Design Space Exploration for a Coarse Grain Accelerator

Farhad Mehdipour, Hamid Noori, Morteza Saheb Zamani*,
Koji Inoue, Kazuaki Murakami

Kyushu University, Fukuoka, Japan
*Amirkabir University of Technology

Outline

- Introduction
- Problem Definition and Basic Concepts
- Hybrid DES Approach for Designing RAC
- Case study: Designing RAC for an Extensible Processor
- Conclusion

Outline

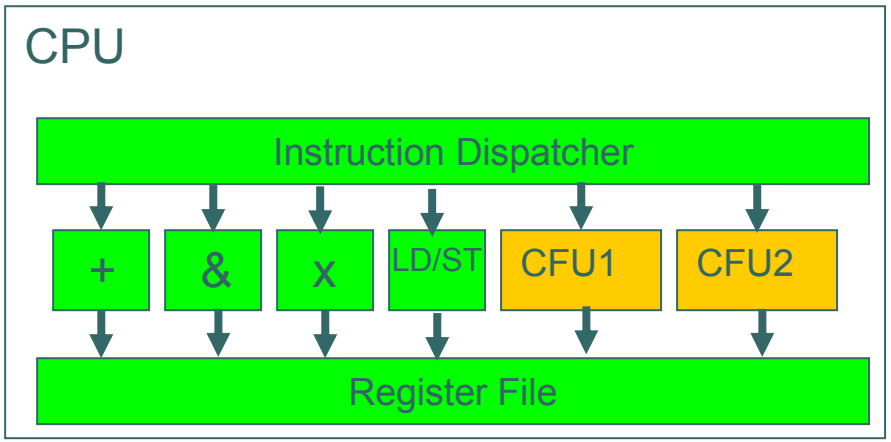
- Introduction
- Problem Definition and Basic Concepts
- Hybrid DES Approach for Designing RAC
- Case study: Designing RAC for an Extensible Processor
- Conclusion

Designing Embedded Systems

- Embedded Microprocessors
- Application-Specific Integrated Circuits (ASICs)
- Application-Specific Instruction set Processors (ASIPs)
- Extensible Processors

LD/ST: Load / Store

CFU: Custom Functional Unit

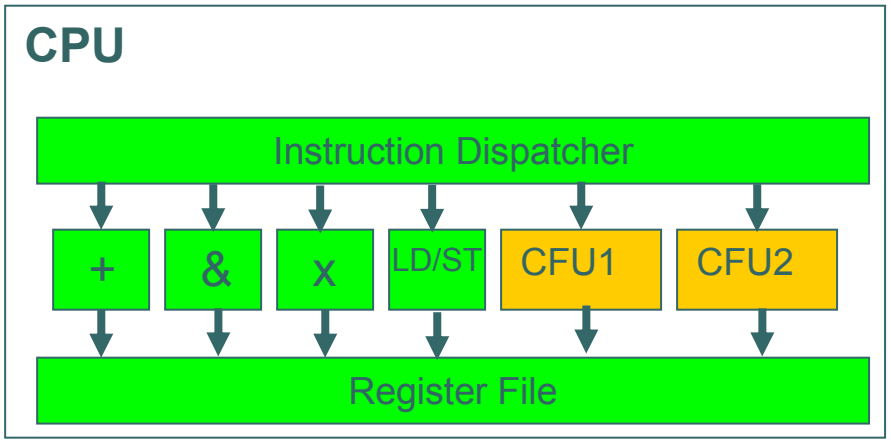


Extensible Processors

- Improving the performance and energy efficiency of the embedded processors
- Maintaining compatibility and flexibility.
  - Using an accelerator for accelerating frequently executed portions
- Accelerator implementations
  - reconfigurable fine/coarse grain hw
  - custom hardware (such as ASIP or Extensible Processors)

LD/ST: Load / Store

CFU: Custom Functional Unit



Custom Instructions

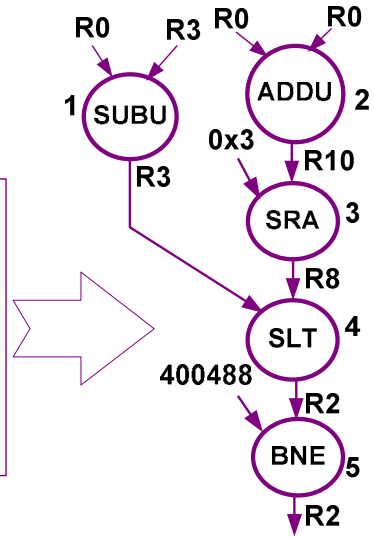
- Instruction set customization ←→ hardware/software partitioning (Identifying critical segments in applications)

- Custom Instructions (CIs) are
  - extracted from critical segments of an application and
  - executed on a Custom Functional Unit (CFU)
- Critical segments→ Most frequently executed portions of the applications

A CI can be represented as a DFG

A Custom Instruction

```
1: SUBU   R3, R0, R3
2: ADDU   R10, R0, R0
3: SRA    R8, R10, 0x3
4: SLT    R2, R3, R8
5: BNE    R0,400488, R2
```

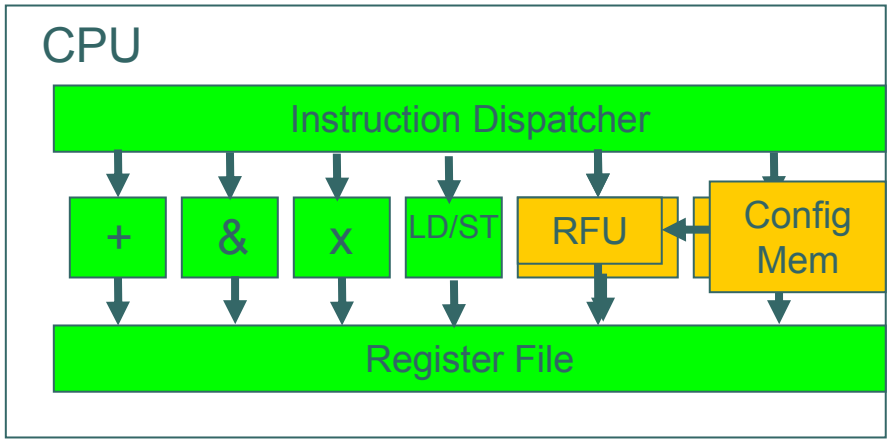


Reconfigurable Processors

- Adding and generating custom instructions after fabrication
- Using a reconfigurable functional unit (RFU) instead of custom functional unit

