A Comparison of Two Pipeline Organizations

Michael Golden
Trevor Mudge

1 December 1994

Talk Outline

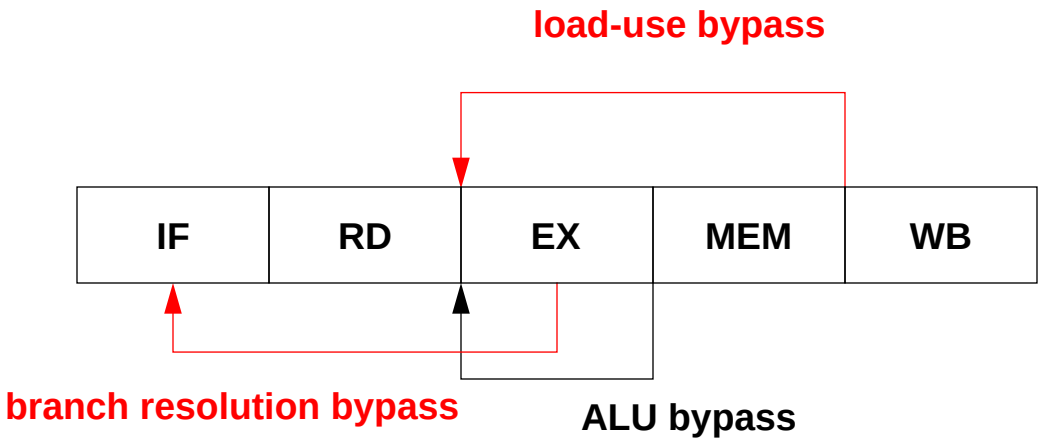
- Background
- Experiments
- Conclusions

Background

- Historically, RISC machines have used variatons of the "classic 5 stage pipeline."
  - MIPS R2000
  - MIPS R4000
  - UltraSPARC
  - Alpha 21064/21164
- Some newer machines have rearranged this pipeline
  - TFP
  - SuperSPARC

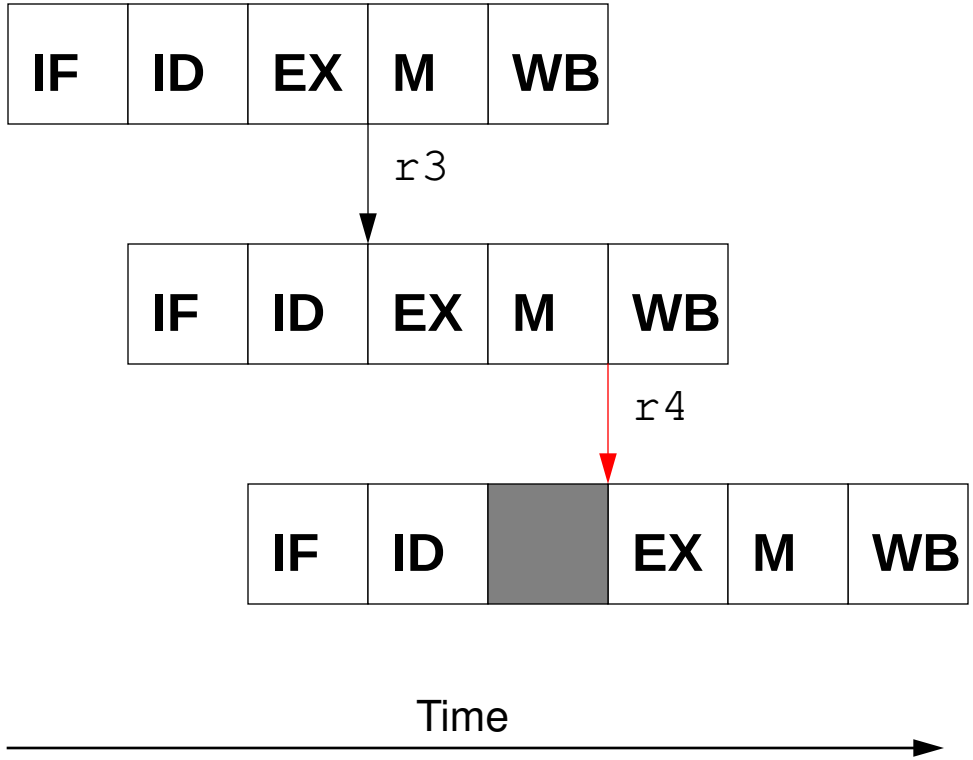
A “Classic 5-Stage Pipeline”

