UNIVERSITAT POLITÈCNICA DE CATALUNYA
Departament d'Arquitectura de Computadors

NUCAR
Northeastern University

CGO'07, San Jose, California - March 2007

# Heterogeneous Clustered VLIW Microarchitectures

Alex Aletà, Josep M. Codina, Antonio González and David Kaeli

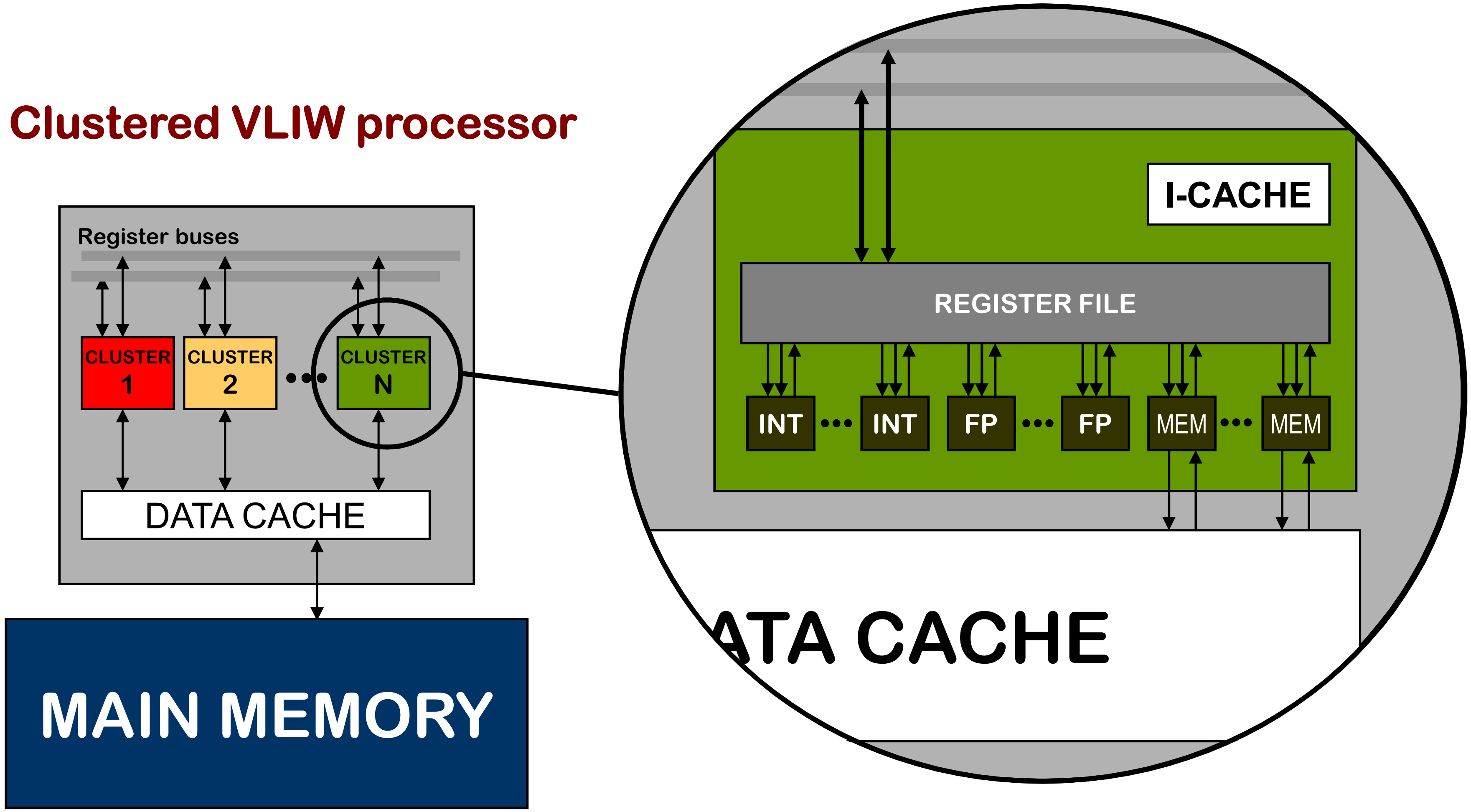
intel

# Clustered Microarchitectures

- **Challenges in processor design**
  - Wire delays
  - Power consumption
- **Clustering: divide the system into semi-independent units**
  - Each unit $\Rightarrow$ Cluster
    - Fast interconnects intra-cluster
    - Slow interconnects inter-clusters
  - Common trend in commercial VLIW processors
    - DSP/embedded domain

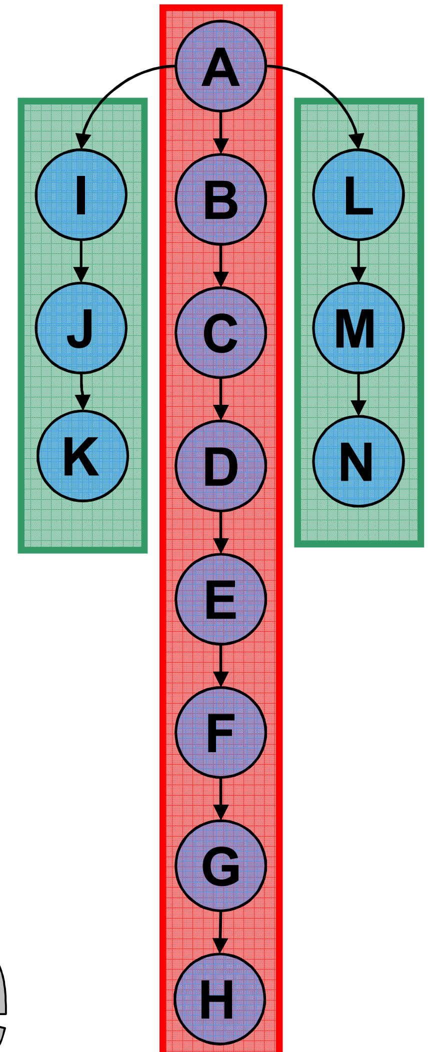
# Clustered VLIW Architecture

## Clustered VLIW processor



# Motivation

- Not all instructions have the same impact on execution time
- Divide resources
  - Performance oriented clusters
    - Higher voltages
    - Faster
    - Place critical instructions
  - Power oriented clusters
    - Lower voltages
    - Consume less power
    - Place non-critical instructions




