IA-64 Architecture Innovations

Abbreviated Version of 2/23/99 IA-64 Architecture Disclosure

intel

HEWLETT PACKARD

Agenda

- Review content disclosed at IDF
  - ◆Focus on benefits
  - ◆Focus on “What’s New?”
- Branch Handling: Predication and Prediction
- Speculation
- Register Rotation & Loop Handling

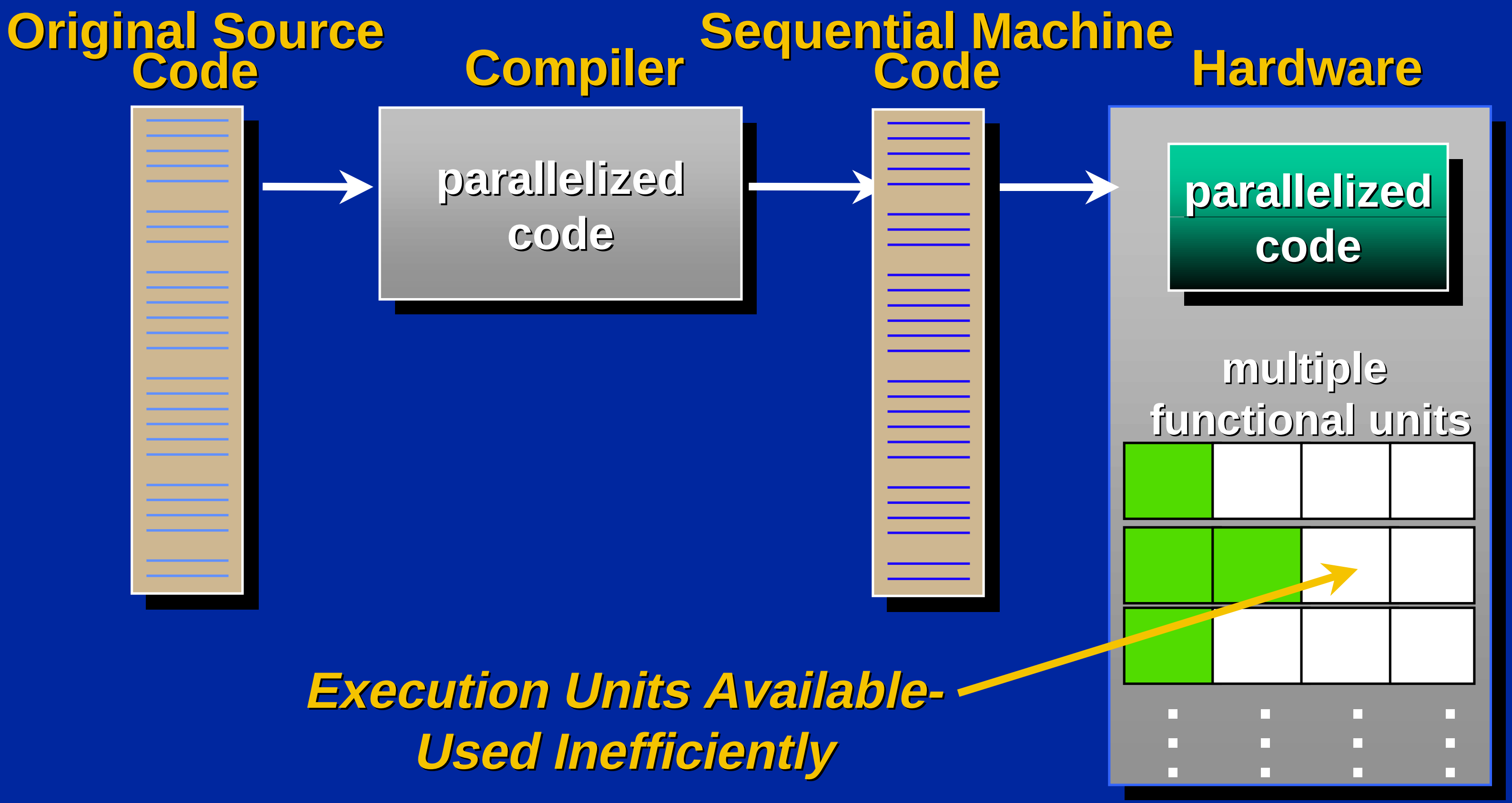
So What’s New?

- Static prediction
  - Improves prediction of always or never taken branches
- Parallel compares
  - Increased parallelism through logical combination of compares
- Hoisting uses
  - Increased scheduling flexibility improves performance
- Data Speculation
  - Moves loads above stores, increasing scheduling flexibility and performance
- Nat bits
  - Enables deferral of exceptions, supports more aggressive speculation
- Multiway branch
  - Executes multiple branches in a single cycle
- Register rotation
  - Enables wider use of software pipelining performance benefits
- Predicate rotation
  - More efficient implementation of software pipelining

