# Concurrent Exploration of Memory and Communication Architecture for MPSoCs

**Nikil Dutt**

**ACES Laboratory**

**Center for Embedded Computer Systems**

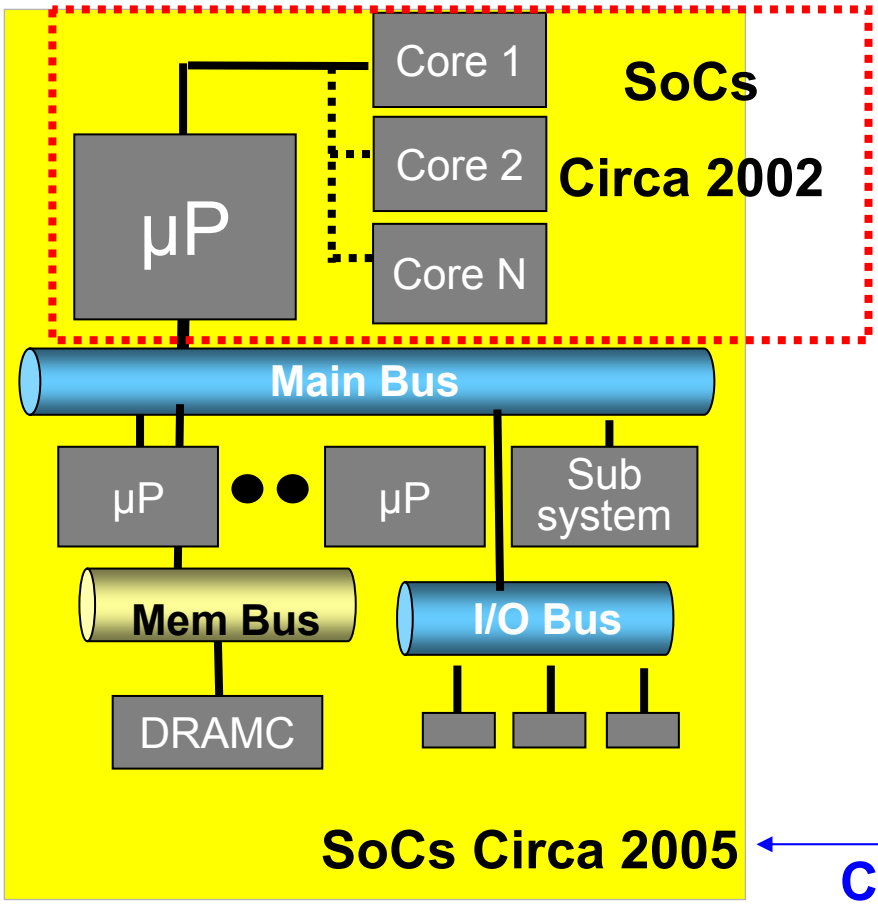
**Donald Bren School of Information and Computer Sciences**

**University of California, Irvine**

**dutt@uci.edu**

**http://www.ics.uci.edu/~aces**

# Data Flow Replacing Data Processing As Major SoC Design Challenge




Critical Decision Was uP Choice

- Exploding core counts requiring more advanced Interconnects
- EDA cannot solve this architectural problem easily
- Complexity too high to hand craft (and verify!)

Critical Decision Is Interconnect Choice

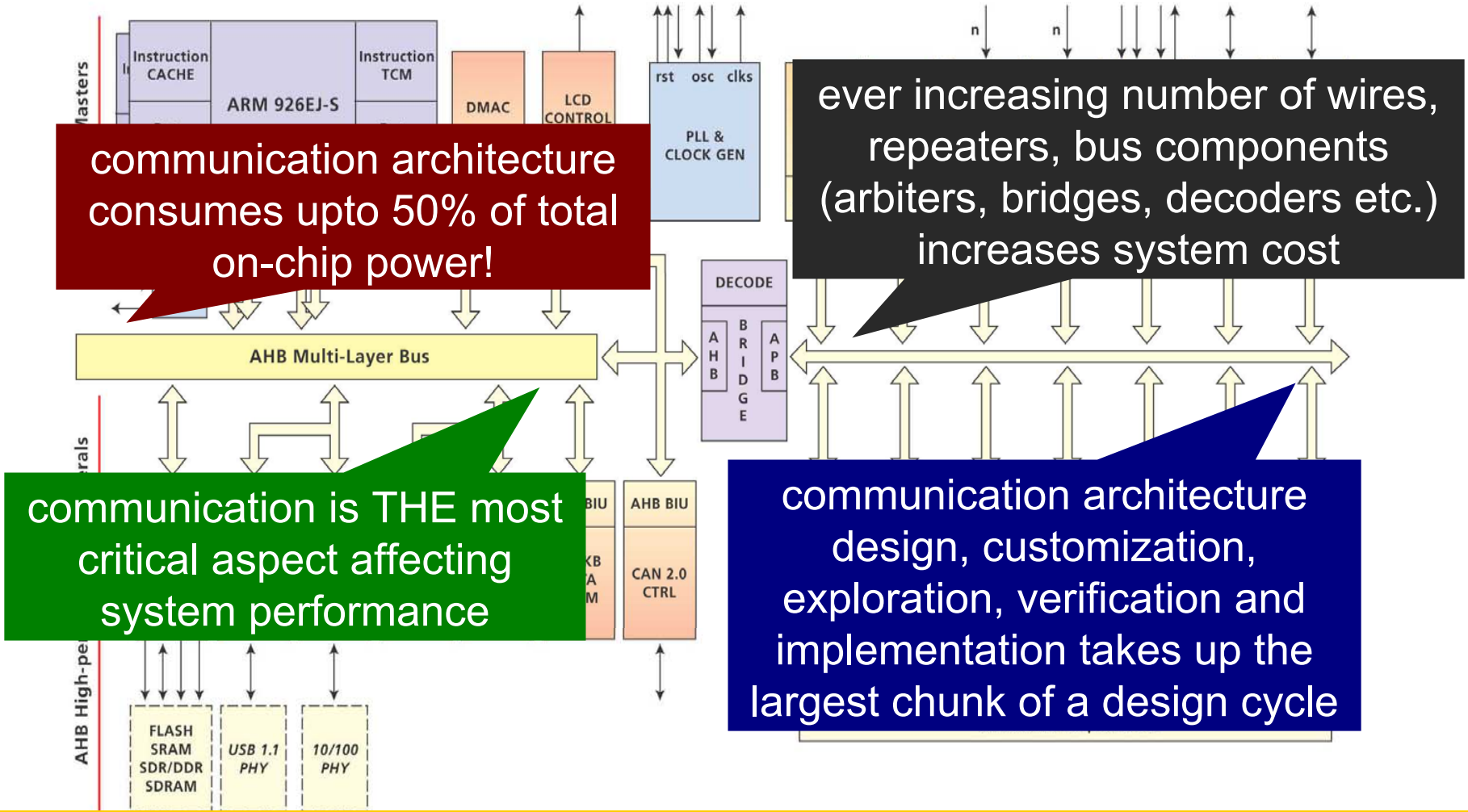
**Communication Architecture Design and Verification becoming Highest Priority in Contemporary SoC Design!**





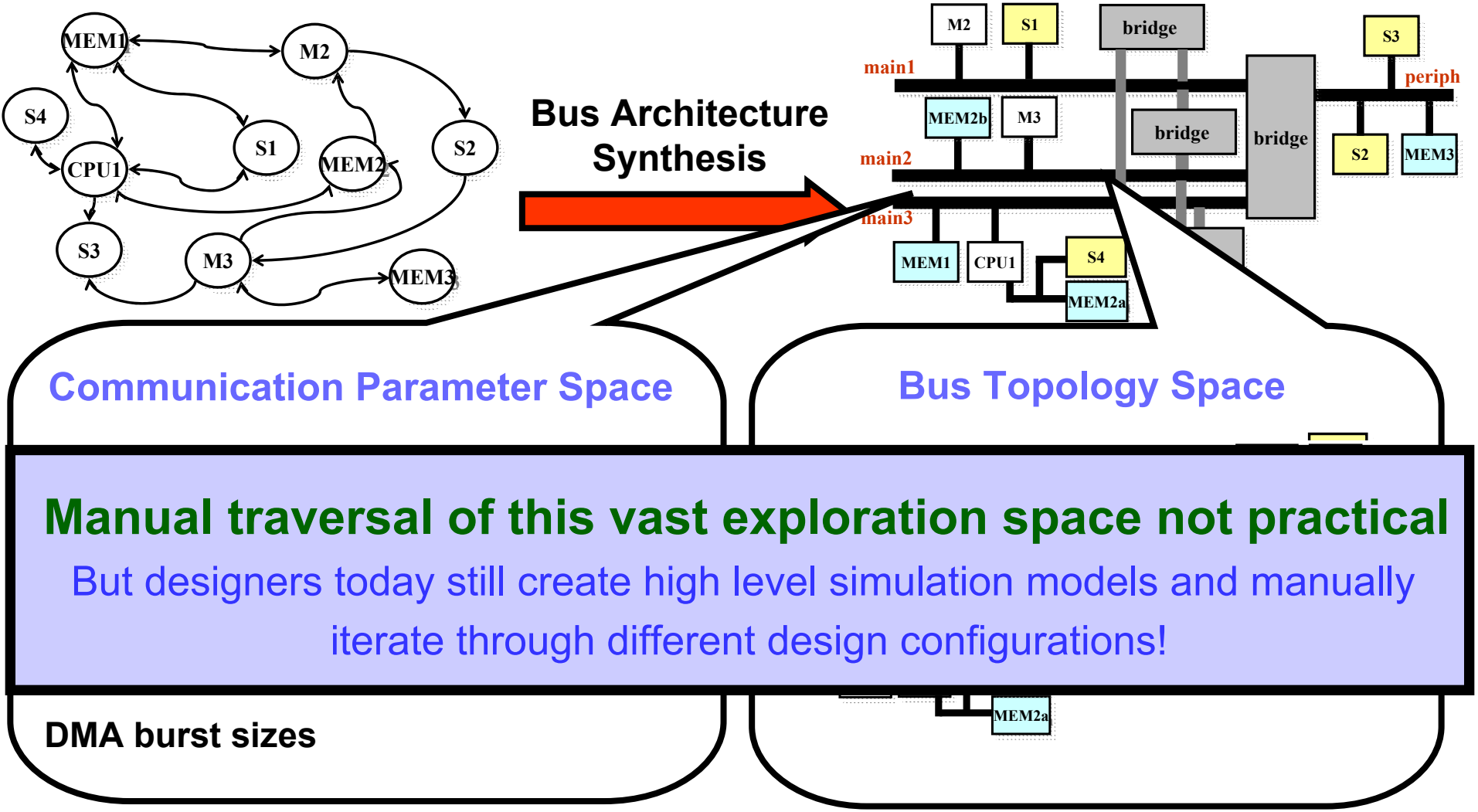
Source: SONICS Inc.