**Heterogeneous Microarchitecture**

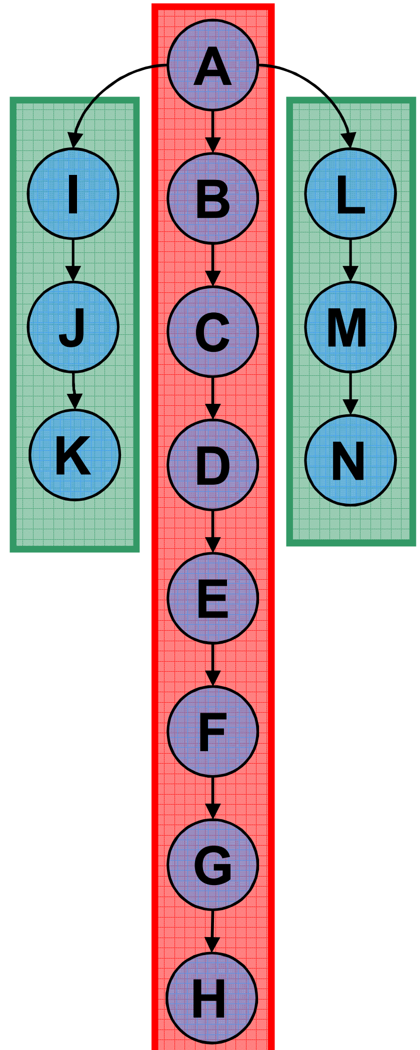
# Motivation

Homogeneous

| | C0 | C1 | C2 | ICN |
|---|---|---|---|---|
| Cycle time | 1 | 1 | 1 | 1 |

Scheduling

| Cycle | C0 | C1 | C2 | Bus |
|---|---|---|---|---|
| 0 | A | | | |
| 1 | B | | | Com A |
| 2 | C | I | L | |
| 3 | D | J | M | |
| 4 | E | K | N | |
| 5 | F | | | |
| 6 | G | | | |
| 7 | H | | | |



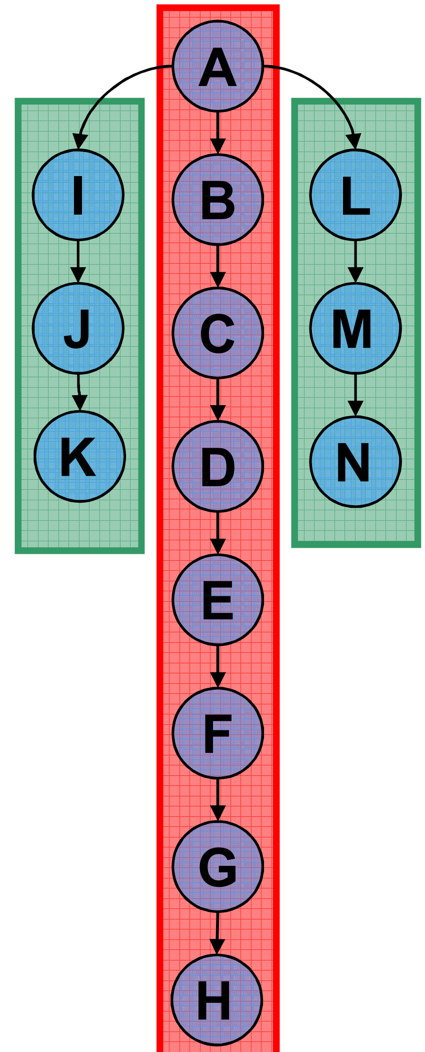
# Motivation

**Heterogeneous**

| | C0 | C1 | C2 | ICN |
|---|---|---|---|---|
| Cycle time | 1 | 2 | 2 | 1 |

**Scheduling**

| Cycle | C0 | C1 | C2 | Bus |
|---|---|---|---|---|
| 0 | A | | | |
| 1 | B | | | Com A |
| 2 | C | I | L | |
| 3 | D | | | |
| 4 | E | J | M | |
| 5 | F | | | |
| 6 | G | K | N | |
| 7 | H | | | |



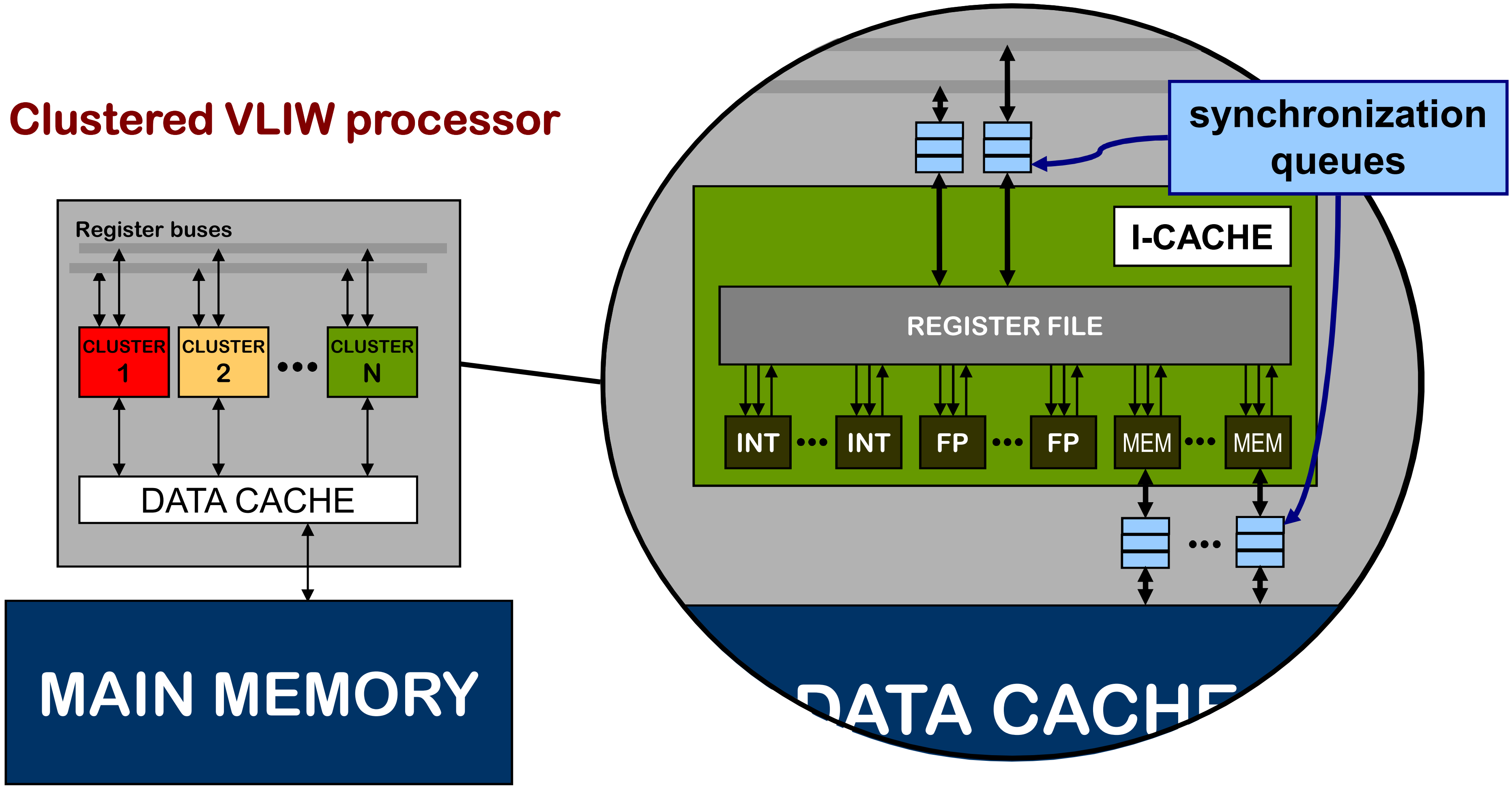
# Talk Outline

- Heterogeneous Clustered Architecture
- Proposed Compiler Techniques
- Experimental Evaluation
- Conclusions