intel®

HEWLETT PACKARD

Traditional Architectures: Limited Parallelism

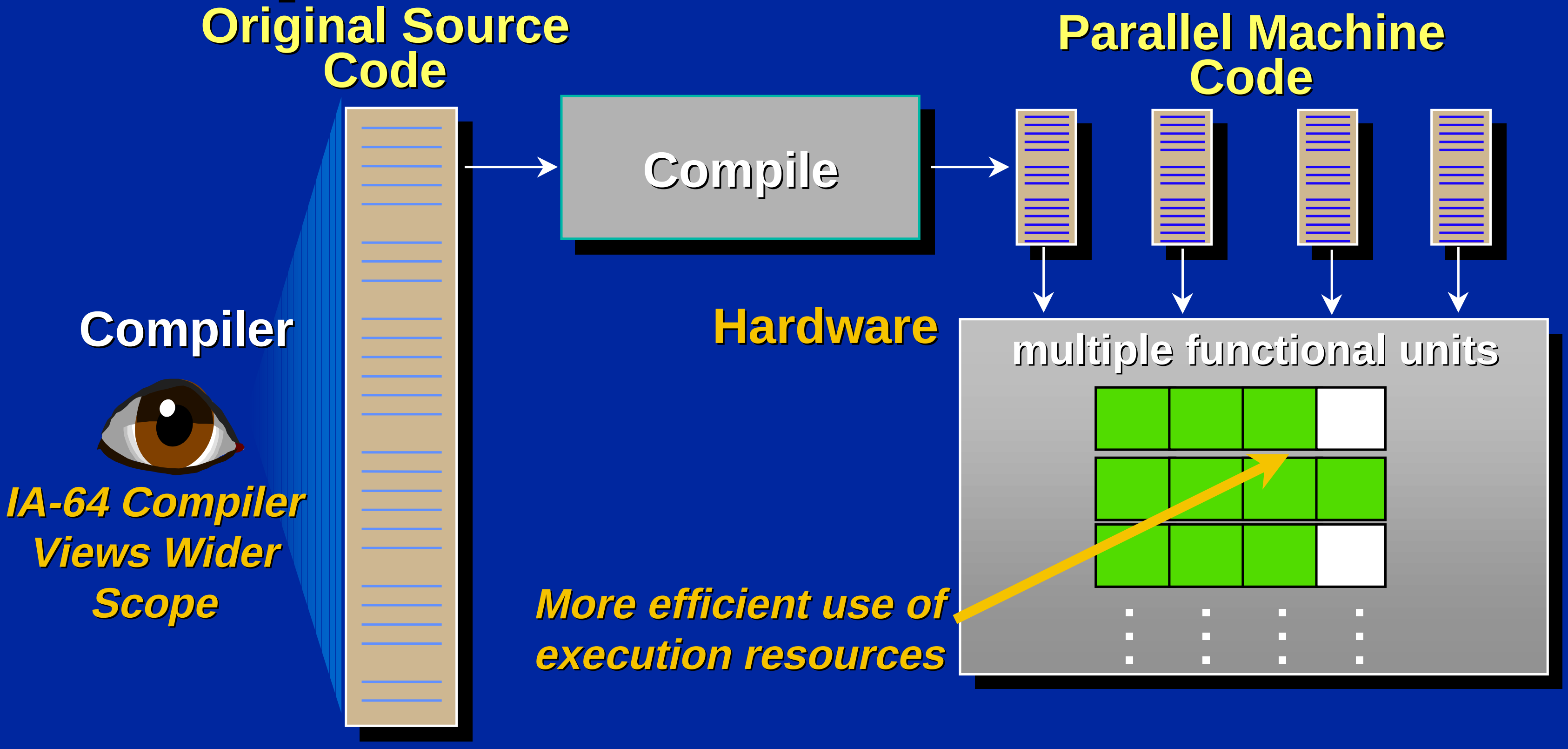


Today's Processors are often 60% Idle

intel

HEWLETT PACKARD

IA-64 Architecture: Explicit Parallelism


Original Source Code
Compile
Parallel Machine Code
Compiler
IA-64 Compiler Views Wider Scope
Hardware
multiple functional units
More efficient use of execution resources

Increases Parallel Execution

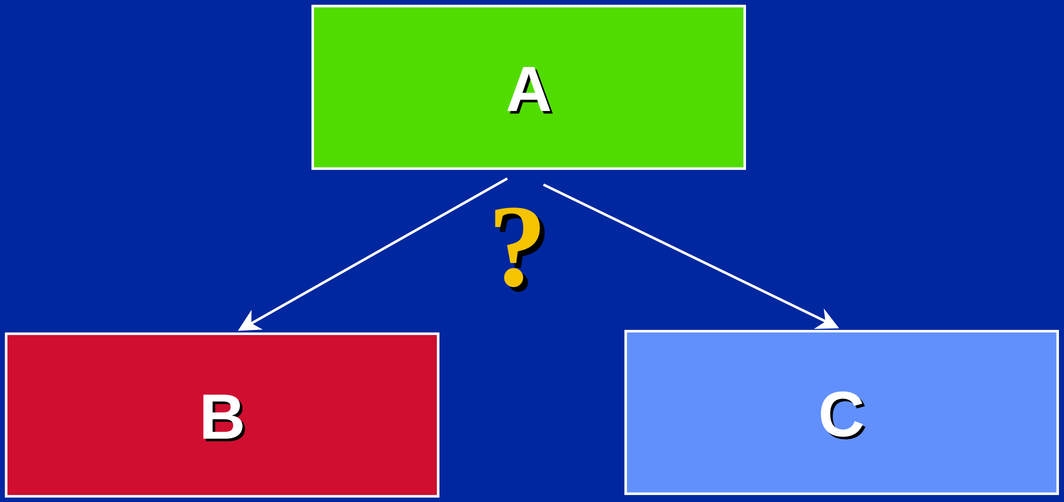
Branch Handling

Traditional Arch		IA-64
✓	Dynamic Prediction	✓
	Static Prediction on a per branch basis	✓
	Predication	✓
		& Others

- Branches are breaks in code
- Can indicate a decision
- Common in wide variety of applications

