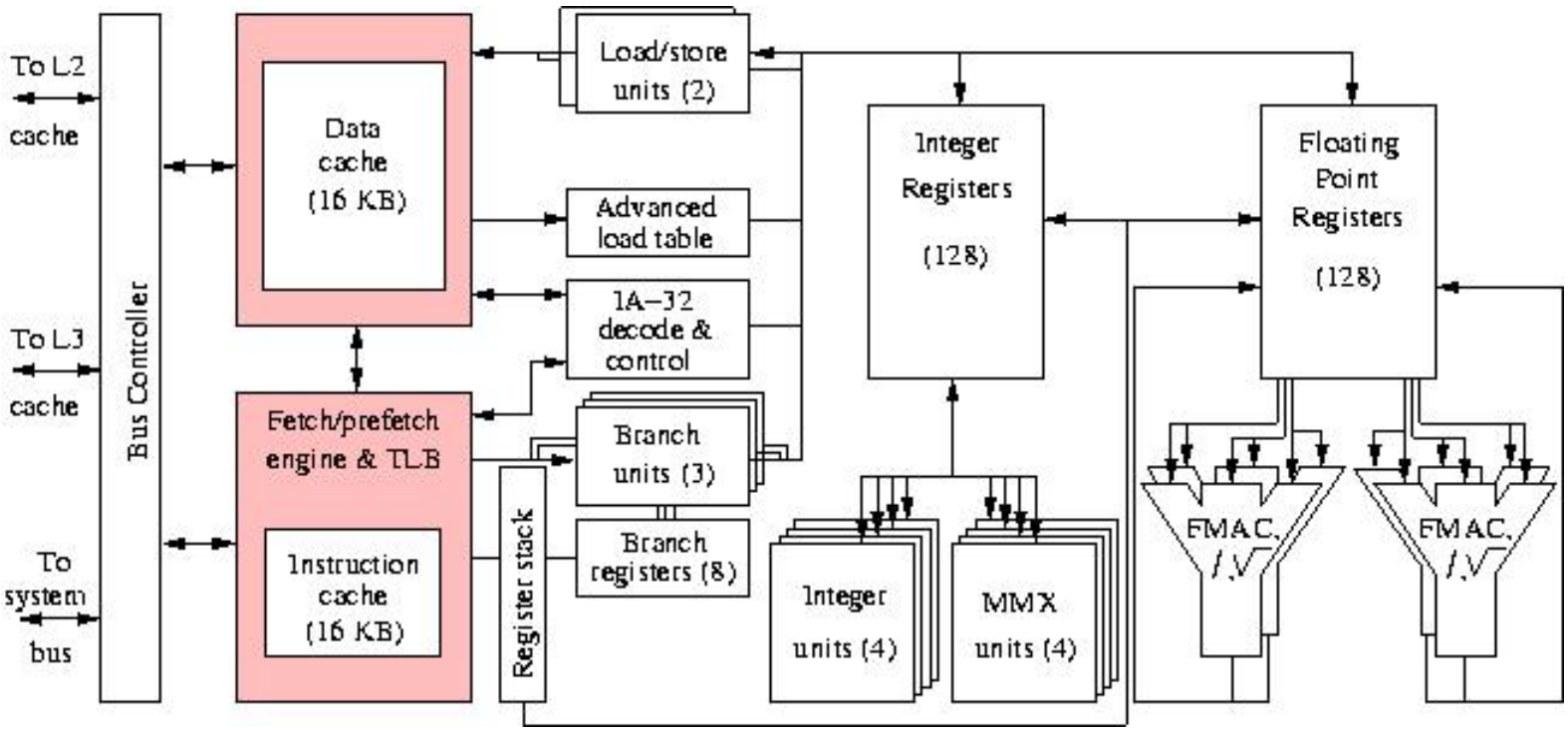
EPIC: IA64 - Intel Itanium