# Heterogeneous Architecture

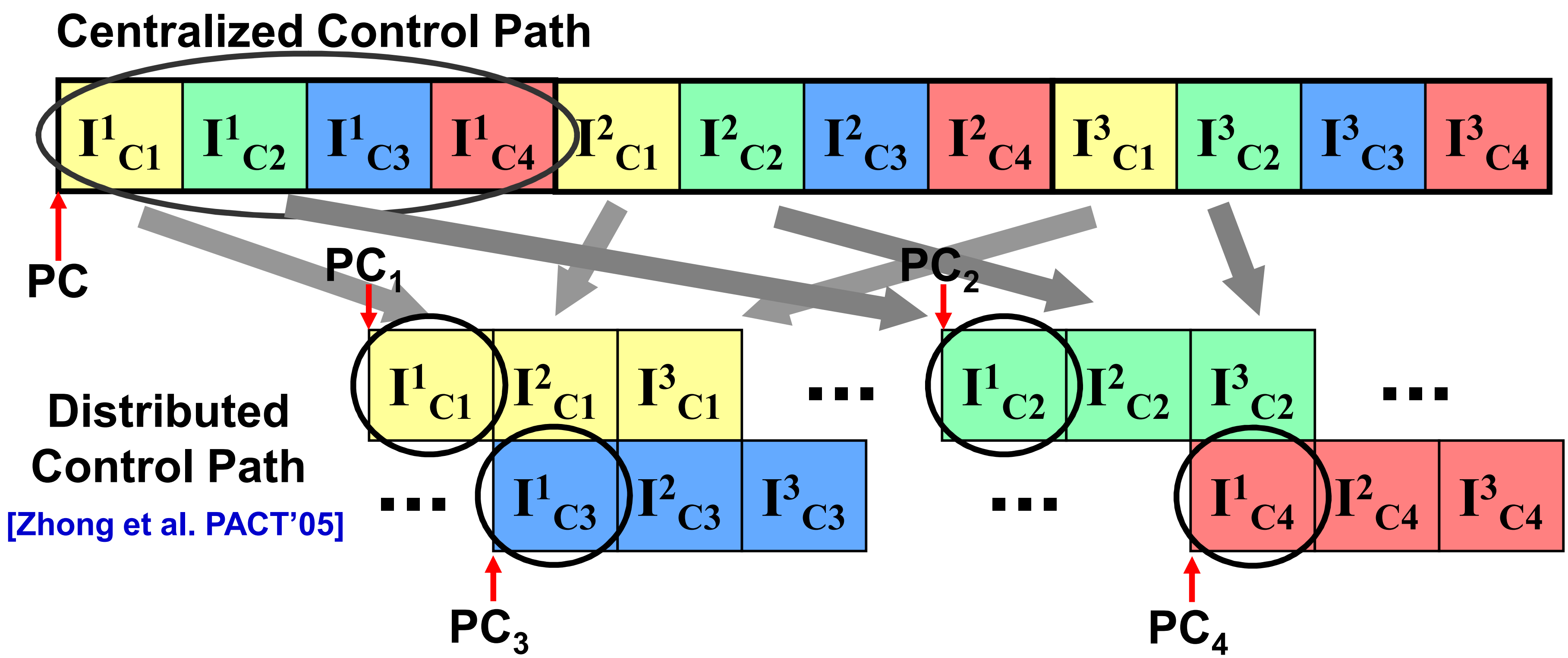
- Configured similar to a multiple clock domain design
  - Domain boundaries:
    - Each cluster
    - Inter-connection Network
    - Memory hierarchy
- Each domain can use a different voltage / frequency
  - Performance oriented: higher voltage/frequency
  - Power oriented: lower voltage/frequency
- Communication between domains: synchronization queues

# Heterogeneous Architecture



# Distributed Control Path

- Fetch and decode units distributed among clusters
  - Instruction lay-out
    - Grouped by cluster



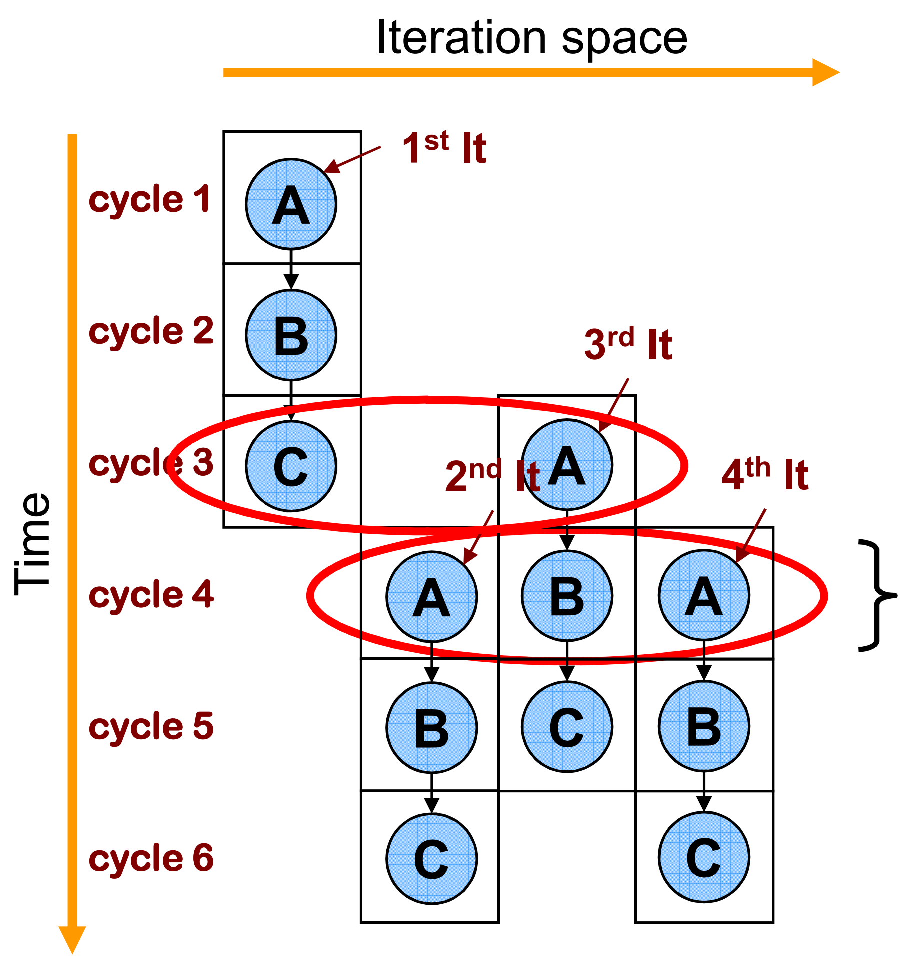
# Talk Outline

- Heterogeneous Clustered Architecture
- Proposed Compiler Techniques
- Experimental Evaluation
- Conclusions