# Need for Communication-centric Design Flow

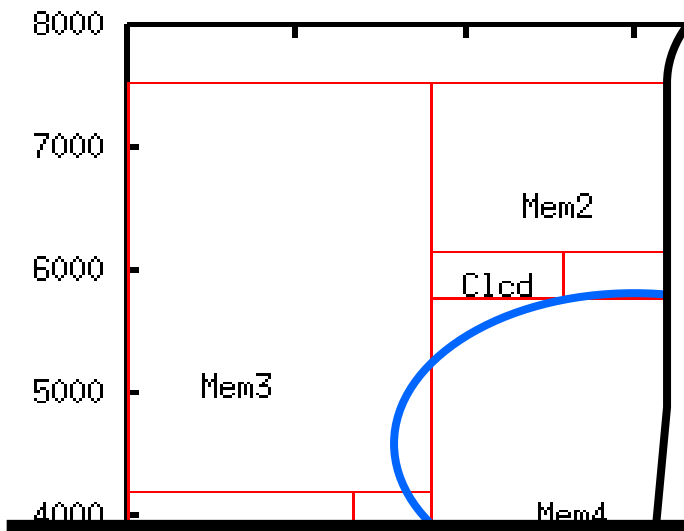



**Communication Architectures** in today's complex systems **significantly** affect performance, power, cost and time-to-market!

# Bus Architecture Synthesis



# Need for Physically-Aware BA Synthesis



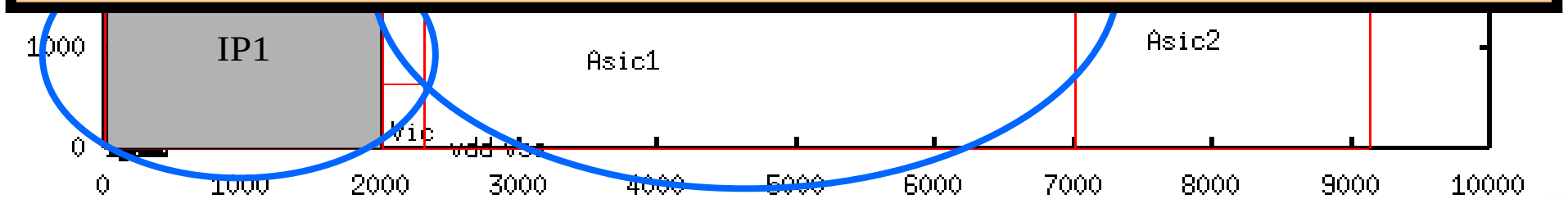

**After physical design, we detect a bus cycle time violation!**

**Such a violation has adverse effect on system cost, complexity and constraint satisfiability**

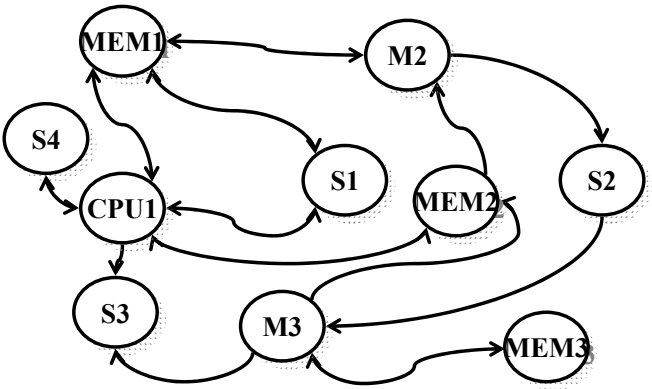
**To eliminate bus cycle violations, designers pipeline busses with latches, register slices …**

**severely effects performance**

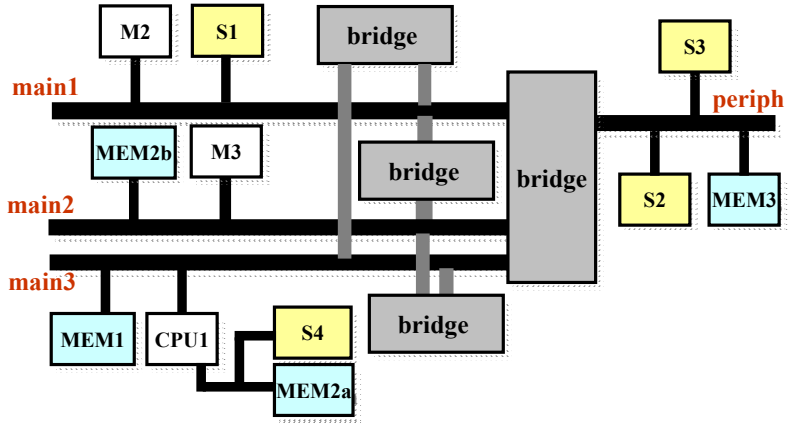
**Since BA synthesis determines effective capacitance on bus, there is a need to make BA synthesis *physically aware***



# Our Approach: FABSYN (DAC-2005)




**Automated**
**Bus Architecture Synthesis**




- early BA exploration and timing violation detection / elimination
  - **verify feasibility of synthesized BA early in the design flow**
  - **saves costly design iterations later**

- increasingly important in the deep submicron era as
  - ***clock speeds*** **increase**
  - **lengthy *propagation delays* cause timing violations**

# COSMECA: Application Specific Co-Synthesis of Memory and Communication Architectures for MPSoC

**Sudeep Pasricha and Nikil Dutt**

**ACES (Architectures and Compilers for Embedded Systems) Lab**

**Center for Embedded Computer Systems (CECS)**

**University of California, Irvine**

**{sudeep,dutt}@cecs.uci.edu**

# Outline

- **Motivation**
- Synthesis of Communication and Memory Architectures
- Problem Formulation
- Related Work
- COSMECA Co-Synthesis Methodology
- Case Studies
- Conclusion and Future Work

# Motivation

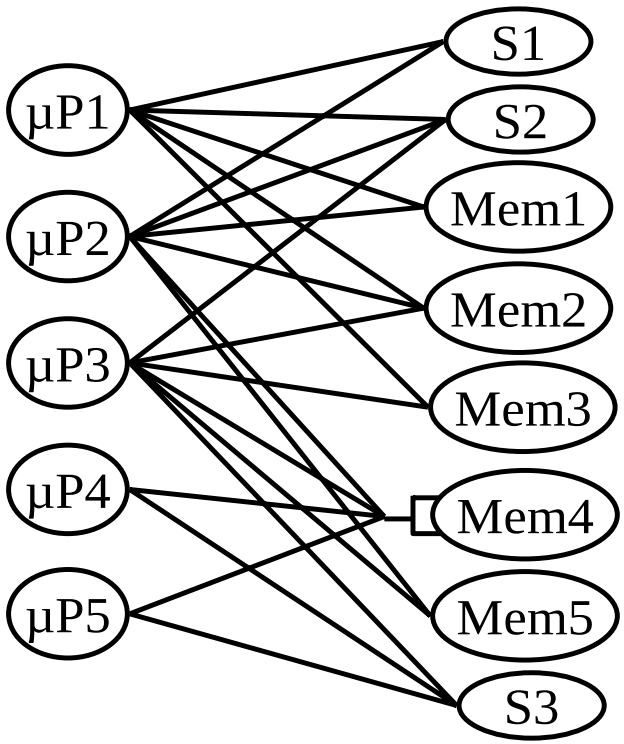
- **Modern multi-processor system-on-chip (MPSoC) designs rapidly increasing in complexity**
  - large bandwidth requirements
  - massive data sets which must be stored and accessed from memories
    - **especially for multimedia and networking applications**
- **MPSoC communication architecture fabric has to cope with entire inter-component traffic**
  - considerably impacting performance; design cycle time
- **Memory architectures dictate most of the data traffic flow in MPSoC**
  - considerably impacting performance; die area
    - **upto 70% of die area today; estimates indicate figure will go upto 90% soon**

**Therefore, it is IMPERATIVE that designers focus on exploration and synthesis of memory and communication architectures early in the design flow**
**e.g. using concepts proposed in *platform-based design***

