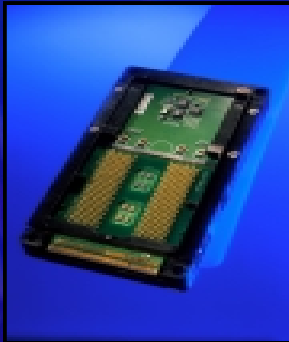
IA-64 Architecture

Overview

A High-Performance Computing Architecture



Stefan.Lamberts@intel.com
Internet Solutions Group EMEA
Technical Marketing

July 2000

intel

Agenda

- Motivation and IA-64 feature overview
- IA-64 features
  - EPIC
  - Data types, memory and registers
  - Register stack
  - Predication and parallel compares
  - Software pipelining and register rotation
  - Control & data speculation
  - Branch architecture
  - Integer architecture
  - Floating point architecture
- Itanium™ processor overview
- Itanium™ processor based systems overview
- Operating systems, tools and programming

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

IA-64: Extending the Intel® Architecture

- **Designed for High Performance Computing**
  - **Scientific**
  - **Technical & Engineering**
  - **Business**
- **New EPIC Technology**
- **IA-64 Architecture uses EPIC**
- **Itanium™ processor is the first implementation of IA-64**

IA-64 Architecture

performance, scalability, availability, compatibility

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Performance Limiters

- Parallelism not fully utilized
  - Existing architectures cannot exploit sufficient parallelism in integer code to feed a wide in-order implementation
- Branches
  - Even with perfect branch prediction, small basic blocks of code do not fully utilize machine width
- Procedure Calls
  - Software modularity is becoming standard resulting call/return overhead
- Memory latency and address space
  - Increasing relative to processor cycle time (larger cache miss penalties) and limited address space

IA-64 *IA-64 overcomes these limitations, and more !*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

IA-64 Architectural Features

Parallelism

Branches

Procedure Calls

Memory Latency

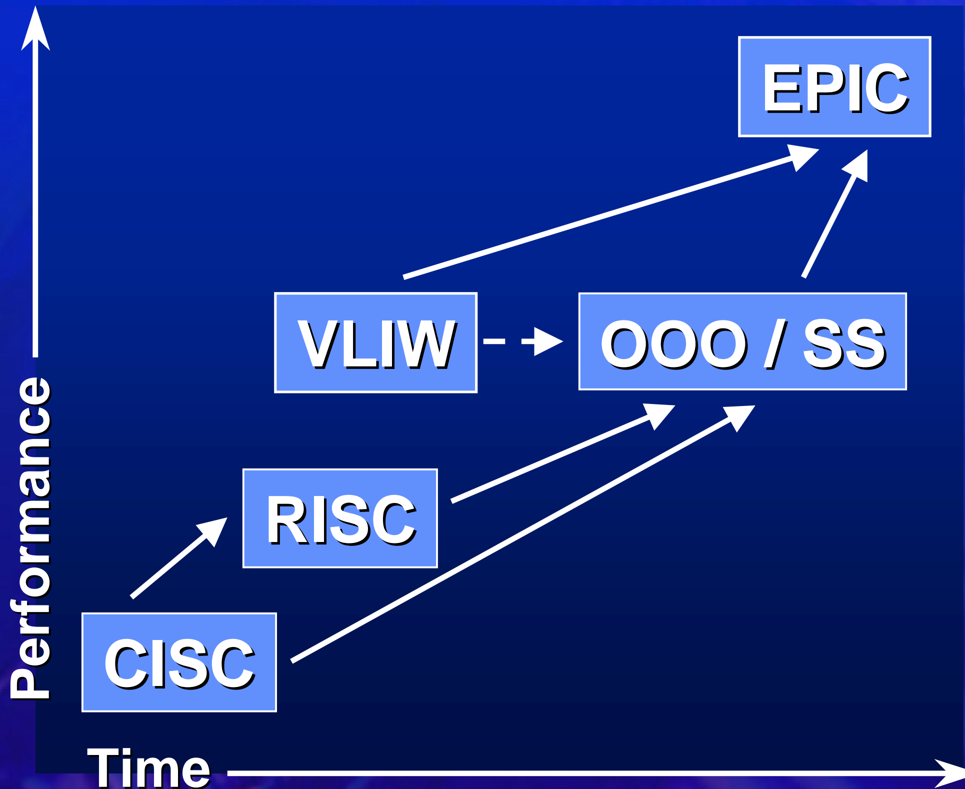
- 64-bit Address Flat Memory Model
- Explicit Parallel Instruction Computing
- Large Register Files
- Automatic Register Stack Engine
- Predication
- Software Pipelining Support
- Register Rotation
- Sophisticated Branch Architecture
- Loop Control Hardware
- Control & Data Speculation
- Cache Control
- Powerful Integer Architecture
- Advanced Floating Point Architecture
- Multimedia Support (MMX™ Technology)

IA-64 *It's more than just 64 bits …*

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Next Generation Architecture



EPIC Design Philosophy

- Maximize performance via hardware & software synergy
- Advanced features enhance instruction level parallelism
  - Predication, Speculation, …
- Massive hardware resources for parallel execution

Beyond Traditional Architectures

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Agenda

- Motivation and IA-64 feature overview
- IA-64 features
  - EPIC
  - Data types, memory and registers
  - Register stack
  - Predication and parallel compares
  - Software pipelining and register rotation
  - Control & data speculation
  - Branch architecture
  - Integer architecture
  - Floating point architecture
- Itanium™ processor overview
- Itanium™ processor based systems overview
- Operating systems, tools and programming

intel®

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

EPIC Instruction Parallelism

Source Code

Instruction Groups (series of bundles)

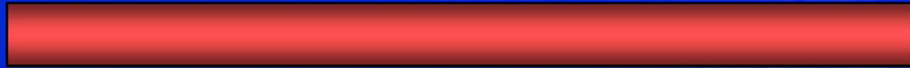
- No RAW or WAW dependencies
- Issued in parallel depending on resources

Instruction Bundles (3 Instructions)

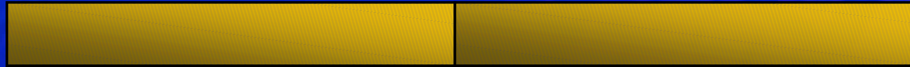
- 3 instructions + template
- 3 x 41 bits + 5 bits = 128 bits

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

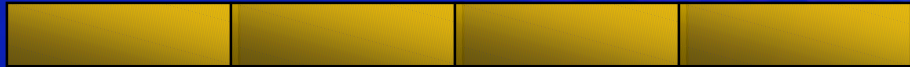
Compiler (HW) Data Types



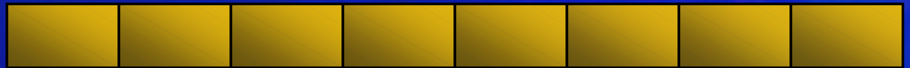
64-bit Integer



2x32-bit SIMD Integer

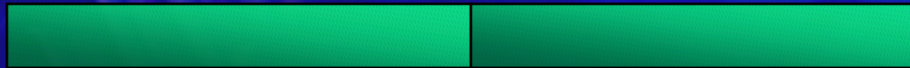


4x16-bit SIMD Integer



8x8-bit SIMD Integer

64-bit DP F.P.



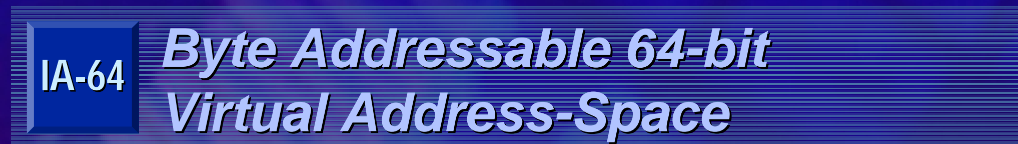
2x32-bit SIMD SP-F.P.

IA-64 *All common data types are supported*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

64-bit Memory Access

- 18 BILLION Giga Bytes accessible
  - 2^{64} == 18,446,744,073,709,551,616
- Byte addressable access with 64-bit pointers
  - 64-bit virtual address space
  - HW support for 32-bit pointers
- Access granularity and alignment
  - 1,2,4,8,10,16 bytes
  - Alignment on naturally aligned boundaries is recommended
  - Instructions are always 16-byte aligned
- Support for both Big and Little endian byte order
- Memory hierarchy control



Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Memory Hierarchy Control

- **Software can explicitly control memory accesses**
  - Specify levels of the memory hierarchy affected by the access
  - Allocation and Flush resolution is at least 32-bytes
- **Allocation (Prefetch)**
  - Allocation implies bringing the data close to the CPU
  - Allocation hints indicate at which level allocation takes place
  - Used in load, store, and explicit pre-fetch instructions
- **De-allocation and Flush**
  - Invalidates the addressed line in all levels of cache hierarchy
  - Write data back to memory if necessary

Three levels of cache (full speed L2 cache, 2/4MB L3-cache) & Atomic operation support

IA-64 *Control over Cache (De)Allocation*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

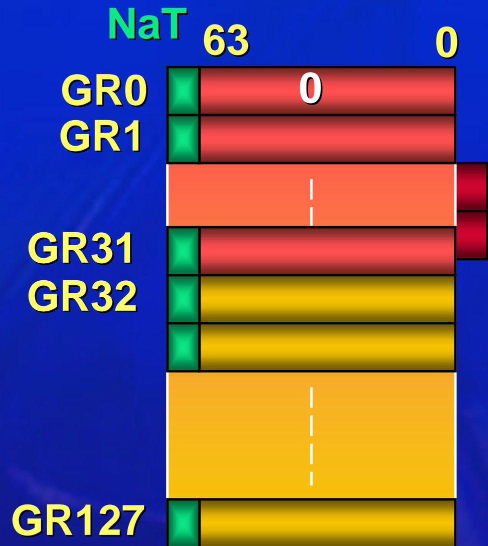
Memory Access Ordering

- Explicit control
  - Memory Fence mf - ensures all prior memory operations are seen prior to all future memory operations
  - Acquire Load ld.acq - ensure I am seen prior to all future memory operations
  - Release store st.rel - ensure that all prior memory operations are seen prior to me
  - Synchronize instruction caches sync.i - Ensure all instruction caches have seen all prior flush cache instructions
- Implicit - applicable to semaphore instructions
  - xchg — Exchange mem and General Register (GR)
  - cmpxchg — Conditional exchange of mem and GR
  - fetchadd — Add immediate to memory
- Strong ordering model is compatible with IA-32 Ordering

