An Architecture Framework for an Adaptive Extensible Processor

Noori, Hamid
Graduate school of Information Science and Electrical Engineering, Kyushu University

An Architecture Framework for an Adaptive Extensible Processor

Hamid Noori

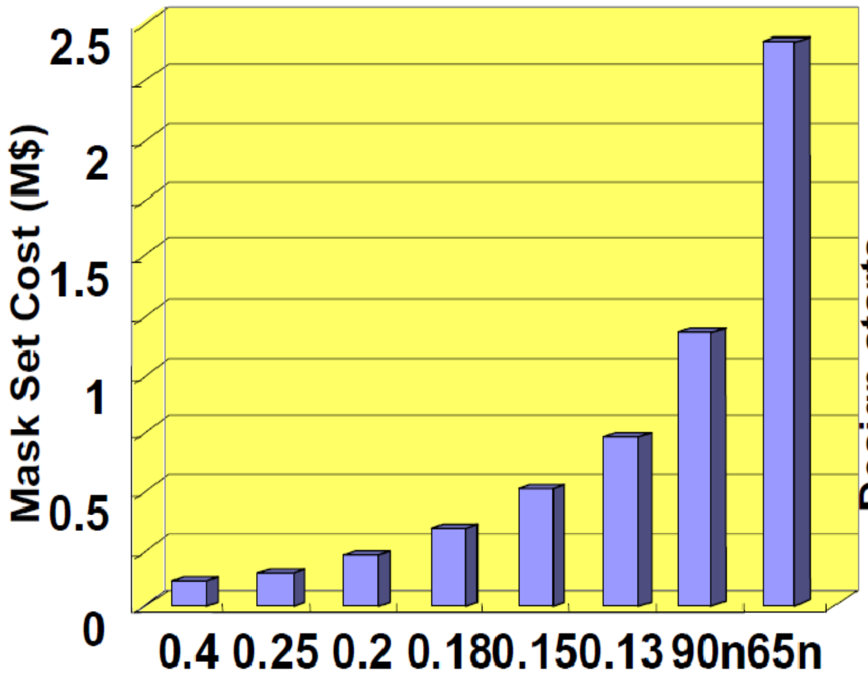
Graduate School of Information Science and Electrical Engineering, Department of Informatics
Kyushu University

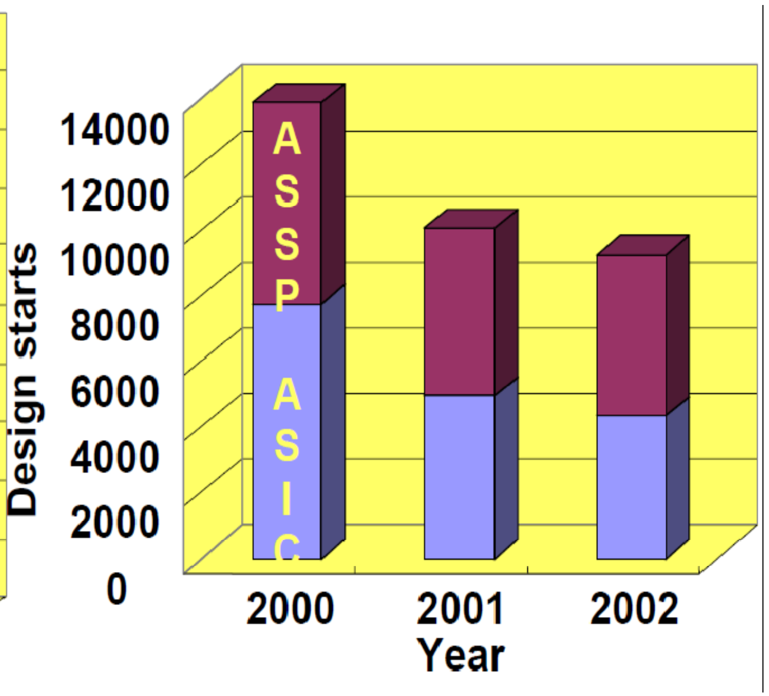
Outline

- Motivations
- Goal
- An Adaptive Extensible Processor
  - Extensible Processors
  - General Overview of Proposed Architecture
  - Generating Custom Instructions
  - Proposed Reconfigurable Functional Unit
- Evaluation Results