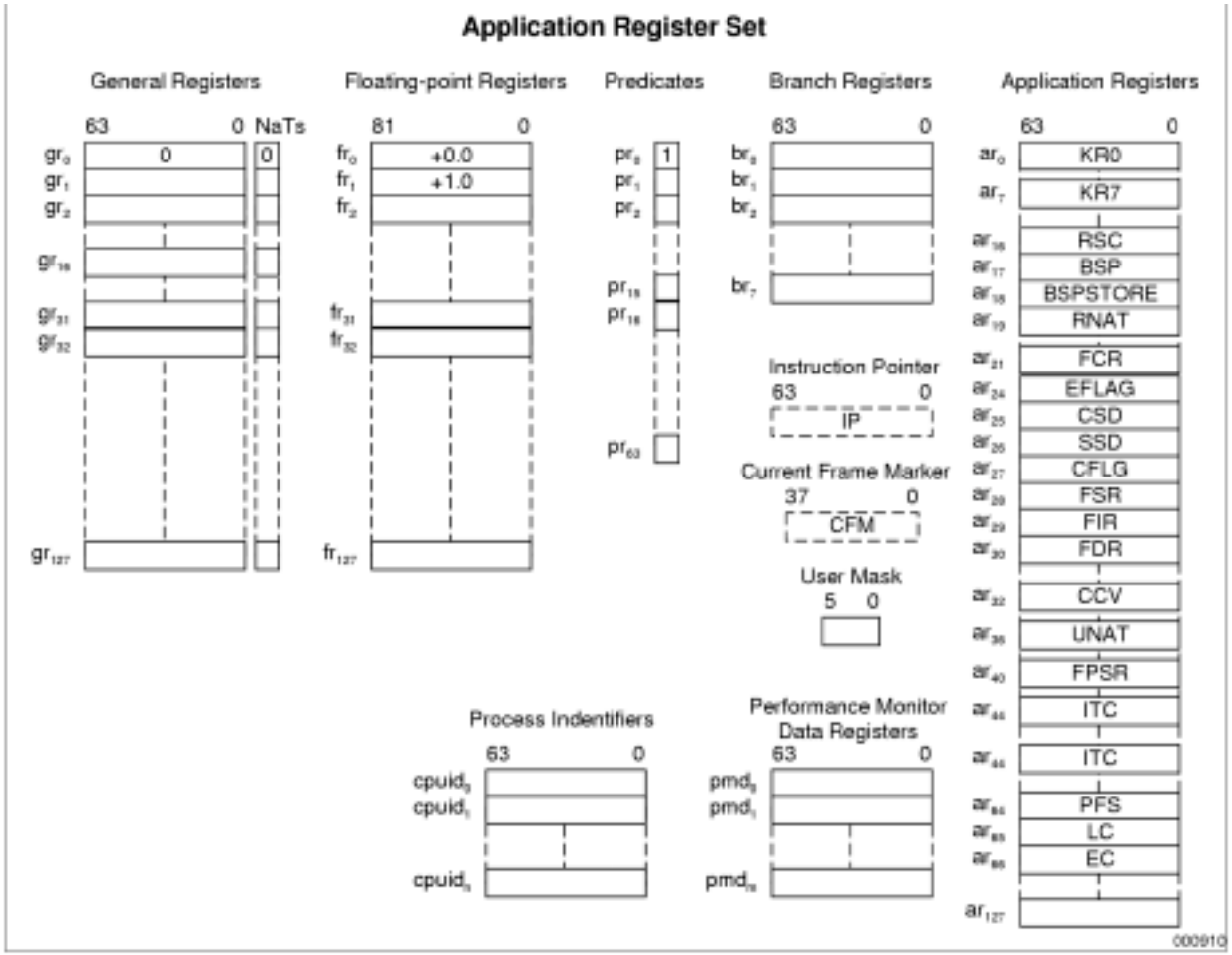


Block diagram of Intel Intanium.