CFU: Custom Functional Unit

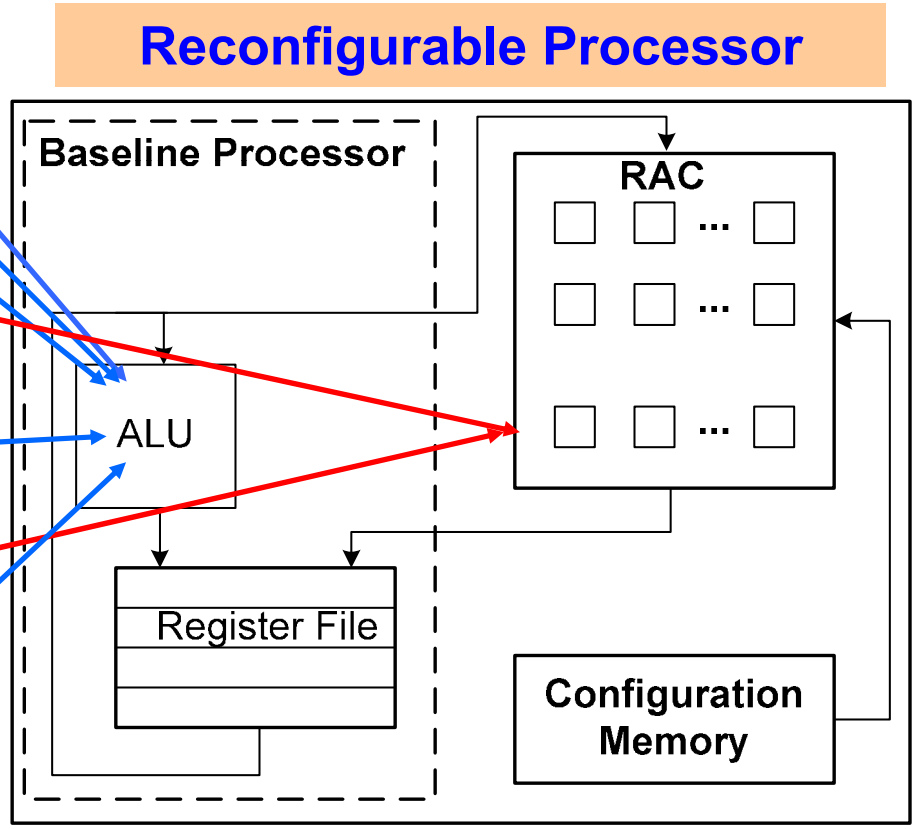
RFU: Reconfigurable Functional Unit



How a Reconfigurable Processor Works

```
400680    subiu    $25,$25,1
400688    lbu      $13,0($7)
400690    lbu      $2,0($4)
400698    sll      $2,$2,0x18
4006a0    sra      $14,$2,0x18
4006a8    addiu    $4,$4,1
4006b0    srl      $8,$2,0x1c
4006b8    sll      $2,$8,0x2
4006c0    addu     $2,$2,$25
4006c8    lw       $2,0($2)
4006d0    xori     $13,$13,1
4006d8    addu     $10,$10,$2
400680    subiu    $25,$25,1
400698    sll      $2,$2,0x18
4006a0    sra      $14,$2,0x18
400688    lbu      $13,0($7)
4006e0    bgez     $10,4006f0
.
.
.
```

Hot Basic Block



GPP: General Purpose Processor

RAC: Reconfigurable Accelerator

Outline

- Introduction
- Problem Definition and Basic Concepts
- Hybrid DES Approach for Designing RAC
- Case study: Designing RAC for an Extensible Processor
- Conclusion

Designing a Reconfigurable Accelerator (RAC)

- Multitude of design parameters
  - e.g. number of functional units, input/output ports, type of functional units and etc.
- Several design parameters
  - high complexity of the RAC design
  - the requirements for a methodological approach
- A major challenge
  - finding the right balance between the different quality requirements (e.g. speedup, area, energy consumption)

Traditional Design Process

- Describing a reference model
- Verifying the model functional correctness
- Obtaining a rough estimates of performance
- Manual or semi-automatic generation of several alternative designs
- Choosing the most suitable design based on various performance metrics

Design Space Exploration

- Design Space Exploration (DSE)
  - the process of analyzing several “functionally equivalent” implementation alternatives to identify an optimal solution
  - Can become too computationally expensive

- Example:
  - a design with four tasks on an architecture with three processing modules and each have four possible configurations results in 500 design alternatives