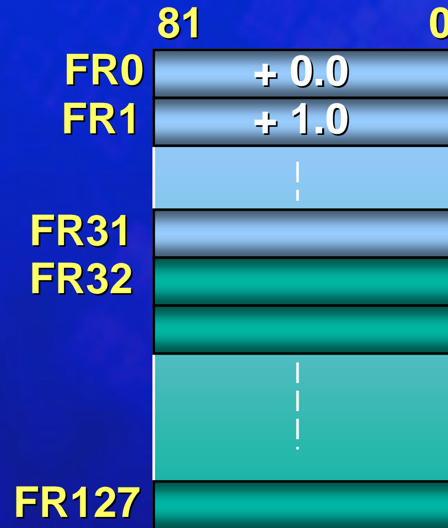
Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Large Register Set

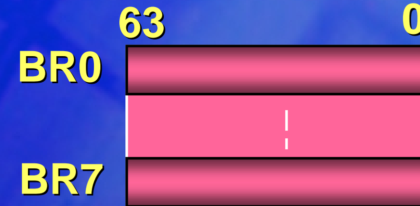
Integer Registers



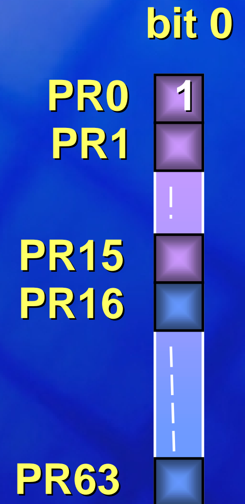
FP Registers



Branch Registers



Predicate Registers



32 Static

96 Framed, Rotating

32 Static

96 Rotating

16 Static

48 Rotating

IA-64 *Remove Resource Bottlenecks*

intel®

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

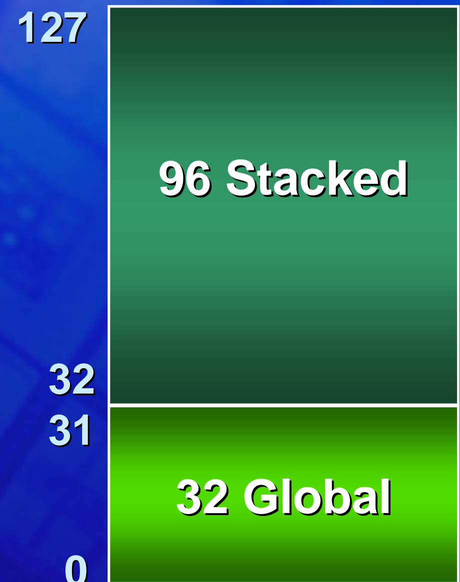
Context Switch

- "Normal"
  - full context switch saves all registers
  - GR and FR
- "Lazy"
  - saves only GR registers or specific range (GR0-31, GR32-GR127)
- "Fast"
  - doesn't save any registers and uses 16 separate banked "shadow" registers '(OS only, GR16'-GR31')
  - e.g. interrupt and exception handling

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Register Stack

- GRs 0-31 are global to all procedures
- Stacked registers begin at GR32 and are local to each procedure
- Each procedure's register stack frame varies from 0 to 96 registers
- Only GRs implement a register stack
  - The FRs, PRs, and BRs are global to all procedures
- Register Stack Engine (RSE)
  - Upon stack overflow/underflow, registers are saved/restored to/from a backing store transparently



IA-64 *Optimized CALL/RETURN and Parameter Passing*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

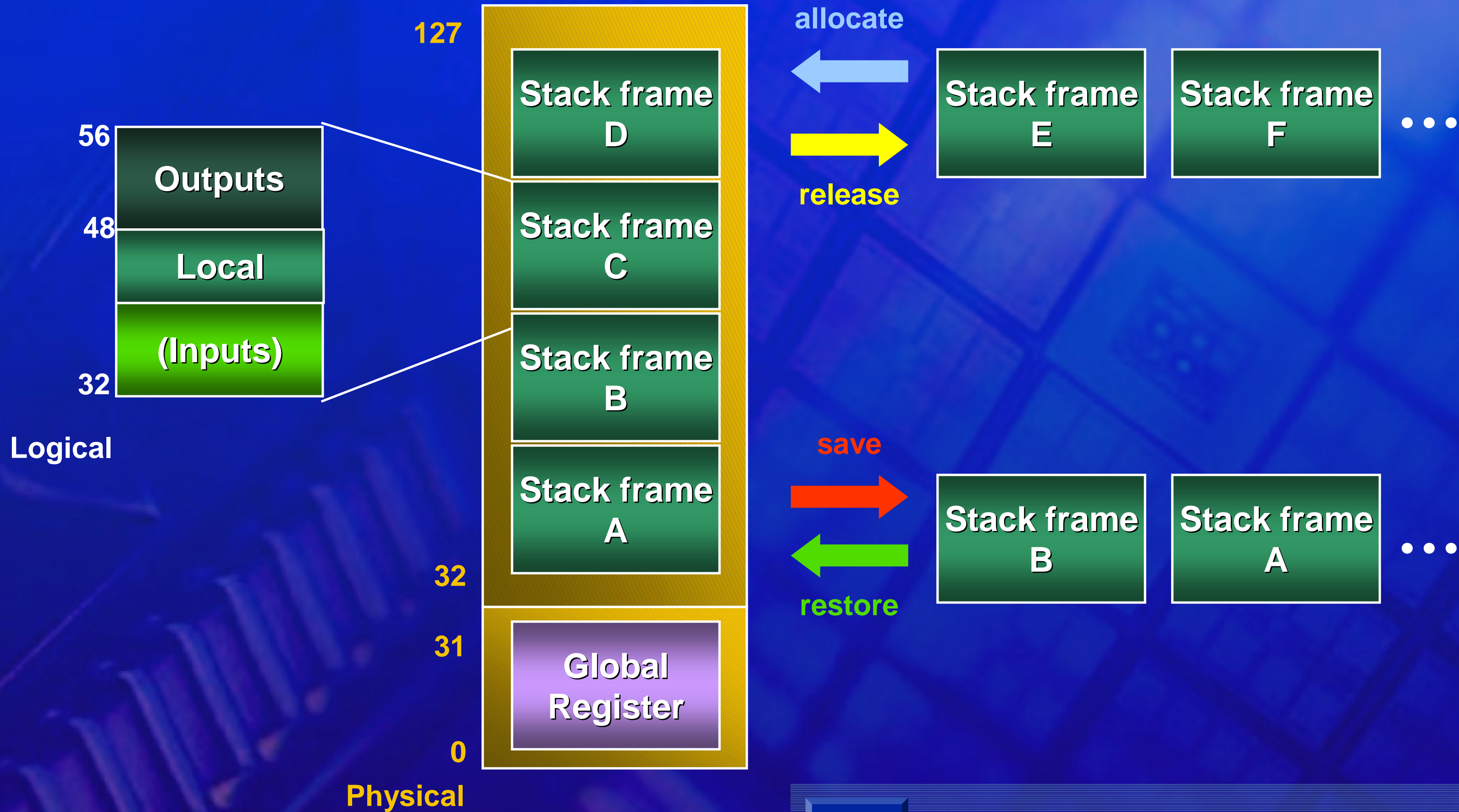
Register Stack in Work

- Call changes frame to contain only the caller's output
- Alloc instr. sets the frame region to the desired size
  - Three architecture parameters: local, output, and rotating
- Return restores the stack frame of the caller

IA-64 *Avoids Register Spill/Fill among Procedure Calls*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Register Stack Engine

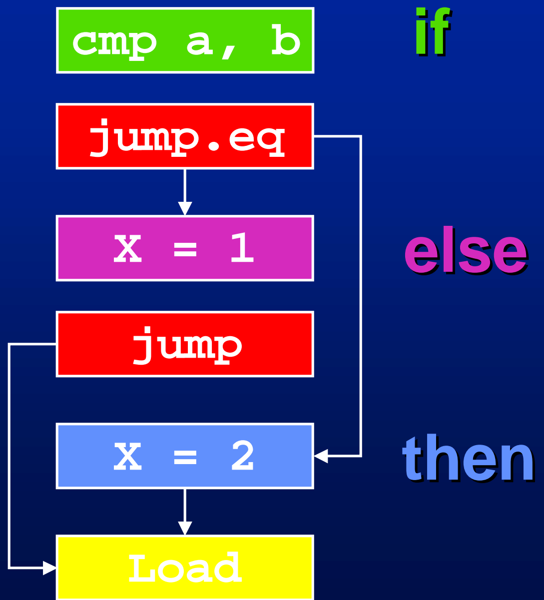


IA-64 *GR (Integer) Registers only*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Predication: Control Flow to Data Flow

Traditional Arch.

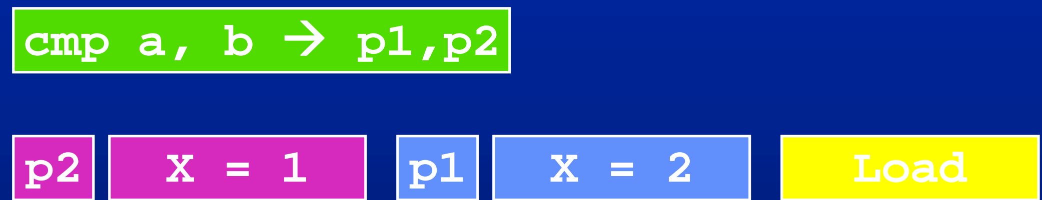


if

else

then

IA-64



- Conditional execution based on qualifying predicate
- 64 predicate registers
- Can be combined with logical operations

IA-64 *Removes/Reduces Branches and Enables Parallel Execution*

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Predication ...

