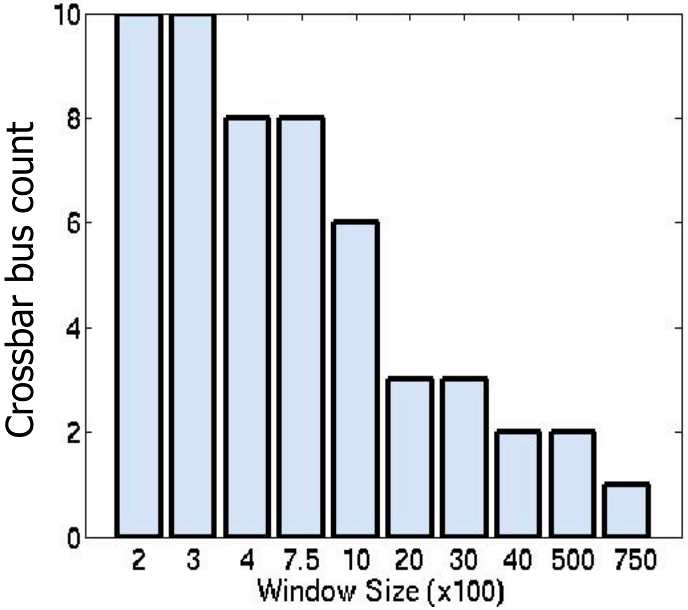


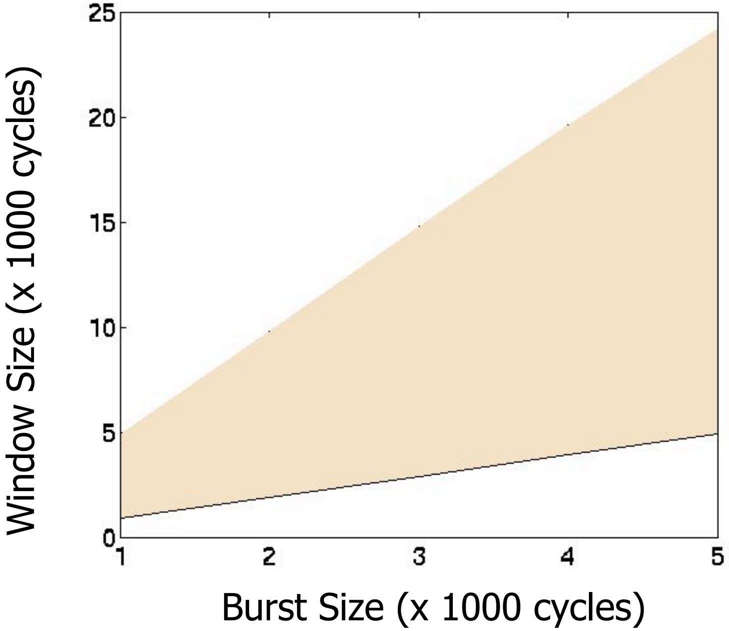
Relative Peak Latency



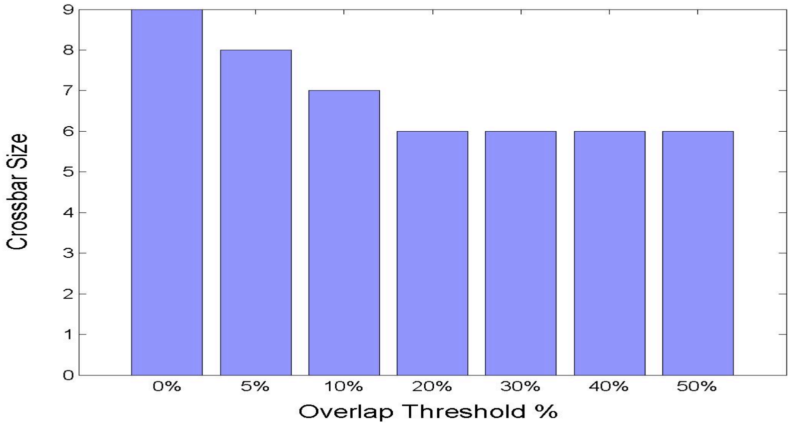
Sensitivity to constraints

- Window size & overlap constraints-parameters
- Trade off conflicts agains HW complexity





Overlap Threshold Setting



- Controls peak and average latencies
- From experiments, threshold value can be set:
  - 10% of window size for conservative designs
  - 30%-40% of the window size for aggressive designs

Conclusions

- Communication architecture should match application characteristics
- Presented methodology for STbus crossbar design
  - Local variations in traffic,
  - Overlap of streams
  - Actual application traffic
- Large savings in components, good performance
- Approach can be extended to other bus designs
- In future: protocol design, power issues

Thank You