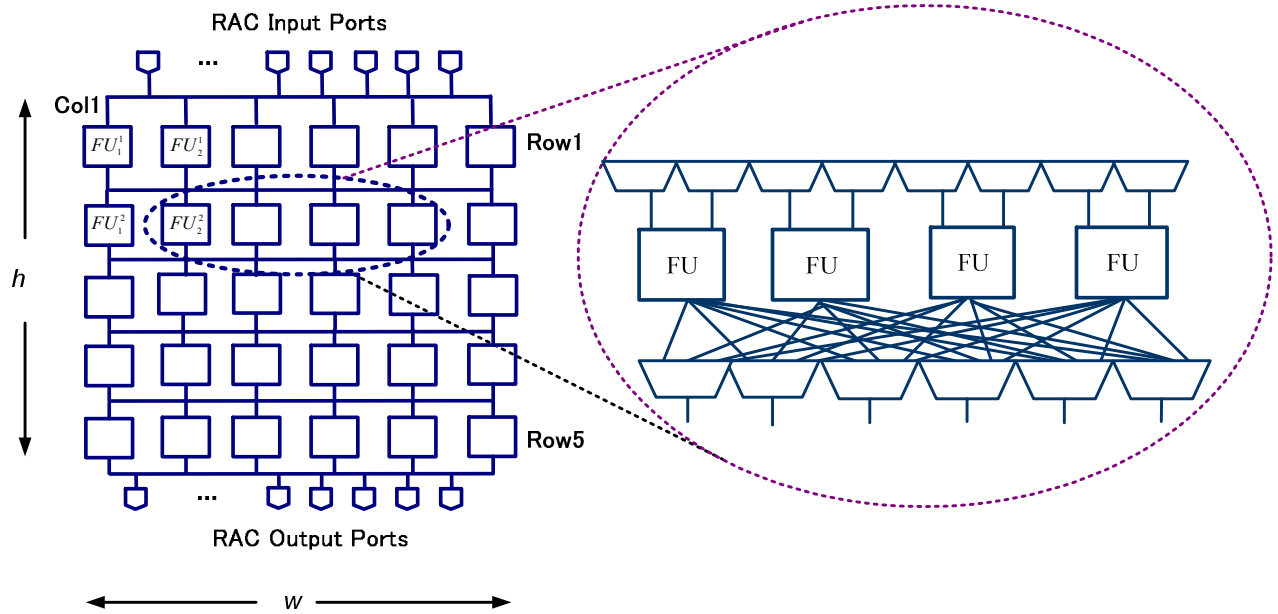
Our Approach: Hybrid (Analytical & Quantitative) approach which drastically reduces the design time & effort

Assumptions

- **RAC**
  - **a matrix of FUs**
  - **the width/height equal to *w/h***
  - **basically has a combinational logic**
- **FUs in RAC are fully connected**
  - **Except the lack of connections from lower to upper rows**

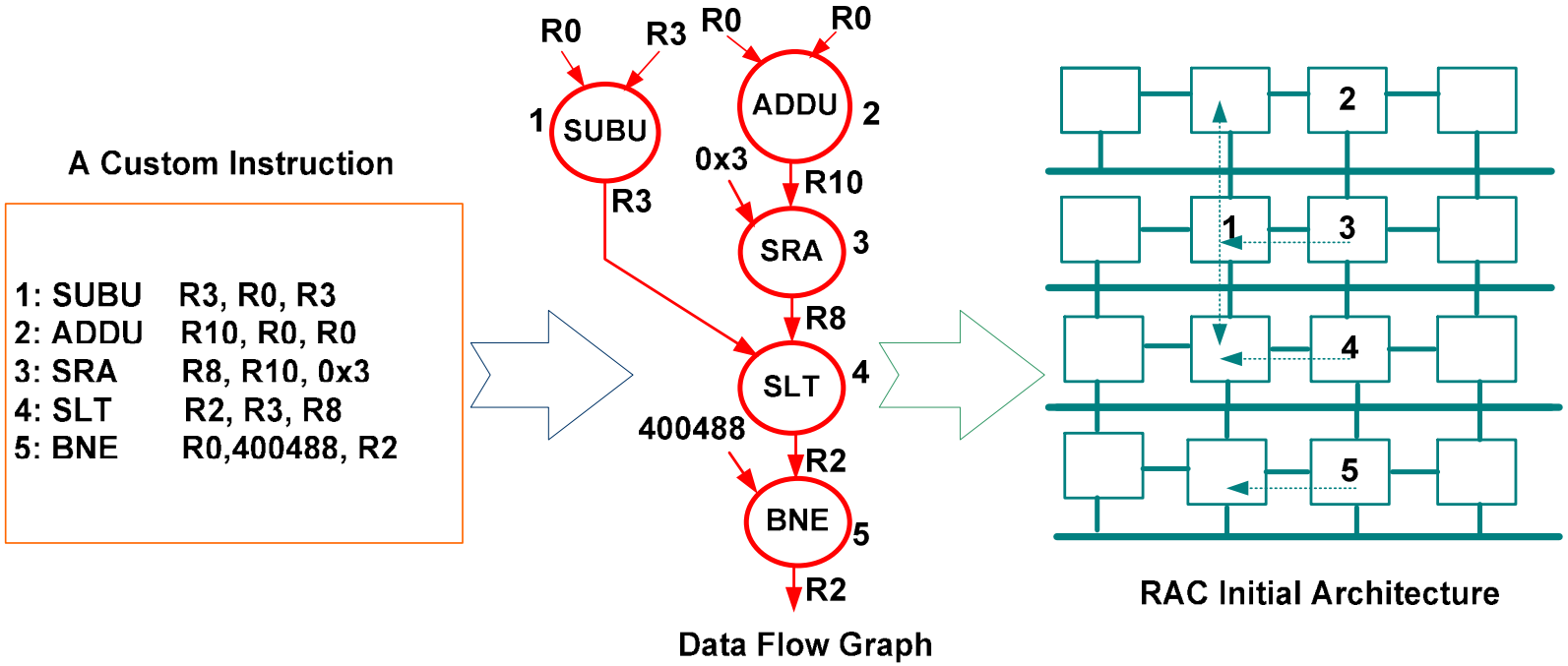
Basic Elements

- Functional Units (logic resources)
- Multiplexers (routing resources)



Assumptions

- CIs (DFGs) are mapped onto the RAC and executed at runtime



Outline

- Introduction
- Problem Definition and Basic Concepts
- Hybrid DES Approach for Designing RAC
- Case study: Designing a RAC for an Extensible Processor
- Conclusion