- **Unpredictable branches removed**
  - Misprediction penalties eliminated
- **Basic block size increases**
  - Compiler has a larger scope to find ILP
- **ILP within the basic block increases**
  - Both "then" and "else" executed in parallel
- **Wider machines are better utilized**

IA-64 *Predication Enables and Enhances ILP*

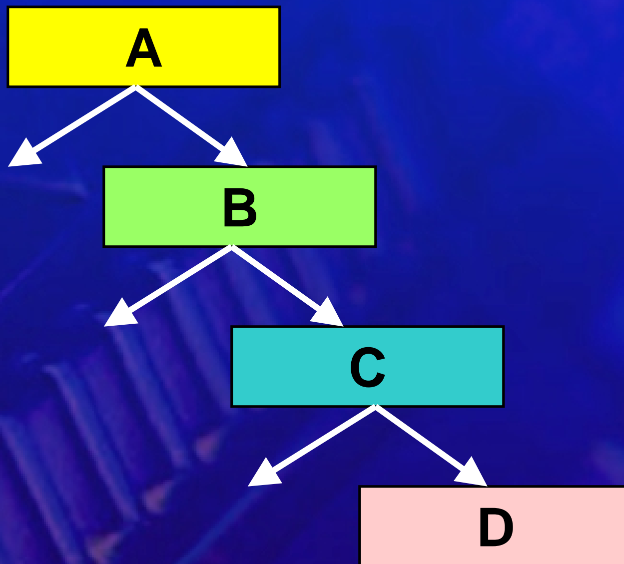
intel ITANIUM intel inside

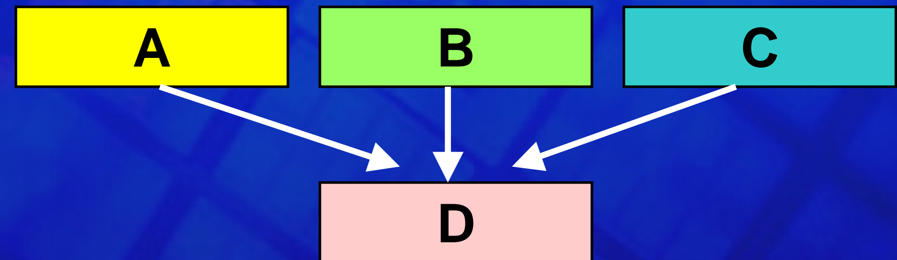
Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Parallel Compares

- **Three new types of compares:**
  - **AND: both target predicates set FALSE if compare is false**
  - **OR: both target predicates set TRUE if compare is true**
  - **ANDOR: if true, stets one TRUE, set other FALSE**

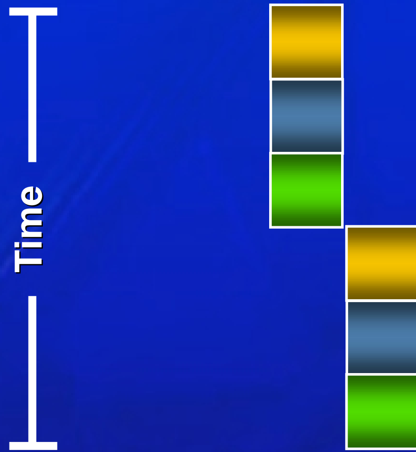


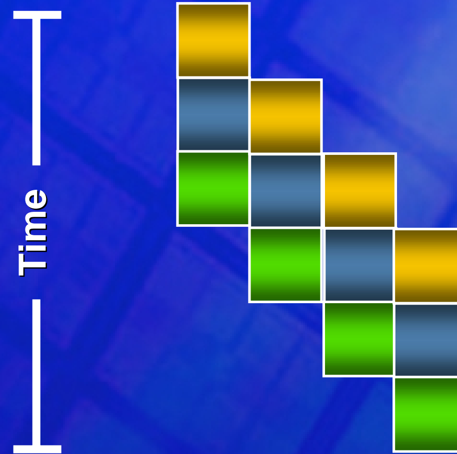


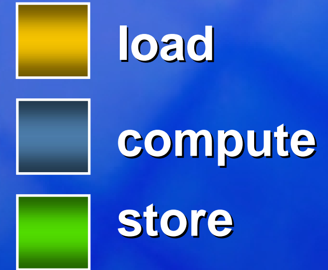
IA-64 *Reduces Critical Path*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Software Pipelining







- Traditional architectures use loop unrolling
  - Results in code expansion and increased cache misses
- IA-64 Software Pipelining uses rotating registers
  - Allows overlapping execution of multiple loop instances
- Predication controls the pipeline stages

IA-64 *Provides Direct Support for Software Pipelining*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Software Pipelining

- **IA-64 features that make this possible**
  - **Full predication to define pipeline stages**
  - **Special branch handling features**
    - Loop branches
    - Special loop registers (LC, EC)
  - **Register rotation: removes loop copy overhead**
  - **Predicate rotation: removes prologue & epilog**
- **Traditional architectures use loop unrolling**
  - **High overhead: extra code for loop body, prologue and epilog**
  - **Consumes a large number of registers**

intel®

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Software Pipelining

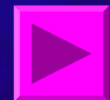
Loop Iteration

p16 (load)	p17	p18 (compute)	p19 (store)
1	0	0	0
1	1	0	0
1	1	1	0
1	1	1	1
1	1	1	1
0	1	1	1
0	0	1	1
0	0	0	1

Prologue

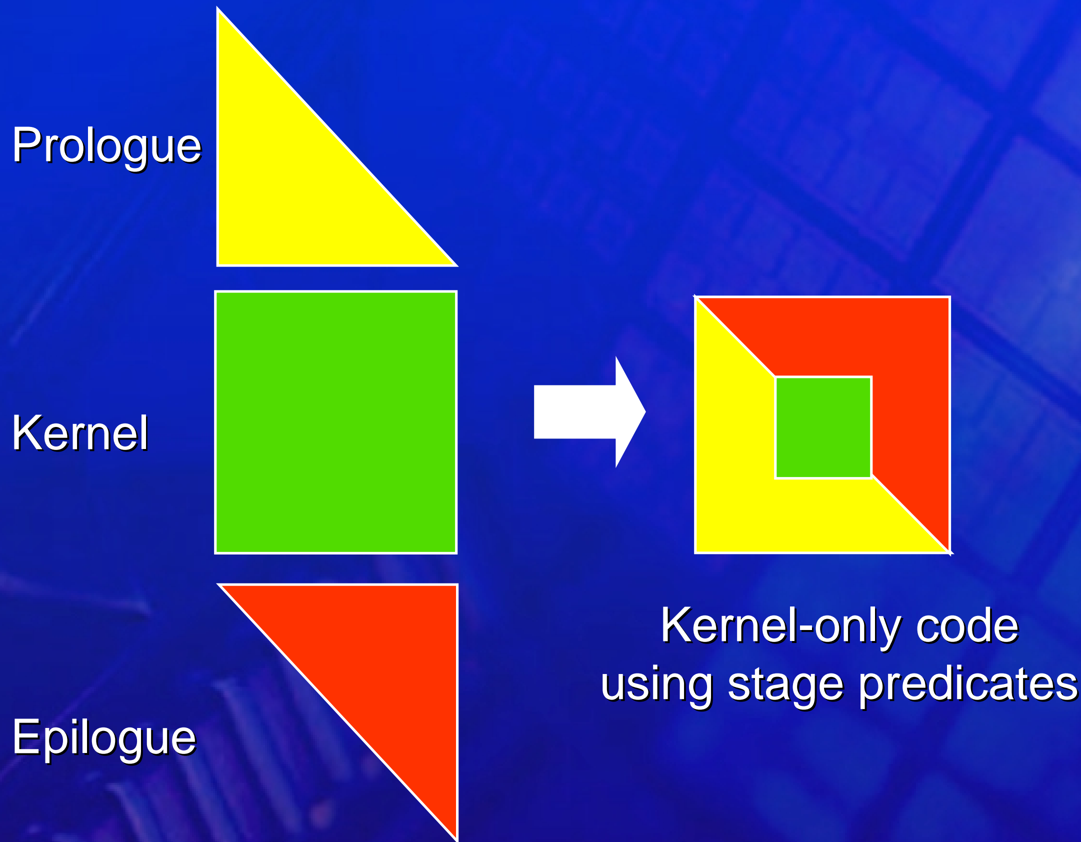
Kernel

Epilogue



Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Software Pipelining



IA-64 Features

- **Rotating Registers**
- **Loop branches**
- **Full predication**
- **Rotating Predicates**

IA-64 ***SWP Applicable to many Loops in IA-64***

intel

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Register Rotation

- GR32-127 and FR32-127 can rotate (specified range)
- Separate rotating register base for each set (GR, FR)
- Loop branches decrement all register rotating bases (RRB)
- Instructions contain a "virtual" register number
  - physical register # = RRB + virtual register #

Predicate register range also rotates.

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Control & Data Speculation

Control Speculation
moves loads above branches / calls

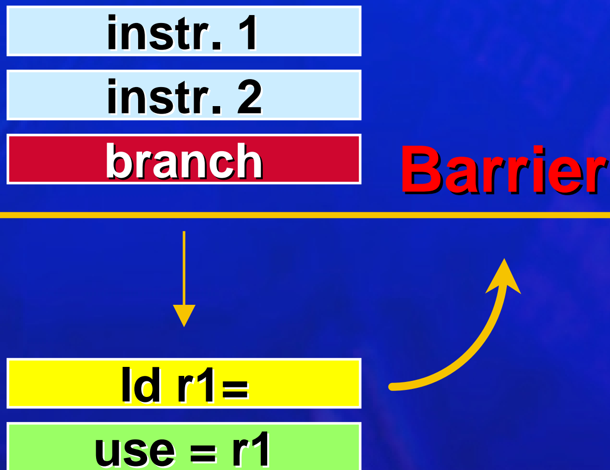
Data Speculation
moves loads above possibly conflicting stores

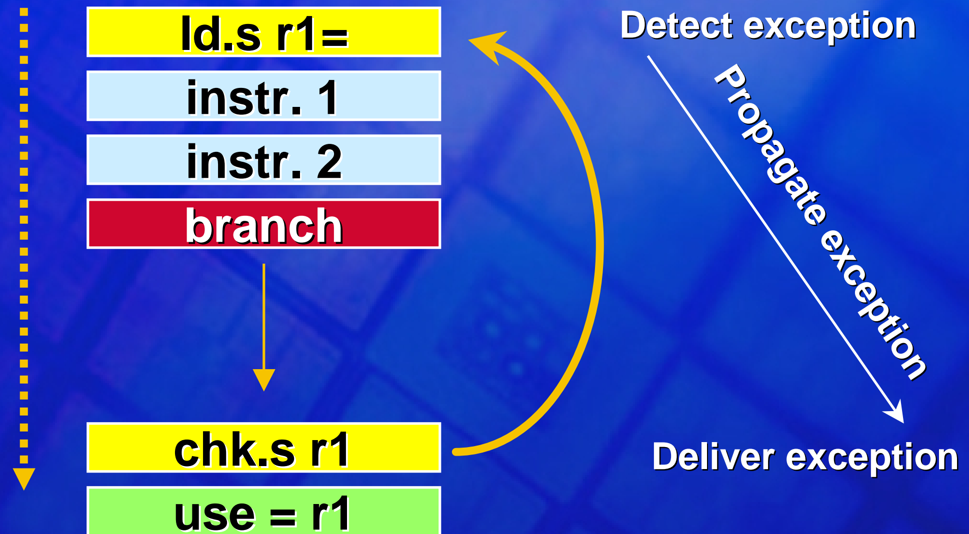
IA-64 *Speculation reduces the impact of memory latency*

intel

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Control Speculation



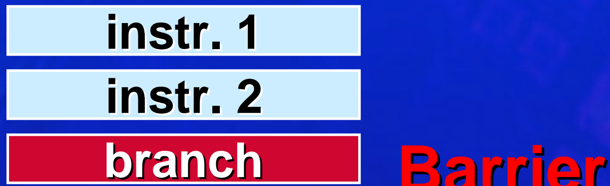


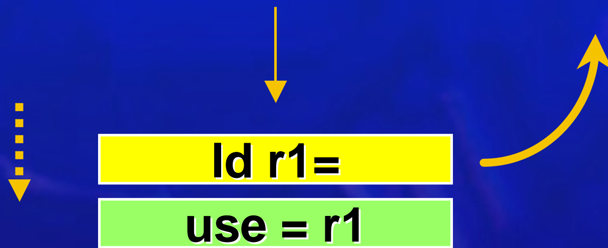
- Control Speculation moves loads above branches
  - Detected exception indicated using NaT bit / NaTVal
- Check raises detected exceptions
- Branch barrier broken to minimize memory latency

