

# **Conceptos de Arquitectura de Computadoras**

---

## **Clase 09**

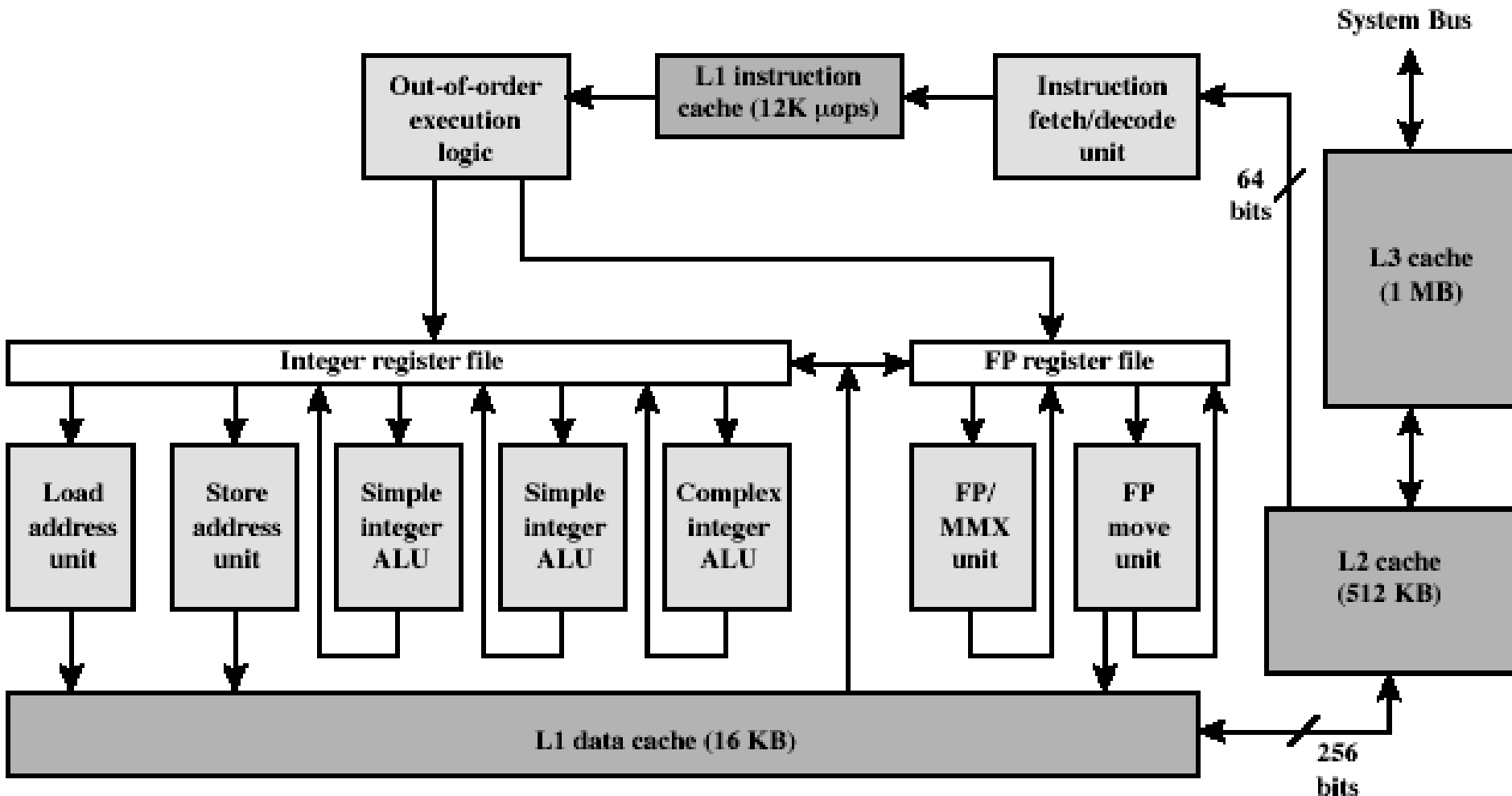
### **Superescalares y mas VLIW**

# Hacia Pentium 4

---

- 80486 - CISC
- Pentium – algún componente superescalar
  - 2 unidades de ejecución de enteros separadas
- Pentium Pro – todo superscalar
- Modelos siguientes refinan y mejoran el diseño superscalar

