Mechanisms: Run-time and Compile-time

Outline

- ***Agents of Evolution***
- ***Run-time***
  - ***Branch prediction***
  - ***Trace Cache***
  - ***SMT, SSMT***
  - ***The memory problem (L2 misses)***
- ***Compile-time***
  - ***Block structured ISA***
  - ***Predication***
  - ***Fast track, slow track***
  - ***Wish branches***
  - ***Braids***
  - ***Multiple levels of cache***

Agents of Evolution

-- Performance

-- Bottlenecks

-- Good Fortune

Evolution of the Process

Pipelining
Branch Prediction
Speculative execution (and recovery)
Special execution units (FP, Graphics, MMX)
Out of order execution (and in-order retirement)
Pipelining with hiccups (R/S, ROB)
Wide issue
Trace cache
SMT, SSMT
Soft errors
Handling L2 misses

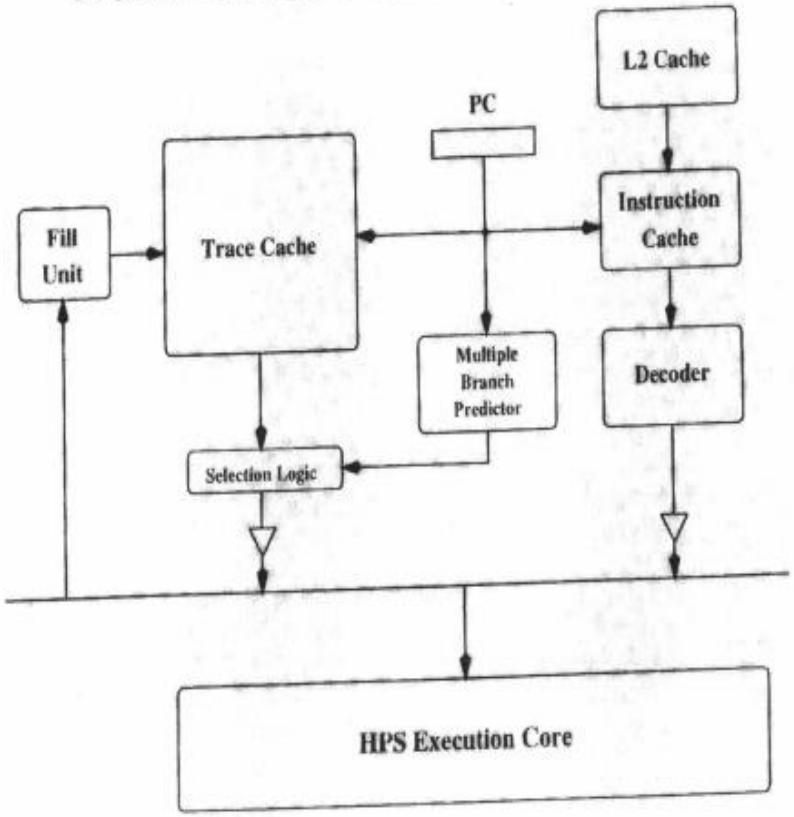
Run-time Mechanisms

- ***Branch prediction***
- ***Trace Cache***
- ***SMT, SSMT***
- ***The memory problem (L2 misses)***

The Conditional Branch Problem

- ***Why? Because there are pipelines. (Note: HEP)***
- ***Mechanisms to solve it***
  - ***Delayed branch***
  - ***Take both paths***
  - ***Eliminate branches (predication, compound predicates)***
- ***Branch Prediction***
  - ***Static: Always taken, BTFN***
  - ***Early dynamic: LT, 2 bit counter***
  - ***2 level → gshare, agree, hybrid***
  - ***Indirect jumps***
  - ***Perceptron, TAGE***
  - ***Expose branch prediction hardware to the software***
  - ***Wish branches***

Trace Cache Fetch Mechanism



Trace Cache Issues

Storage partitioning (and set associativity)