# Why Memory and Communication Architecture Co-Synthesis?

- Traditionally, in platform-based design, memory synthesis is performed before communication architecture synthesis
  - mainly due to tractability issues
  - this can lead to sub-optimal design decisions
    - **resulting in higher cost systems**

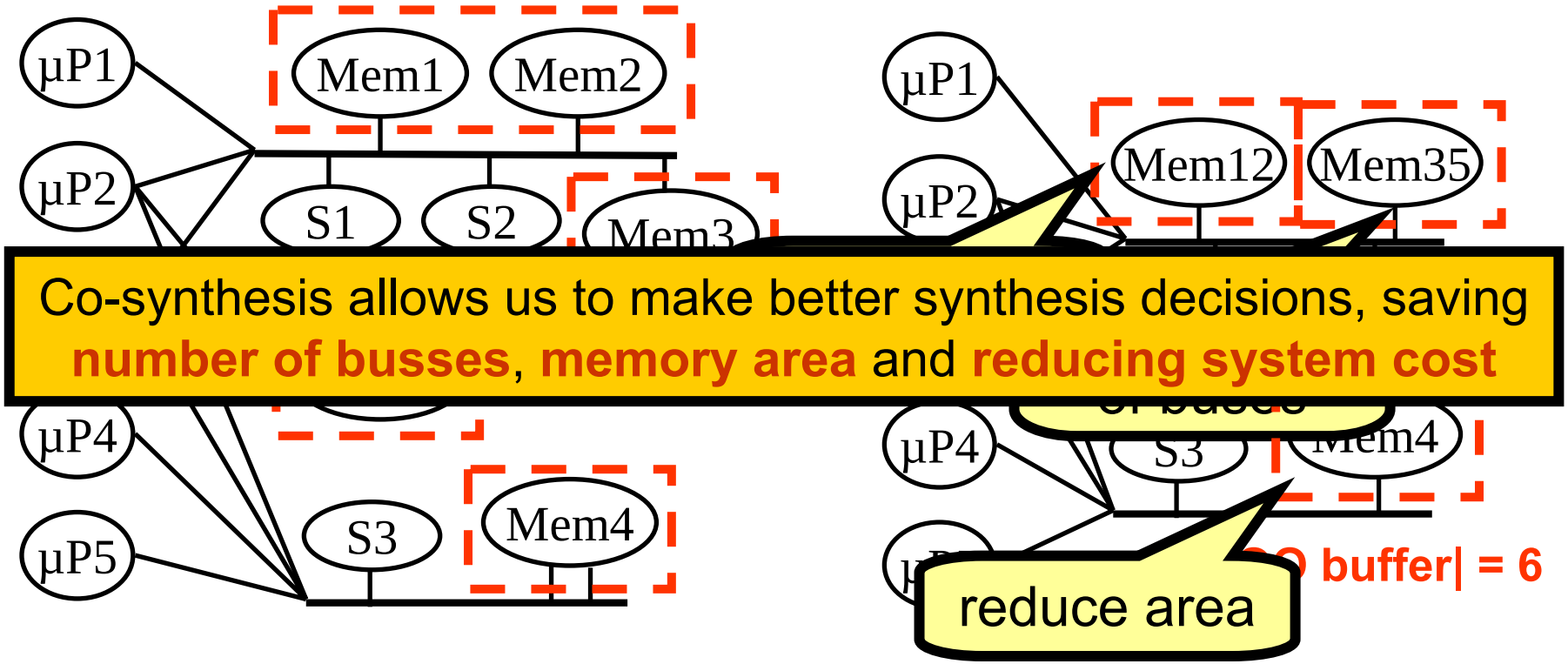



**consider synthesis of memory, communication architectures for the MPSoC shown**

# Why Memory and Communication Architecture Co-Synthesis?

**Typical separate synthesis case**

**Co-synthesis case**




Co-synthesis allows us to make better synthesis decisions, saving **number of busses**, **memory area** and **reducing system cost**

***mem area = 34.12 mm²***
***|bus|=11***

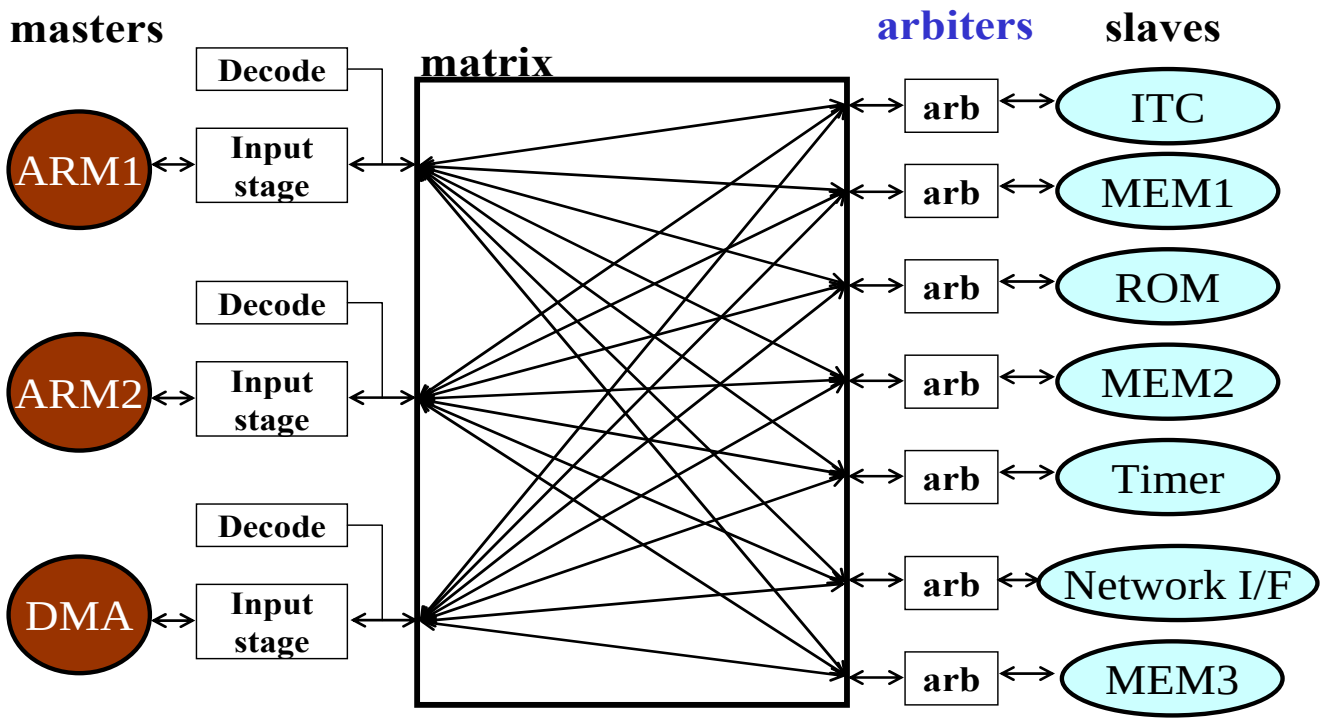
***mem area = 25.93 mm²***
***|bus|=9***

# Outline

- Motivation
- **Synthesis of Communication and Memory Architectures**
- Problem Formulation
- Related Work
- COSMECA Co-Synthesis Methodology
- Case Studies
- Conclusion and Future Work

# Bus Matrix Communication Architectures

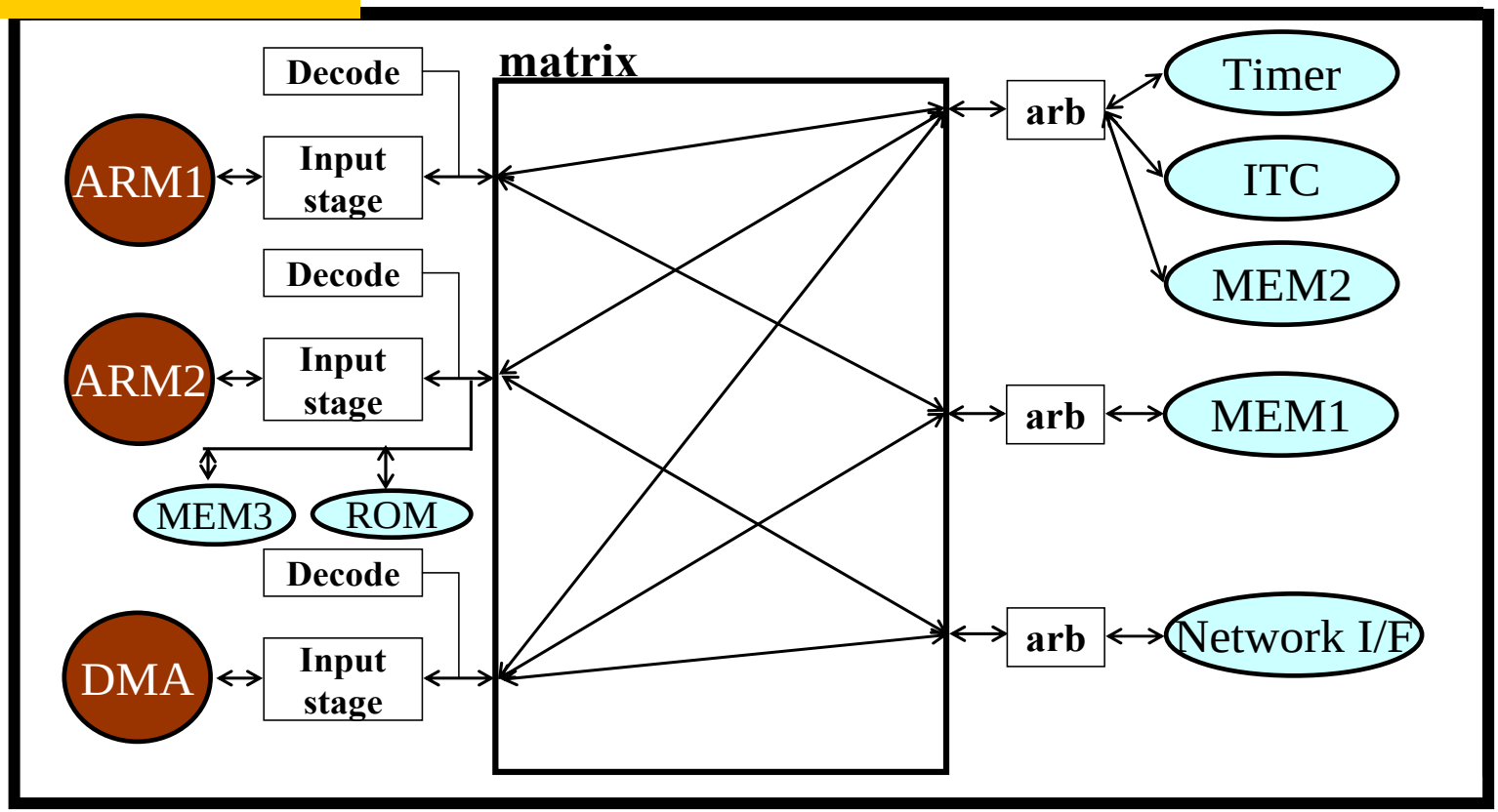
- Recent trend has been to use Bus Matrix communication architectures to support high bandwidths for modern MPSoC systems
  - AMBA, CoreConnect, STBus all support matrix configurations