# Pentium 4: diagrama de bloques



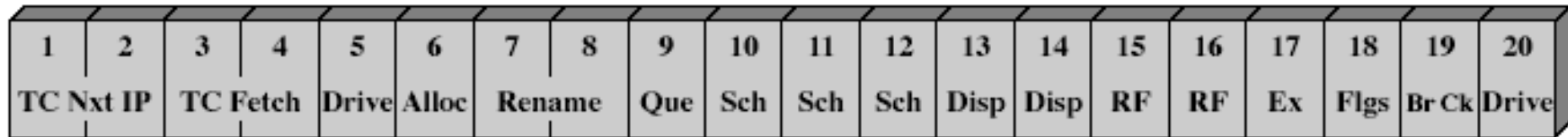
# Pentium 4. Operación

---

- Busca instrucciones en memoria en el orden del programa estático
- Traduce instrucciones en 1 o + instrucciones RISC
  - micro-operaciones
- Ejecuta las micro-ops en un cauce superscalar
  - micro-ops pueden ser ejecutadas fuera de orden
- Entrega resultados de micro-ops a los registros en el orden de flujo de programa original
- Caparazón CISC con núcleo RISC
- Cauce del núcleo interno de al menos 20 etapas
  - Algunas micro-ops requieren múltiples etapas de ejecución