Review

Dynamic Branch Prediction



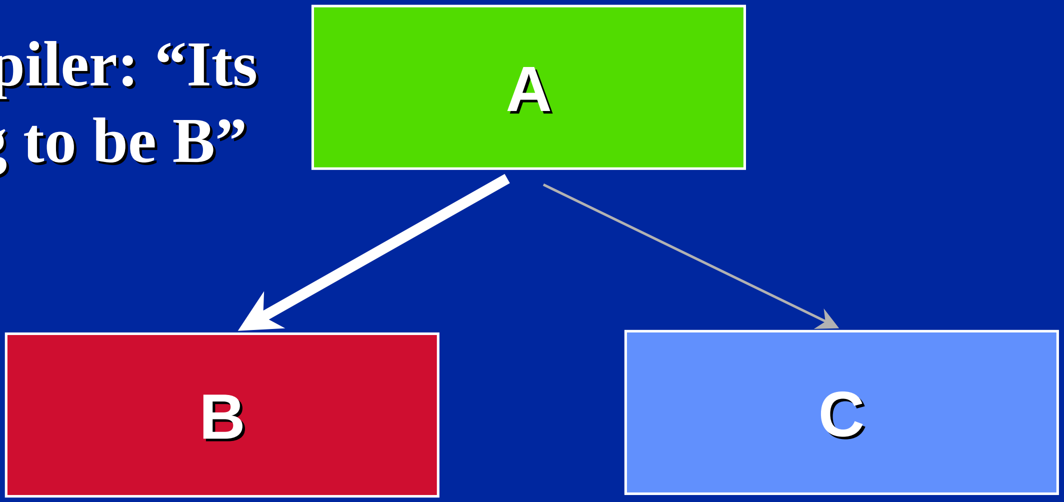
Traditional Architectures

- Guess either B or C
- Suffer performance penalty when mispredicted
- 5-10% mispredict rate can cost 40+% performance

New!

Static Branch Prediction

Compiler: "Its going to be B"

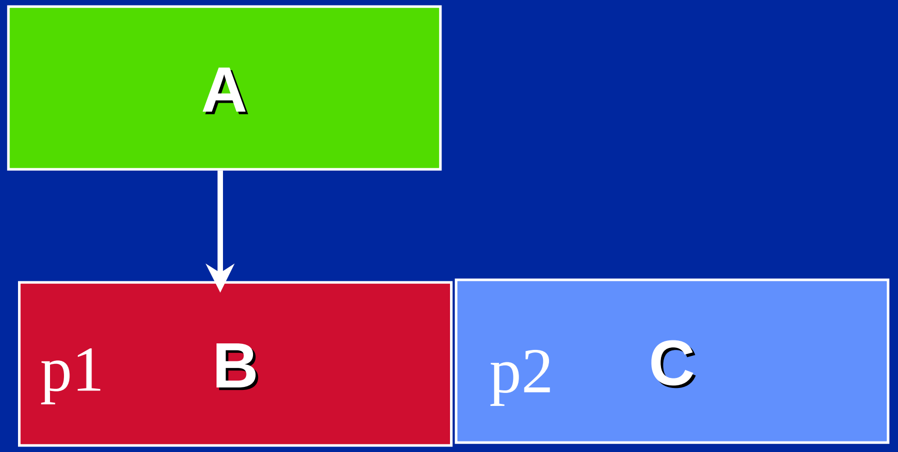


- Compiler knows which one is almost always taken or never taken
- Removes guesswork for processor, reduces mispredict penalties
- Concentrates hardware resources on other difficult to predict branches

intel®

HEWLETT PACKARD

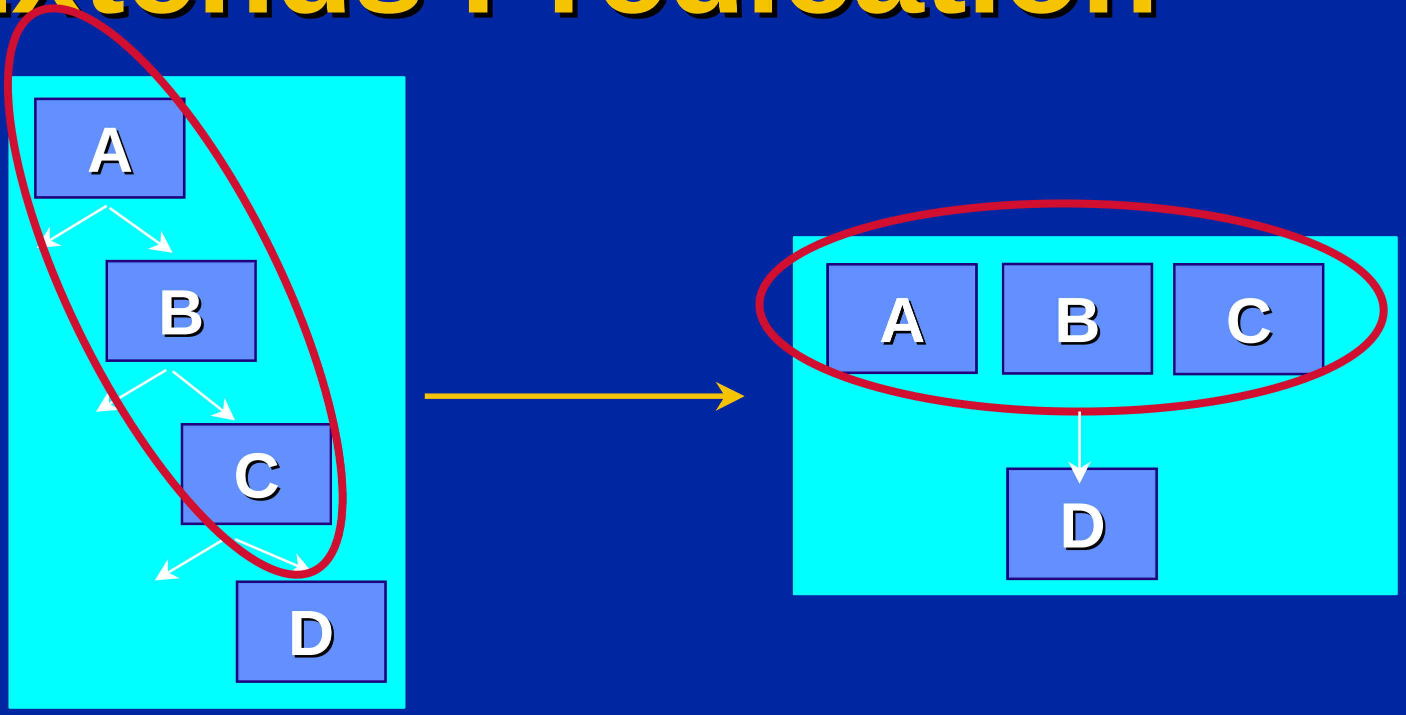
Predication



- Removes Branch, executes B&C in parallel
- Avoids possibility of mispredict
- Predicates (p1 & p2) are bits that turn on/off B & C
- Biggest benefit to code w/ hard to predict branches
    - Large server apps
    - Data sorting

New!