Motivations (1/2)

- Exploding NRE Costs

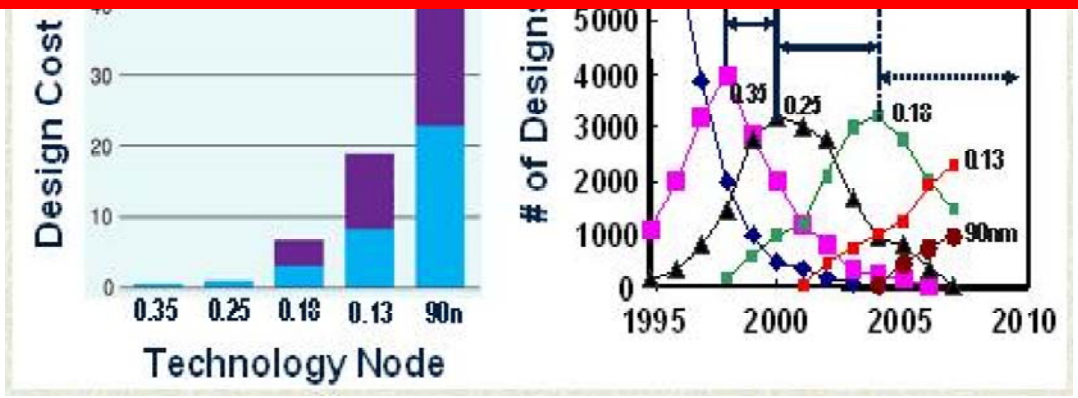




Motivations (2/2)

- Higher design costs due to more complex SoCs
- More complex applications → more computation and pr
- S

This has led to the quest for a flexible and reusable embedded processor that still must achieve the required performance and energy efficiency levels.