Path Associativity

Partial Matching

Inactive Issue

Branch Promotion

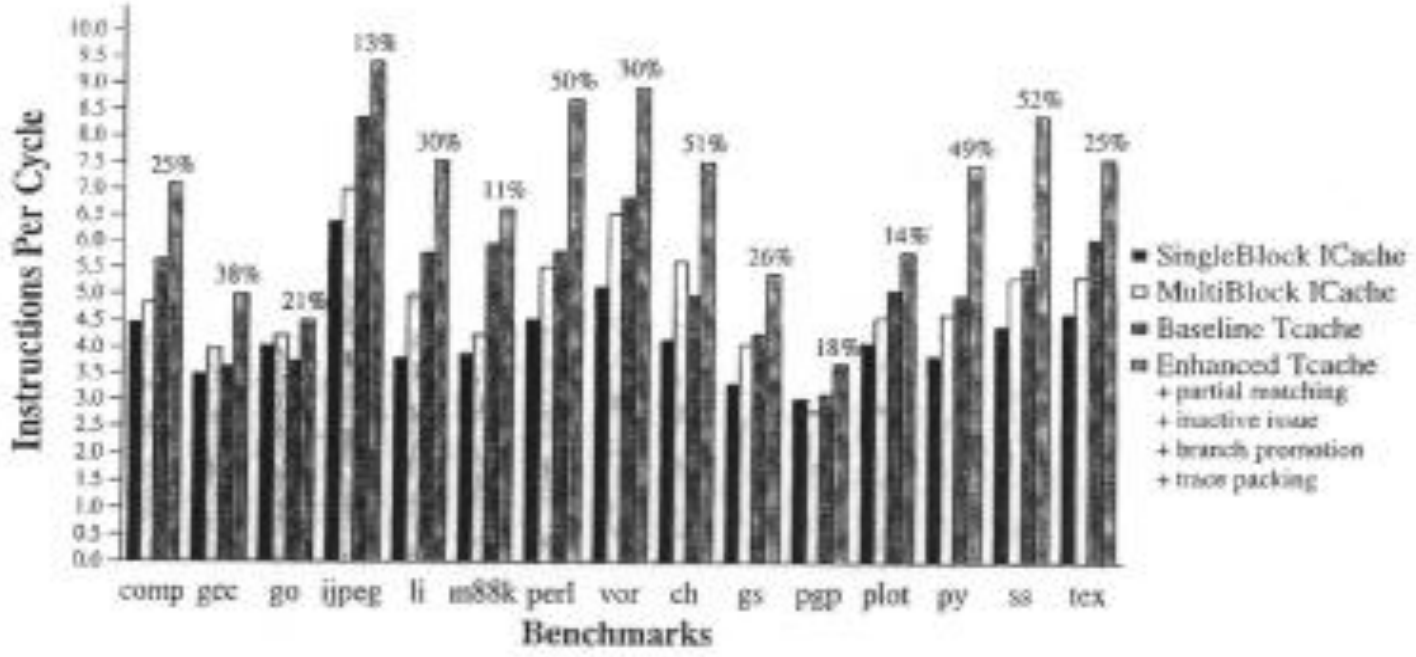
Trace Packing

Dual Path Segments

Block collection (retire vs. issue)

Fill unit latency

Overall Performance (aggressive core)



SMT and SSMT

- ***SMT***
  - ***Invented by Mario Nemirovsky, called DMS (HICSS-94)***
  - ***Multiple threads sharing common back end***
- ***SSMT (aka "helper threads)***
  - ***Proposed in ISCA 99 (Rob Chappell et.al)***
  - ***Simultaneously invented by M. Dubois, USC tech report***
  - ***When the application only has one thread***
  - ***Subordinate threads work on chip infrastructure***
  - ***Subordinate threads: in microcode or at ISA level***