Parallel Compares: Extends Predication



- Reduces critical path, further increasing performance
- Enables reduction from 7 to 4 cycles on queens loop

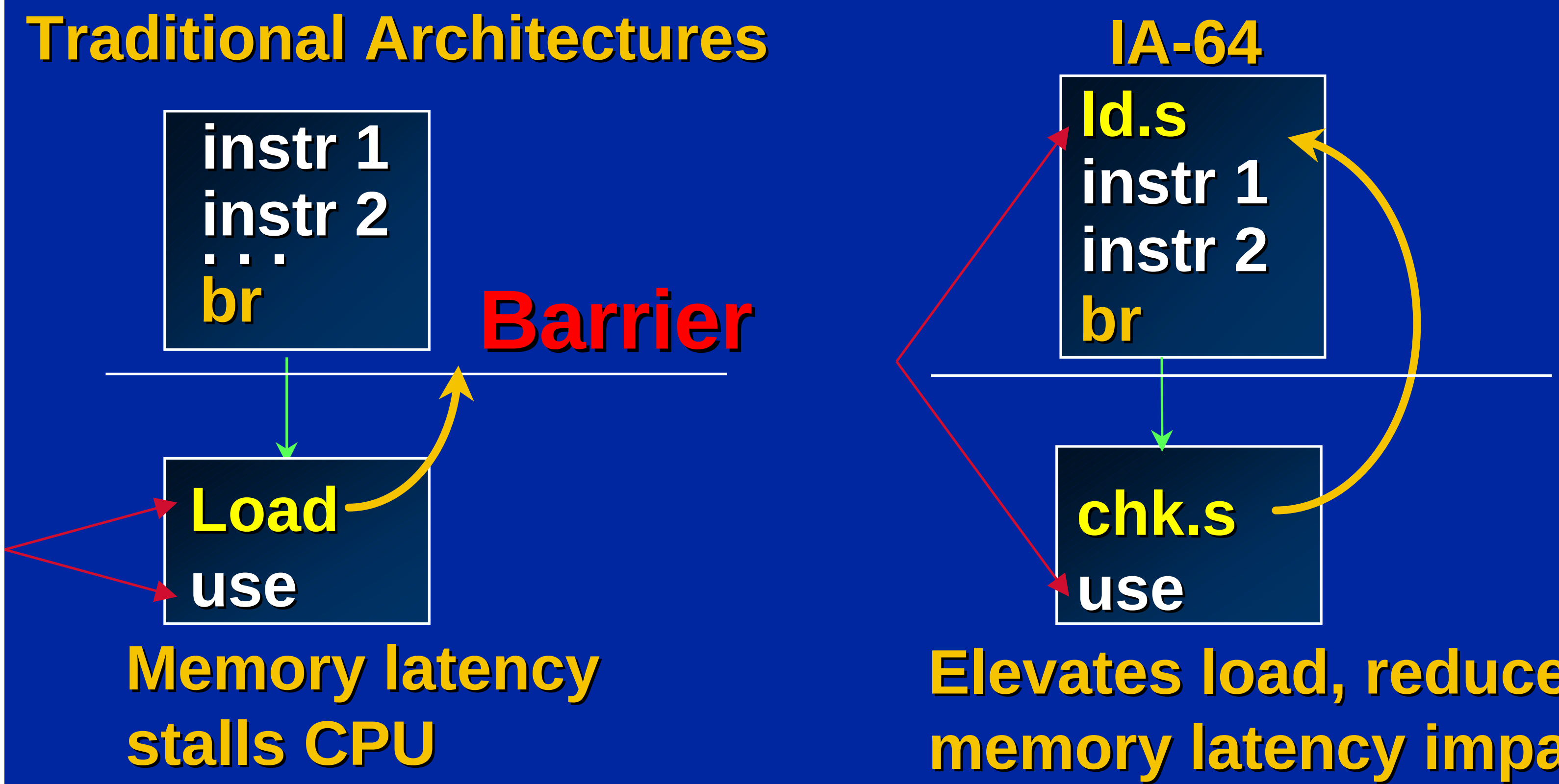
Unique feature to IA-64

Predication Benefits

- Reduces branches and mispredict penalties
  - 50% fewer branches and 37% faster code*
- Parallel compares further reduce critical paths
- Greatly improves code with hard to predict branches
  - Large server apps- capacity limited
  - Sorting, data mining- large database apps
  - Data compression
- Traditional architectures' "bolt-on" approach can't efficiently approximate predication
  - Cmove: 39% more instructions, 30% lower performance*
  - Instructions must all be speculative