LUI Pipeline



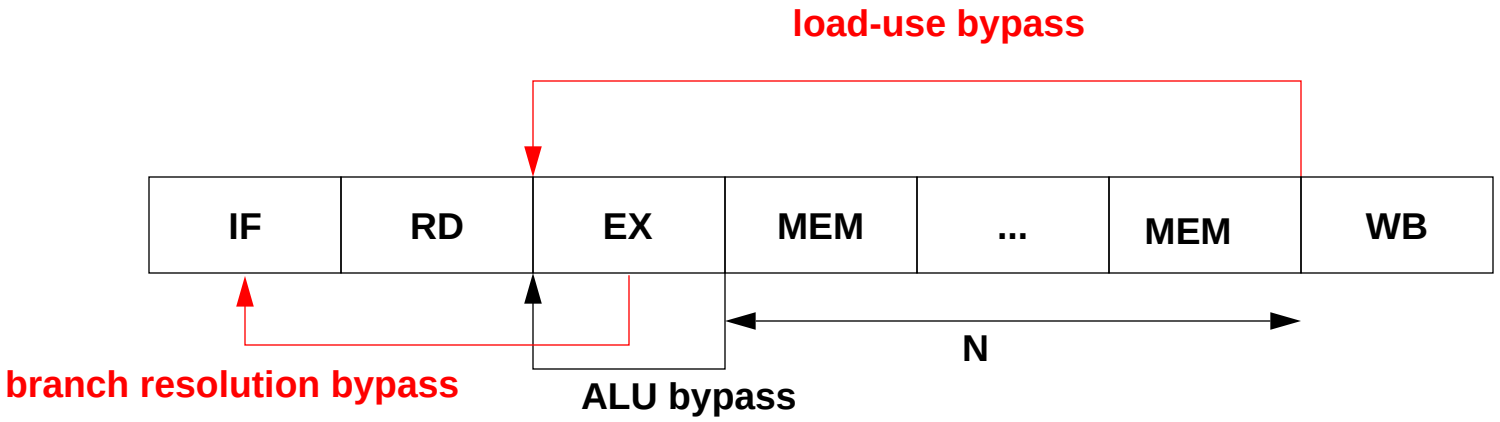
Load-Use Hazard

```
add     r3, r1, r2
lw      r4,4(r3)
add     r6, r4, r5
```