IA-64 Architecture

- Support for
  - branch prediction (minimize the cost of branches),
  - predicated execution (remove branches),
  - control speculation and data speculation (reduce memory latency effects)
  - cache hints (compiler prediction of spatial or temporal locality of the memory area being accessed; can be used for placement of cache lines in the cache hierarchy by the processor)
- Register sets:
  - 128 64-bit general purpose registers: GR0-GR127
  - 128 80-bit floating-point registers: FR0-FR127
  - 64 1-bit predicate registers: PR0-PR63
  - 8 64-bit branch registers: BR0-BR7
  - features: register stack and rotating registers

IA-64 Architecture



IA-64 Architecture

- Each IA-64 instruction is categorized into 6 types and may be executed on one or more execution unit types.
- 4 functional unit categories:
  - I unit (integer)
  - F unit (floating-point)
  - M unit (memory)
  - B unit (branch)
- 6 microoperation categories:

– Integer ALU (A-type)	executed on M- or I units
– Non-ALU Integer (I-type)	executed on I units
– Memory (M-type)	executed on M units
– Floating-point (F-type)	executed on F units
– Branch (B-type)	executed on B units
– Extended (L/X-type)	executed on I- or B units

IA-64 Architecture

- The IA-64 defines 128-bit long instruction words (called bundles) consisting of three 41-bit microoperations and a 5-bit template field. Multiple bundles can be issued per clock cycle (number is defined by implementation).

- Template field:
  - helps decode and route instructions
  - indicates the location of stops that mark the end of groups of microoperations that can execute in parallel

- Dispersal: process of sending instructions to functional units.

IA-64 Architecture

- There are 12 basic bundle types: MII, MI_I, MLX, MMI, M_MI, MFI, MMF, MIB, MBB, BBB, MMB, MFB. Each basic type has two versions, one with a stop after the third slot and one without.

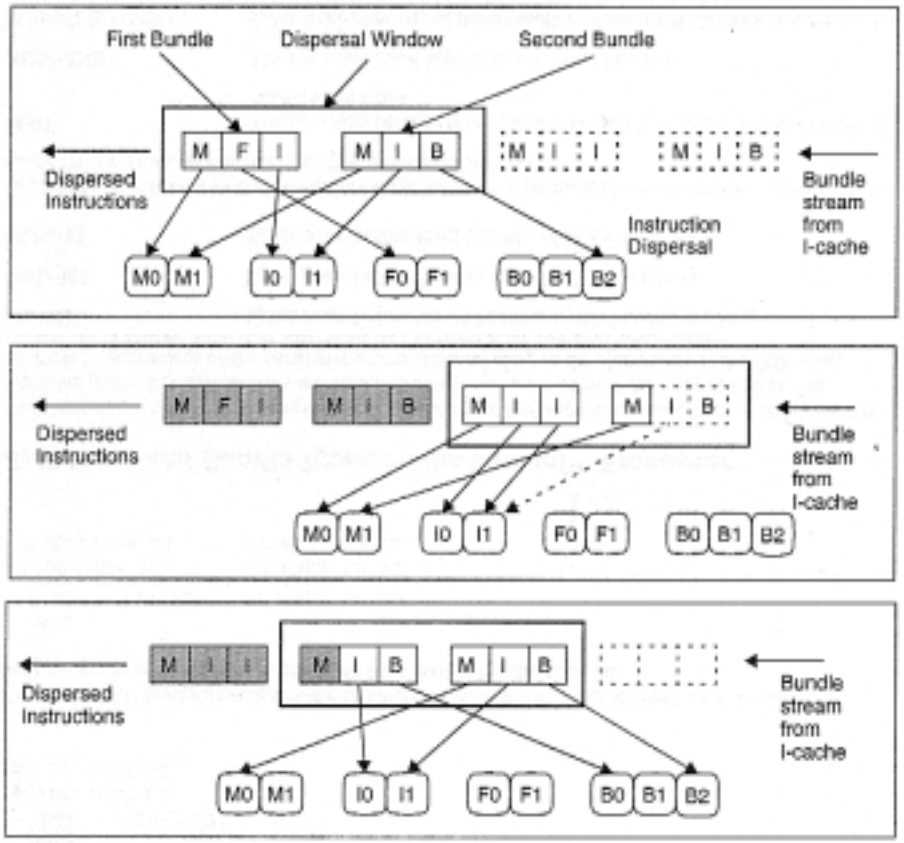
- An instruction group is a sequence of instructions starting at a given bundle address up to the first stop, taken branch, break.b instruction fault or illegal operation fault.