* Source: S. Mahlke, 1995

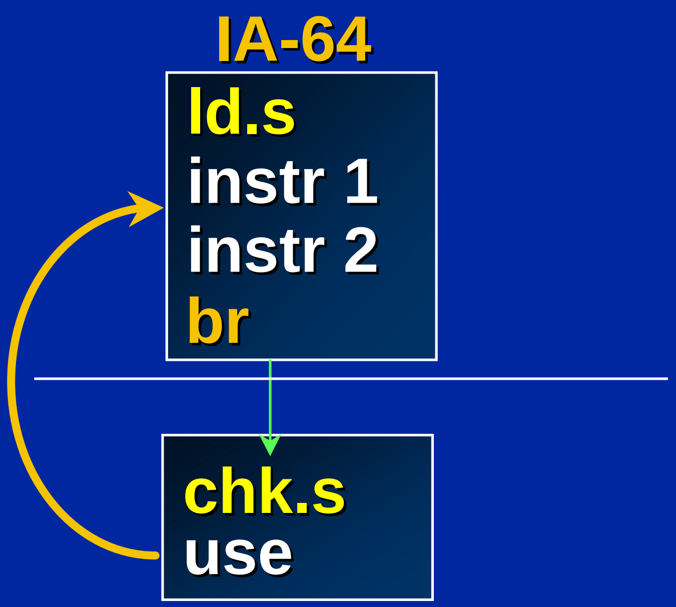
Speculation Review


Traditional Architectures
instr 1
instr 2
. . .
br
Barrier
Load
use
Memory latency stalls CPU
IA-64
ld.s
instr 1
instr 2
br
chk.s
use
Elevates load, reduces memory latency impact

Speculation overcomes the memory latency bottleneck in today's & tomorrow's systems

New!

Hoisting Uses

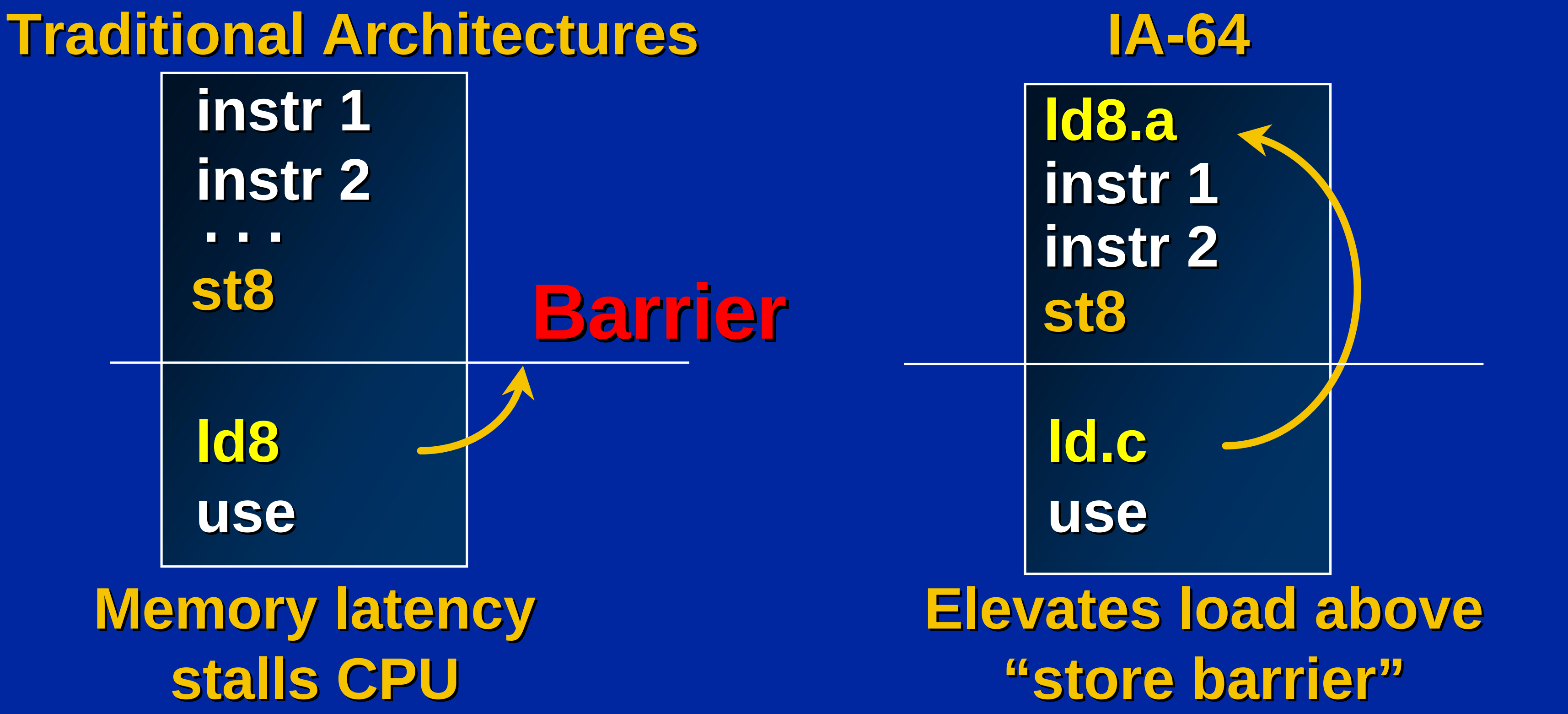


- The uses of speculative data can also be executed speculatively
- Provides additional scheduling flexibility to achieve greater parallelism

New!

Introducing Data Speculation

- Compiler can issue a load prior to a preceding, possibly-conflicting store


Traditional Architectures
instr 1
instr 2
. . .
st8
Barrier
ld8
use
Memory latency stalls CPU
IA-64
ld8.a
instr 1
instr 2
st8
ld.c
use
Elevates load above "store barrier"