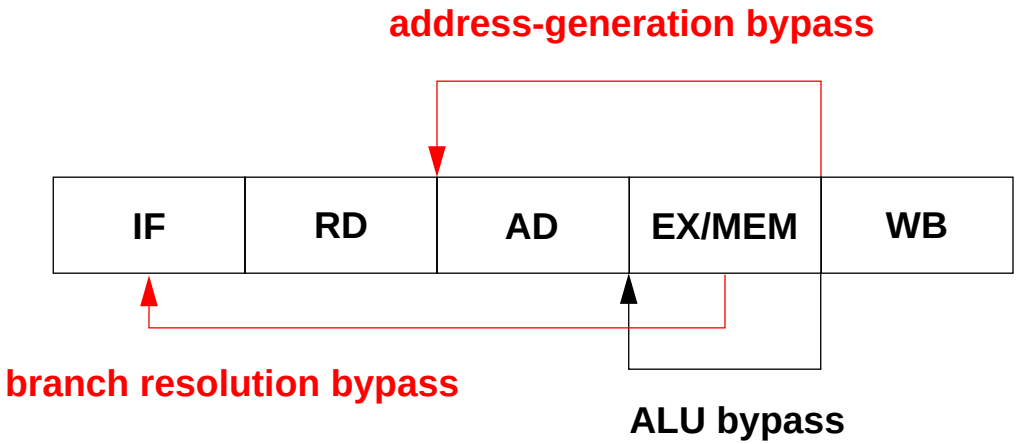
Longer Cache Access

LUI-N Pipeline

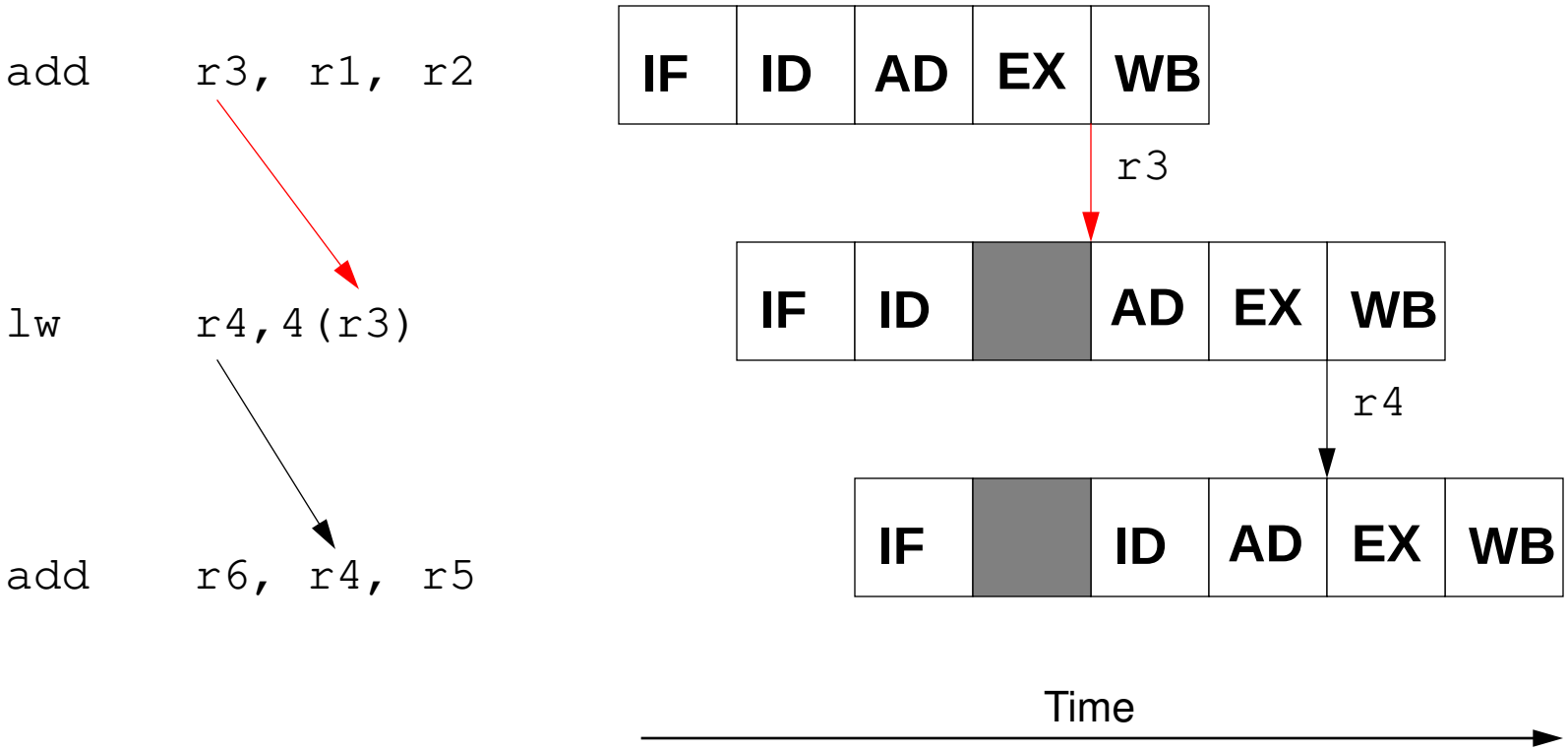


A Different Approach

AGI Pipeline

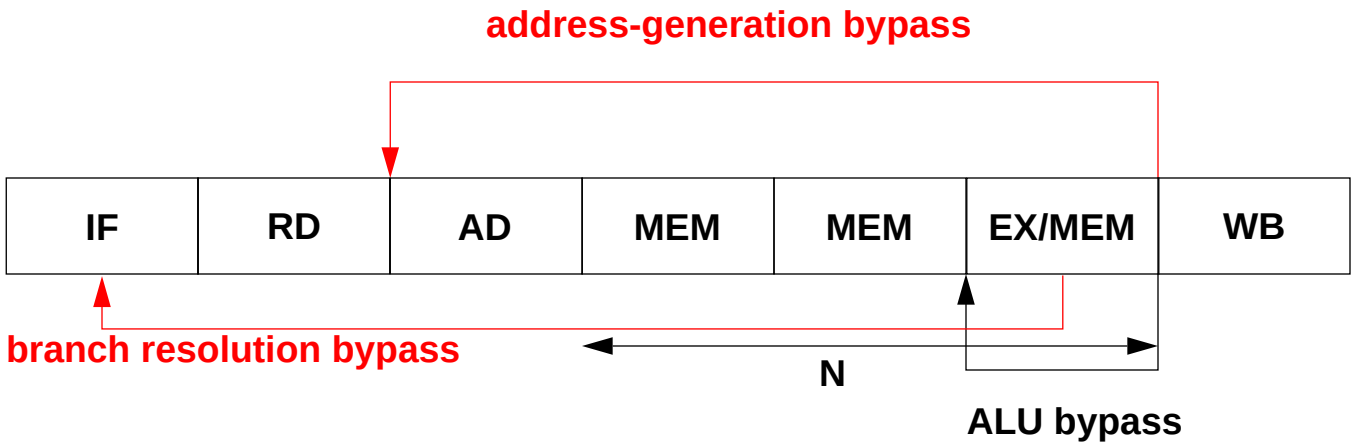


Address-Generation Hazard



Longer Cache Access Time

AGI-N Pipeline



Pipeline Comparison

LUI-N Pipeline	Inefficiency	AGI-N Pipeline
✔	Load-Use Hazard	
	Address-Generation Hazard	✔
0 cycles (+ delay slot)	Branch Penalty	N Cycles (+ delay slot)

Talk Outline

- Background
- Experiments
- Conclusions

Trace-driven simulation

- Trace produced by pixie and consumed by xsim.
- SPEC 92 integer benchmarks executed to completion.

Benchmark Name	Input File	Base Execution Time in Cycles	Average Basic Block Size
compress	reference	79 192 765	5.1
eqntott	reference	1 381 970 038	3.0
espresso	bca.in	493 384 704	5.6
gcc	stmt.i	133 778 490	5.0
sc	loada1	436 172 261	4.6
xlisp	short	1 171 528 797	3.0

Compiler Support

- GCC - The Gnu C Compiler
- Modified local scheduling algorithm
- No provision for global scheduling

Machine Model

- MIPS instruction set
  - Load delay slot replaced with a load interlock or an address-generation interlock
  - All other delay slots honored
- Memory System
  - Perfect caches
  - Single cycle to access instruction cache
  - Data cache access time varies from 1 to 7 cycles
- Branch Prediction
  - Constant branch prediction accuracy