The Problem

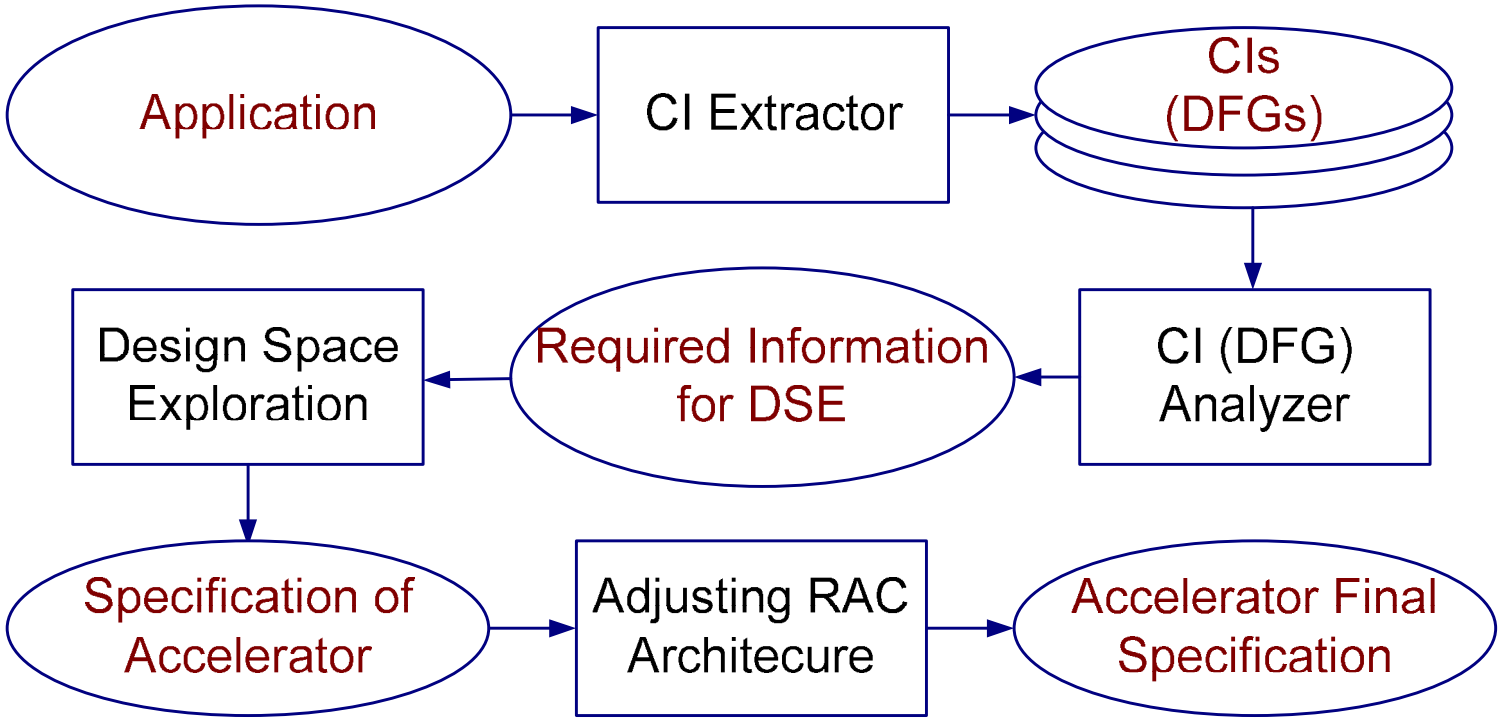
- Determining a RAC while optimizing Speedup & Area
- Design parameters
  - the RAC's width and height
  - the number of FUs,
  - the number of input and output ports

- Speedup

$$Speedup = \frac{TotalClockCyclesOnBase\,\mathrm{Pr}\,ocessor}{TotalClockCyclesOnRAC}$$

Design Methodology- Tool Chain

Our Approach: A Hybrid (Analytical & Quantitative) DSE Approach



Problem Formulation

The number of CCs required for executing *DFG(i,j)* on the base processor

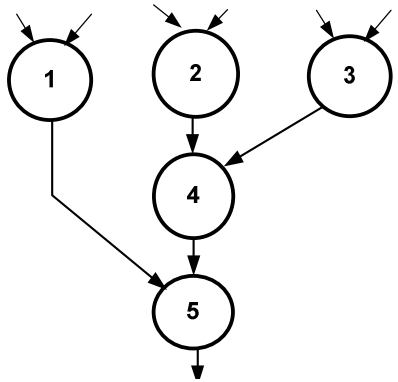
$$CC(CPU) = \sum_{i=1}^{W}\sum_{j=1}^{H} CC_j^i(CPU) \times \alpha_j^i$$

the fraction of all DFGs with the width of *i* and height of *j (DFG(i,j))*

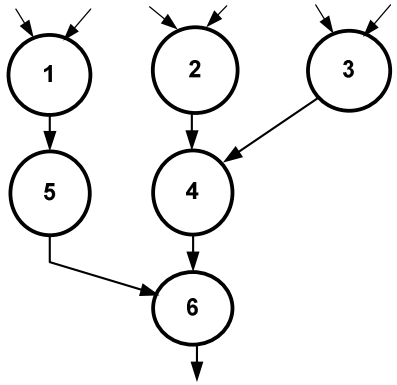
$\alpha_3^4 = 7\%$: percentage of execution time concerns to all DFGs with the width of 4 and height of 3 is 7%.

$$CC_j^i(CPU) = \gamma_j^i(FU) \times i \times j$$

Average number of instructions in all *DFG(i,j)s*



$\gamma_3^3 = 5$



$\gamma_3^3 = 6$

Problem Formulation

$$CC(RAC_h^w) = \sum_{i=1}^{W}\sum_{j=1}^{H} CC_j^i(RAC_h^w) \times \alpha_j^i$$

the number of Clock Cycles for executing *DFG(i,j)* on a *RAC (w,h)*

When one or both dimensions of a DFGs are greater than RAC's dimensions:

- Temporal Partitioning: Divides a DFG into time exclusive smaller DFGs
- Reconfiguration overhead time for loading subsequent partitions of a DFG from the configuration memory onto the RAC