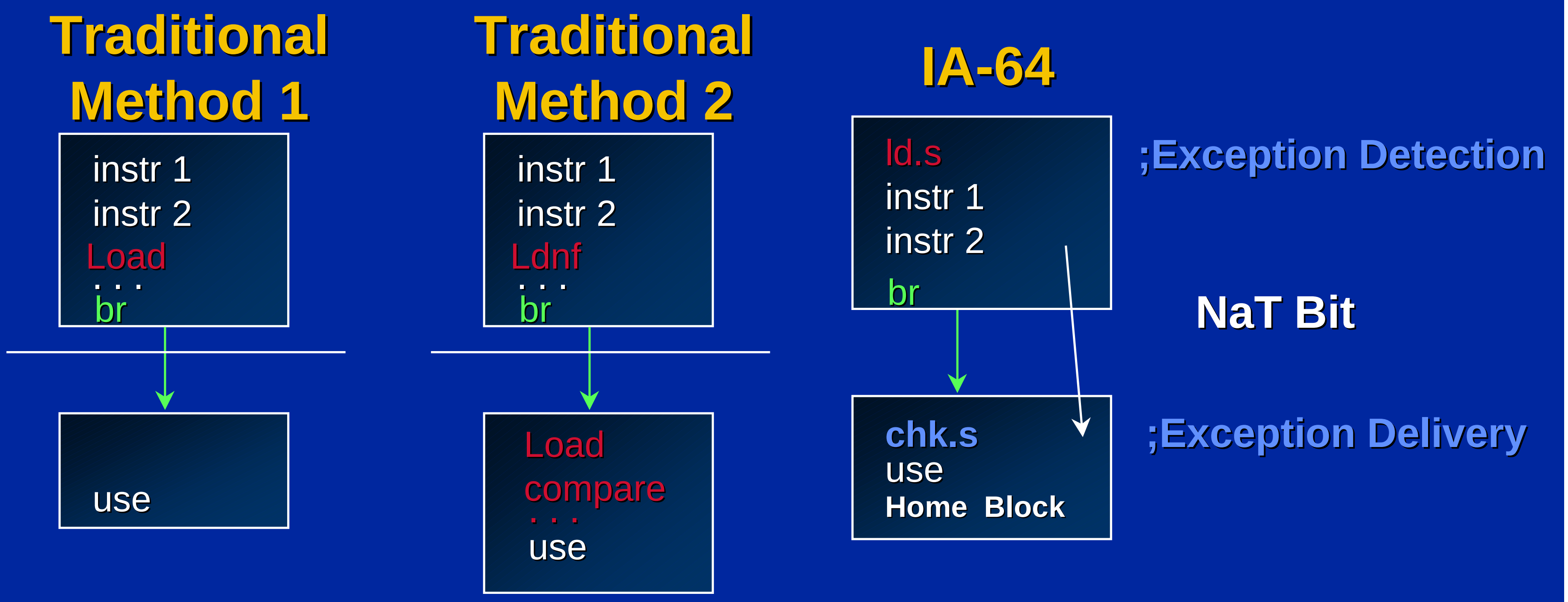
Unique feature to IA-64

intel

HEWLETT PACKARD

New!

Exception Handling



Only in limited instances

Extra baggage to ensure integrity & can't hoist uses (e.g. non-faulting load)

Widely applicable, NaT bit ensures integrity of data without penalty

intel

HEWLETT PACKARD

Speculation Benefits

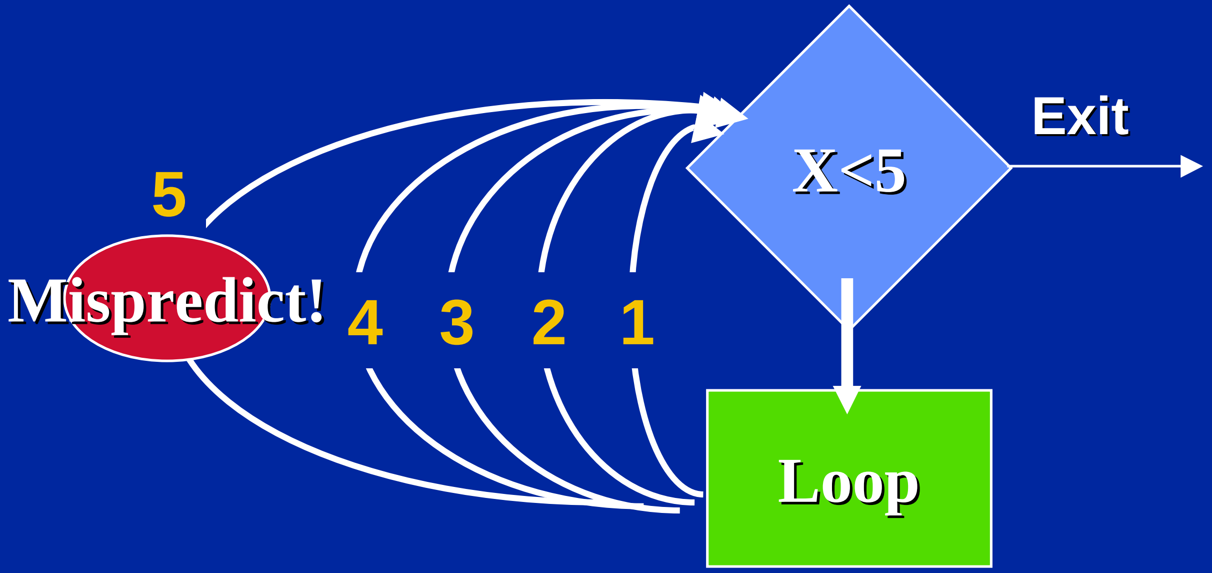
- Reduces impact of memory latency
  - Performance improvement at 79% when combined with predication*
- Greatest improvement to code with many cache accesses
  - Large databases
  - Operating systems
- Scheduling flexibility enables new levels of performance headroom

* August et.al., 1998

New!