Full Bus Matrix

# Bus Matrix Communication Architecture Synthesis

partial bus matrix




- Goal is to automatically synthesize a partial bus matrix with minimal number of buses, and which meets all performance requirements of the application

# Memory Architectures

- Variety of different memory types available to satisfy storage requirements in MPSoC applications
  - DRAMs, SRAMs, EPROMs, EEPROMs etc.
- Typically
  - DRAMs -> larger memory requirements, slower, cheaper
  - SRAMs -> smaller memory requirements, faster, expensive
  - EPROMs and EEPROMs -> read-only data
- Several tradeoffs during memory architecture synthesis
  - SRAM vs. DRAM
    - **cost vs. performance vs. area**
  - ports vs. number of memory blocks
  - …

# Memory Architecture Synthesis

- *COSMECA* selects memory blocks from a library populated by several types of memories
  - on-chip SRAMs
  - on-chip DRAMs
  - EPROMs and EEPROMs

- Each memory type can have variants in library, having different
  - ***capacities***, ***areas***, ***ports, operating frequencies*** and ***access times***

- Memory synthesis in *COSMECA* essentially maps application arrays and scalars to physical memories from the library

- Goal is to automate memory architecture selection to **meet memory area bounds**, while satisfying performance requirements

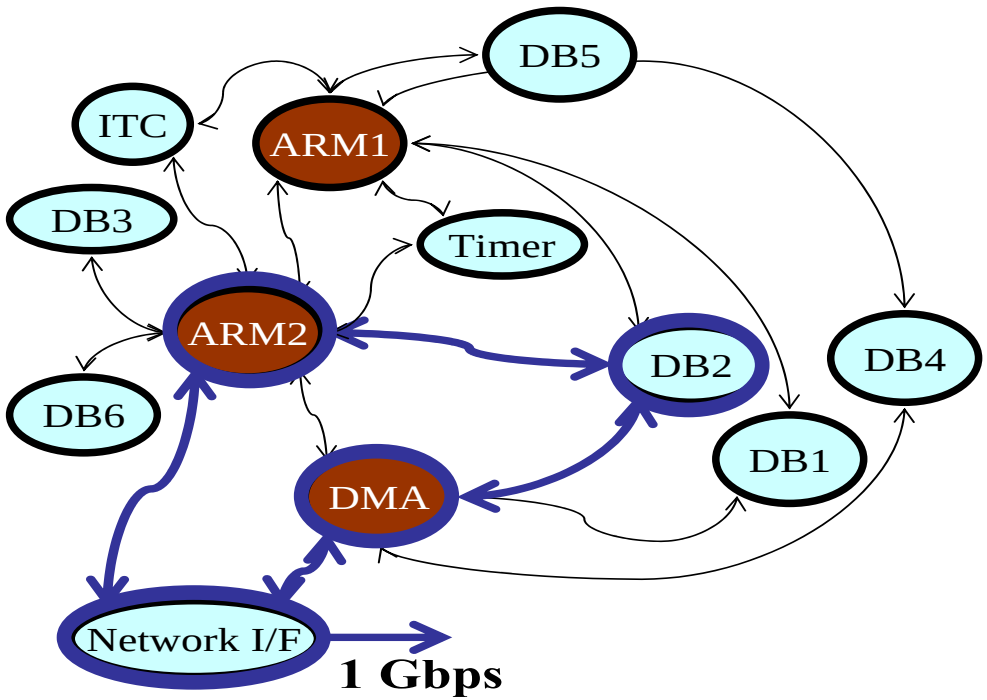
# Outline

- Motivation
- Synthesis of Communication and Memory Architectures
- **Problem Formulation**
- Related Work
- COSMECA Co-Synthesis Methodology
- Case Studies
- Conclusion and Future Work

# MPSoC Performance Constraints

- MPSoC designs have performance constraints that can be represented in terms of ***Data Throughput Constraints***
- **Communication Throughput Graph, *CTG = G(V,A)*** incorporates SoC components and throughput constraints
- **Throughput Constraint Path (TCP)** is a CTG sub-graph



# Problem Formulation

- **Given:**
  - an MPSoC with performance, memory area constraints
  - a target bus matrix communication architecture (e.g. AMBA, STBus)

- **Assumptions:**
  - hardware-software partitioning has been done already
    - except memories → represented as abstract DBs
  - IPs are non-modifiable "black box" components
    - ex

Data Block (DB) – arrays, scalars

- **Goals:**
  - automatically synthesize bus matrix AND memory architectures
  - satisfy all throughput AND memory area constraints in design
  - primary objective -> minimize number of busses in matrix
  - secondary objective -> minimize memory area

# Outline

- Motivation
- Synthesis of Communication and Memory Architectures
- Problem Formulation
- **Related Work**
- COSMECA Co-Synthesis Methodology
- Case Studies
- Conclusion and Future Work

# Related Work

- Plenty of work in the area of shared/hierarchical bus-based communication architecture synthesis
  - Lahiri et al. [ICCAD 2000], Lyonnard et al. [DAC 2001]
  - Pinto et al. [DAC 2002], Ryu et al. [DATE 2003]
  - **Pasricha et al. [ASPDAC 2005], [DAC 2005]**

- However, very few research efforts have looked at bus matrix synthesis
  - Ogawa et al. [DATE 2003] proposed a transaction based simulation environment to explore and design a bus matrix
    - **need to manually specify topology, arbitration scheme, memory mapping → too time consuming**
  - Murali et al. [DATE 2005] proposed an automated application specific bus matrix synthesis approach
    - **work focuses on topology synthesis only**
  - **Pasricha et al. [ASPDAC 2006]** proposed an automated application specific bus matrix synthesis approach
    - **synthesize BOTH topology and parameter values**

# Related Work

- Only a few approaches have attempted to simultaneously explore memory and communication subsystems
  - Grun et al. [DATE 2002] considered connectivity topology in conjunction with memory exploration
    - **for simple processor-memory systems**
  - Kim et al. [CODES+ISSS 2004] deal with bus topology and arbitration exploration
    - **to determine best memory port-to-bus mapping**
  - Other memory synthesis approaches use static estimates of communication
    - **e.g. Knudsen et al. [CODES 1998]**

- However, these techniques only address a small part of the problem
  - don't really focus on co-synthesis of memory and communication architectures