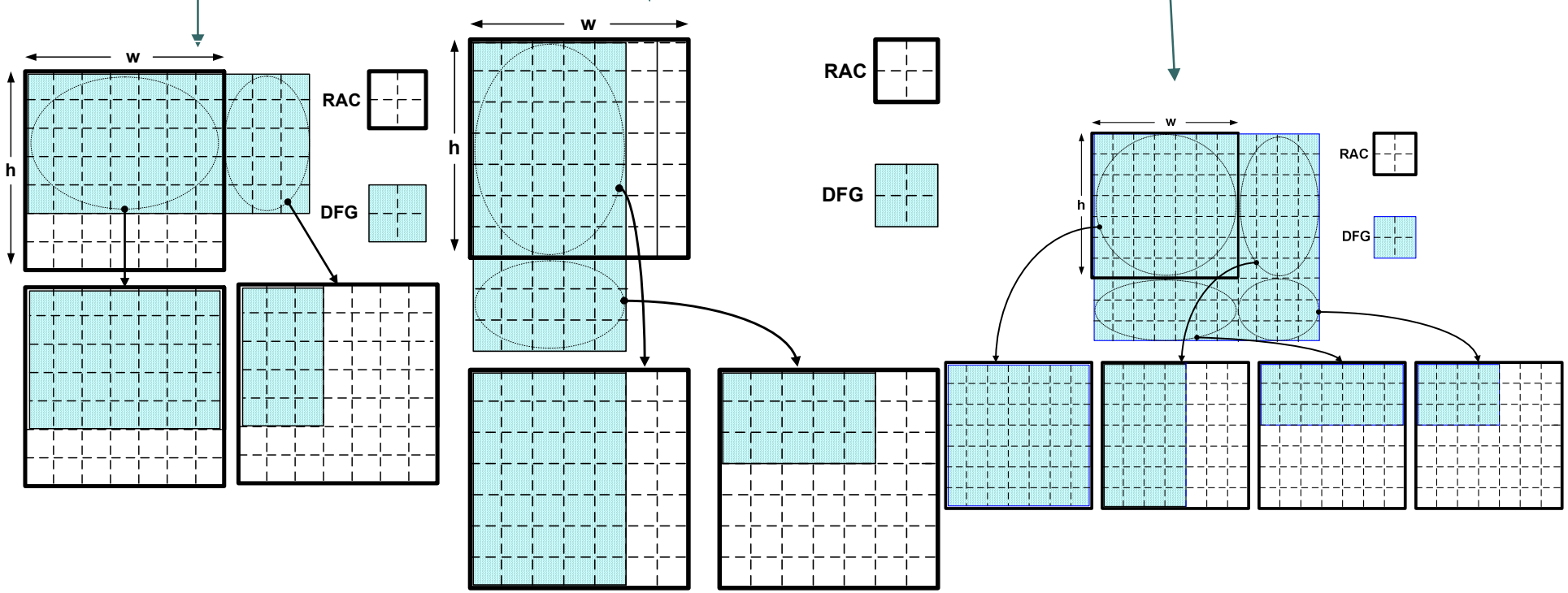
Problem Formulation

$$1) i \le w, j \le h \longrightarrow CC_j^i(RAC_h^w) = D(RAC_h^w)$$

$$2) i > w, j \le h$$

$$3) i \le w, j > h$$

$$4) i > w, j > h$$



Delay of RAC

Delay of FU*(k,i)*

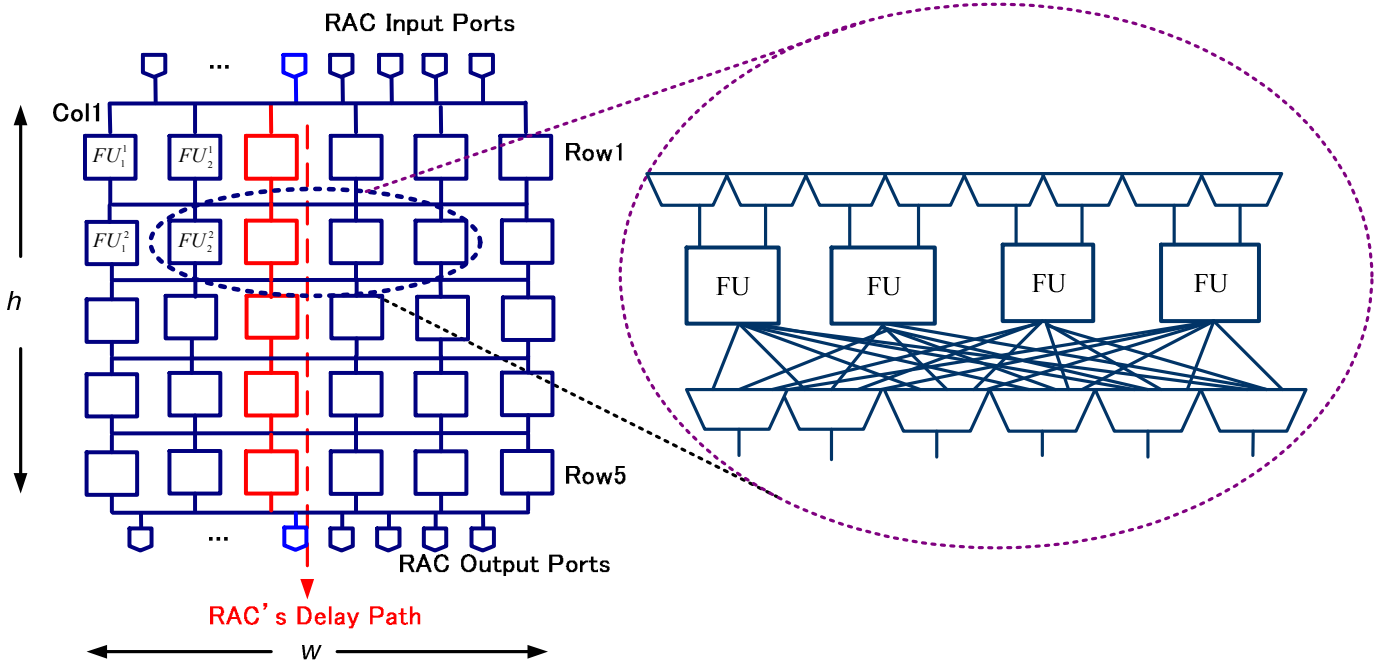
Delay of MUX*(k,i)*

$$D(RAC_h^w) = \sum_{i=1}^{h} D(FU_i^k) + \sum_{i=1}^{h-1} D(MUX_i^k), k \in \{1,\ldots,w\}$$