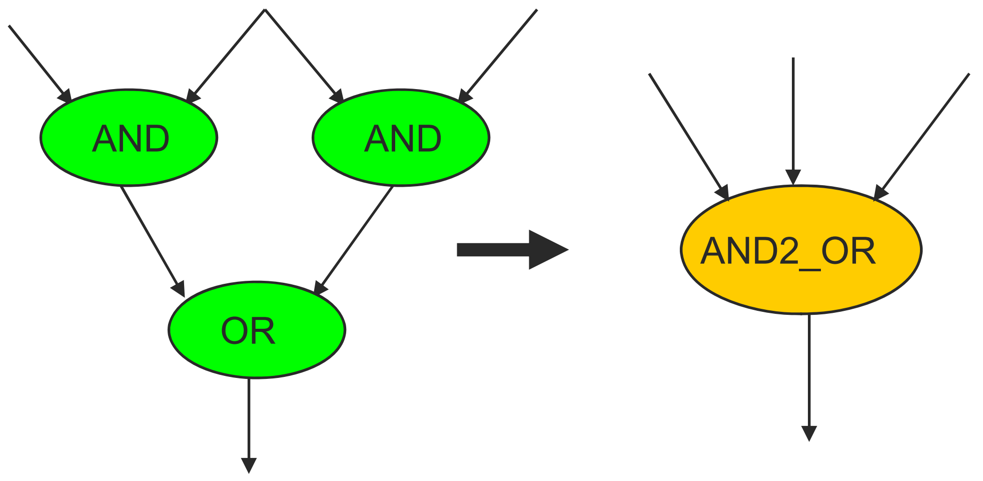
Keynote @ DATE 2007

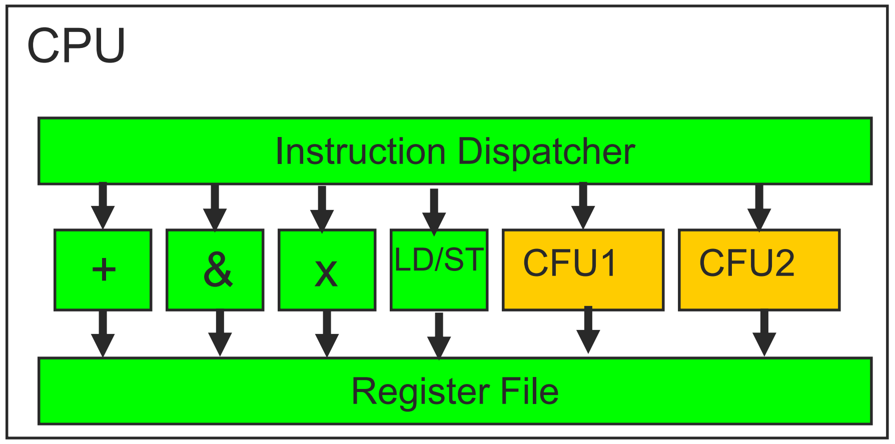
Goal

- Improving the performance and energy efficiency of embedded processors, while maintaining binary compatibility, and flexibility of embedded processors.

An Adaptive Extensible Processor

- Introduction to Extensible Processors





LD/ST: Load / Store

CFU: Custom Functional Unit

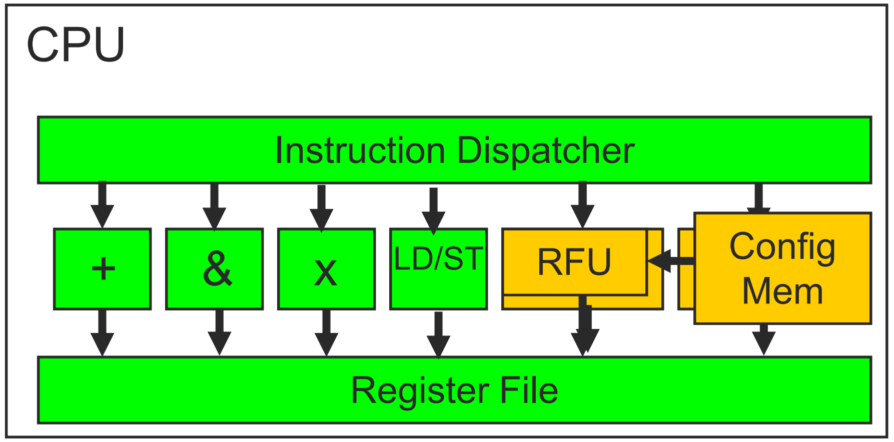
Custom Instructions

- Improve performance efficiency
  - Increasing parallelism
  - Reducing the latency of critical path
  - Reducing number of intermediate results written to the register file
  - Reducing cache misses

- Improve energy efficiency
  - Reducing accesses to
    - Instruction cache
    - Register file
    - Decoder
    - ALU
    - Cache misses
  - Reducing execution time (clock energy)