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

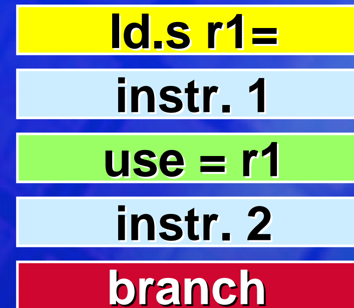
Hoisting Uses

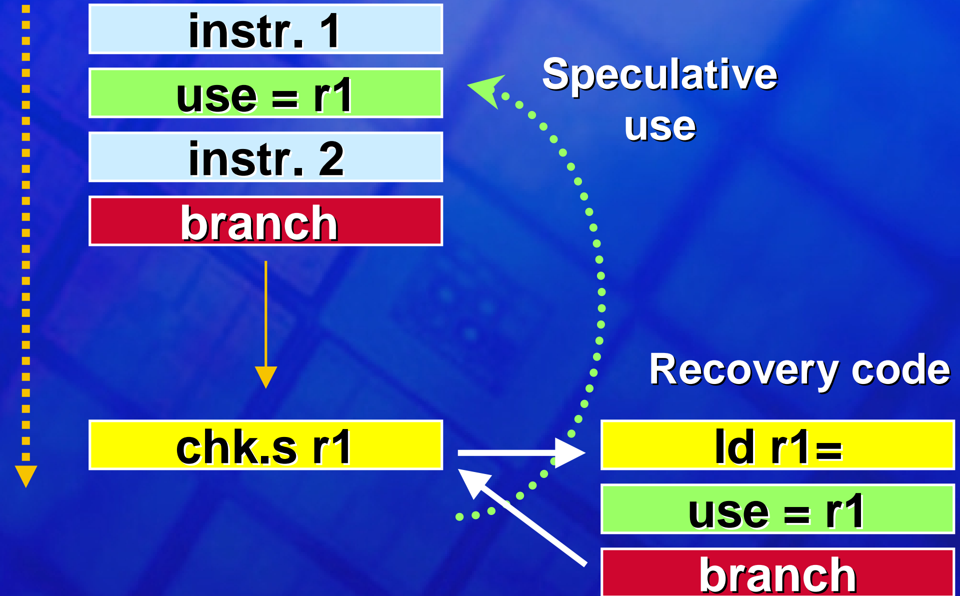
Traditional Arch.





IA-64





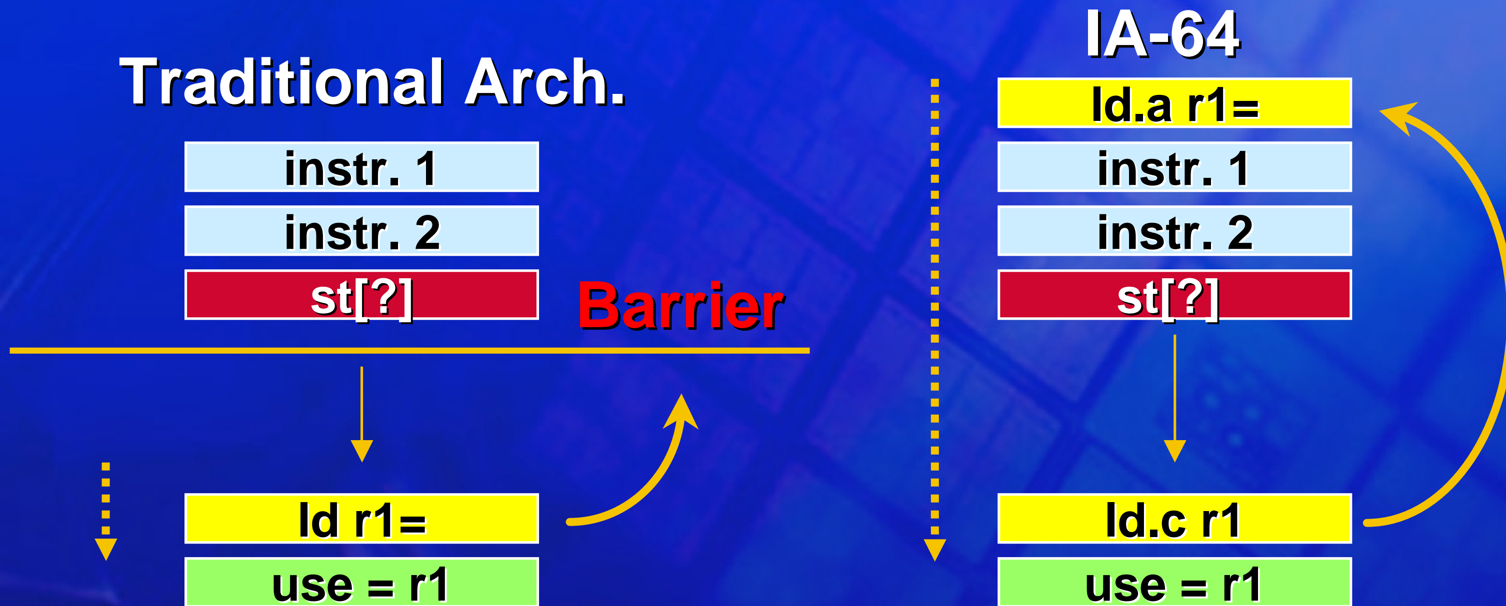
- All computation instructions propagate NaTs to reduce number of checks to allow single check on results
- Compares also propagates when writing predicates

intel

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Data Speculation



- Data Speculation moves loads above possibly conflicting stores
  - Keeps track of load addresses used in advance (ALAT)
- Advanced-loaded data can be used speculatively

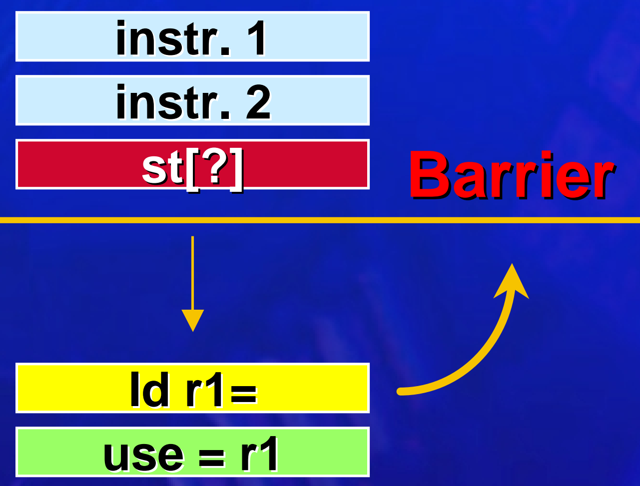
intel

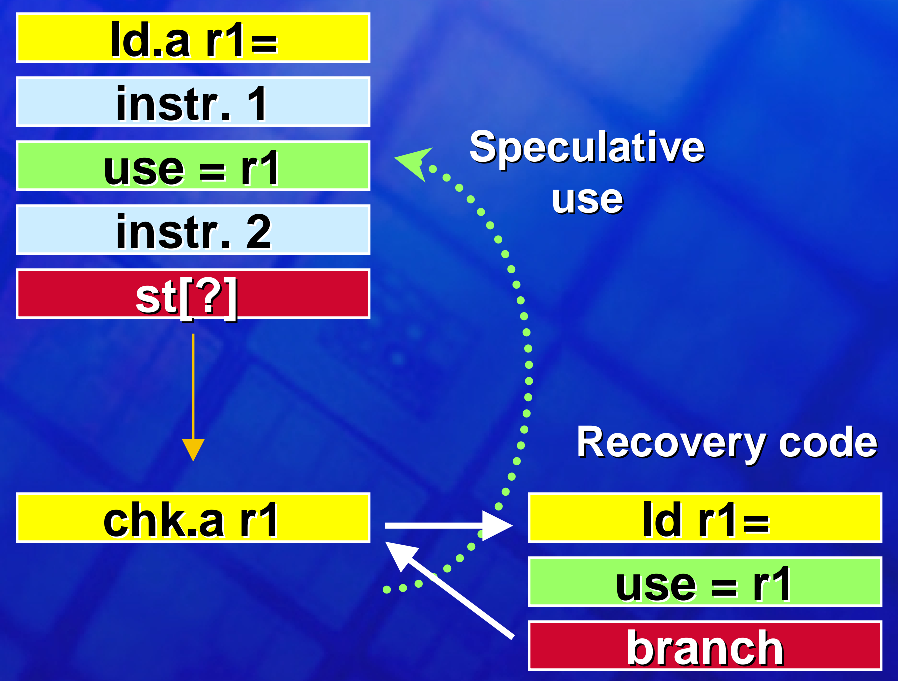
ITANIUM

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Hoisting Uses





IA-64 *Data and Control Speculation Can be combined*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Advanced Load Address Table - ALAT

- ld.a inserts entries
- Conflicting stores remove entries
  - also ld.c.clr, chk.a.clr
- Presence of entry indicates success
  - chk.a branches when no entry is found

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Branch Architecture

- **Branch types**
  - IP-offset branches (21-bit disp.)
  - Indirect branches via 8 branch registers
  - HW-supported counted loop control instr.
- **Branch Predict hints**
  - Advance information on downstream branches and branch conditions
  - Branch hints can be static or dynamic
- **Multi-way branches**
  - Bundle 1-3 branches in a bundle
  - Allow multiple bundles to participate