Loop Handling

“Do loop 4 times”

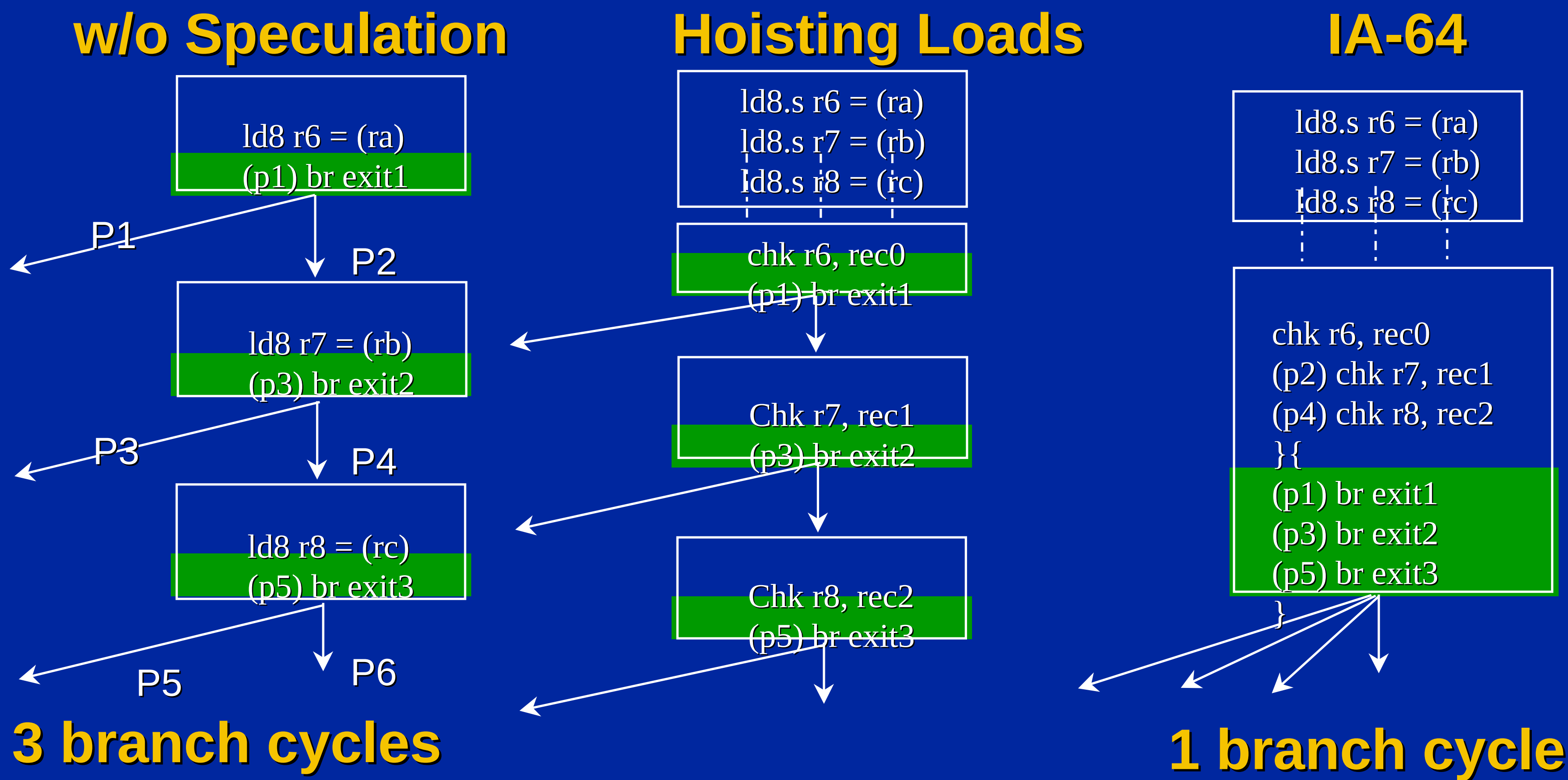


IA-64 has special register to accelerate loops & avoid mispredicts: called Loop Counter (LC)

Improves integer code performance

New!

Multi-way Branch



- Multiway branches: more than 1 branch in a single cycle
  - Chaining multiway branches allows n-way branching

intel

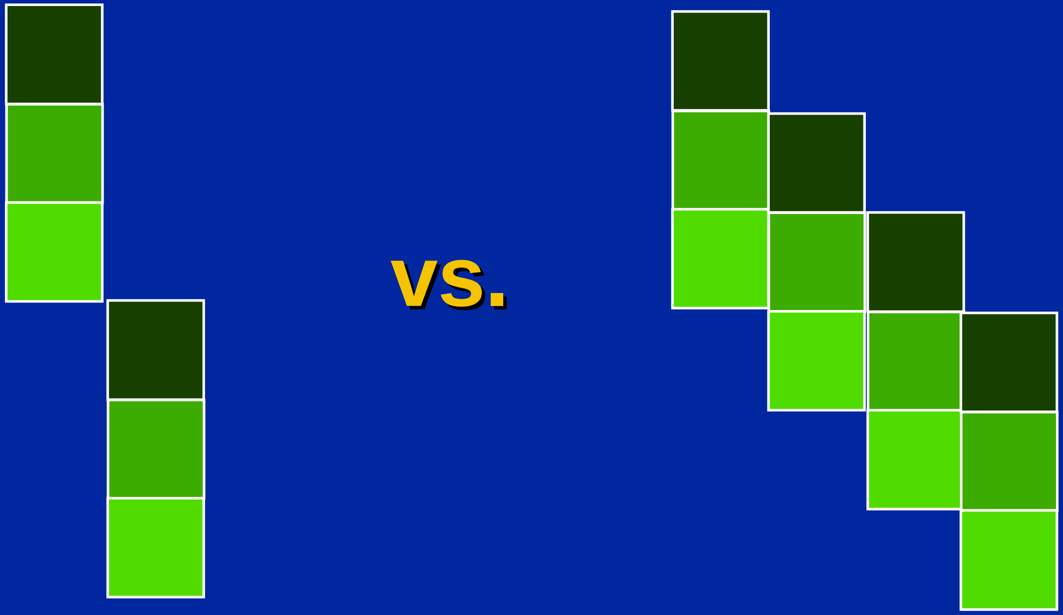
HEWLETT PACKARD

01/04/99

New!

