ILP Speedups in the 10’s

or: Reducing the Ill Effects of Branches

(“ILP”: Instruction Level Parallelism)

Prof. Augustus (Gus) K. Uht
Dept. of Electrical and Computer Engineering

UNIVERSITY OF RHODE ISLAND
"A 21st-century microprocessor may well [issue] up to dozens of instructions [per cycle, peak]..."

Contributions of the Work

• New form of speculative execution (DEE)
  – Optimal, low cost, high performance:

  *Speedup factors of 26-31 (2,600% - 3,100%)*

• New machine model devised for DEE:
  \text{Levo} (target ILP: x 20)
  – On single chip in 4-5 years (by 2000 AD!)
Acknowledgements

• This work supported by the Intel Corp. through a grant from the Intel Research Council. Other support from URI and NSF.
• Simulations by Vijay Sindagi
• Other contributors to the work:
  – Sajee Somanathan
  – Sridhar Mahankali
Current Contributors

- Wei Tan
- Jenny Xing
- Christy Julian
- Prof. Qing Yang & Co.
Rest of Talk

• Introduction- the name of the game is: Speed
• Other Background
  – ILP limits, Branch Effect Reduction Techniques
• **Disjoint Eager Execution (DEE)**
  – Theory
  – Heuristic
  – Performance evaluation
• The prototype: Levo
Ways to Improve Computer Performance

• Technology: *increase speed of transistors*
• Circuits: *faster gates*
• Algorithms: *reduce computational complexity*
• Compiler: *better optimizations*
• **Architecture: parallelism:**
  – pipelining
  – multiprocessors & distributed computers
  – *Instruction Level Parallelism (ILP)*
Instruction Level Parallelism (ILP)

- Execute more than 1 instruction per cycle
- Example:

1. \( A = B + C \)
2. \( D = E + F \)
3. \( G = A + H \)

instructions 1 and 2 can execute in parallel;
1 and 3 cannot (data dependency)
State of the ILP World

• The Problem:
  – GP code ILP speedups of only 2-3 in both machines and other research

• Constraints:
  – Machine code compatibility
  – Source code not available

• Trends:
  – Transistor densities to 50-100M/chip by 2000
  • How to best use this hardware?
Other Background

• Oracle ILP speedups:
  – Riseman and Foster (1972), harmonic mean speedup $S = 25$;
  – Lam and Wilson (1992): $S = 159$; & others....

• w/ realistic constraints, only get: $S = 2$ to $3$ (to date, using SPECint92’s)

• Branches are the problem!
Branch Effect Reduction Techniques (*BERT*’s)

- Both hardware and software can be used
- Branch Target Buffer
- VLIW, Software pipelining
- Minimal Control Dependencies
- Speculative Execution:
  - *Conditionally execute past branch(es) before value of condition is known.*
Minimal Control Dependencies
(Uht85, Ferrante87, Uht91)

• Classic model: *restrictive control dependencies*

• Can be relaxed: w/MCD, 3 & 4 ind. of 1

1. if (a<8) {
2. b=c+d;
3. x=y+z;
4. if (p>5) {...}
Speculative Execution

- Given: \( l \) is depth of greatest speculation
- **Single Path (SP)** - \( O(l) \) cost, but low performance: cumulative prob. \( (cp) \) --> 0
- **Eager Execution (EE)** - best performance, w/ infinite resources, but high cost: \( O(2^l) \)
- Need something better, with good features of both SP and EE:

  **Disjoint Eager Execution (DEE)**
SP and EE Models

Single Path
($l = 6$)

Eager Execution
($l = 2$)
DEE Theory

• *Branch Path* (resources) definition: dynamic code between branches (PE’s to execute the code in the path as concurrently as possible)

• Rule of Greatest Marginal Benefit:

  *Assign resources to most likely paths, over all pending paths*

• Optimal for constrained resources

• Cost: $O(kl^2) ; k<1$
Assigning Resources

① .7
Assigning Resources
Assigning Resources
Assigning Resources

1. 0.7
2. 0.49
3. 0.34
4. 0.3
Assigning Resources
Assigning Resources
Assigning Resources
Assigning Resources
Assigning Resources
Assigning Resources
DEE in Practice

• Problem: hard to compute “true” cumulative probabilities dynamically
• Solution: DEE *static tree* heuristic
  – Use average branch prediction accuracy (bpa or $p$) for all branches
  – Static tree shape determined as part of machine design
  – Resources are fixed to the static tree
  – Cost: still $O(kl^2)$ ; $k<1$
Typical Static Tree

A number on a path is the overall or cumulative probability of the path being executed.