Delay of RAC(w,h)

$$D(MUX_i^k) = \text{delay of mux}(\ 2^{s_j^k} \text{ to } 1)$$

$$s_j^k = \left\lceil \log_2^{m_j^k} \right\rceil \qquad m_j^k = (j-1)\times w + (w-1)$$



Optimization Problem

$$S(RAC_h^w) = \frac{CC(CPU)}{CC(RAC_h^w)} = \frac{\sum_{i=1}^{W}\sum_{j=1}^{H} CC_j^i(CPU)\times\alpha_j^i}{\sum_{i=1}^{W}\sum_{j=1}^{H} CC_j^i(RAC_h^w)\times\alpha_j^i}$$

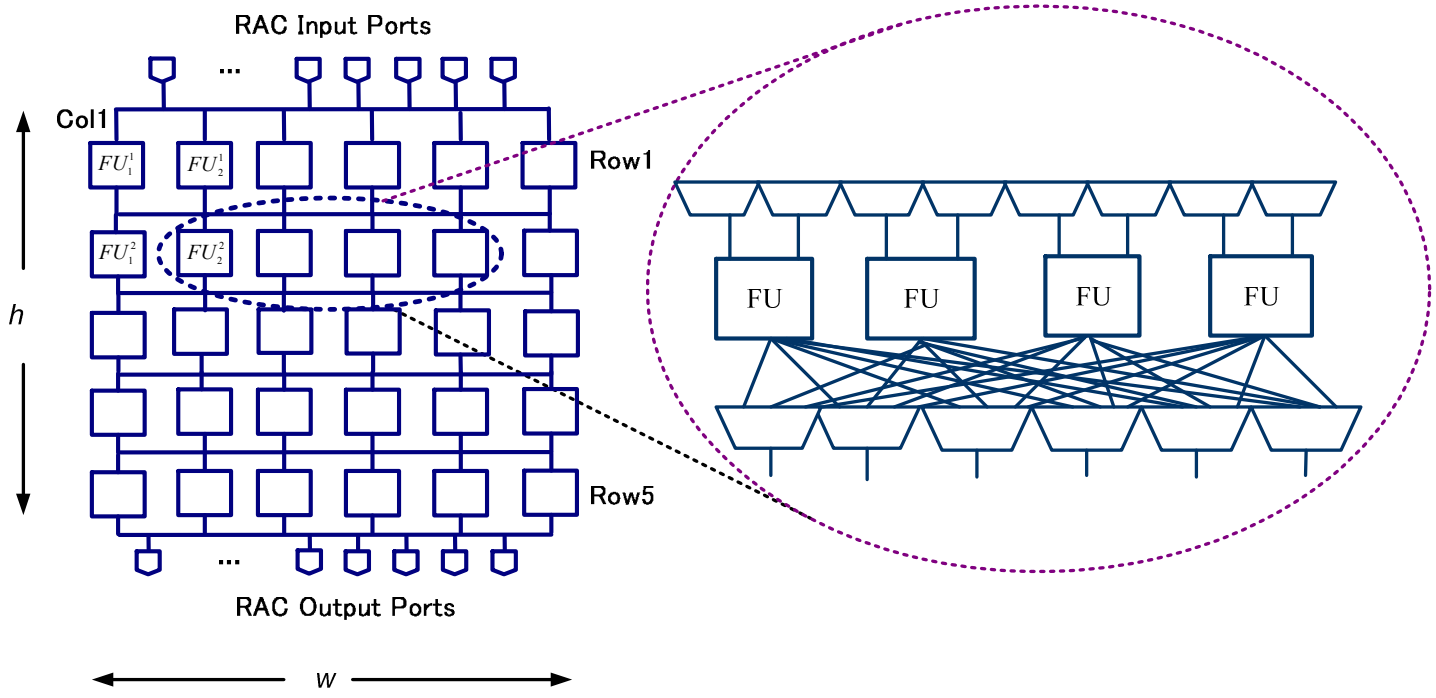
Objective: Maximize($S(RAC_h^w)$)

Area of RAC

Area is a secondary optimization goal

$$A(RAC_h^w) = \sum_{i=1}^{w}\sum_{j=1}^{h} A(FU_j^i) + 2 * \sum_{i=1}^{w}\sum_{j=1}^{h-1} A(MUX_j^i)$$