# Cauce segmentado de Pentium 4

---



TC Next IP = trace cache next instruction pointer

TC Fetch = trace cache fetch

Alloc = allocate

Rename = register renaming

Que = micro-op queuing

Sch = micro-op scheduling

Disp = Dispatch

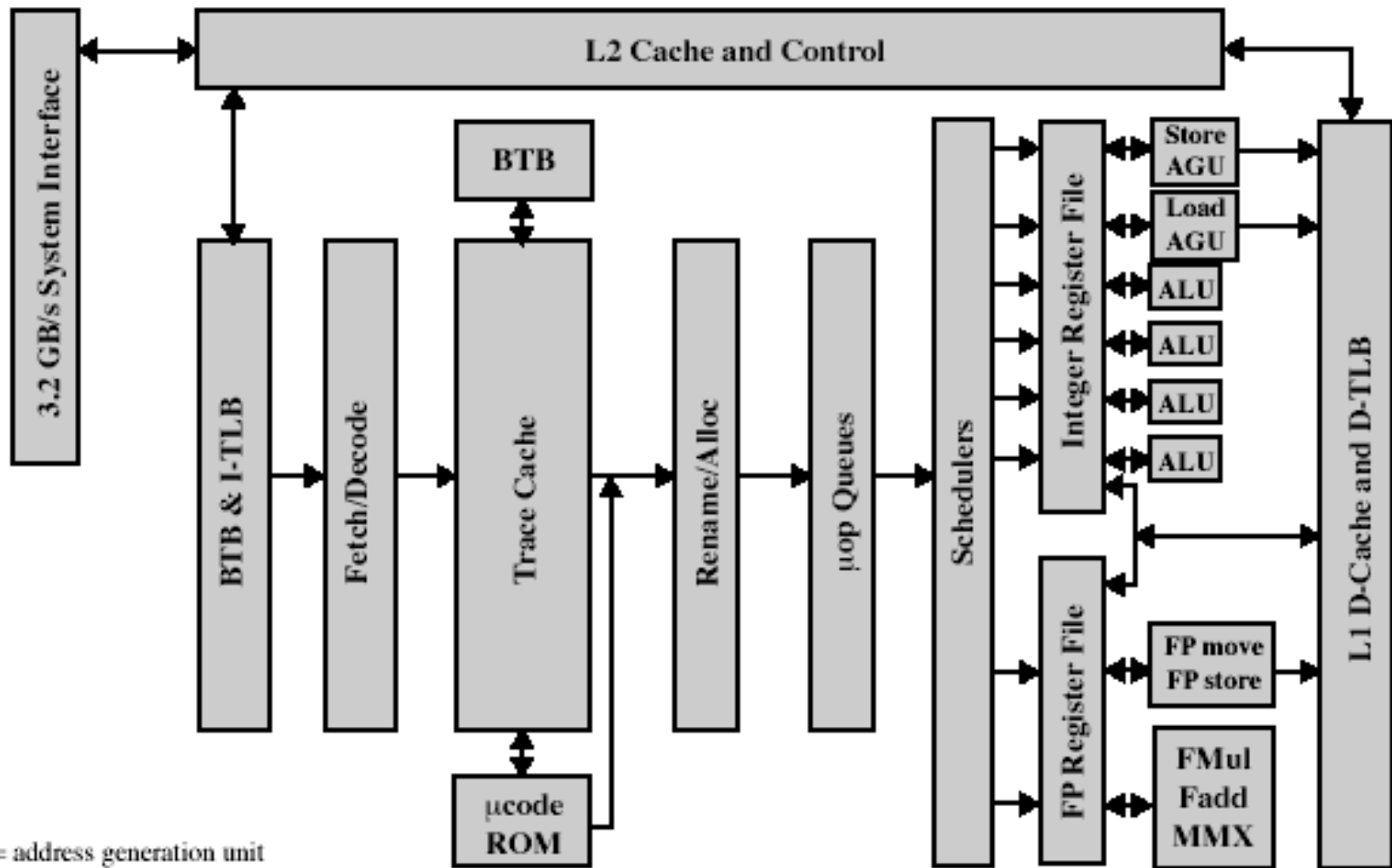