# Statically Scheduled Processors

- **Performance relies on the compiler**
  - Instruction scheduling
  - Register allocation
  - Clustered microarchitectures
    - Cluster assignment
    - Communications
- **Multimedia and numeric code**
  - Majority of the execution in loop bodies
  - Modulo scheduling

# Modulo Scheduling



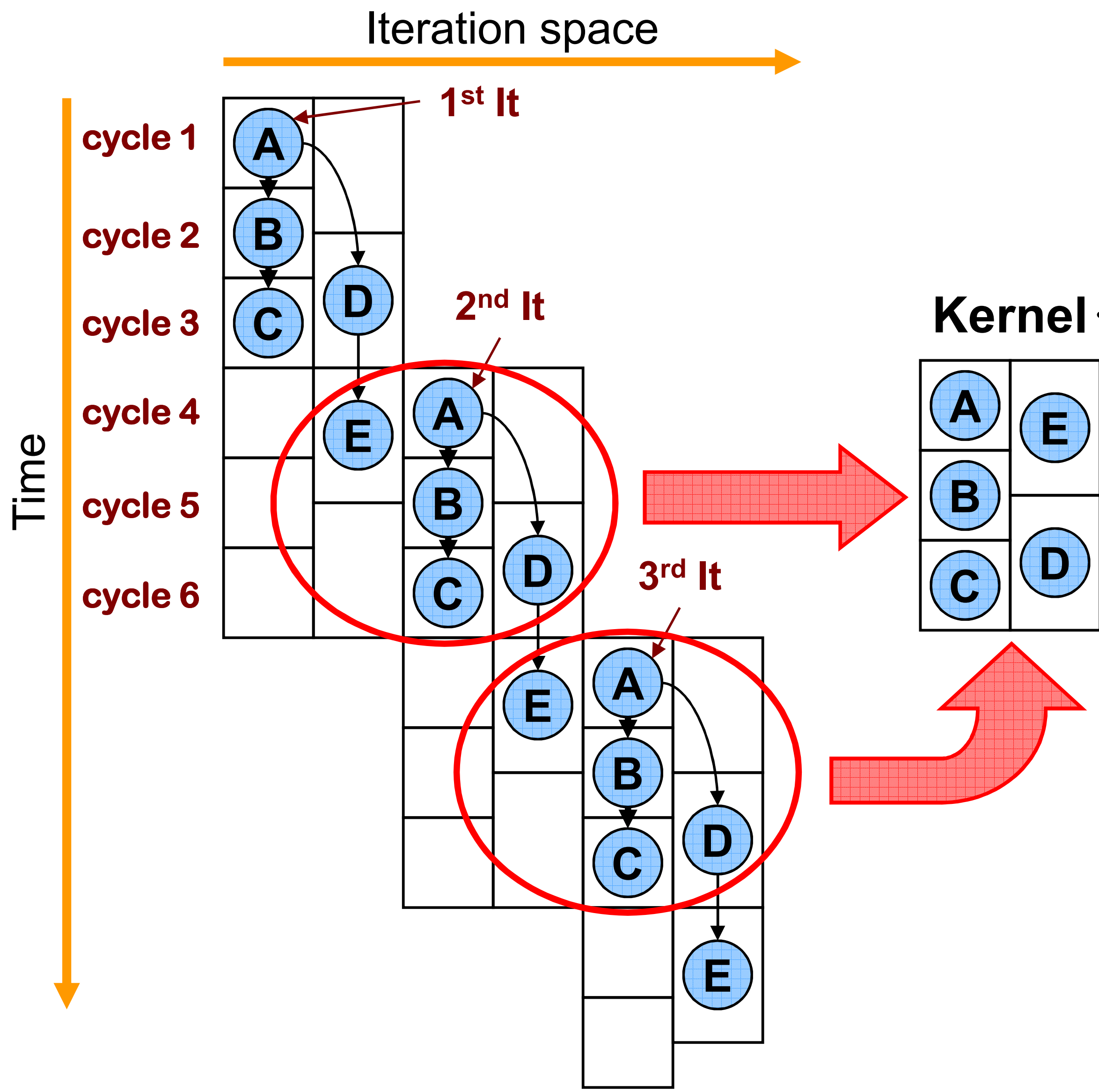

- **Effective technique for scheduling loops**
  - **Overlaps loop iterations**
  - **Increases parallelism**

Kernel: pattern repeated every II cycles

II ≥MII (Minimum Initiation Interval)

- Constrained by resources and recurrences
- MII = max {resMII, recMII}

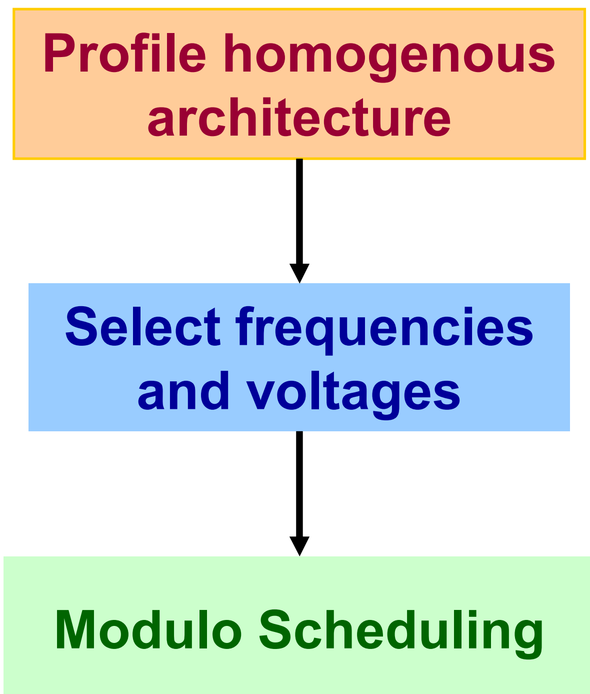
# MS for Heterogeneous




- Each cluster has its own II
- Constant Iteration Time (IT)

MIT = max {resMIT, recMIT}

# Proposed Technique



# Select Voltages and Frequencies

- For each domain
- At program level
- Consider different delays between fast and slow domains
  - Estimate execution time
  - Estimate energy consumption
  - Select minimum $ED^2$

# Estimate Execution Time

- Use profiling
  - $T_{exec} = n_{iters} \cdot (II + SC - 1) \cdot T_{cycle}$
- Estimated IT
  - Enough to accommodate
    - All instructions
    - All recurrences
    - Value lifetimes of the profiled homogeneous
    - Communications required by the profiled homogeneous