IA-64 Architecture

- Instruction groups can extend over an arbitrary number of bundles and are determined by the location of stops in assembly code.

- Instruction execution phases:
  - fetch: read the instruction from memory
  - read: read architectural state
  - execute: perform the specified operation
  - update: update architectural state, if necessary

IA-64 Architecture: Bundling and Dispersal on Itanium



IA-64 Architecture

- Between instruction groups: instructions behave as though read accesses occur after the update of all instructions from the previous group.

- Within instruction groups: instructions behave as though
  - memory read accesses occurred after the update of the memory of all prior instructions in the same group.
  - register read accesses occurred before the update of the register state by any instruction (prior or later) in the same instruction group – with the exception of some dependence requirements (see next slide).

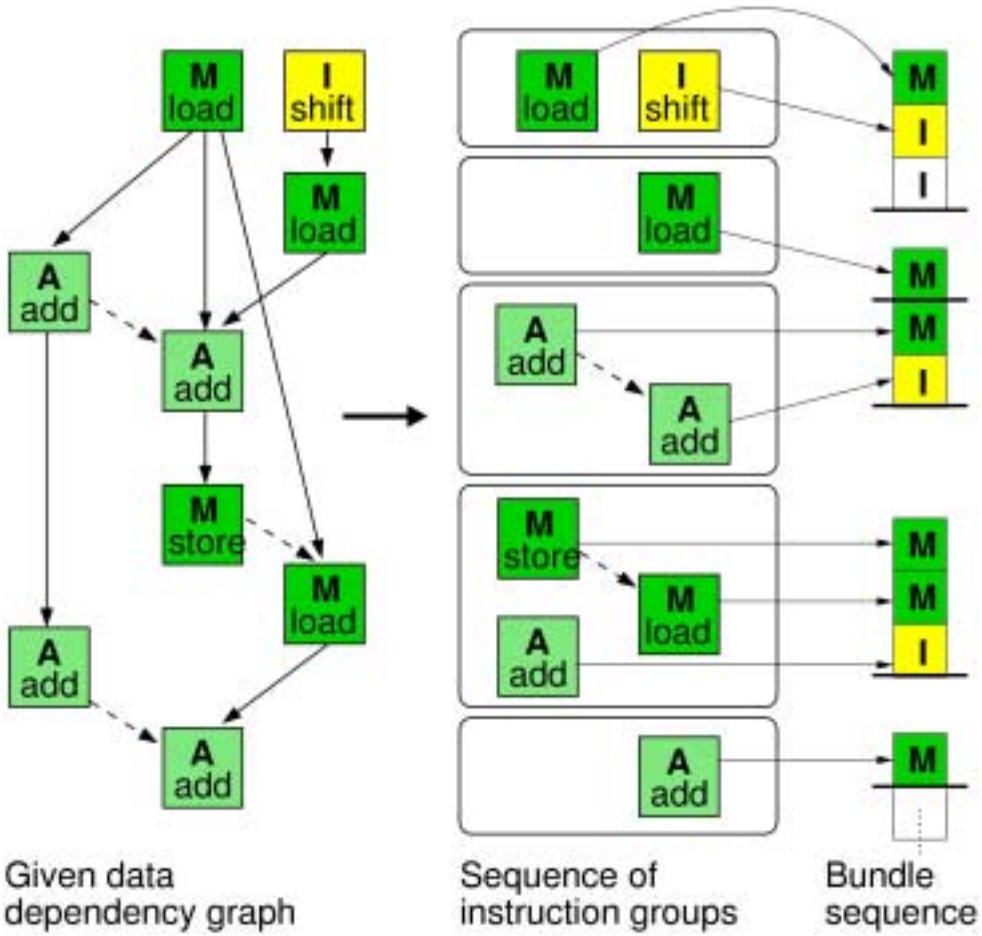
IA-64 Architecture

- Register dependencies: Within an instruction group, read-after-write (RAW) and write-after-write (WAW) register dependencies are not allowed (with respect to explicit and implicit register accesses), except for some special cases. Write-after-read register dependencies (WAR) are allowed, except for some special cases.