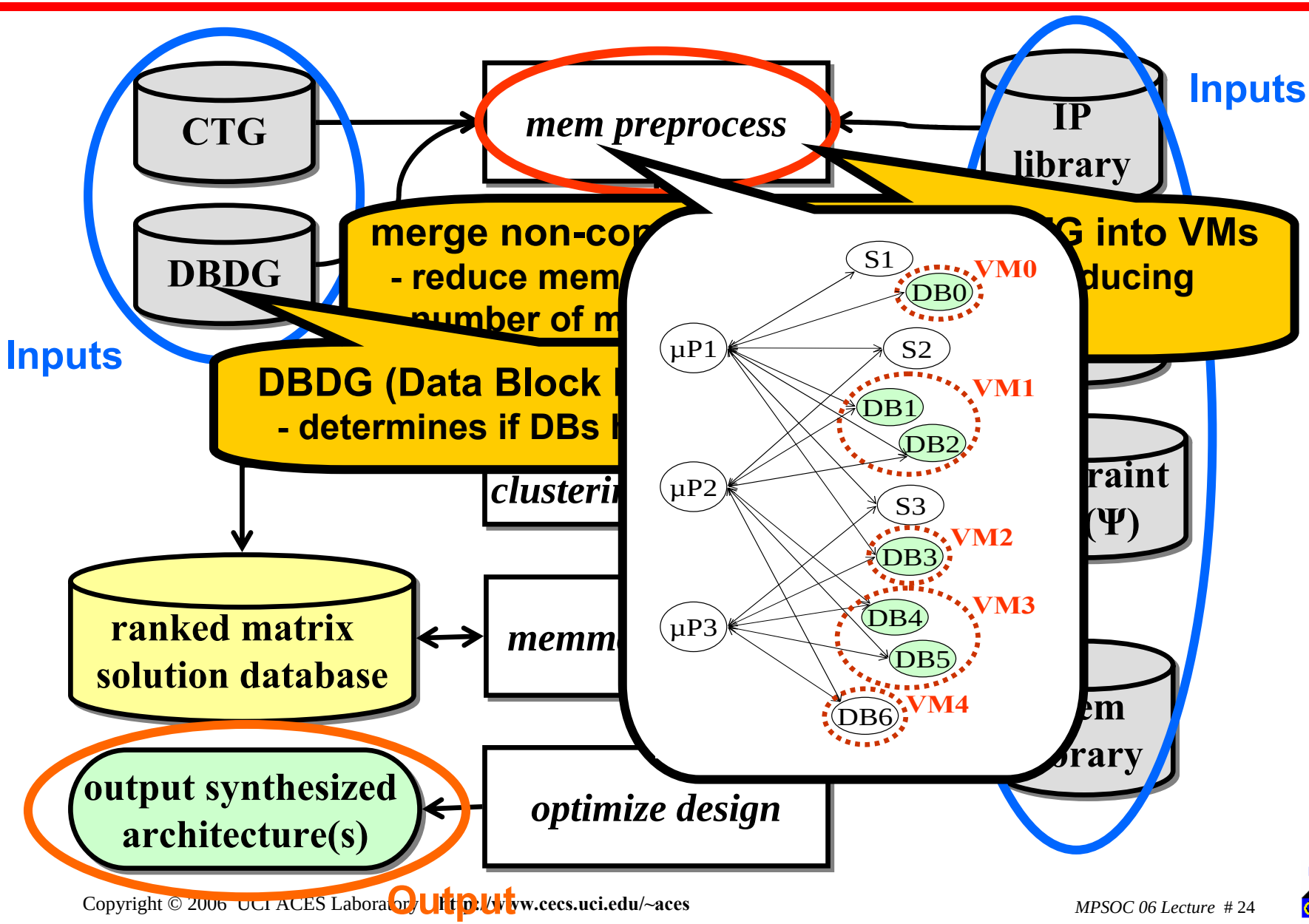
- ***COSMECA*** is a novel co-synthesis approach which attempts to automatically
  - synthesize bus matrix topology AND parameter values
  - simultaneously determine a mapping of data to physical memories while also deciding number, size, ports and type of memories

# Outline

- Motivation
- Synthesis of Communication and Memory Architectures
- Problem Formulation
- Related Work
- **COSMECA Co-Synthesis Methodology**
- Case Studies
- Conclusion and Future Work

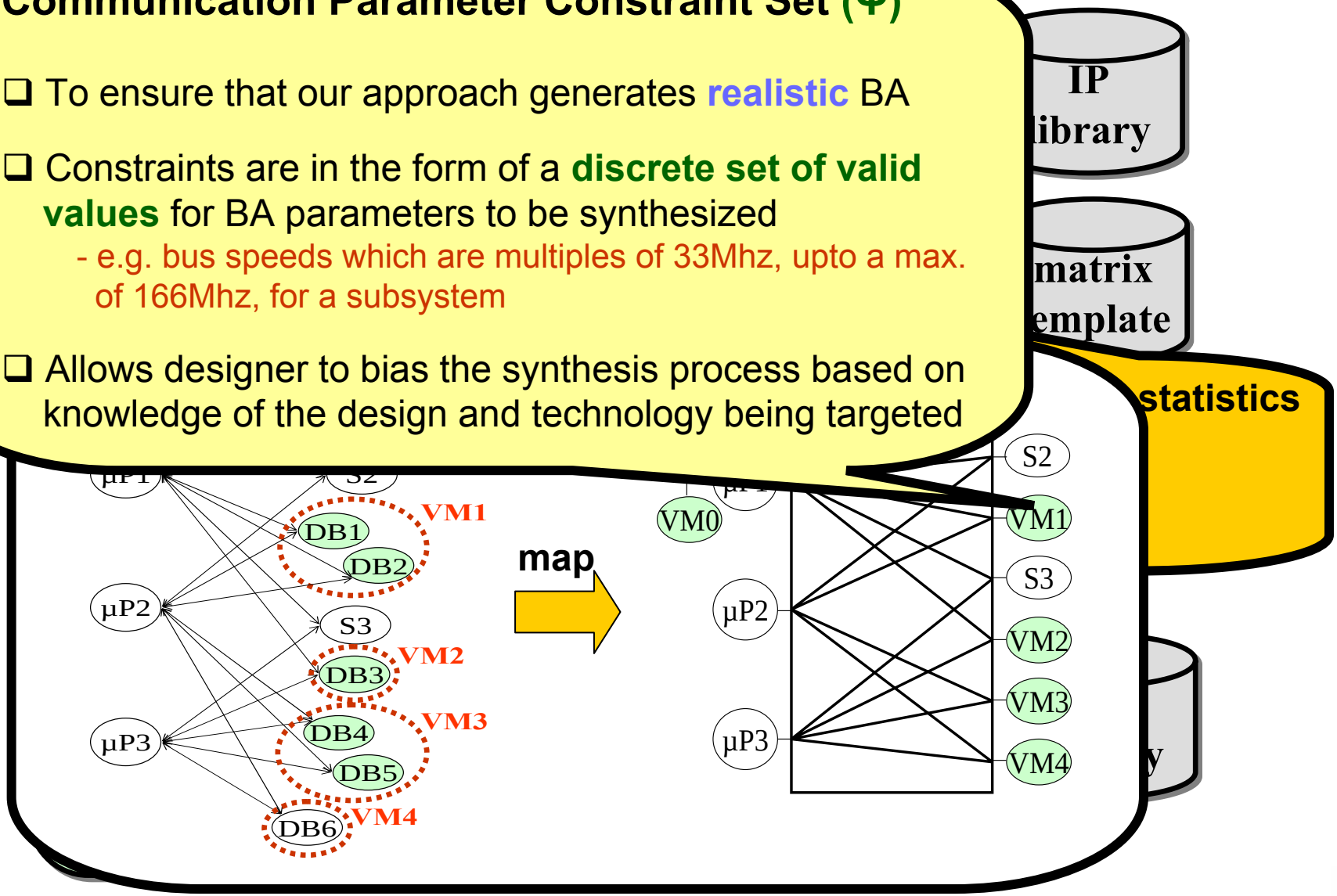
# COSMECA Co-Synthesis Flow



# COSMECA Co-Synthesis Flow

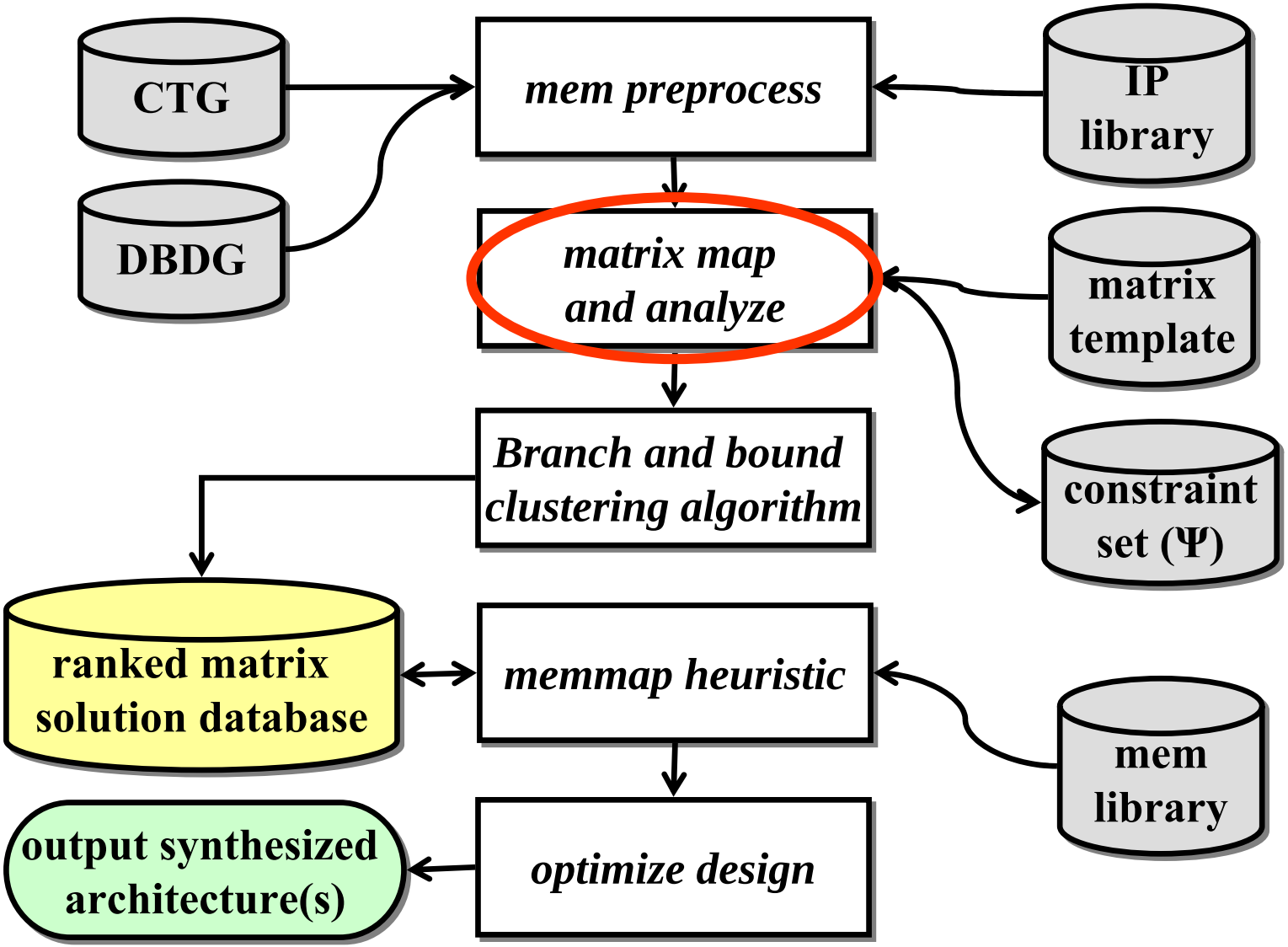
**Communication Parameter Constraint Set (Ψ)**

