UPC

Design of Clustered Superscalar Microarchitectures

Joan-Manuel Parcerisa[1]

jmanel@ac.upc.es

Antonio González[1,2]

antonio@ac.upc.es

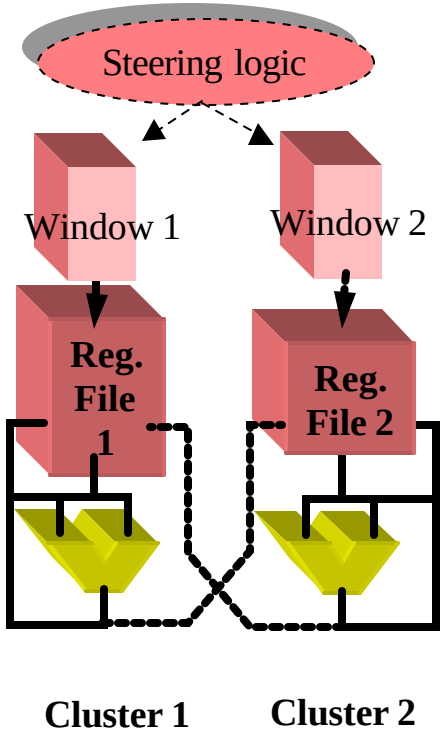
[1] Departament d'Arquitectura de Computadors

[2] Intel-UPC Barcelona Research Center

Universitat Politècnica de Catalunya – Barcelona, Spain

Dynamic cluster assignment schemes

- **Main goals**
  - Minimize communication penalty
  - Maximize workload balance
  - ☞ there is a trade-off
- **Slice-based schemes**
  - Loads (branches) are likely critical
  - Keep load (branch) backward slice in same cluster
  - Assign slices to clusters according to workload



R.Canal, J.-M.Parcerisa and A.González. “Dynamic Cluster Assignment Mechanisms”, HPCA6, 2000

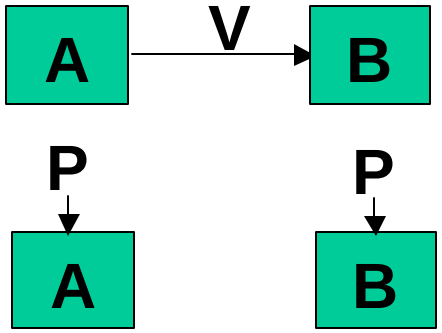
Dependence-based steering

- **Two basic rules**
  - 1- To minimize communication penalties
    - select clusters with highest number of operands mapped
  - 2- To maximize workload
    - choose the least loaded of the above selected clusters
- **Give priority to critical operands**
  - If a source register is unavailable (likely critical)
    - rule 1 chooses its producer cluster
- **Avoid high imbalances**
  - If imbalance > threshold
    - choose the least loaded cluster (ignore rule 1)
    - or better: discard overloaded clusters (before applying rules 1, 2)

R.Canal, J.M.Parcerisa and A.González “Dynamic Code Partitioning for Clustered Architectures” IJPP 29(1), 2001

Reducing wire delay penalty through VP

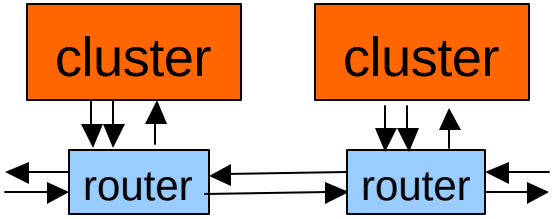
- **VP hides wire delays**
  - Let A and B have the same prediction P
  - A checks prediction locally
- **Apply to cluster communications**
  - Speculate on remote source registers that must be communicated
- **Shallower dependence graph**
  - 50% less communications
  - VP-aware steering improves workload balance

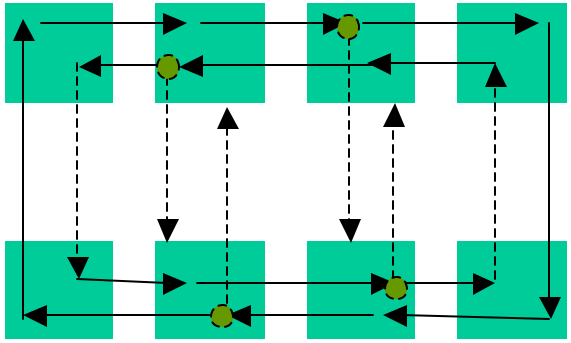


J.-M.Parcerisa and A. González “Reducing Wire Delay Penalty through Value Prediction”, Micro33, 2000

Efficient cluster interconnects

- **All-to-all interconnects do not scale**
- **Point-to-point vs. buses**
  - Local link arbitration, faster wires
- **Requirements**
  - Low latency: source routing, no intermediate buffers
  - Low BW: 1 input/cluster, 2 links/cluster-pair
- **4 and 8 cluster interconnects**
  - Buses
  - Sinchronous Ring
  - Partially Asynch. Ring, Mesh and Torus

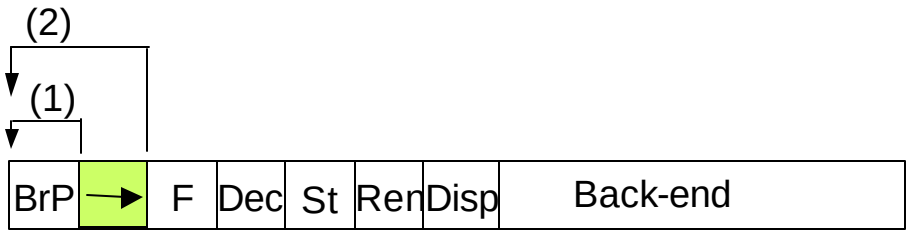


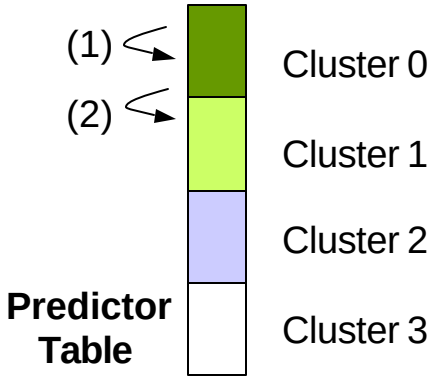


J.-M.Parcerisa, J.Sahuquillo, A.González and J.Duato, “Efficient Interconnects for Clustered Microarchitectures”, PACT 2002

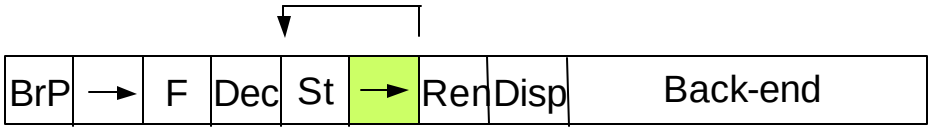
Clustering front-end structures

- **Clustered branch predictor**
  - Pipeline prediction ahead of I-cache + interleave by hi-bits
  - Bubble only when high level interleave boundary crossed (2)





- **Clustered steering**
  - Make assignments as if instructions were independent
  - Check dependences, and override assignment, if dependence was violated



J.-M.Parcerisa, A. González and J.E.Smith. “A Clustered Front-End for Superscalar Processors”, Tech. Report, 2004