- If the program violates the dependency rules within an instruction group, processor behavior is undefined.

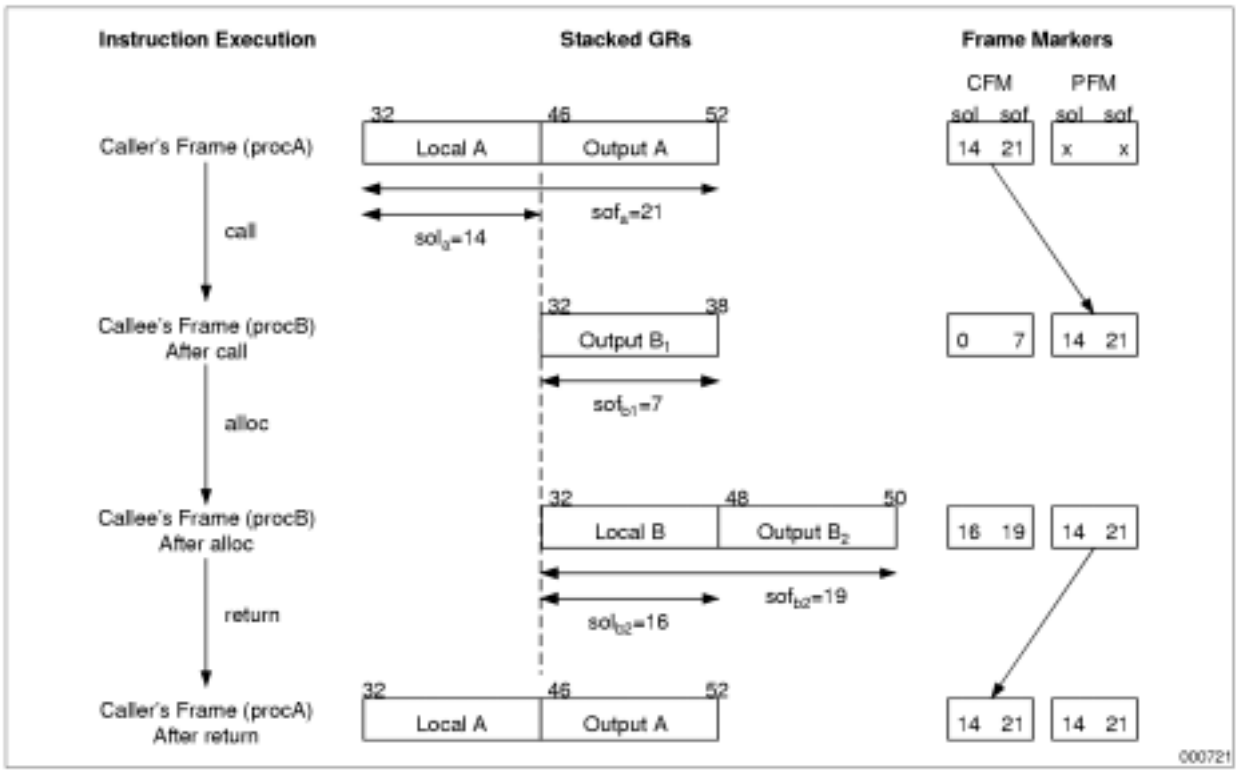
- Memory dependencies: Within an instruction group, RAW, WAW, and WAR memory dependencies are allowed. A load will observe the results of the most recent store to the same memory address.

IA-64 Architecture



IA-64 Architecture

- Register Stack Behavior on Procedure Call and Return



IA-64 Architecture

- Compare instructions and predication:
  - A compare instruction tests for a single specified condition and generates a boolean result which is written to predicate registers. The predicate registers can be used as conditions for conditional branches and as qualifying predicates for predication.
  - Most compare instructions set two predicate registers.
  - Compare instructions can have 5 types: Normal, Unconditional, AND, OR, or DeMorgan. The type defines how the instruction writes its target predicate registers.
  - AND, OR, and DeMorgan types are termed "parallel" compare types since they allow multiple simultaneous compares to target a single predicate register.