Proposed Approach

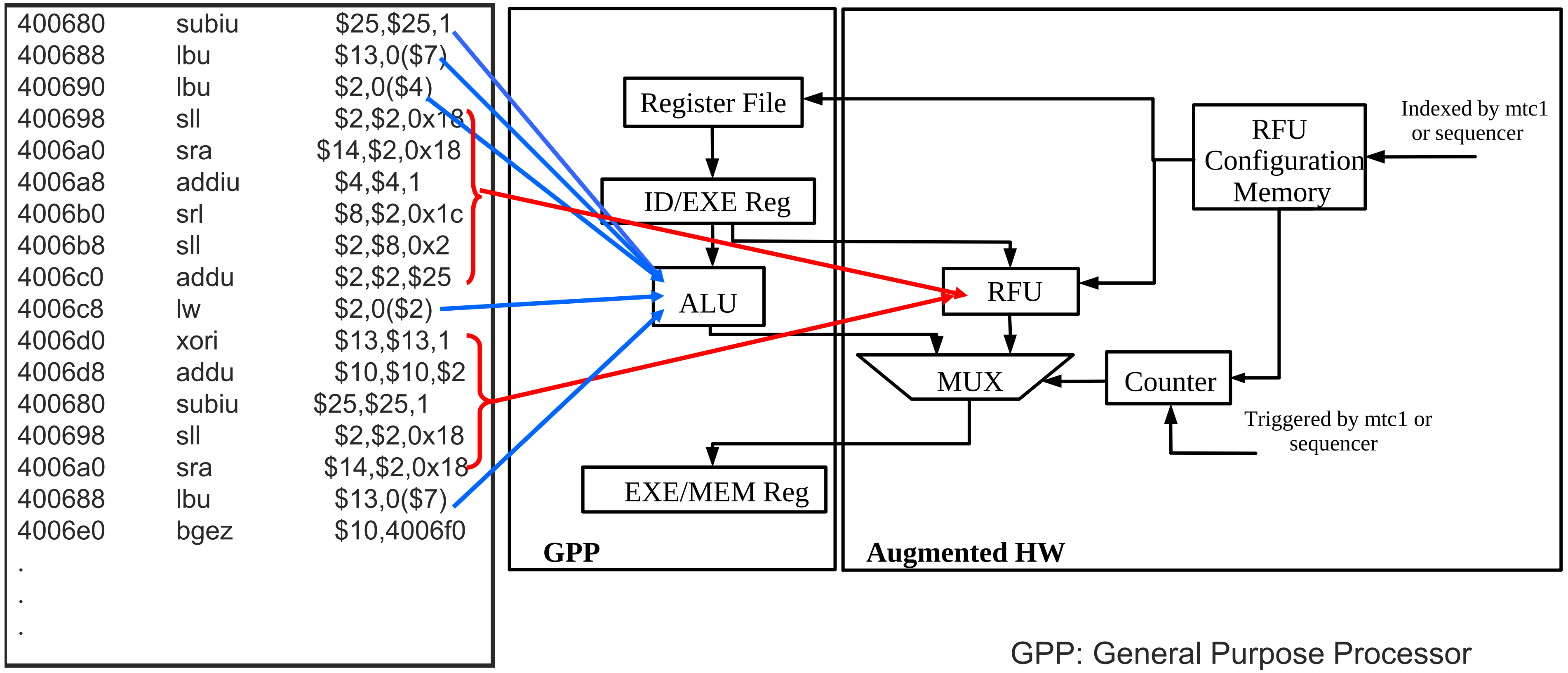
- An Adaptive Extensible Processor (ADEXOR)
  - Adding and generating custom instructions after fabrication
  - Using a reconfigurable functional unit (RFU) instead of custom functional unit



CFU: Custom Functional Unit

RFU: Reconfigurable Functional Unit

General Overview of the Proposed Architecture



```
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
.
.
.
```