RF = register file

Ex = execute

Flgs = flags

Br Ck = branch check

# Pentium 4: segmentación



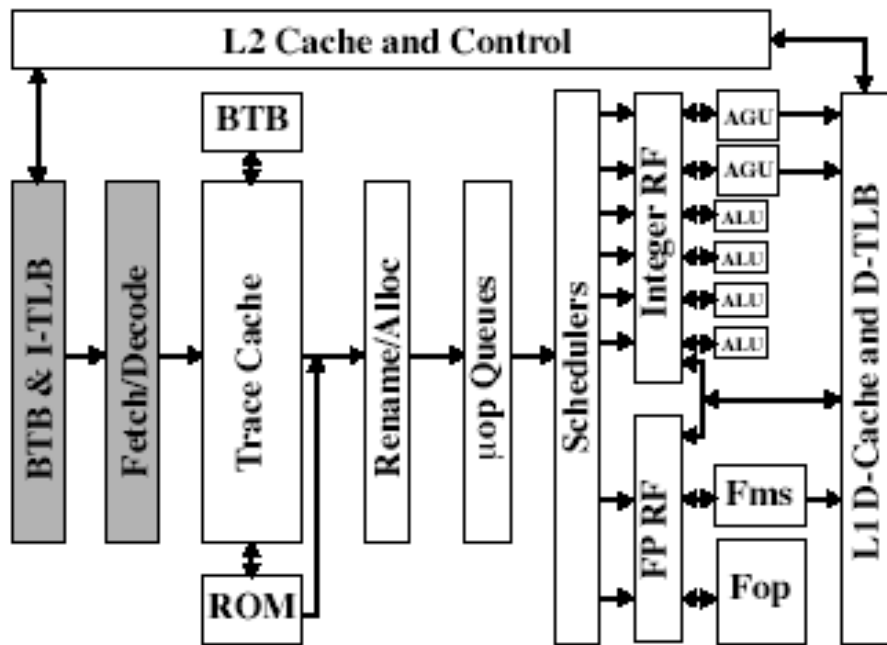
AGU = address generation unit

BTB = branch target buffer

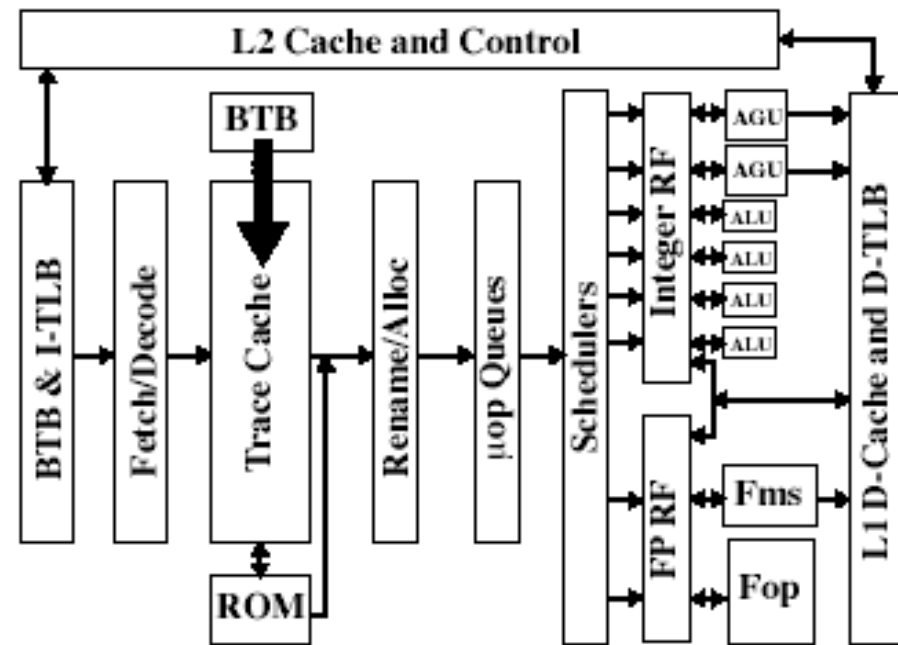
D-TLB= data translation lookaside buffer