# Estimate Energy Consumption

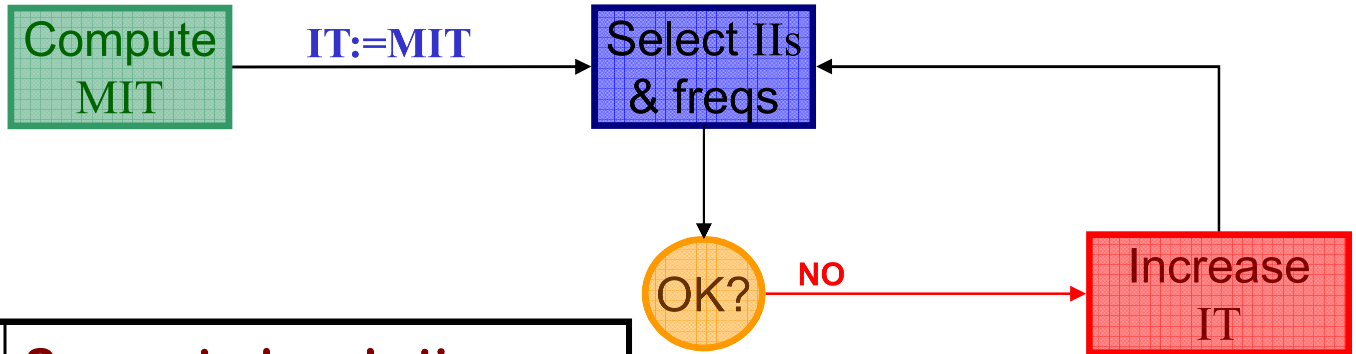
- **Same microarchitecture, different voltages**
  - **Relative dynamic power**

$$\frac{P_{dyn_1}}{P_{dyn_2}} = \frac{p_t \cdot f_1 \cdot C_L \cdot V_{dd1}^2}{p_t \cdot f_2 \cdot C_L \cdot V_{dd2}^2} = \frac{f_1 \cdot V_{dd1}^2}{f_2 \cdot V_{dd2}^2}$$

  - **Relative static power**

$$\frac{P_{stat1}}{P_{stat2}} = \frac{{}^{I_0}\!/_{W_0} \cdot W_t \cdot 10^{-V_{th1}/S} \cdot V_{dd1}}{{}^{I_0}\!/_{W_0} \cdot W_t \cdot 10^{-V_{th2}/S} \cdot V_{dd2}} = 10^{\frac{V_{th2} - V_{th1}}{S}} \cdot \frac{V_{dd1}}{V_{dd2}}$$

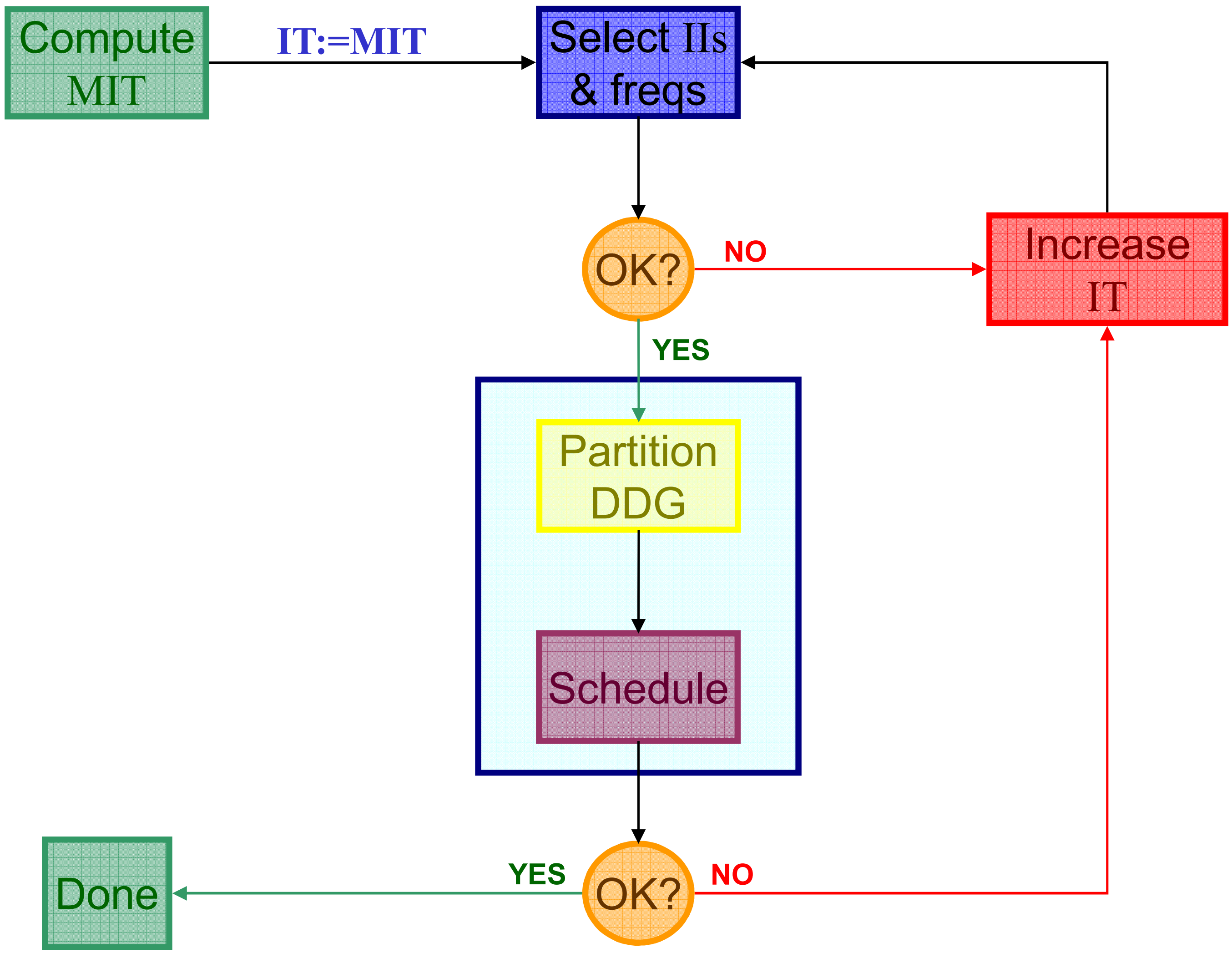
# MS Algorithm




| Cluster | Supported cycle times |
|---|---|
| C0 | 1ns |
| C1 | 5/4 ns ; 4/3 ns ; 3/2 ns |

| | | |
|---|---|---|
| IT= 7ns | II(C0)= 7 /1 = 7 cycles | II(C1)= 7 / 1.25 = 5.6<br>II(C1)= 7 / 1.3333 = 5.25<br>II(C1)= 7 / 1.5 = 4.667 |
| IT= 8ns | II(C0)= 8 / 1= 8 cycles | II(C1)= 8 / 1.25 = 6.4<br>II(C1)= 8 / 1.3333 = 6 |