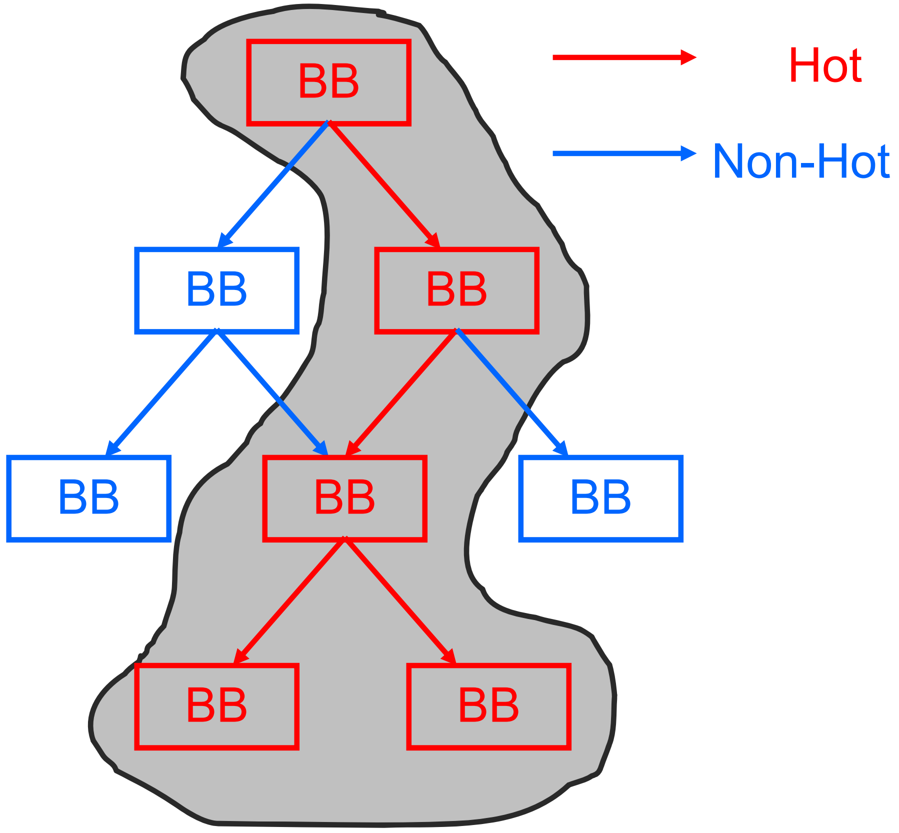
Hot Basic Block

GPP: General Purpose Processor

RFU: Reconfigurable Functional Unit

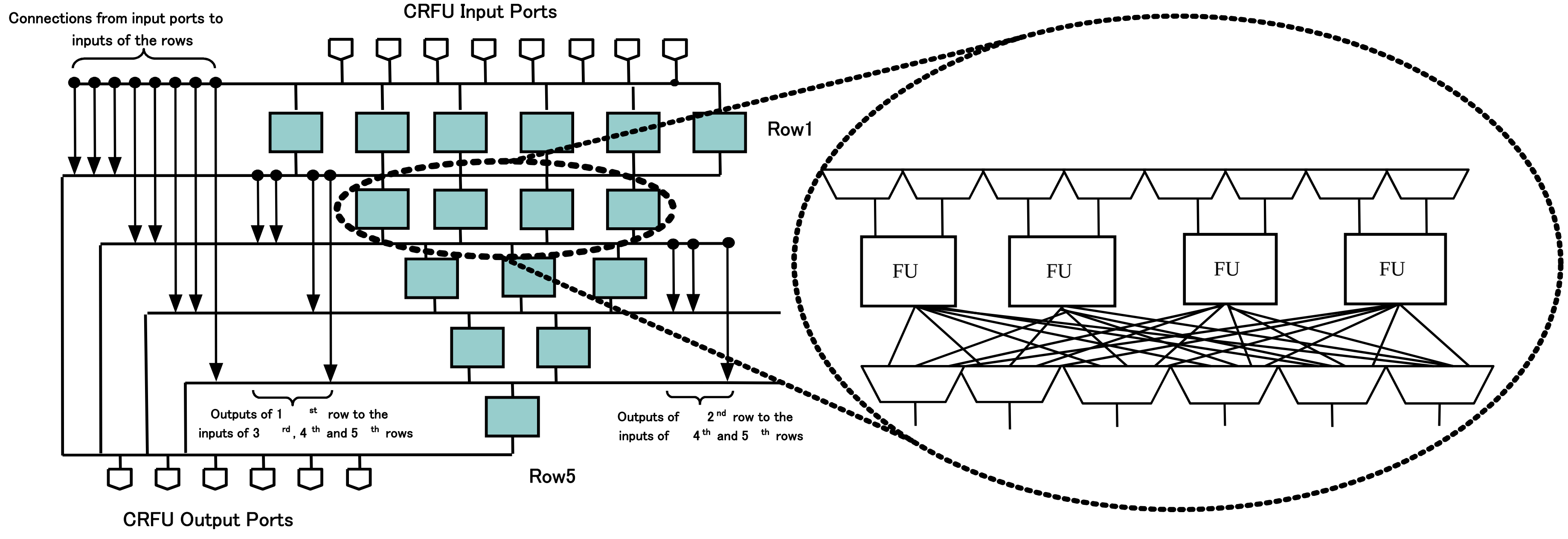
Generating Custom Instructions

- Load, divide, multiply, floating point operations are not supported



BB: Basic Block