- To ensure that our approach generates **realistic** BA
- Constraints are in the form of a **discrete set of valid values** for BA parameters to be synthesized
  - e.g. bus speeds which are multiples of 33Mhz, upto a max. of 166Mhz, for a subsystem
- Allows designer to bias the synthesis process based on knowledge of the design and technology being targeted




 **http://www.cecs.uci.edu/~aces**

# COSMECA Co-Synthesis Flow

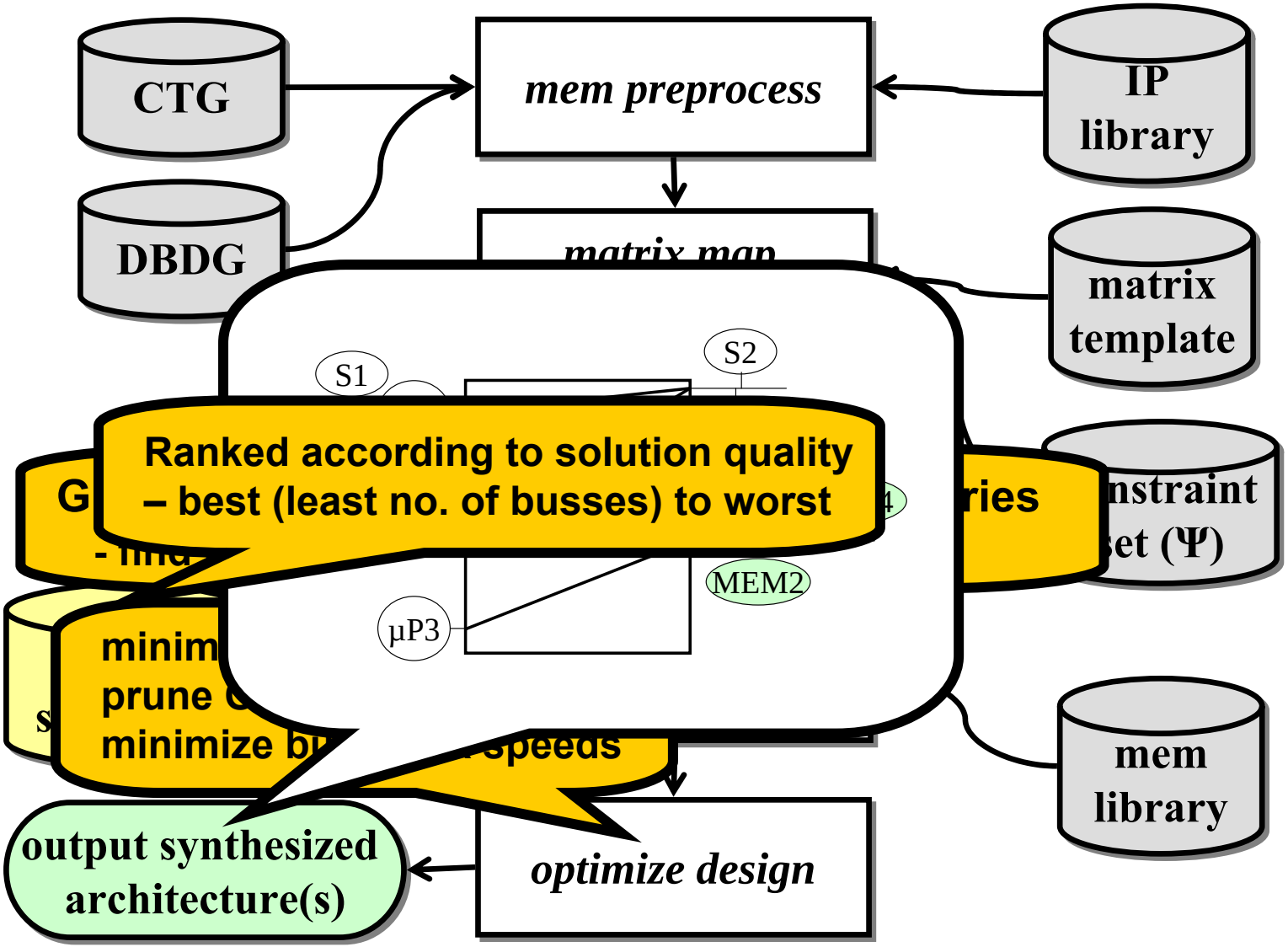


# Branch and Bound Clustering Algorithm

- Goal: cluster slave modules to minimize matrix cost
- Start by clustering two slave clusters at a time
  - Initially, each slave cluster has only one slave
- However, the total number of clustering configurations possible for n slaves is $\boldsymbol{(n! \times (n-1)!)/2^{(n-1)}}$
  - Extremely large number for even medium sized SoCs!
- Solution: use a *Bounding* function for pruning
  - *Called after every clustering operation*
  - *Uses lookup table to discard duplicate clustering ops*
  - *Discards non-beneficial clustering (i.e. no savings in no. of busses)*
  - *Discards incompatible clustering*
    - ***e.g. mergers of busses with conflicting bus speeds***
  - *Discards clustering which violates b/w requirements*

# COSMECA Co-Synthesis Flow




 http://www.cecs.uci.edu/~aces

# Outline

- Motivation
- Synthesis of Communication and Memory Architectures
- Problem Formulation
- Related Work
- COSMECA Co-Synthesis Methodology
- **Case Studies**
- Conclusion and Future Work

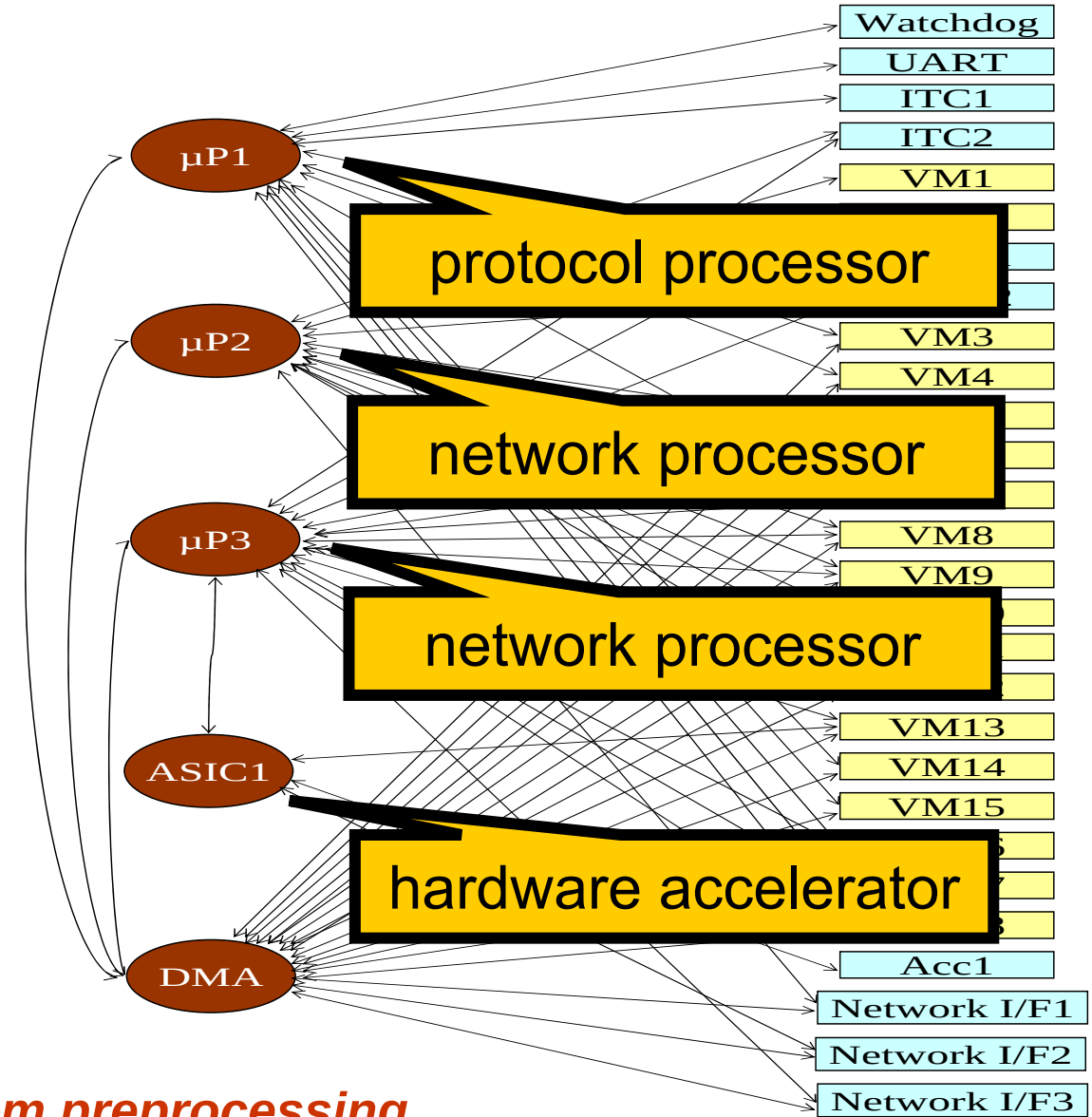
# Case Studies

- To evaluate effectiveness of our co-synthesis approach, we applied it to 4 MPSoC applications from networking domain
  - PYTHON, SIRIUS – variants of existing industrial strength applications
  - VIPER2, HNET8 – larger systems derived from next-gen MPSoCs