$$A(MUX_j^i) = \text{delay of mux(}\ 2^{s_j^i}\ \text{to 1)}$$



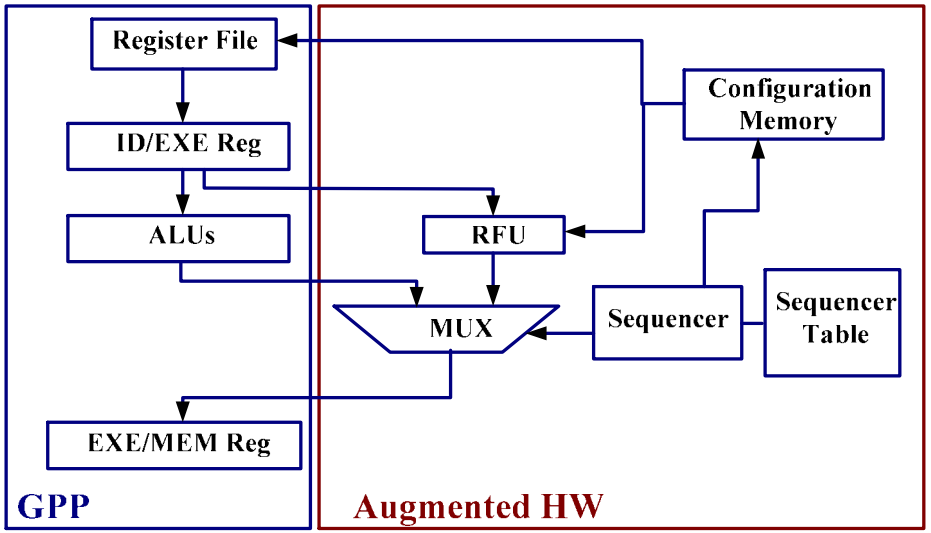
Outline

- Introduction
- Problem Definition and Basic Concepts
- Hybrid DES Approach for Designing RAC
- Case study: Designing a RAC for an Extensible Processor
- Conclusion

Designing RAC for a Reconfigurable Processor

- AMBER: a reconfigurable processor targeted for embedded systems
- Main components
  - a base processor (general RISC processor)
  - Sequencer and
  - a coarse-grained reconfigurable functional unit (RFU)

AMBER's Architecture



Quantitative Approach

- 22 applications of Mibench were attempted
- Applications were executed on Simplescalar and profiled
- Hot segments and DFGs were extracted from the applications
- No limitation in the RFU's initial architecture
- DFGs were mapped on the initial RFU architecture

Specification of the designed architecture for AMBER's RFU (Quantitative Approach)

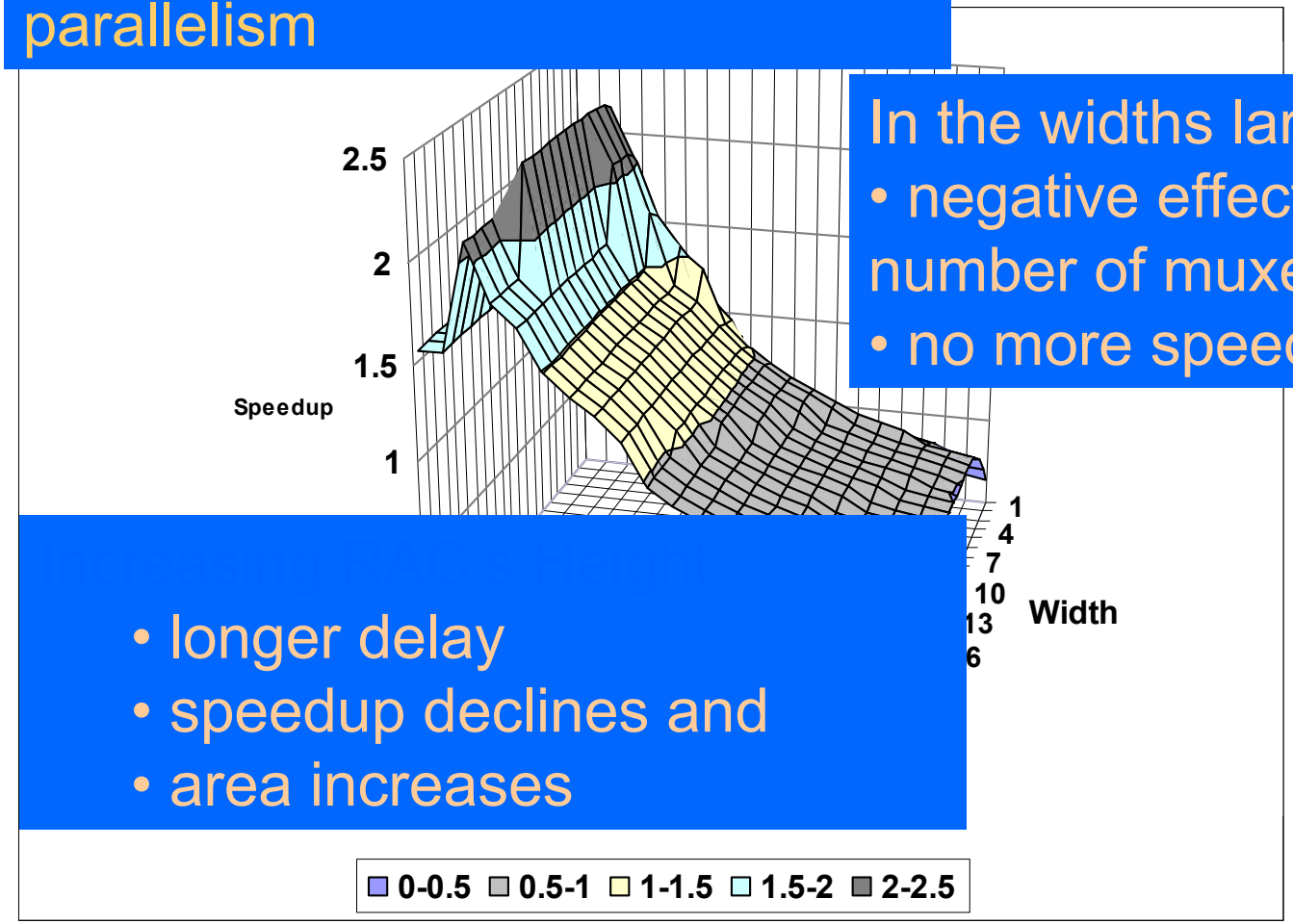
No. of FUs	16
Width/Height	6/5
No. of Input/Output ports	8/6
Number of FUs in each row	6, 4, 3, 2, 1

Hybrid DSE Approach

- FU and various size multiplexers
  - Synthesized using Hitachi 0.18um
  - Measuring delay and area
- DFGs are analyzed to extract required information quantitatively
- Reconfiguration penalty time: 1 clock cycle
- The base processor clock frequency: 166MHz