# MS Algorithm

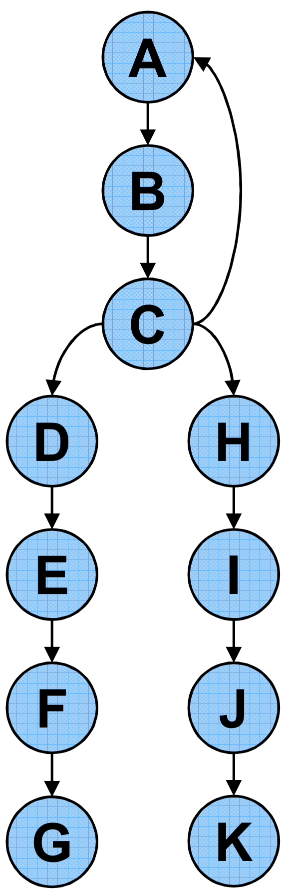


# MS for Heterogeneous

- **Multilevel graph partitioning algorithm**
  - Coarsening
    - Place critical recurrences in fast clusters
  - Refinement
    - Estimate execution time and energy consumption
    - Optimize for $ED^2$
- **Scheduling**
  - Instructions delayed due to synchronization hazards

# Recurrence Constrained Loops

- **Large benefits expected**
  - A small number of instructions are critical for execution time




- 2-cycle latency instructions
- 4 clusters, 1 FU per cluster

- **Homogeneous**
  - Recurrence: 6 cycles
  - II= 6 cycles (all clusters!) ⇨ lots of unused slots
- **Heterogeneous**
  - 1 fast cluster, cycle time= 1ns
  - 3 slow clusters, cycle time= 2ns

IT= 6ns
II(fast cluster)= 6
II(slow clusters)= 3

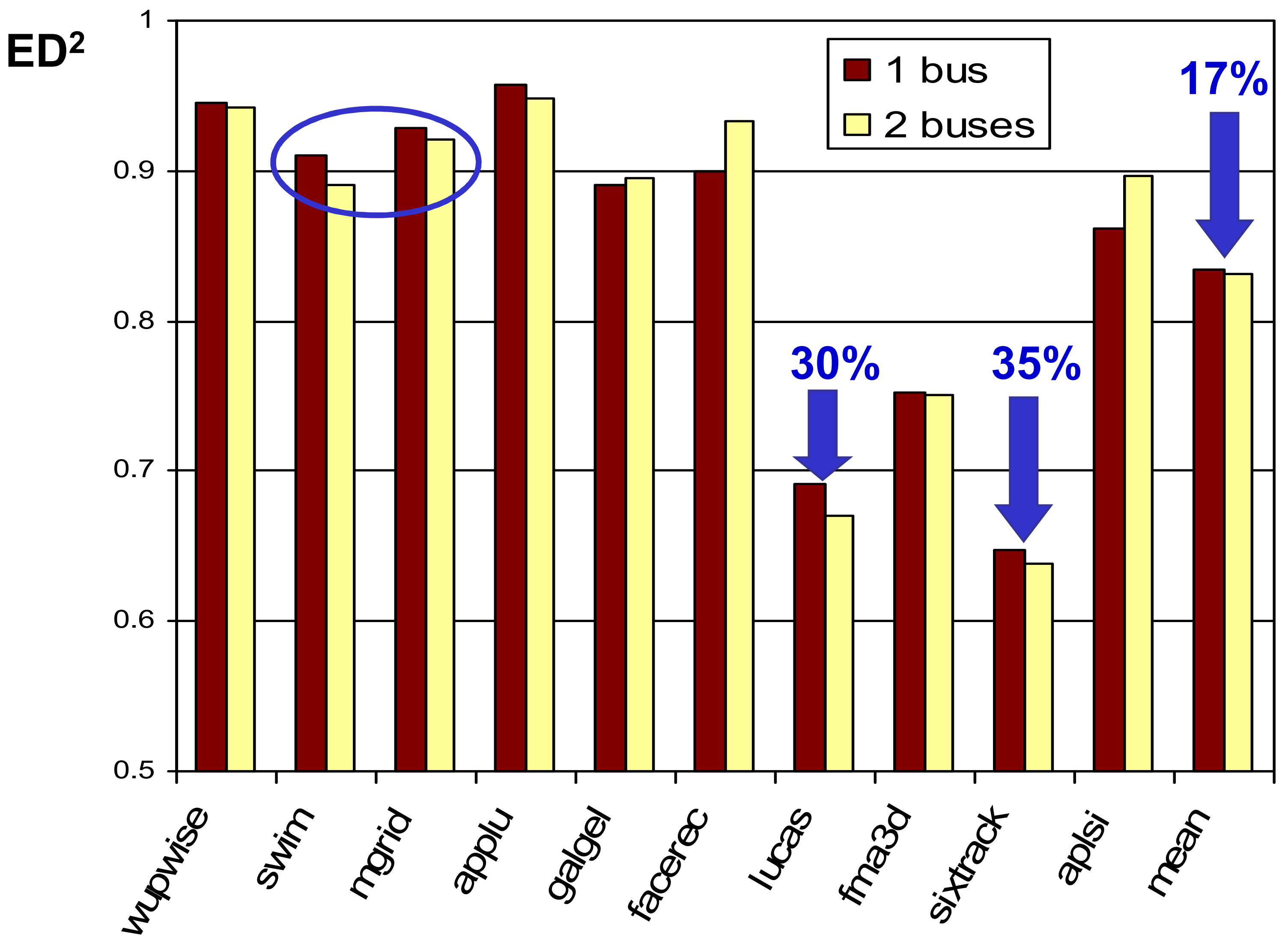
# Talk Outline

- Heterogeneous Clustered Architecture
- Proposed Compiler Techniques
- Experimental Evaluation
- Conclusions

# Experimental Environment

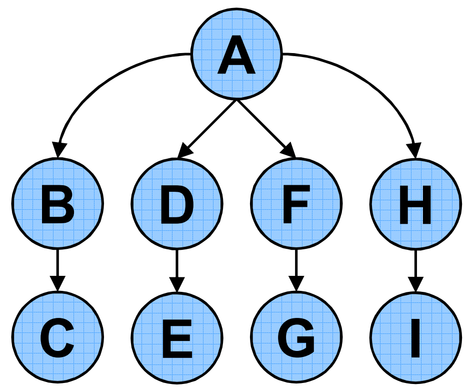
- **Microarchitecture**
  - Clusters
    - 4 clusters
    - 1 FP-unit, 1 INT-unit, 1 memory port, 16 registers per cluster
  - Inter-connection Network
    - 1-cycle latency broadcast buses
  - Heterogeneity
    - Clusters: 1 performance oriented / 3 power oriented
- **Benchmarks:**
  - SpecFP2k Fortran programs
  - Loops obtained with ORC
- **Baseline: homogeneous architecture**

# Results



ED²
1
0.9
0.8
0.7
0.6
0.5
1 bus
2 buses
17%
30%
35%
wupwise
swim
mgrid
applu
galgel
facerec
lucas
fma3d
sixtrack
aplsi
mean