Software Pipelining

- Overlapping execution of different loop iterations



- More iterations in same amount of time

IA-64 Offers an Innovative Approach

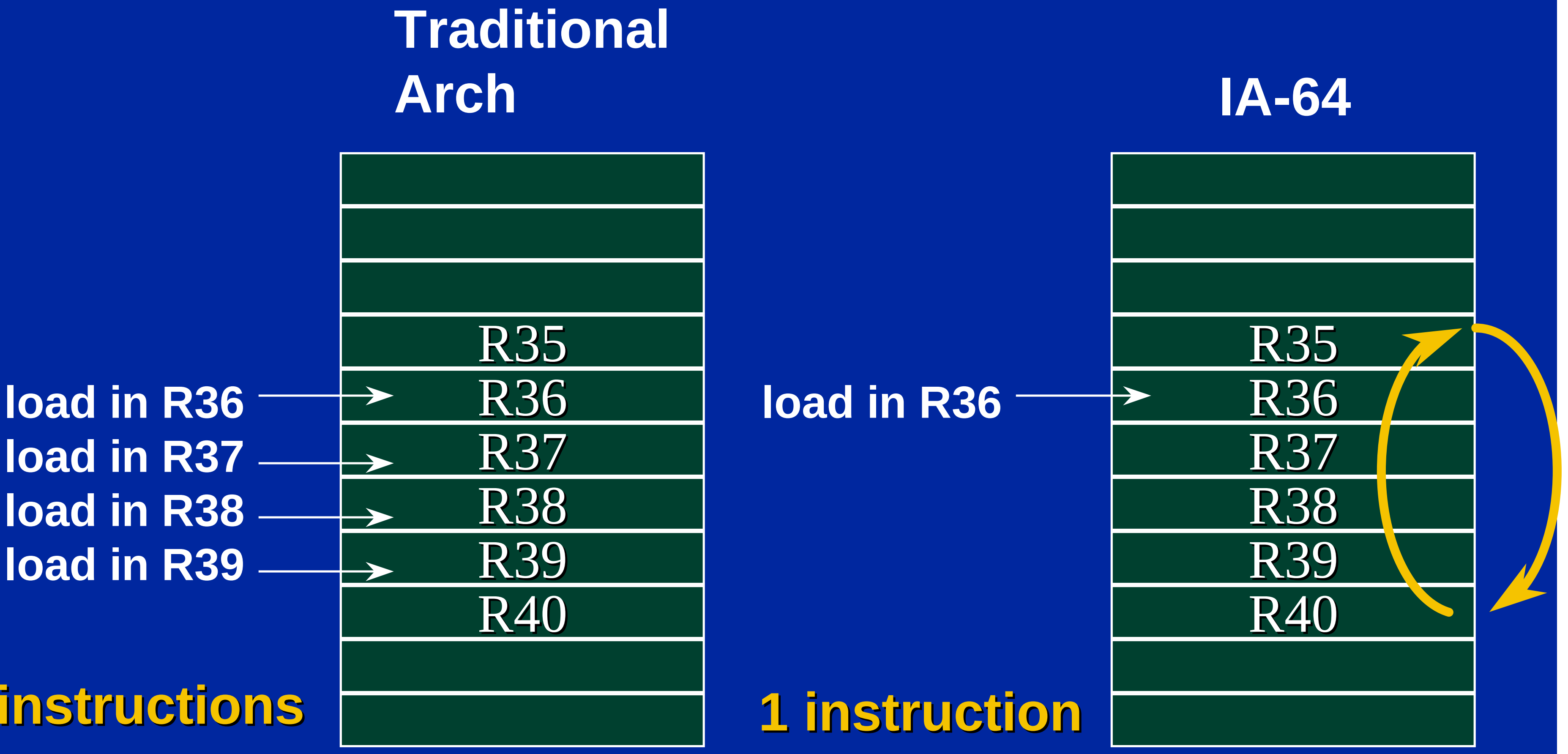
Software Pipelining

- IA-64 features that make this possible
  - Full Predication
  - Special branch handling features
  - Register rotation: removes loop copy overhead
  - Predicate rotation: removes prologue & epilogue
- Traditional architectures use loop unrolling
  - High overhead: extra code for loop body, prologue, and epilogue

Especially Useful for Integer Code With Small Number of Loop Iterations

New!

Introducing Rotating Registers



Improves performance without code expansion

intel

HEWLETT PACKARD

Software Pipelining Benefits

- Loop pipelining maximizes performance; minimizes overhead
  - Avoids code expansion of unrolling and code explosion of prologue and epilogue
  - Smaller code means fewer cache misses
  - Greater performance improvements in higher latency conditions
- Reduced overhead allows S/W pipelining of small loops with unknown trip counts
  - Typical of integer scalar codes

Reviewing What’s New:

- Parallel compares
- Tbit
- Nat bits
- Deferral
- Hoisting uses
- Propagation
- Branch instructions
- Static prediction
- Loop branches
- LC register
- EC register
- Multiway branch
- Branch registers
- Register rotation
- Predicate rotation
- RRBs

Feature Comparison

Traditional Architectures	IA-64
Dynamic branch prediction	Branch specific Static Prediction and Predication enhance dynamic prediction to reduce mispredict penalties
Conditional moves limited in applicability and require additional instructions	Predication widely applicable, parallel compares further enhance benefit
Non faulting loads limited to certain conditions or require additional instructions	Control and data speculation enable greater scheduling freedom of loads
Software pipelining limited to large loops due to code size explosion	Rotating registers and rotating predicates allow wide application of software pipelining performance benefits without the code growth

Summary

- Predication removes branches
  - ◆Eliminates branches & mispredicts, increases ILP
  - ◆Good for large database applications
- Speculation reduces memory latency
  - ◆Enhances ILP and scalability
  - ◆Good for variety of server applications
    - (databases, OLTP, etc.)
- S/W pipelining support enables broad usage
  - ◆Performance for small integer loops with unknown trip counts as well as monster FP loops