Speedup Evaluation

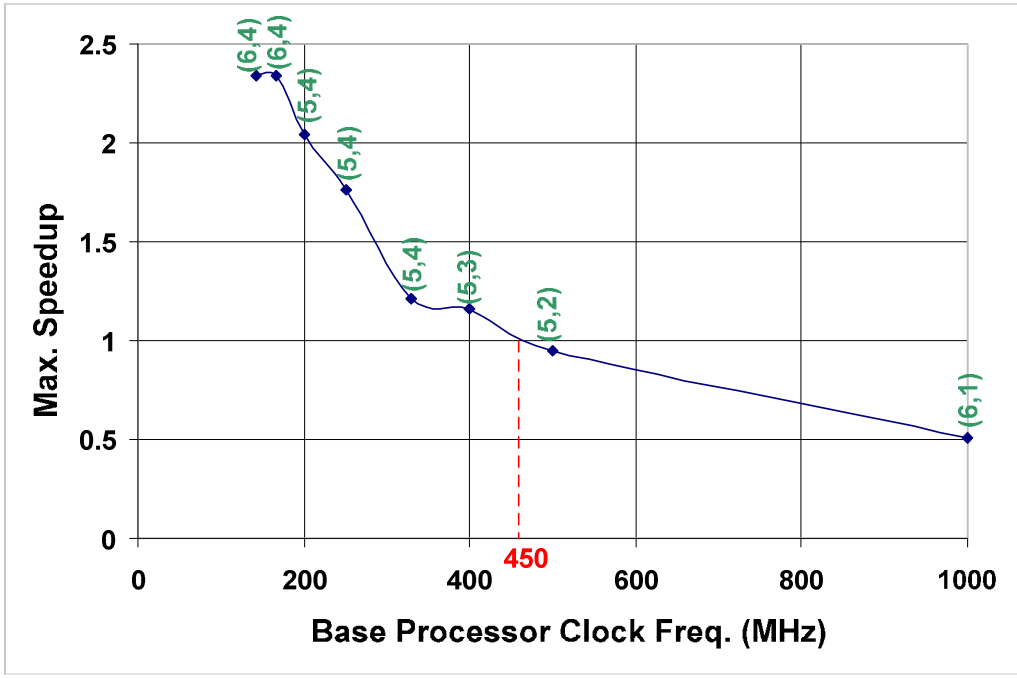
Increasing RAC's width→ more parallelism



In the widths larger than 6:
- negative effect of growing the number of muxes and their sizes
- no more speedup achievable

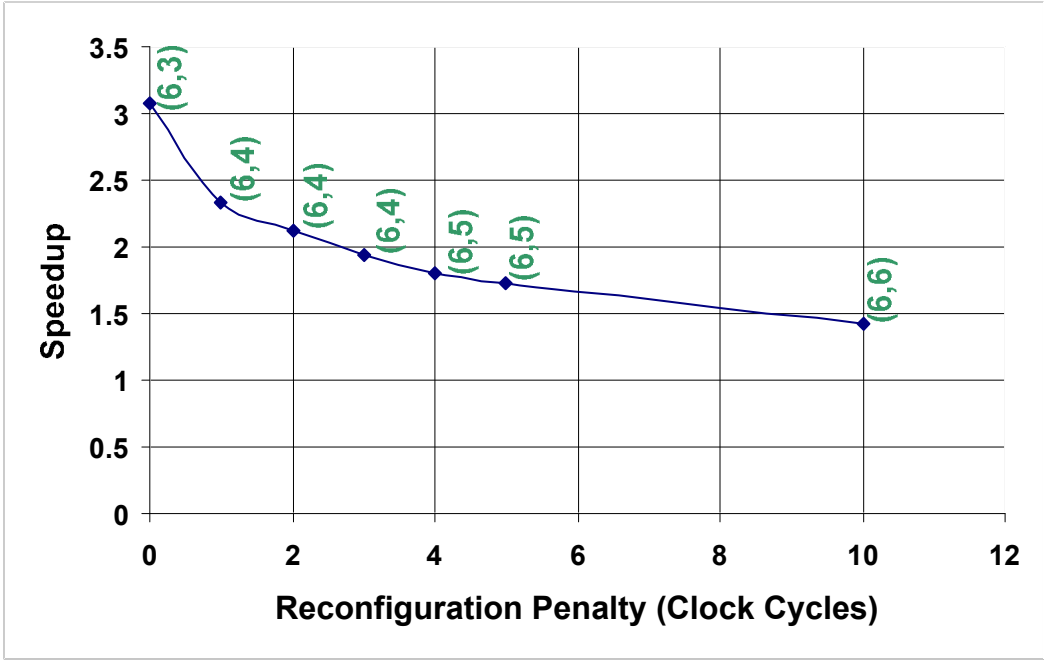
- longer delay
- speedup declines and
- area increases

Effect of the Base Processor Clock Frequency



- Increasing clock frequency→ Reduction in the maximum achievable speedup
- no more speedup in the clock frequencies more than 450MHz

Effect of the Reconfiguration Overhead Time



By increasing the reconfiguration overhead time

- The maximum achievable speedup degrades
- Height of RAC grows and longer DFGs are mappable

Comparison

	Design Method	*Design Time & Effort*	*Basic Design Parameters*	Flexibility*
Clark et al.	Quantitative & Synthesis	High	Mapping rate	Low
Ours (previous)	Quantitative	High	Mapping rate	Low
Yehia et al.	Synthesis & Simulation	Very High	No. of operations, inputs/outputs	Low
Ours (current)	Hybrid	Low	Speedup & Area	High

Outline

- Introduction
- Problem Definition and Basic Concepts
- Hybrid DES Approach for Designing RAC
- Case study: Designing a RAC for an Extensible Processor
- Conclusion

Conclusion

- Hybrid DSE approach
  - Uses realistic data from the attempted applications
  - Substantially reduces design time and designer efforts
  - Can be used for shrinking a large design space
  - Easily extendable to apply new design parameters
  - More suitable where the new applications are added

Thanks for your attention!