Traditional Architecture:

```
compute0
compute1
compute2
compare (a==b)
B: branch_if_eq → Target
...
Target:
```

IA-64 Architecture:

```
hint B,Target   (early hint)
compute0
compute1
compute2
compare(a==b)
B: branch_if_eq → Target
...
Target:
```

IA-64

Aggressive branch prediction
Decoupled front end with code prefetch,
Branch hints reduce misprediction and overhead

intel

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

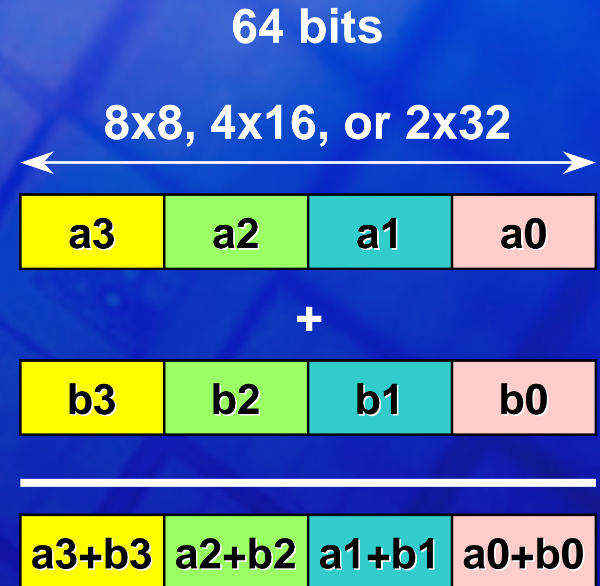
Integer Architecture

- 128 general registers (64 bit; 1s+63i)
- Full 64-bit support (as well as 8-16-32-bit)
- XMA: Integer Multiply-Add instruction (l = i * j + k)
- Integer multiply is executed in the floating-point unit
- Data transfer
  - load, store, GR ←→ FR conversion
- SIMD Integer operations
- Divide / remainder deferred to software
  - Based on floating-point operations
  - High throughput achieved via pipelining

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

IA-64 SIMD - Integer

- Exploits data parallelism with SIMD (*Single Instruction Multiple Data*)
- Performance boost for audio, video, imaging, streaming etc. functions
- GRs treated as 8x8, 4x16, or 2x32 bit elements
- Several instruction types
  - Addition and subtraction, multiply
  - Pack/Unpack
  - Left shift, signed/unsigned right shift
- Compatible with Intel® MMX™ Technology



IA-64 *Performance Boost for all Data Parallel Apps*

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Floating-Point Architecture

- **Fused Multiply Add Operation**
  - An efficient core computation unit
  - Greater precision, faster than independent multiply and add
- **Abundant Register resources**
  - 128 registers (32 static, 96 rotating)
- **High Precision Data computations**
  - 82-bit unified internal format for all data types
  - Full IEEE.754 support
- **Software divide/square-root**
  - High throughput achieved via pipelining
- **Wide (Speculative) Memory Access**
  - Dual Load-Pair support
  - Address memory latency
  - Pre-fetch support

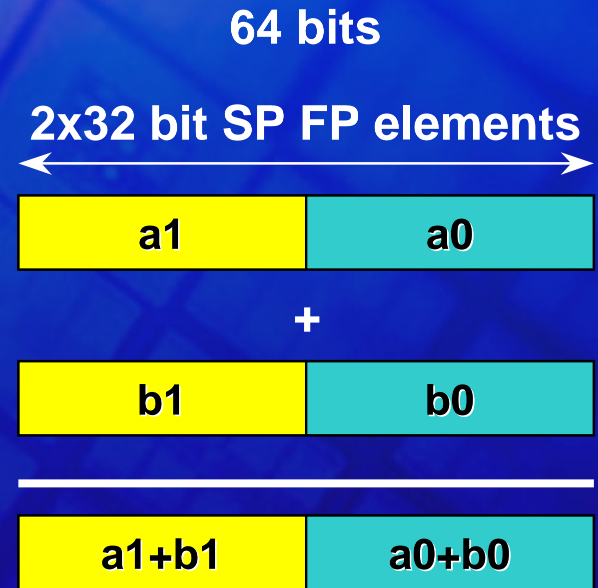
IA-64 *Excellent Workstation & HPC Application Performance*

2 independent FP Units
Up to 4 DP FP operations per clock
Up to 4 DP FP operands loaded per clock (from L2 cache)

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

IA-64 SIMD – F.P.

- Exploits data parallelism with SIMD (Single Instruction Multiple Data)
- Up to 2x performance boost
- F.P. Registers treated as two 32 bit single precision elements
  - Full IEEE.752 compliance
  - Availability of fast divide (non IEEE)
- Compatible with Intel® Streaming SIMD Extensions (SSE)



IA-64 *Enables World Class 3D Graphics Performance*

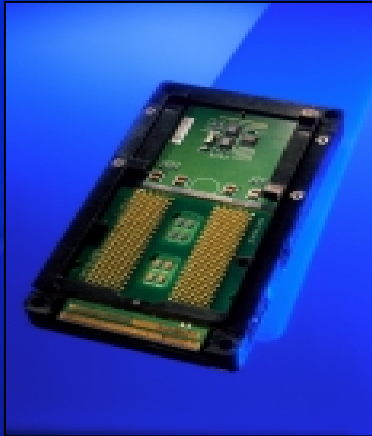
Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

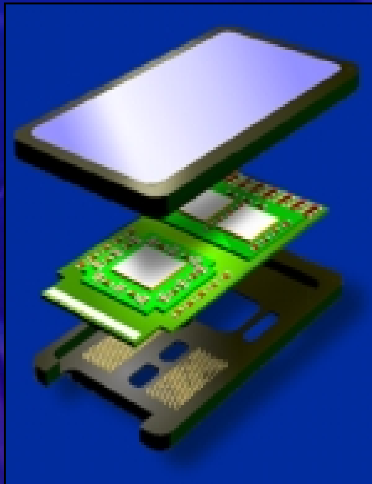
Floating-Point Status Register

- **Contains dynamic control/status for FP operations**
- **Trap/Fault disable bits**
  - trap disables for IEEE exception events
  - trap disable “D” for denormal operand exception
- **4 separate status fields → 4 computational env.**
  - Each field specifies precision/rounding mode, Trap disables, flush to zero, widest range exponent
  - Each field reports sticky exception flags

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Intel® Itanium™ Processor





- IA-64 starts with Itanium processor
- Platform with Intel® 460GX chipset
- Solid progress following first silicon
  - More than 4 OS running today
  - Demonstrated real IA-64 Windows 2000 and Linux applications on real hardware
  - Engineering samples shipping to OEMs, IHVs and ISVs
- Comprehensive validation underway

320M transistors: 25M in CPU, 295M in L3 cache

More and better Capacity & Capability

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Extending Intel® Architecture

System Performance

Extends IA Headroom, Scalability and Availability for the Most Demanding Environments

Madison* IA-64 perf

Deerfield* IA-64 price/perf

McKinley*

IA-64

IA-32

* Intel code names

Future IA-32

Foster*

Outstanding Performance for 32 Bit Volume Apps

'99 '00 '01 '02

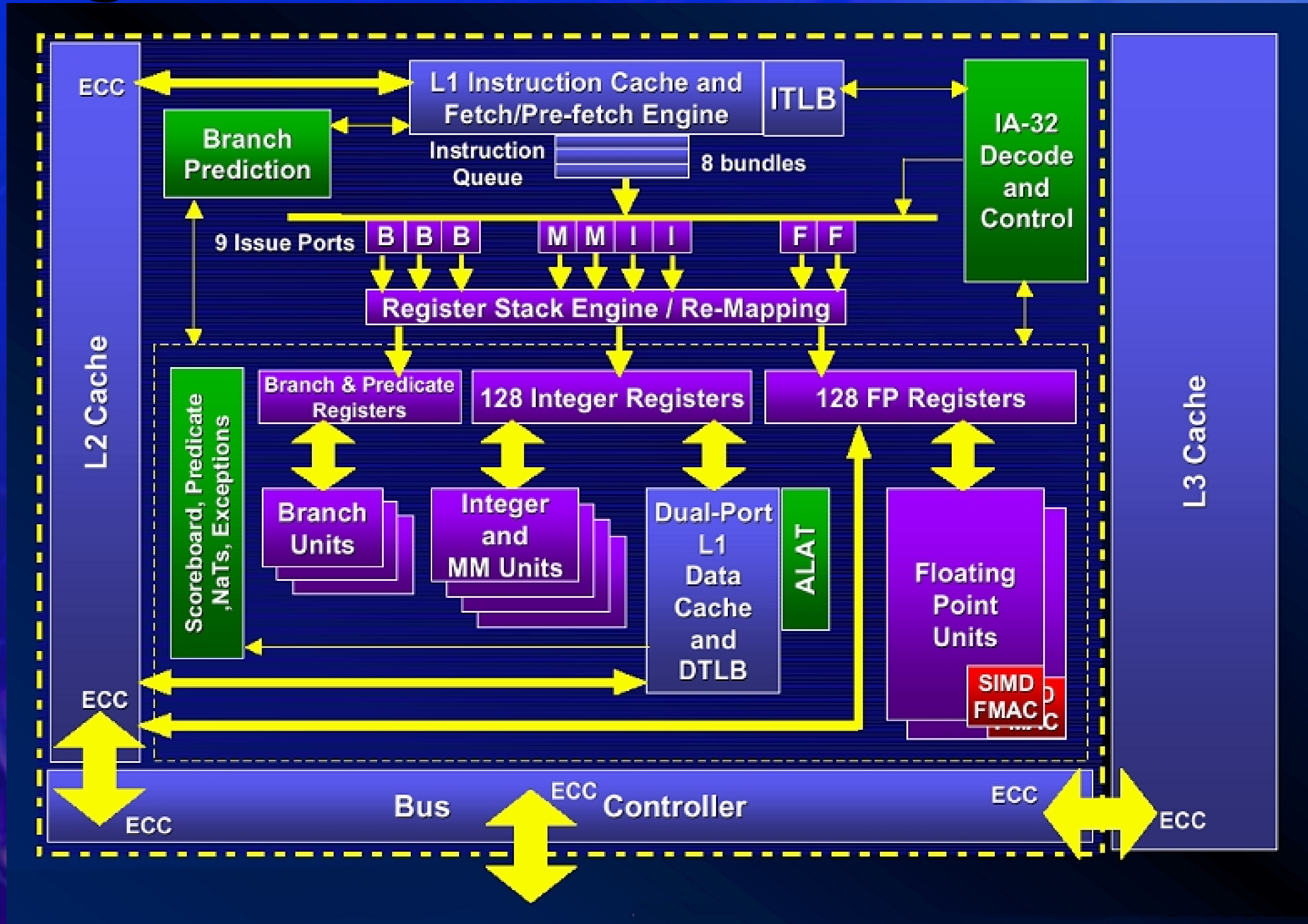
.25µ .18µ .13µ

All dates specified are target dates provided for planning purposes only and are subject to change.