IA-64 Architecture

- Compare Types:

Compare Type	Completer	Operation	
		First Predicate Target	Second Predicate Target
Normal	*none*	if (qp) {target = result}	if (qp) {target = !result}
Unconditional	unc	if (qp) {target = result} else {target = 0}	if (qp) {target = !result} else {target = 0}
AND	and	if (qp && !result) {target = 0}	if (qp && !result) {target = 0}
	andcm	if (qp && result) {target = 0}	if (qp && result) {target = 0}
OR	or	if (qp && result) {target = 1}	if (qp && result) {target = 1}
	orcm	if (qp && !result) {target = 1}	if (qp && !result) {target = 1}
DeMorgan	or.andcm	if (qp && result) {target = 1}	if (qp && result) {target = 0}
	and.orcm	if (qp && !result) {target = 0}	if (qp && !result) {target = 1}

IA-64 Architecture

- Parallel compare instructions:

```
if (r1==1 || r2==2 || r3==3 || r4==4) then s;

cmp.ne p1,p0 = r0, r0;;        //init
cmp.eq.or p1,p0 = 1,r1
cmp.eq.or p1,p0 = 2,r2
cmp.eq.or p1,p0 = 3,r3
cmp.eq.or p1,p0 = 4,r4;;
(p1) s
```

IA-64 Architecture

- Speculation: premature execution of an operation
  - conditions:
    - low probability of required recovery
    - enhancement of instruction-level parallelism by speculative execution.
- Control speculation: Execution of an operation before the branch which guards it.
- Data speculation: Execution of a memory load prior to a store that precedes it in the operation sequence and that may alias with it.

IA-64 Architecture

Control speculation:

```
(p1)   br.cond label
       ld8 r4=[r3];;
       add r4=8,r4;;
```



```
       ld8.s r4=[r3];;
// ...
(p1)   br.cond label;;
       chk.s r4, recovery;;
back:  add r4=8,r4;;
// ...
recovery:
       ld8 r4=[r3];;
       br back;;
```

IA-64 Architecture

- Data Speculation:

```
//...
st8 [r4]=r12
ld8 r6=[r8];;
add r5=r6,r7;;
st8 [r18]=r5
```



```
        ld8.a r6=[r8];;
        // ...
        add r5=r6,r7;;
        // ...
        st8 [r4]=r12
        chk.a.clr r6, recovery
back:   st8 [r18]=r5
        // ...
recovery:
        ld8 r6=[r8];;
        add r5=r6,r7
        br back;;
```

IA-64 Architecture

- Multimedia instructions supporting packed arithmetic: a general purpose register is considered as concatenated of 8 8-bit, 4 16-bit, or 2 32-bit elements. Operations:
  - parallel modulo addition (padd), parallel modulo subtraction (psub), parallel compare (pcmp), etc.
  - Data arrangement: interleave odd elements from both sources (mix.l), convert from larger to smaller elements with signed saturation (pack.uss), etc.
- Integer (fixed-point) multiply is executed in the floating-point unit (multiply/accumulate). Example: xma f1=f3, f4, f2 computes f1:=f2+f3*f4.

Intel Itanium

- First IA-64 implementation.
- Functional units:
  - 2 M-type
  - 2 I-type
  - 2 F-type
  - 3 B-type
- All functional units are fully pipelined.
- Functional units of the same type are not identical; seldomly used operations usually can only be executed on the first unit of each type.

Intel Itanium

- Dispersal window contains two bundles allowing up to 6 microoperations to be issued in one clock cycle, among which at most 2 I-type, 2 M-type, 2 F-type, 3 B-type operations.
- The functional unit a microoperation is executed by is determined by its type and its position in the bundle.
- Execution time:
  - 1-2 cycles for most integer operations
  - 2-7 cycles for F-type operations
- 10-stage pipeline, divided into two parts:
  - the 3-stage *front end* reads the instruction stream
  - the 7-stage main pipeline executes the instructions
  - *front end* and main pipeline are decoupled; they are connected by a FIFO buffer that can contain up to 8 bundles.