I-TLB = instruction translation lookaside buffer

# Operación del cauce

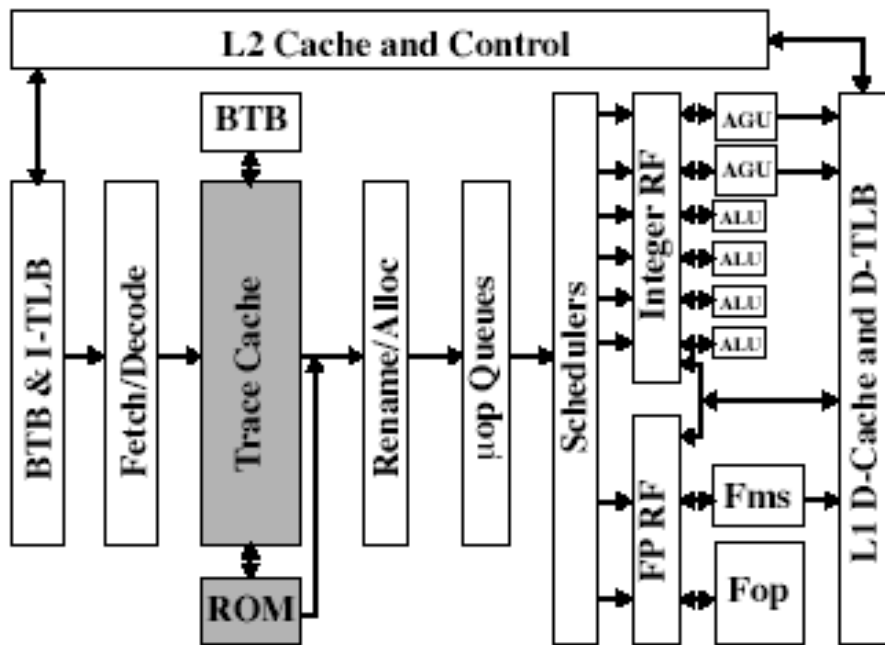


(a) Generation of micro-ops

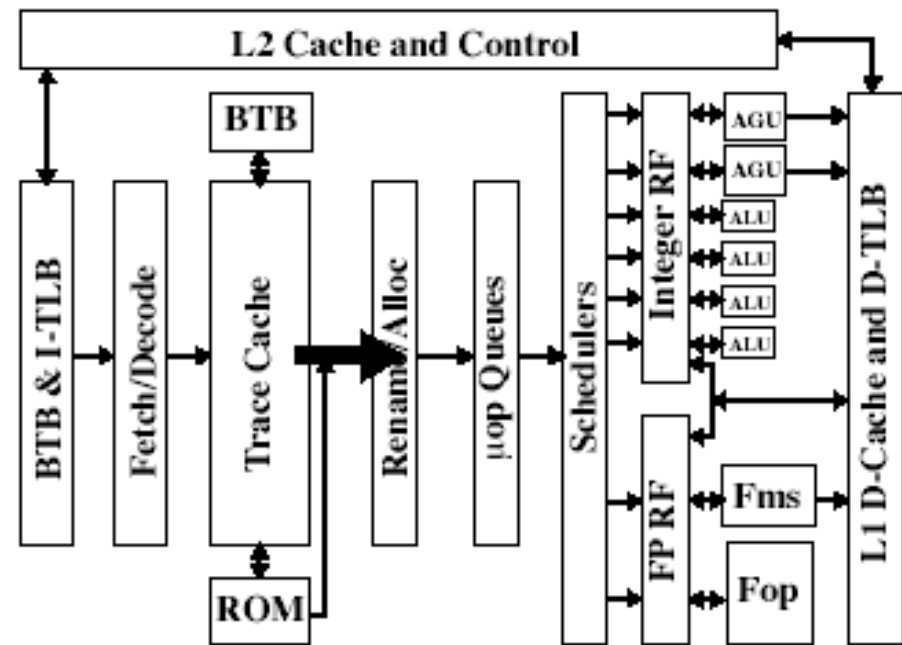


(b) Trace cache next instruction pointer

# Operación del cauce (2)



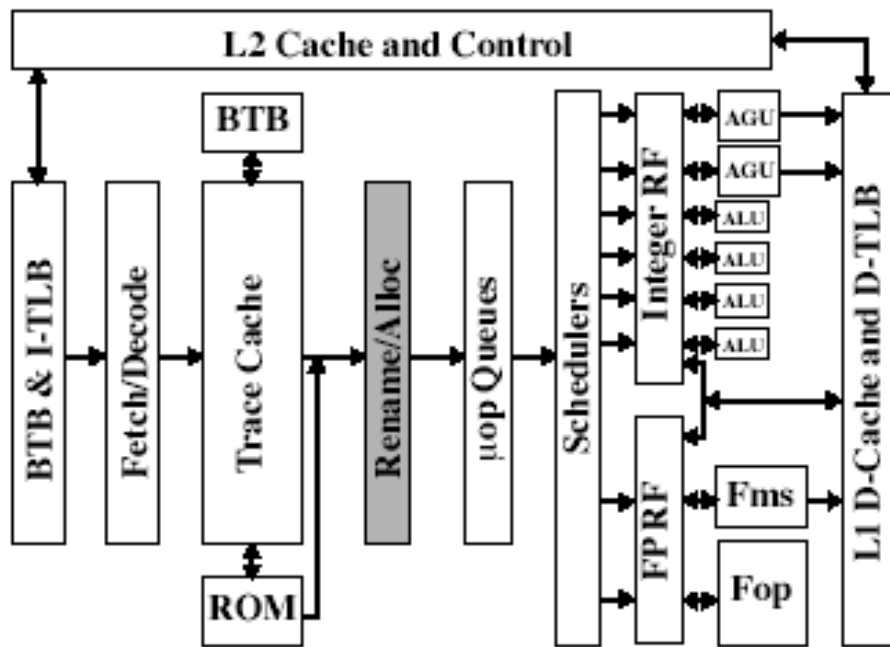