# High ILP Programs




- All instructions have a similar impact on execution time
  - No benefit using different frequencies
  - Higher IPC
    - Dynamic energy accounts for a majority of the total energy consumption
    - Use lower $V_{dd}$ and lower frequencies
  - Memory hierarchy and inter-connection network can use different voltages

# Talk Outline

- Heterogeneous Clustered Architecture
- Proposed Compiler Techniques
- Experimental Evaluation
- Conclusions

# Conclusions

- **Heterogeneous clustered architectures**
  - **Clusters run at different voltages / frequencies**
    - **Instructions that impact execution time scheduled in fast clusters**
    - **Remaining instructions in power-oriented clusters**
- **Proposed compiler techniques**
  - **Algorithm to select the voltages / frequencies**
  - **MS for heterogeneous configurations**
- **$ED^2$ : 15% improvement on average**
  - **Up to 35% for selected programs**

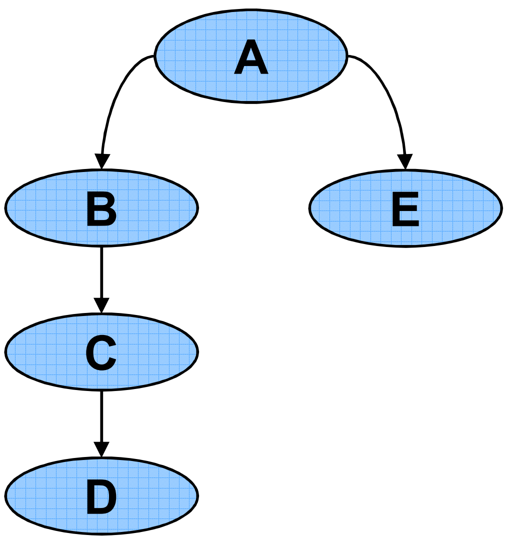
UNIVERSITAT POLITÈCNICA DE CATALUNYA
Departament d'Arquitectura de Computadors

CGO'07, San Jose, California - March 2007

# Heterogeneous Clustered VLIW Microarchitectures

Alex Aletà, Josep M. Codina, Antonio González and David Kaeli

# Motivating Example



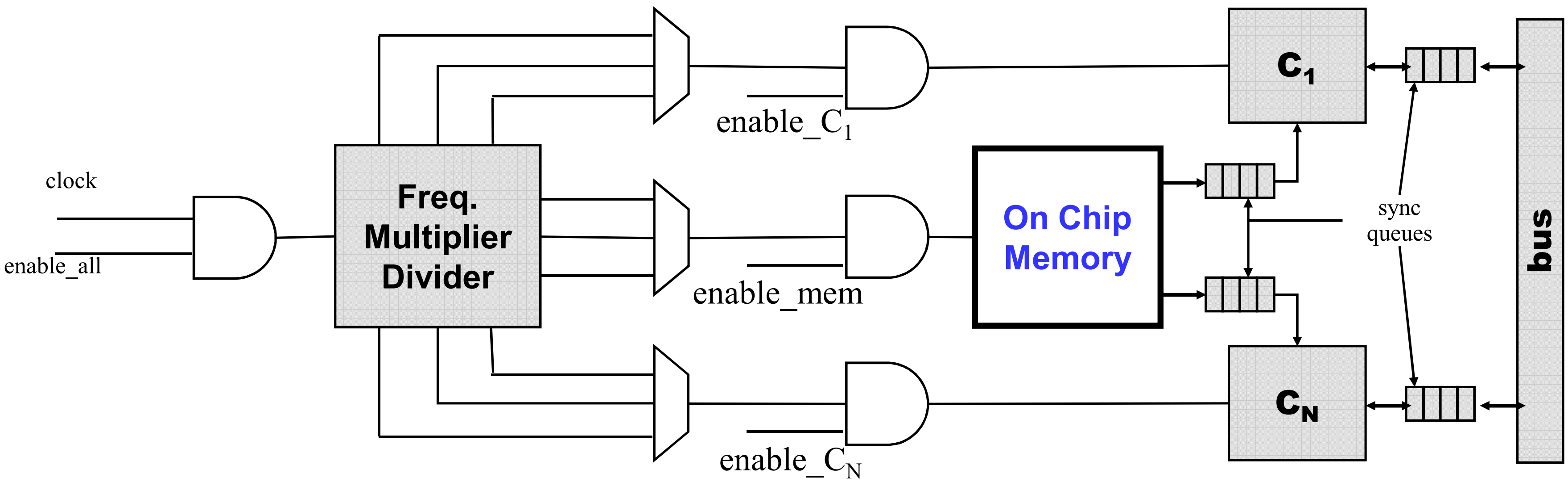

| | C1 | C2 |
|---|---|---|
| Cycle time | 1 ns | 2 ns |

| | C1 | Bus | C2 |
|---|---|---|---|
| 0 | A | | |
| 1 | B | Comm A | |
| 2 | C | | E |
| 3 | D | | |

| | C1 | Bus | C2 |
|---|---|---|---|
| 0 | A | | |
| 1 | B | Comm A | |
| 2 | C | | E |
| 3 | D | | |