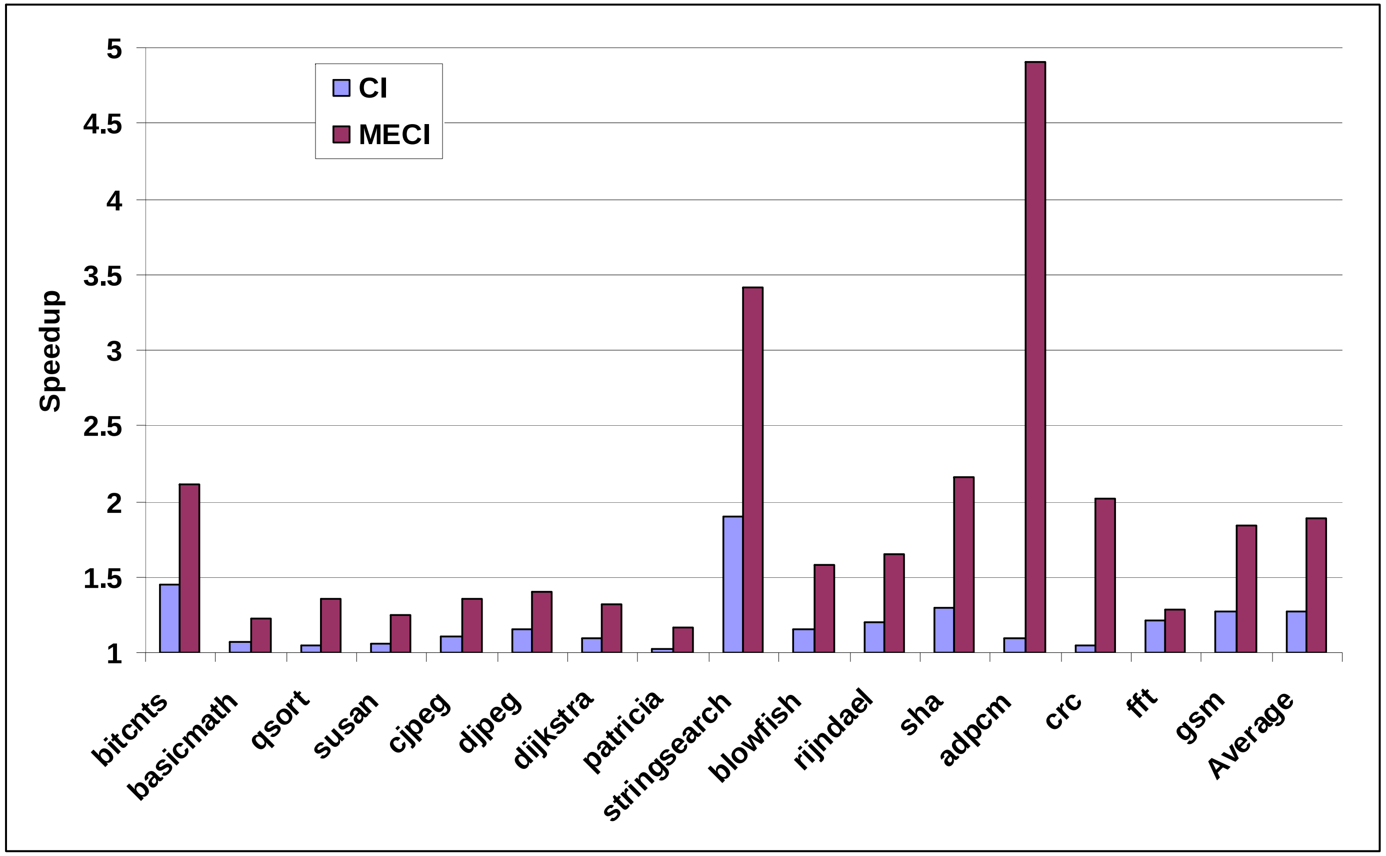
Architecture of the RFU


Connections from input ports to inputs of the rows
CRFU Input Ports
Row1
FU
FU
FU
FU
Outputs of 1st row to the inputs of 3rd, 4th and 5th rows
Outputs of 2nd row to the inputs of 4th and 5th rows
Row5
CRFU Output Ports