## Number of cores in MPSoC applications

| Applications | Processors | Masters | Slaves |
|---|---|---|---|
| ***PYTHON*** | 2 | 3 | 8 |
| ***SIRIUS*** | 3 | 5 | 10 |
| ***VIPER2*** | 5 | 7 | 14 |
| ***HNET8*** | 8 | 13 | 17 |

# CTG for SIRIUS MPSoC




CTG after mem preprocessing

# SIRIUS MPSoC Design Goals and Constraints

## Design Goal: Throughput Constraint Paths (TCPs)

| IP cores in Throughput Constraint Path (TCP) | TCP constraint |
|---|---|
| µP1, VM3, VM4, DMA, VM16, VM17, VM18 | 640 Mbps |
| µP1, VM5, VM6, VM14, VM15, DMA, Network I/F2 | 480 Mbps |
| µP2, Network I/F1, VM8, VM9 | 5.2 Gbps |
| µP2, VM10,VM11,VM12, DMA, Network I/F3 | 1.4 Gbps |
| A | |
| µ | |

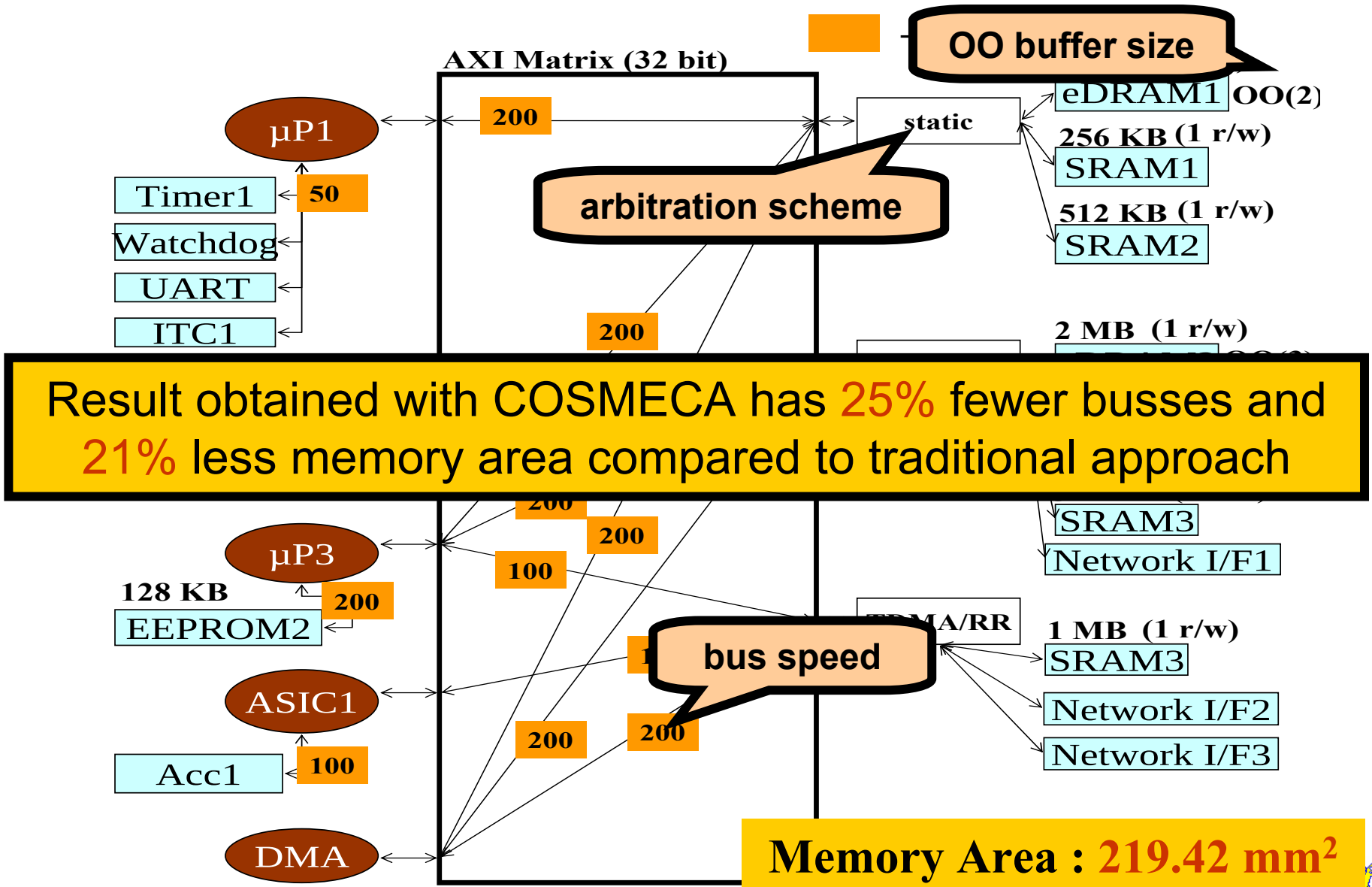
**Target bus matrix architecture: AMBA3 AXI bus matrix**
**Memory Area Constraint: 225 mm²**

## Communication Parameter Constraint Set

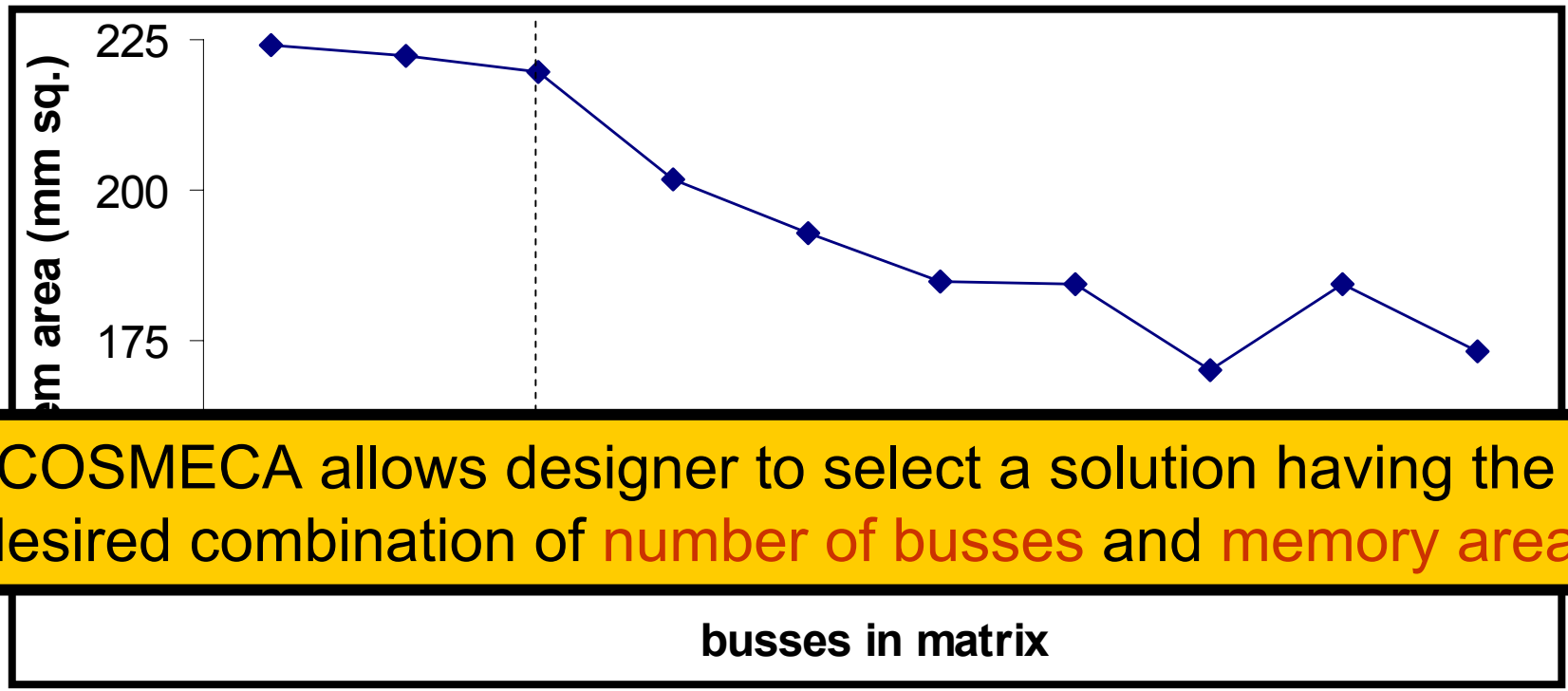
| Set | Values |
|---|---|
| ***bus speed*** | 25, 50, 100, 200, 300, 400 |
| ***arbitration strategy*** | static, RR, TDMA/RR |
| ***OO buffer size*** | 1 – 8 |
| ***mem mapping*** | VM16,VM17=>*DRAM*; VM1,VM2=>*EEPROM* |