  - $\text{Penalty} = N \times (1 - b) \times C_b$

Results

- Execution time normalized to a machine with 0-cycle cache access time.
  - Base machine has no load-use or address-generation interlocks
  - Large numbers indicate poor performance

Summary

- The AGI pipeline has better performance, even on existing binaries.
- The performance benefits increase as cache access time increases.

Summary

- Good branch prediction is necessary.

Branch Prediction Schemes

- Static Methods
  - Young and Smith, “Improving the Accuracy of Static Branch Prediction using Branch Correlation,” ASPLOS 94

    85%-95% on various benchmarks
  - Ball and Larus, “Branch Prediction for Free,” Technical Report #1137, University of Wisconsin, Feb. 1993

    Perfect Predictor: 91% (std. dev. = 5%)

    Heuristics: 80% (std. dev. = 10%)
- Dynamic Hardware Methods
  - Lee and Smith, “Branch Prediction Strategies and Branch Target Buffer Design,” IEEE Computer, January 1984

    80-98% on various benchmarks
  - Yeh and Patt, “A Comparison of Dynamic Branch Predictors that use Two Levels of Branch History,” ISCA 93

    96-97% on SPEC benchmarks (int and fp)

Conclusions

- As cache latency increases, performance benefits of the AGI pipeline increase, even without recompilation.
- Good branch prediction is required.

Future Work

- Add cache effects.
- Add multi-issue effects.
- Add real compiler support.