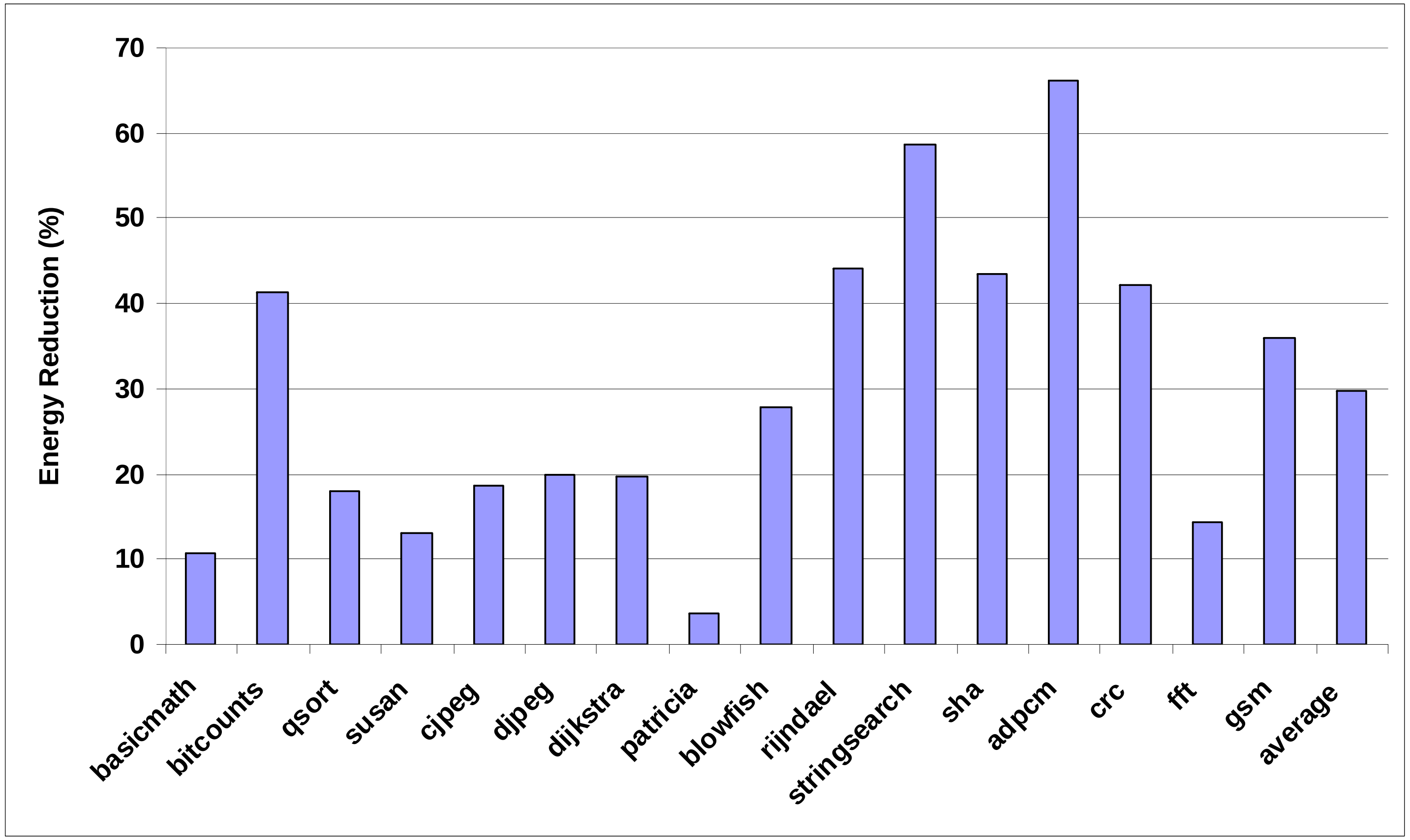
Evaluation Results

Issue	1-way
L1- I cache	16K, 2 way, 1 cycle latency for hit, 20 cycles for miss
L1- D cache	16K, 4 way, 1 cycle latency for hit, 20 cycles for miss
Execution units	1 integer unit, 1 floating point unit , 1 divider (8 cycles), 1 multiplier (5 cycles)
Branch predictor	bimodal
Branch prediction table size	256
Extra branch misprediction latency	3
Clock frequency	135 MHz

Performance Evaluation



Energy Evaluation



Area Overhead

- Base Processor 0.18μm (4.5mm^2) + Cache (3.2mm^2) = 7.7mm^2
- Area Overhead: 37%

Conclusions

- An architecture framework for an adaptive extensible processor
- Generating and adding custom instructions after chip-fabrication
- No new compiler, no source code modification and recompiling
- Evaluation
  - Speedup: max. 4.9 and average 1.9
  - Energy saving: max. 67% and 30% in average
  - Area overhead: 36%

Thank you for your attention