# SIRIUS Synthesized Output



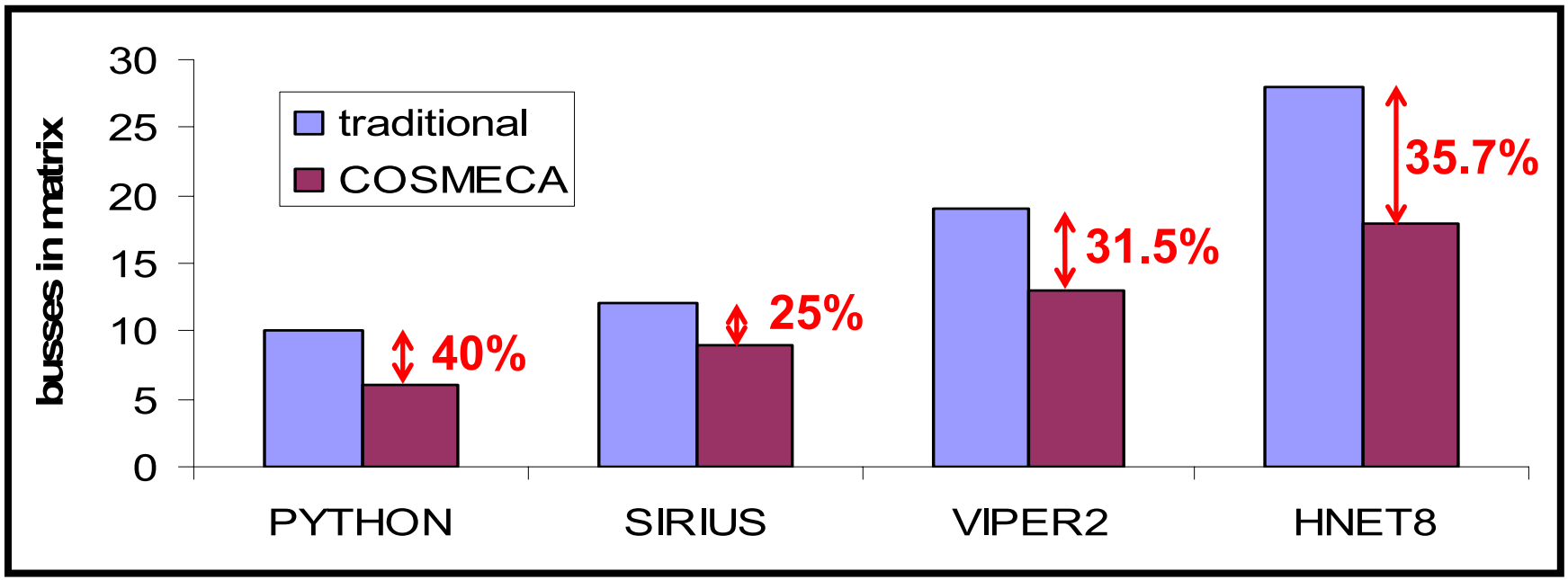
# Solution Tradeoffs for SIRIUS




COSMECA allows designer to select a solution having the desired combination of number of busses and memory area

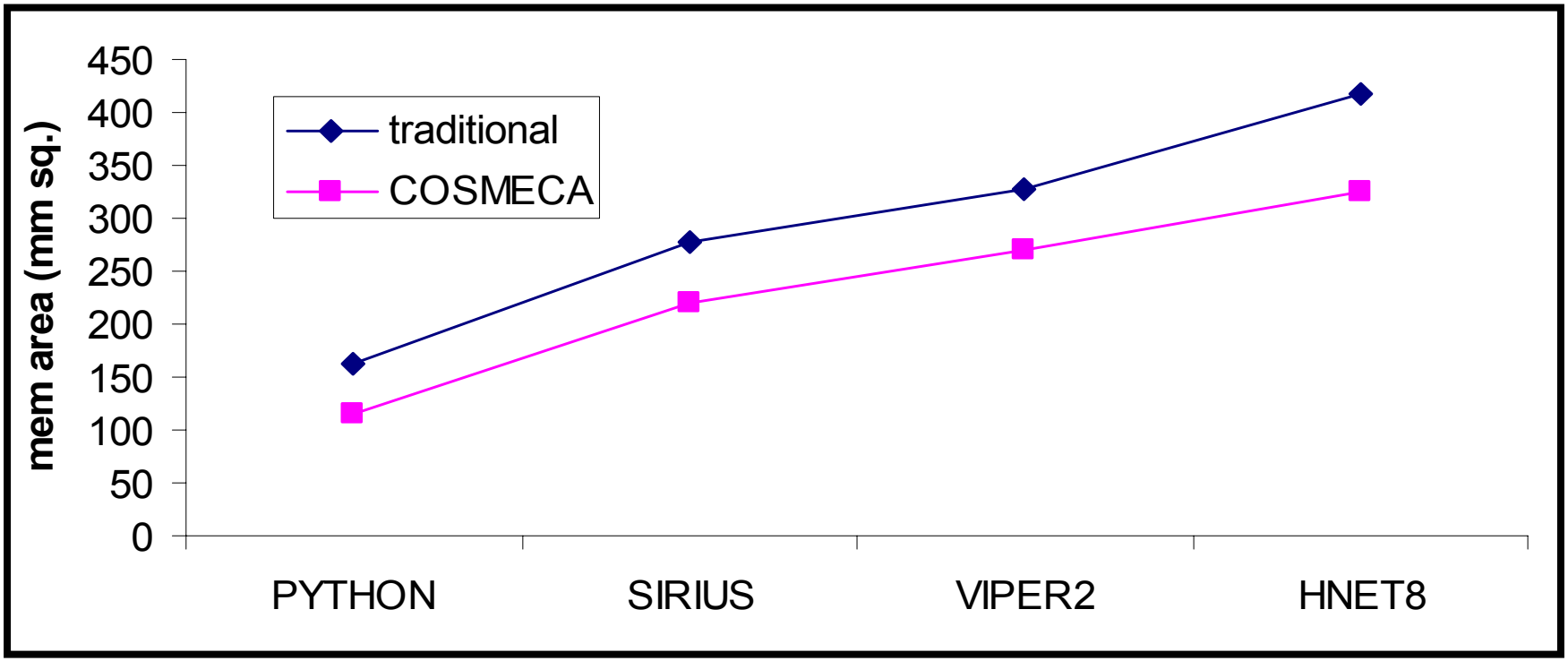
variation in memory area and number of busses for the ten best solutions (*N=10*)

# Bus Matrix Density Comparison




COSMECA saves **25 – 40%** in the number of busses in the matrix compared to the traditional approach

# Memory Area Comparison




COSMECA saves **17 – 29%** in memory area compared to the traditional approach

# Outline

- Motivation
- Synthesis of Communication and Memory Architectures
- Problem Formulation
- Related Work
- COSMECA Co-Synthesis Methodology
- Case Studies
- **Conclusion and Future Work**

# Conclusion

- **We presented an approach for the automated co-synthesis of memory and communication architectures (*COSMECA*)**

- ***COSMECA*** couples the decision making process during memory and communication architecture synthesis, to generate better solutions
  - **lower number of busses**
  - **lower memory area**
  - **lower cost**

- **Results of applying *COSMECA* to 4 industrial strength MPSoC applications indicate a saving of**
  - **upto 40% in number of busses, and**
  - **upto 29% in memory area**

# Future Work

- Incorporating cache customization in the memory synthesis sub-framework
- Crossbar/matrix generation for STBus, CoreConnect standards, in the communication architecture synthesis sub-framework
- Incorporating power as another metric to guide COSMECA co-synthesis methodology

# Acknowledgements


- COSMECA research done jointly with
  - PhD student Sudeep Pasricha

- Sponsors
  - Conexant, Inc. and UC MICRO program
  - NSF
  - SRC

Publications, etc:

- http://www.cecs.uci.edu/~aces

# Thank you!

# Related Publications


**[1]** S. Pasricha, N. Dutt, M. Ben-Romdhane, "Extending the Transaction Level Modeling Approach for Fast Communication Architecture Exploration, **DAC 2004**

**[2]** S. Pasricha, N. Dutt, M. Ben-Romdhane, "Fast Exploration of Bus-based On-Chip Communication Architectures", **CODES+ISSS 2004**

**[3]** S. Pasricha, N. Dutt, M. Ben-Romdhane, "Automated Throughput-driven Synthesis of Bus-based Communication Architectures", **ASPDAC 2005**

**[4]** S. Pasricha, N. Dutt, E. Bozorgzadeh, M. Ben-Romdhane, "Floorplan-aware Automated Synthesis of Bus-based Communication Architectures", **DAC 2005**

**[5]** S. Pasricha, N. Dutt, M. Ben-Romdhane, "Constraint-Driven Bus Matrix Synthesis for MPSOCs", **ASPDAC 2006**

**[6]** S. Pasricha, N. Dutt "COSMECA: Application Specific Co-Synthesis of Memory and Communication Architectures for MPSoC ", **DATE 2006**