The memory problem

Simply: off-chip, on-chip access time imbalance (assuming bandwidth is not a problem)

What to do about it

- ***MLP replacement policy***
- ***DIP replacement***
- ***VWay cache***
- ***Runahead execution***
- ***Address, value prediction***
- ***3D Stacking***
- ***Two-level register file***
- ***Denser encoding of instructions, data***
- ***Better organization of code, data***

Compile-time mechanisms

- ***Predication (and wish branches)***
- ***Block structured ISA***
- ***Fast track, slow track***
- ***Braids***
- ***Multiple levels of cache***

Wish Branches
(Predicated execution to the next level)

- ***Compile-time and Run-time***

- ***At compile time:***
  - ***Does predication even make sense***
  - ***If no, regular branch and branch prediction***
  - ***If yes, mark wish branch and defer to run-time***

- ***At run time:***
  - ***Prediction accuracy low: predicate***
  - ***Prediction accuracy high: branch predict***

Block-structured ISA

References:

- ***Stephen Melvin dissertation (Berkeley, 1991)***
- ***Eric Hao dissertation (Michigan, 1997)***
- ***ISC paper (Melvin and Patt, 1989)***
- ***ISCA paper (Melvin and Patt, 1991)***
- ***Micro-29 paper (Hao, Chang, Evers, Patt, 1996)***

The Atomic Unit of Processinng

Enlarged Blocks

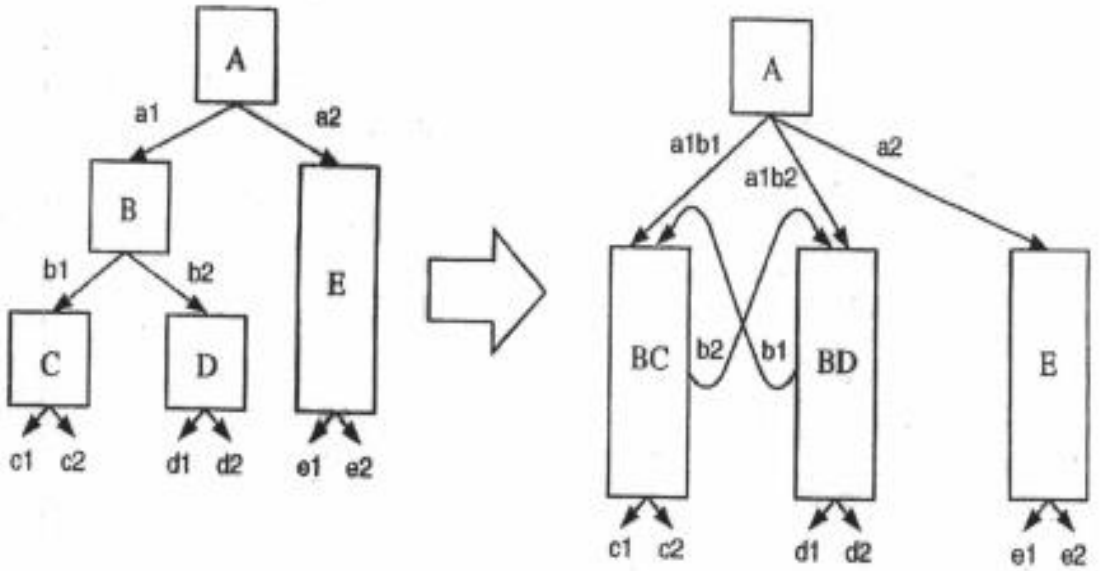
Savings on Register Pressure

Faulting Branches, Trapping Branches

Serial Execution on Exceptions

Comparison to Superblocks, Hyperblocks

Enlarging Basic Blocks



Compile-time mechanisms

- ***Predication (and wish branches)***
- ***Block structured ISA***
- ***Fast track, slow track***
- ***Braids (Francis Tseng, ISCA 2008)***
- ***Multiple levels of cache***