# Heterogeneous Architecture

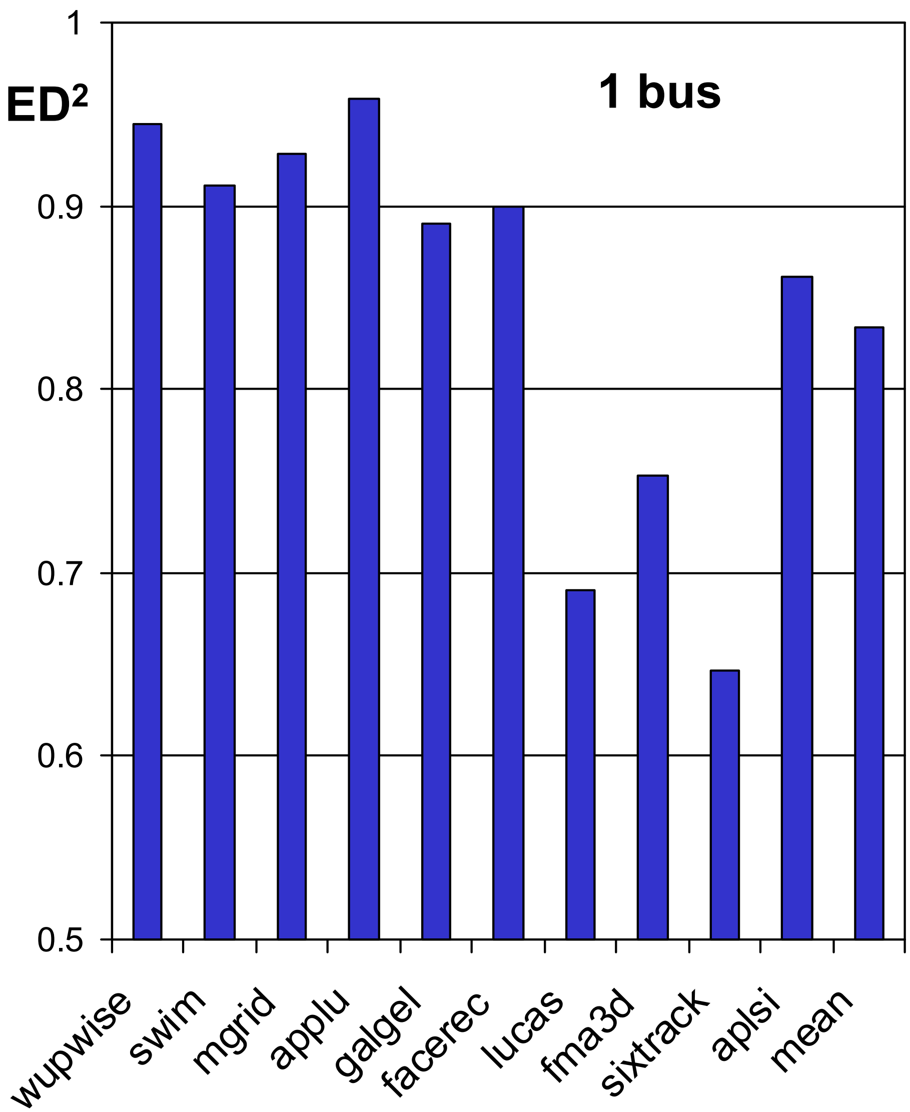
- Signals

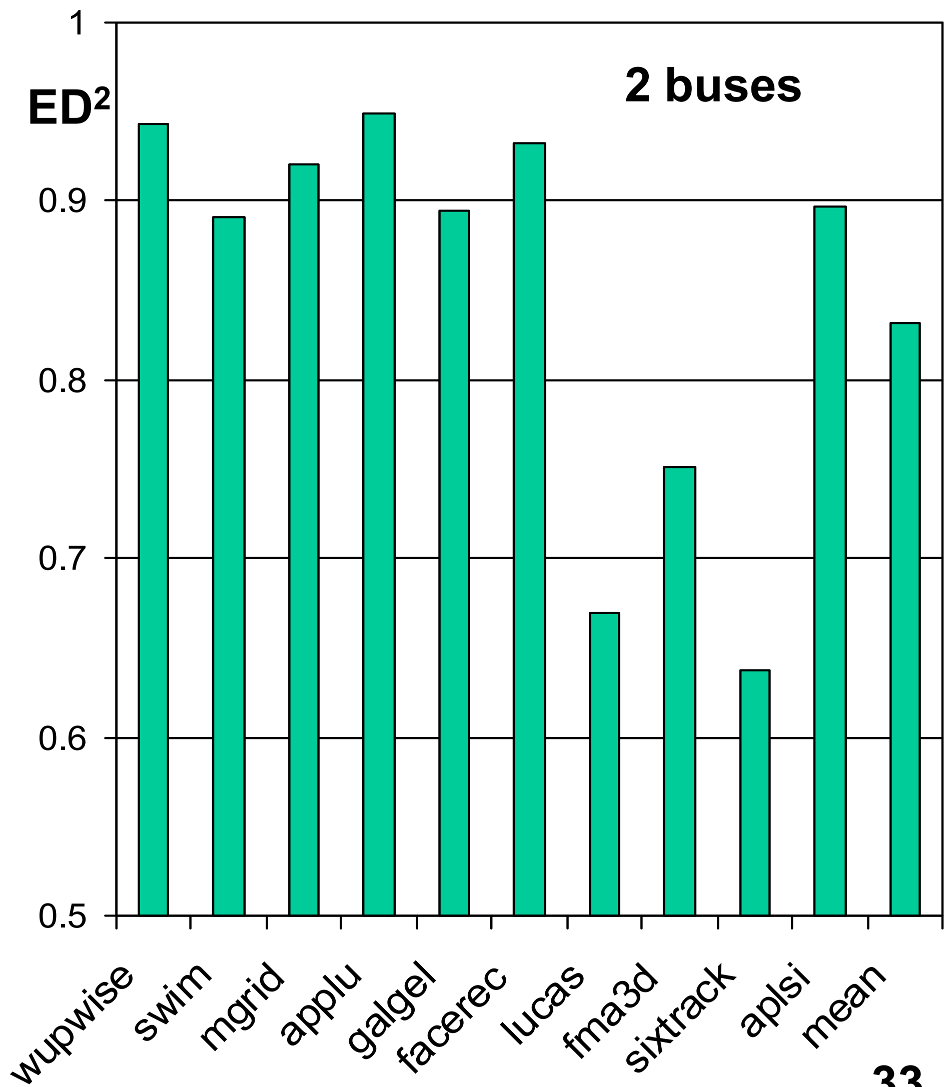


# Branch Instructions

- **Branches decoupled in several instructions (Unbundled Branch Architecture)**
  - Branch target computation
    - Independent in each cluster
  - Branch condition evaluation
    - Computed in one cluster
    - Broadcasted to the rest
  - Control transfer
    - Different in each cluster

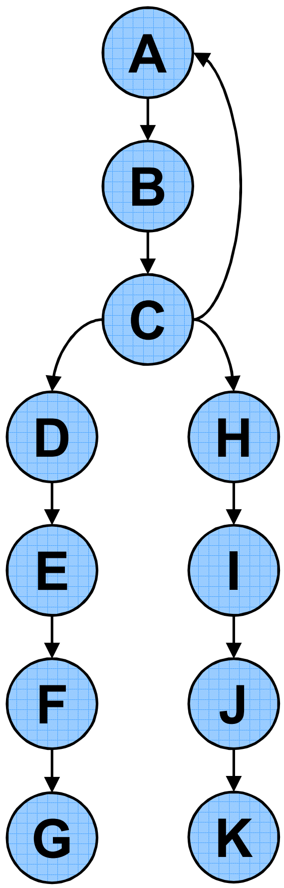
# Results


1 bus
ED2
1
0.9
0.8
0.7
0.6
0.5
wupwise
swim
mgrid
applu
galgel
facerec
lucas
fma3d
sixtrack
aplsi
mean


2 buses
ED2
1
0.9
0.8
0.7
0.6
0.5
wupwise
swim
mgrid
applu
galgel
facerec
lucas
fma3d
sixtrack
aplsi
mean

# Recurrence Constrained Loops

- Largest benefits obtained
  - A small number of instructions are critical for execution time
  - Improvement: 30% - 35% for 189.lucas and 200.sixtrack




- 2-cycle latency instructions
- 4 clusters, 1 FU per cluster

- Homogeneous
  - Recurrence: 6 cycles
  - II= 6 cycles (all clusters!) ⇨ lots of unused slots
- Heterogeneous
  - 1 fast cluster, cycle time= 1ns
  - 3 slow clusters, cycle time= 2ns

} IT= 6ns
II(fast cluster)= 6
II(slow clusters)= 3

# Smallest Benefits

- **Around 5%**
  - 168.wupwise
    - Loops have different characteristics
  - 173.applu
    - Low number of iterations
    - Schedule length has a big impact