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Itanium™ Processor Block Diagram



Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

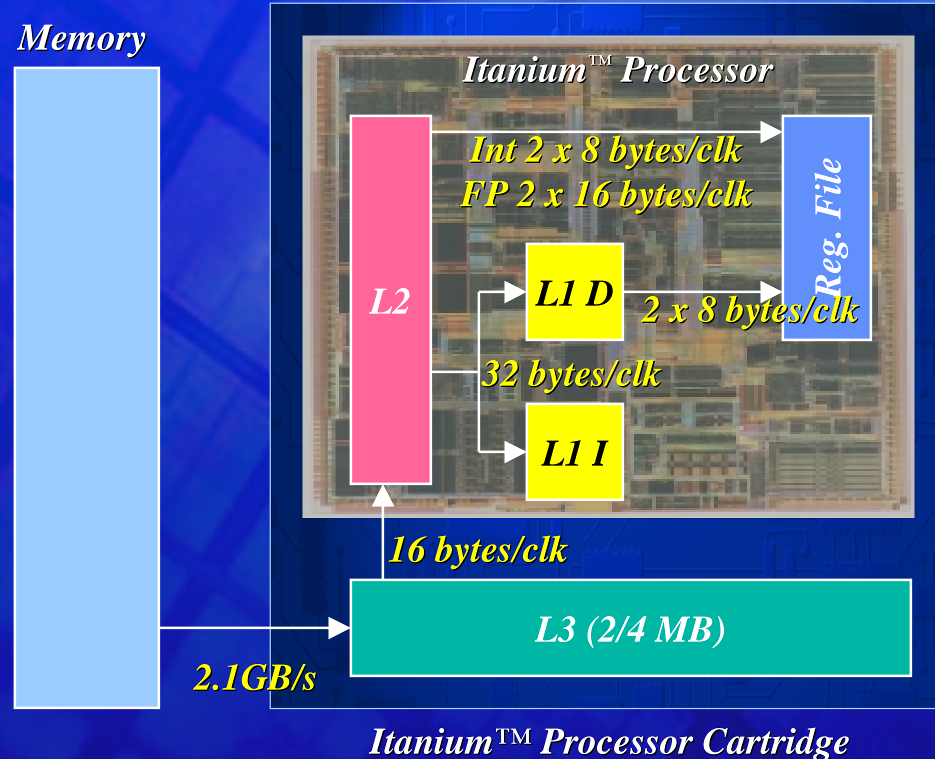
Itanium™ Processor Features

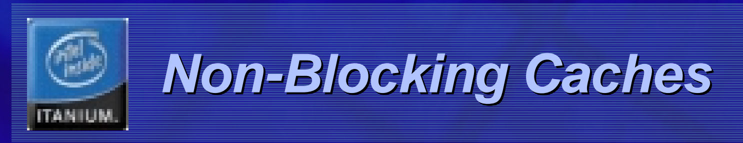
- Up to 6 instructions issued per clock
- 9 instruction issue ports
- 2 floating point units
- 4 integer units
- 3 branch units
- 3 levels of cache at full speed
- L1 and L2 on-chip, L3 (2/4 MB) on cartridge
- 10-stage in-order pipeline

intel
Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Itanium™ Processor Memory Hierarchy

- **L1 Caches (on-chip)**
  - Data Cache
    - 4-way, 32 byte cache lines
    - FP loads bypassed to L2
  - Instruction Cache
    - 4-way, 32 byte cache lines
- **L2 Cache (on-chip)**
  - Unified instr. & data cache
    - 6-way, 64 byte cache lines
- **L3 Cache (on cartridge)**
  - Full speed unified
    - 2/4 MBytes
    - 4-way, 64 byte cache lines
- **Memory**
  - Frontside Bus
    - 2.1 GBytes/sec





intel
Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Itanium™ Processor Pipeline

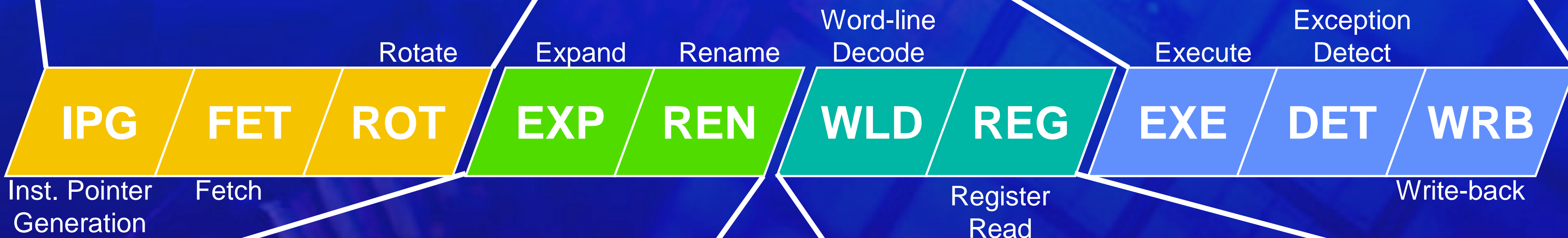
- 6-wide EPIC hardware under compiler control
  - Parallel hardware and control for predication & speculation
  - Efficient mechanism for enabling register stacking & rotation
  - Software-enhanced branch prediction
- 10-stage in-order pipeline designed for:
  - Single cycle ALU (4 ALUs globally bypassed)
  - Low latency from data cache
- Dynamic support for run-time optimization
  - Decoupled front end with prefetch to hide fetch latency
  - Non-blocking caches, register scoreboard to hide load latency
  - Aggressive branch prediction to reduce branch penalty

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

10 Stage In-Order Core Pipeline

Front-End
- Pre-fetch/fetch up to 6 instructions/cycle
- Hierarchy of branch prediction
- Decoupling buffer

Execution
- 4 single cycle ALUs, 2 ld/str
- Predicate delivery and branch
- NAT / Exception / Retirement



Instruction Delivery
- Dispersal of up to 6 instructions on 9 ports
- Register remapping
- Register stack engine

Operand Delivery
- Register read + bypasses
- Register scoreboard
- Predicated dependencies

intel

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Selected Instruction Latencies

Instruction Class	Description	Latency (Cycles)
FMAC	Floating arithmetic	5
FMISC	Floating min, max, …	5
IALU	Integer ALU	1
FLD,FLDP	FP load (L2 hit)	9
	FP load (L3 hit)	24
LD	Integer load except ld.c (L1 hit)	2
	Integer load except ld.c (L2 hit)	6
	Integer load except ld.c (L3 hit)	21

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Itanium™ Processor Based Platform Features

- 1-4 Itanium processor SMP system
- High clock speed target 800 MHz
- 6-way instruction issue and execution
- Up to 64 GB SDRAM memory (460GX)
- 4.2 GB/s memory bandwidth (peak)
- 2.1 GB/s system bus
- 2.1 GB/s I/O bandwidth (peak)
- 1.0 GB/s AGP Pro graphics bus
- 3.2 GFLOPS DP-F.P. peak perf. (6.4 in SP)

SHV Workstation platform: 2-way
SHV Server platform: 4-way

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Server Platform

528MB/s

Address Bus

4x PCI 64/66

WXB

WXB

WXB

SAC

SDC

Data Bus

528MB/s

PCI 64/66

PXB

Addr/Ctrl

Data

FWH

64 SDRAM DIMMs (64GB max.)

AUDIO

528MB/s

LAN

PCI 64/66

DMI

264MB/s

PCI 64/33

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Workstation Platform

Address Bus

Data Bus

AGP-4x

1GB/s

GXB

WXB

SAC

SDC

PXB

PCI 64/66

528MB/s

PCI 64/66

264MB/s

PCI 64/33

FWH

AUDIO

LAN

DMI

Addr/Ctrl

Data

16 SDRAM DIMMs
(16GB max.)

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

HPC with Intel® Architecture:
From Top to Bottom

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Itanium™ Processor Based System Designs

1-8 way SMP

32 way SMP

256-way cc:NUMA

16-way cc:NUMA

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

HPC Market Segment is Changing

Open Industry Standards using Building Blocks

- WindowsNT*, Linux*
- OpenMP*
- MPI*, PVM*

Proprietary Solutions

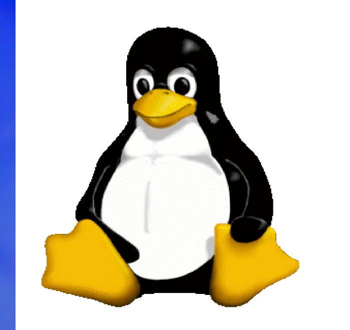
Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

IA-64 Operating Systems

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

IA-64 Linux* (Trillian* Project)



- Team includes VA Linux, IBM*, Intel, HP*, SGI*, Cygnus*, CERN*, Red Hat*, SuSE*, TurboLinux*, and Caldera*
- Running applications
  - Demonstrated on Itanium™ processor system at IDF (8/99)
  - Major applications ported to date include Apache* and Sendmail
  - Development version release available
  - Full development OS releases from distributors available
- Open source OS and compilers available
- http:/www.linuxia64.org

intel

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

C/C++Data Models

OS Implements the Data Models

ILP32

- int, long and ptr are 32 bits
- Used by 32-bit OSs

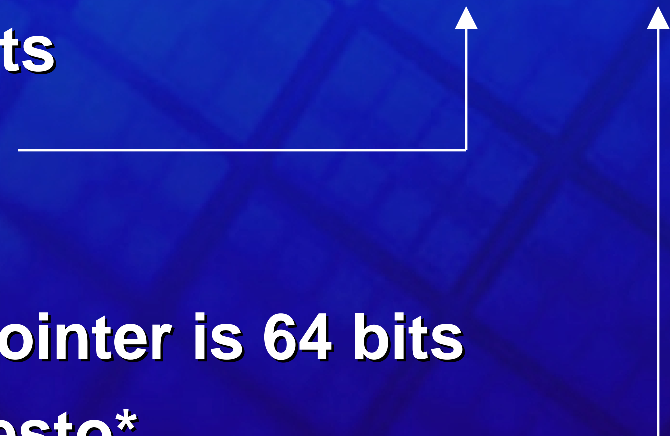
LP64

- int is 32 bits
- long and pointer are 64 bits
- Used by 64-bit UNIX OSs

P64 (or LLP64)