BPA = 90% = 0.90

Total number of branch paths is 34.
DEE Performance Evaluation

- **Method:** pixie and modified dsim used
- **Assumptions:**
  - Unit latency
  - Dynamic Instruction Stream
  - MIPS R3000 instruction set
  - Practical version (heuristic) of DEE modelled
Harmonic Mean Summary

- 5 of 6 SPECint92 benchmarks used:
  - cc1
  - compress
  - eqntott
  - espresso
  - xlisp
  <=100 million instructions each
- 2-bit saturating counter predictor (Smith81)
- “CD-MF” = “Minimal Control Dependencies”
- “DEE-CD-MF” is DEE with MCD; used in Levo
Harmonic Mean

Oracle Speedup: 53.82

Resources (branch paths)

Speedup (factor X sequential)

DEE-CD-MF
SP-CD-MF
DEE-CD
SP-CD
DEE
SP
EE

<<< Levo
Harmonic Mean Summary

Resources (branch paths)

Speedup (factor)

ILP Model

DEE

SP

DEE-CD

SP-CD

DEE-CD-MF

SP-CD-MF

Harmonic Mean

Summary
Comments on Results

• Speedup factors of 26-31 demonstrated with limited resources and DEE-CD-MF
• Combination of DEE and minimal control dependencies is necessary
• Speedup of 20 potentially achievable with Levo
Levo

• Revised CONDEL-2 (Uht85, Uht92) + DEE
  – From CONDEL-2:
    • IQ: Instruction Queue: static instruction window
    • SSI: register and memory renaming registers
    • ISA: storage addresses, one per SSI
  • Implements: DEE-CD-MF
  • 1-to-1 correspondence with ML and DEE paths of static tree
Levo

- **DEE branch instance**
- **logical DEE path**

**SSI**

- **SSI^0 (ML)**
- **SSI^1**
- **SSI^2**
- **SSI^3**
- **SSI^4**

**IQ contents:**
- branch
- branch
- branch

Connections for ISA, etc., are similar.

a. - Broadcast bus for copying of ML state to DEE paths.
b. - Update bus for copying a DEE path state to ML path, upon a DEE branch resolving as mispredicted.

Note: a. and b. can be combined into a single bidirectional bus.
Summary

• **Disjoint Eager Execution (DEE):**
  – Optimal speculative execution
  – Realizes high ILP’s even with hard-to-predict-branch-intensive general-purpose code
  – Achieves 59% of oracle performance
  – Ideas useful elsewhere:
    • Multiprocessors
    • VLIW / software-based ILP machines
Future Work

• Simulate Levo microarchitecture in detail
  – incorporate value prediction (Lipasti96): another x10, for total ILP of x200?
• Finish design, and simulate scheduling logic
• Design and simulate critical path in VLSI
• Build a prototype
Conclusions

- Need to increase ILP to improve general-purpose computer performance
- Branches are main inhibitors of ILP
- Many BERT’s available
- DEE is a very promising new BERT....

Stay tuned!
URL:

http://www.ele.uri.edu/faculty/uht.html

(or auger down from http://www.uri.edu/)
Harmonic Mean

Oracle Speedup: 53.82

Speedup (factor X sequential)

Resources (branch paths)
Oracle Speedup: 23.22

Resources (branch paths)

Speedup (factor X sequential)

- DEE-CD-MF
- SP-CD-MF
- DEE-CD
- SP-CD
- DEE
- SP
- EE
Oracle Speedup: 25.86

Resources (branch paths)

Speedup (factor X sequential)

- DEE-CD-MF
- SP-CD-MF
- DEE-CD
- SP-CD
- DEE
- SP
- EE

compress
Oracle Speedup: 2810.48

Resources (branch paths)

Speedup (factor X sequential)

- DEE-CD-MF
- SP-CD-MF
- DEE-CD
- DEE
- SP
- EE

eqntott
Oracle Speedup: 815.62

Resources (branch paths)

DEE-CD-MF
SP-CD-MF
DEE-CD
SP-CD
DEE
SP
EE

Speedup (factor X sequential)