- int and long are 32 bits; pointer is 64 bits
- Used by Win64* and Modesto*

	ILP32 size (bits)	LP64 size (bits)	P64 size (bits)
int	32	32	32
long	32	64	32
pointer	32	64	64

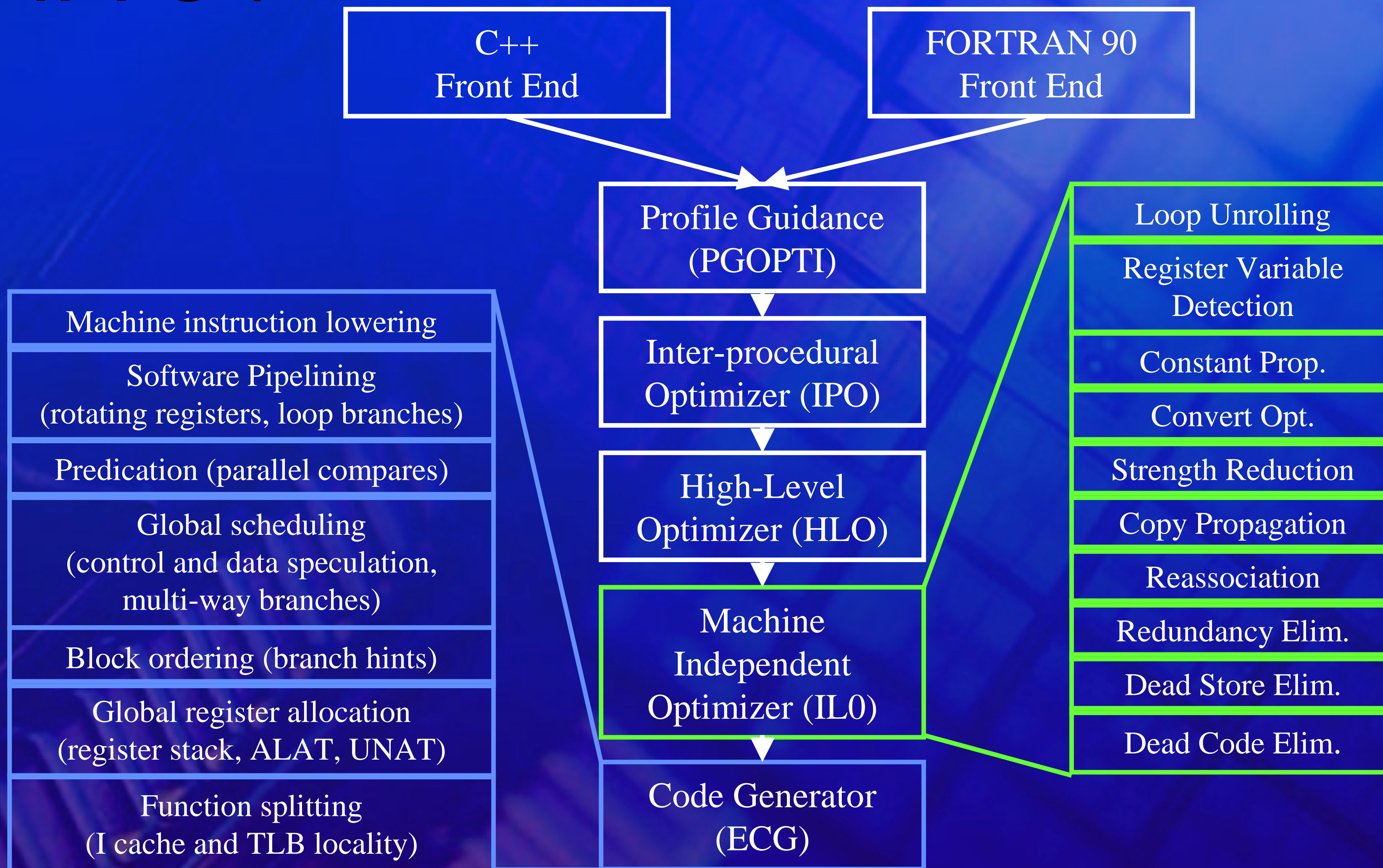


* Third party names and brands are the property of their respective owners

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Intel® Compiler for IA-64



Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

IA-64 HPC Compilers & Tools

- C/C++, FXX, Java, …
- OpenMP
- MPI, PVM
- Performance Libraries
- Vtune, …

intel

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

IA-64 Application Benefits

Outstanding Performance

- Removes performance bottlenecks
  - Large register files
  - High Parallelism
  - Predication
  - SW pipelining support
  - Memory latency hiding
- 64-bits allows bigger address space
- IEEE-accurate floating point
- Ability to run IA-32 applications

intel

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

IA-64 User Benefits

- Big in-memory data structures and DB
- Large file system and data files
- Efficient large integer calculations
- Fast 64-bit F.P. calculations
- Fast Security processing
- More and faster transactions
- More services
- Higher throughput
- Improved availability and manageability

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Intel: More Than Just Microprocessors

Intel WebOutfitter Service

Intel® Online Services

intel online services

Intel® e-Business Center

The Platform for e-Business

Intel® ISP Program

Web Hosting Services

technology

Intel® Application Solution Centers

The Best Place for Optimizing Software Perfo... Worl...

IA-32
Performance Tuning
Optimizing
Collaboration

Intel® Teamstation™ System

Intel® ProShare® Video Syst...

Intel® Play™ QX3™ Computer Microscope

Intel Architecture Labs

Intel® Intercast® Viewer

TV and the Web combining forces

Intel® Create & Share™ Camera Pack

VTune Performance Enhancement Environment

C/C++ Compiler

Fortran Compiler

start up
skill up
team up

Intel® e-Business Network

Intel® Internet Media Services

...rved.
...and names are the property of their respective owners

IA-64 : The Future for HPC

http://developer.intel.com/

http://developer.intel.com/design/ia64/

http://developer.intel.com/technology/itj/q41999.htm

intel

Glossary

intel®

Glossary

- ALAT (Advanced Load Address Table) **- cache used for data speculation which stores the most recent advanced load addresses**
- ALoad/Acheck **- advanced load/check (Data Speculation)**
- Basic Block **- code which is between two branches; if one instruction in the block of code executes, then all instructions in that block will also execute**
- Control Speculation **- the execution of an operation before the branch which guards it; used to hide memory latency**
- Data Speculation **- the execution of a memory load prior to a store that precedes it, and that may potentially alias it; used to hide memory latency**

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Glossary

- IA-32 **- the name for Intel's current ISA (32-bit and 16-bit)**
- IA-32 System Environment **- the system environment of an IA-64 processor as defined by the Pentium® processor and Pentium® Pro processor**
- IA-64 **– Intel® 64-bit Architecture is composed of the 64-bit ISA and IA-32; IA-64 integrates the two into a single architectural definition**
- IA-64 Firmware **- the Processor Abstraction Layer and the System Abstraction Layer**
- IA-64 System Environment **- IA-64 operating system with privileged resources along with capability to support the execution of existing IA-32 applications**
- Instruction Set Architecture (ISA) **- defines application level resources which include: user-level instructions, addressing modes, segmentation, and user visible register files**

intel®

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Glossary

- NaT bit/NaT Value (Not a Thing) **- used with control speculation to indicate that a number stored in a general or floating-point register is not valid**
- Predication **- the conditional execution of an instruction; used to remove branches from code**
- Processor Abstraction Layer (PAL) **- the IA-64 firmware layer which abstracts IA-64 processor features that are implementation dependent**
- Sload/SCheck **- speculative load/check (control speculation)**
- System Abstraction Layer (SAL) **- the IA-64 firmware layer which abstracts IA-64 system features that are implementation dependent**
- System Environment **- defines processor specific operating system resources which include: exception and interruption handling, virtual and physical memory management, system register state, and privileged instructions**

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Backup

intel

SW Pipelined Loop Example

- **DAXPY inner loop : dy[i] = dy[i] + (da * dx[i])**
  - 2 loads, 1 fma, 1 store / iteration
- **Machine assumptions**
  - can do 2 loads, 1 store, 1 fma, 1 br / cycle
  - load latency of 2 clocks
  - fma latency of 1 clock (not realistic, but good for example)
- **Special Registers**
  - LC: Loop Counter

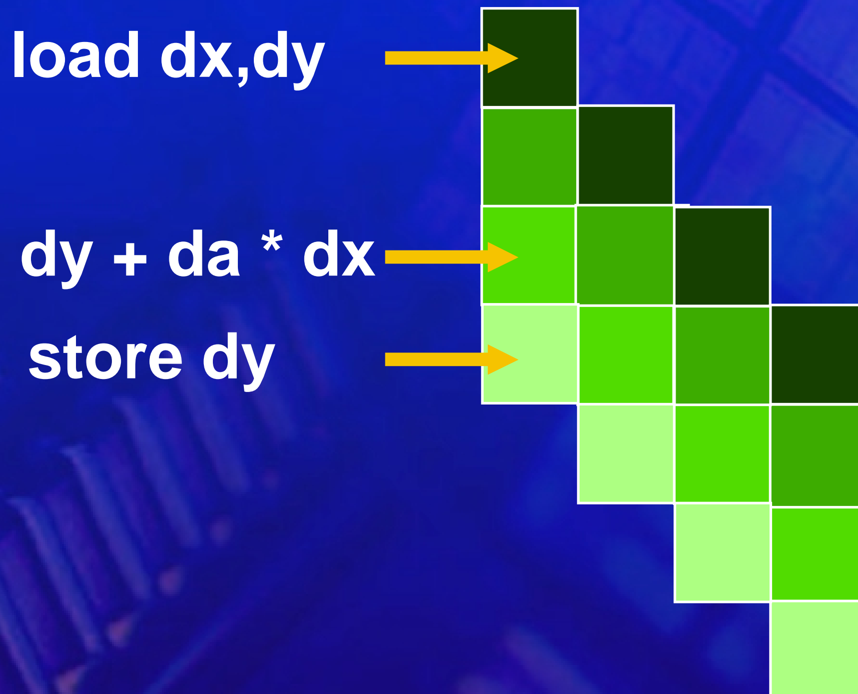
intel®

Copyright © 2000, Intel Corporation. All rights reserved.

*Other brands and names are the property of their respective owners

Example: Pipeline

- **Each column represents 1 source iteration**



Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Example Code

```
.rotf dx[3], dy[3], tmp[2]

        mov   ar.lc = 3          // #iterations-1
        mov   ar.ec = 4          // #stages
        mov   pr.rot = 0x10000
        ;;
looptop:
  (p16)       ldfd  dx[0] = [dxsp],8
  (p16)       ldfd  dy[0] = [dysp],8
  (p18)       fma.d tmp[0] = da, dx[2], dy[2]
  (p19)       stfd  [dydp] = tmp[1],8
              br.ctop looptop
        ;;
```

intel ITANIUM

Copyright © 2000, Intel Corporation. All rights reserved.
*Other brands and names are the property of their respective owners

Loop Execution

Execution Sequence

...		
19:	?	(p19)
18:	?	(p18)
17:	?	
16:	?	(p16)
63:	?	(p63)
.		

RRB=0

LC=? EC=?

Before Initialization

*Other brands and names are the property of their respective owners

Loop Execution

Execution Sequence

(p16) ld_x (p16) ld_y (p18) fma (p19) st

...	
19: 0	(p19)
18: 0	(p18)
17: 0	
16: 1	(p16)
63: 0	(p63)
.	

RRB=0

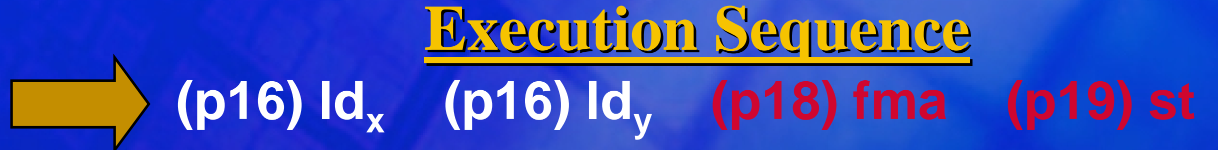
LC=3 EC=4

Initialization

Copyrig

*Other brands and names are the property of their respective owners

Loop Execution



Register	Value	Predicate
...		
19:	0	(p19)
18:	0	(p18)
17:	0	
16:	1	(p16)
63:	0	(p63)
.		

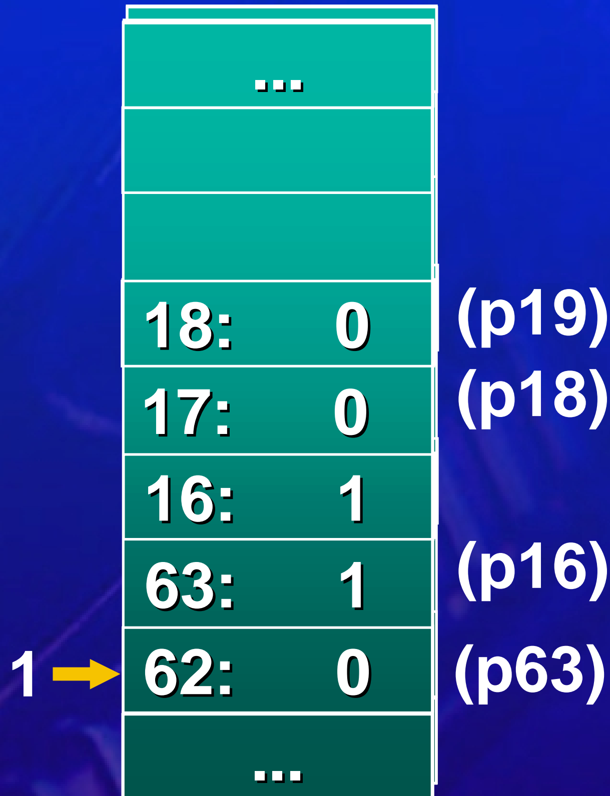
RRB=0

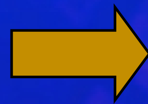
LC=3 EC=4

Prologue

Copyrig...
*Other brands and names are the property of their respective owners

Loop Execution





Execution Sequence

(p16) ld_x (p16) ld_y (p18) fma (p19) st

(p16) ld_x (p16) ld_y (p18) fma (p19) st

RRB=-1

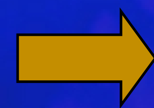
Branch 1

LC=2 EC=4

Copyrig…

*Other brands and names are the property of their respective owners

Loop Execution



Execution Sequence

(p16) ld_x (p16) ld_y (p18) fma (p19) st

(p16) ld_x (p16) ld_y (p18) fma (p19) st

(p16) ld_x (p16) ld_y (p18) fma (p19) st

RRB=-2

Branch 2

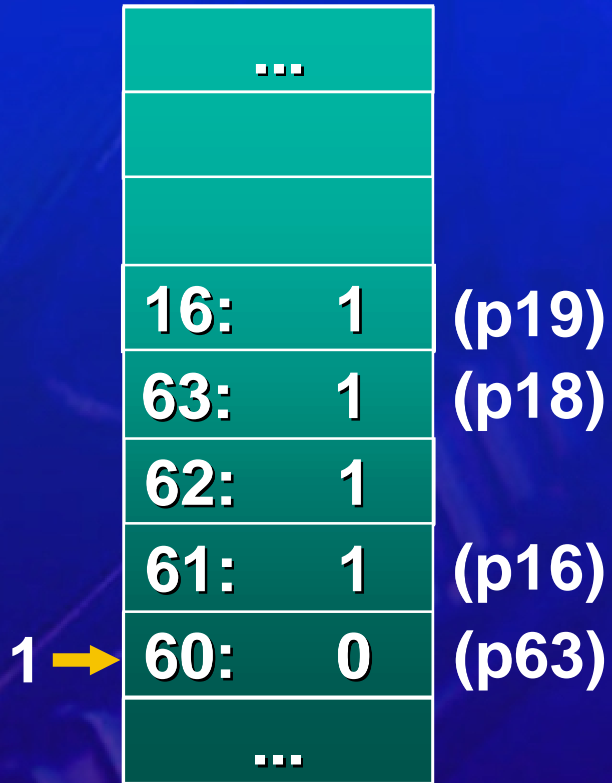
LC=1 EC=4

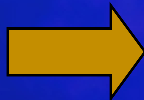
intel®

Copyrig…

*Other brands and names are the property of their respective owners

Loop Execution





Execution Sequence

(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st

RRB=-3

Branch 3

LC=0 EC=4

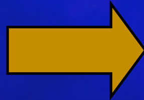
Copyrig...
*Other brands and names are the property of their respective owners

Loop Execution

...	
63: 1	(p19)
62: 1	(p18)
61: 1	
60: 0	(p16)
59: 0	(p63)
...	

0 →

RRB=-4



Execution Sequence

(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st

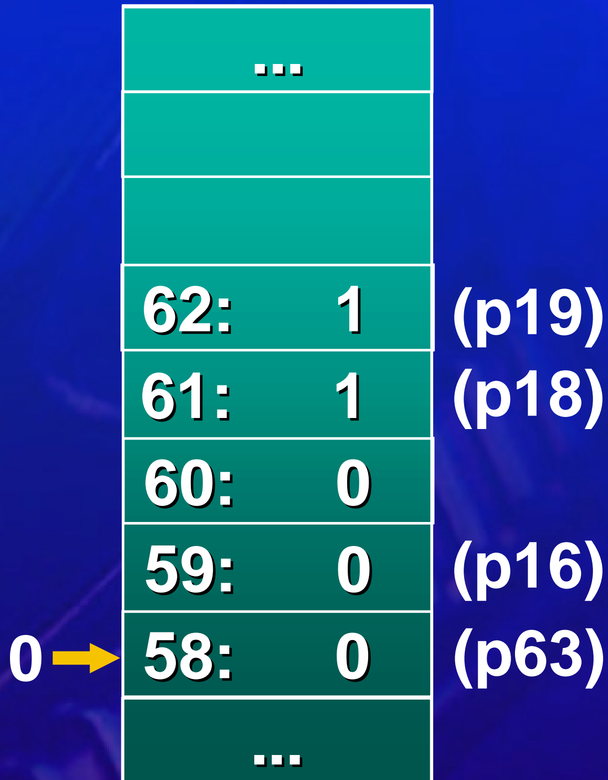
LC=0 EC=3

Branch 4

Copyrigh

*Other brands and names are the property of their respective owners

Loop Execution



(p19)
(p18)
(p16)
(p63)

Execution Sequence

(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st

LC=0 EC=2

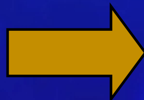
Branch 5

Copyrigh
*Other brands and names are the property of their respective owners

Loop Execution

...	
61: 1	(p19)
60: 0	(p18)
59: 0	
58: 0	(p16)
0 → 57: 0	(p63)
...	

RRB=-6



Execution Sequence

(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st

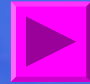
LC=0 EC=1

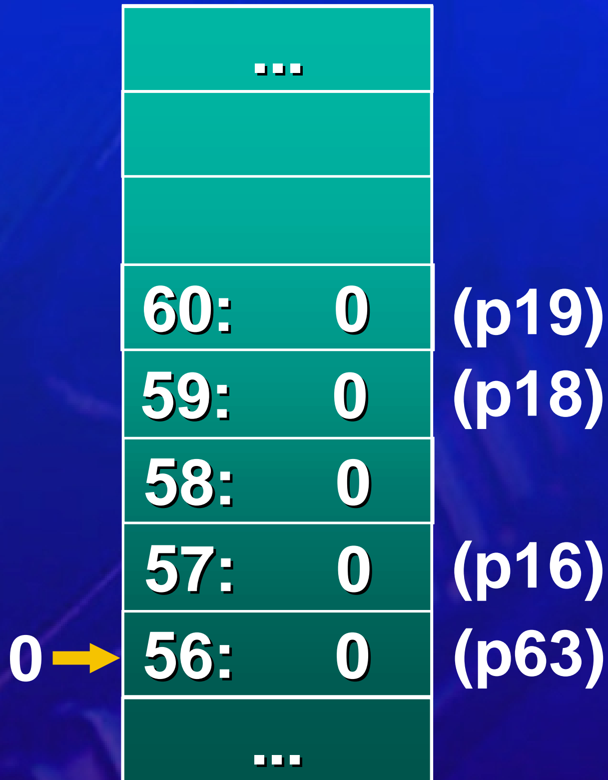
Branch 6

Copyrigh

*Other brands and names are the property of their respective owners

Loop Execution





Execution Sequence

(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st
(p16) ld_x	(p16) ld_y	(p18) fma	(p19) st

fall through

RRB=-7

Branch 7

LC=0 EC=0

Copyrigh

*Other brands and names are the property of their respective owners