(c) Trace cache fetch



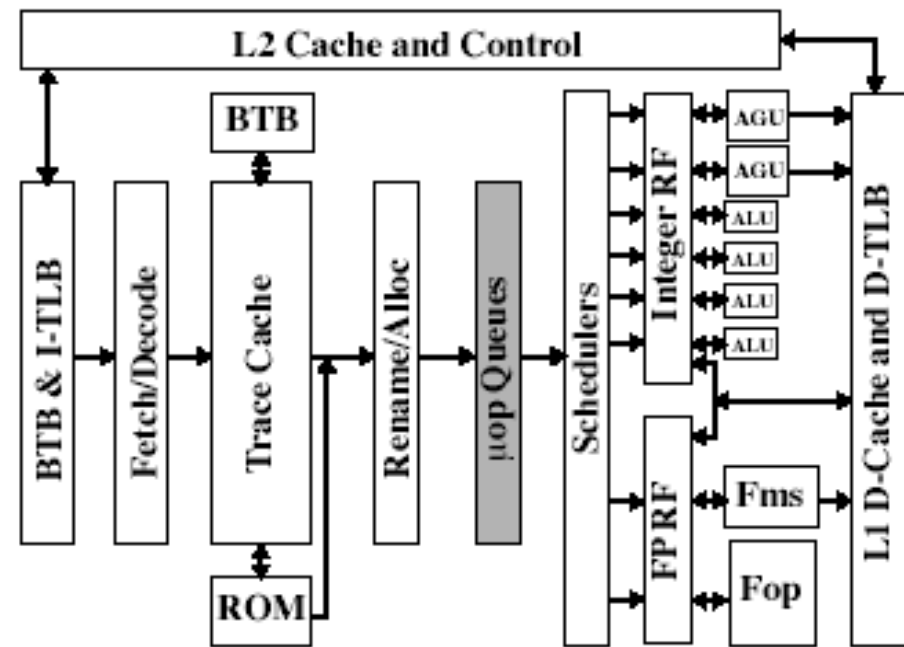
(d) Drive



# Operación del cauce (3)

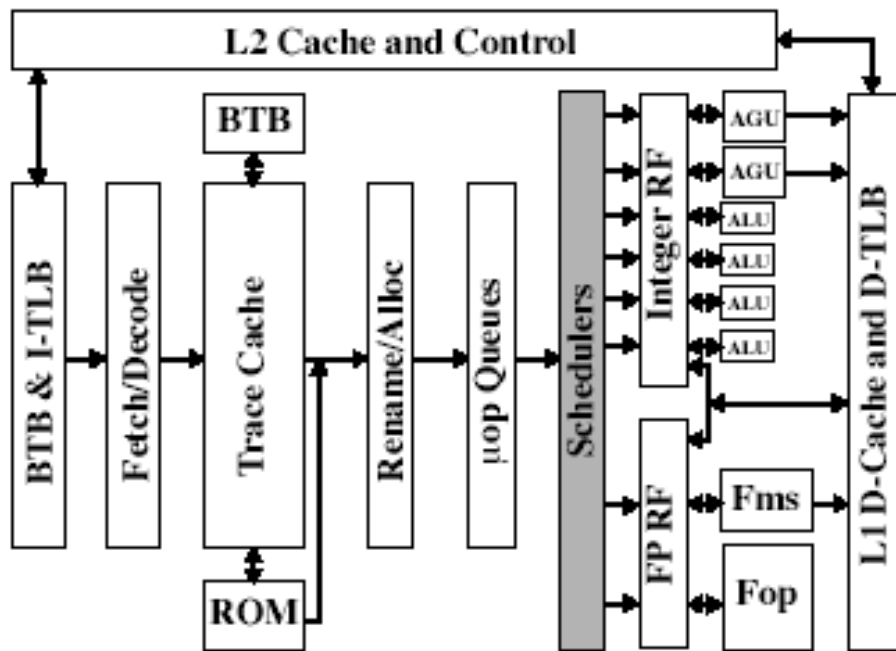


(e) Allocate; Register renaming

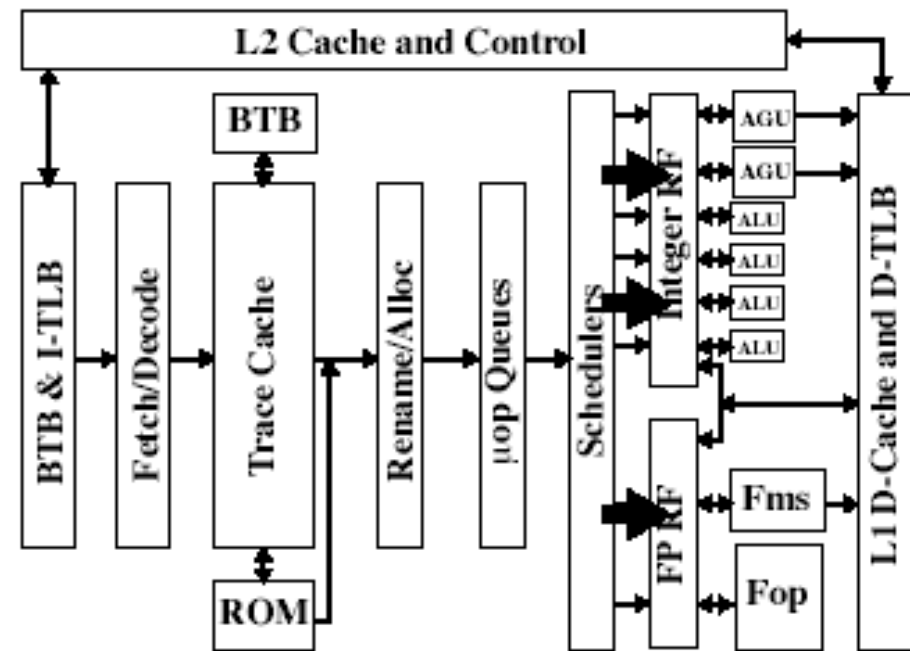


(f) Micro-op queuing

# Operación del cauce (4)

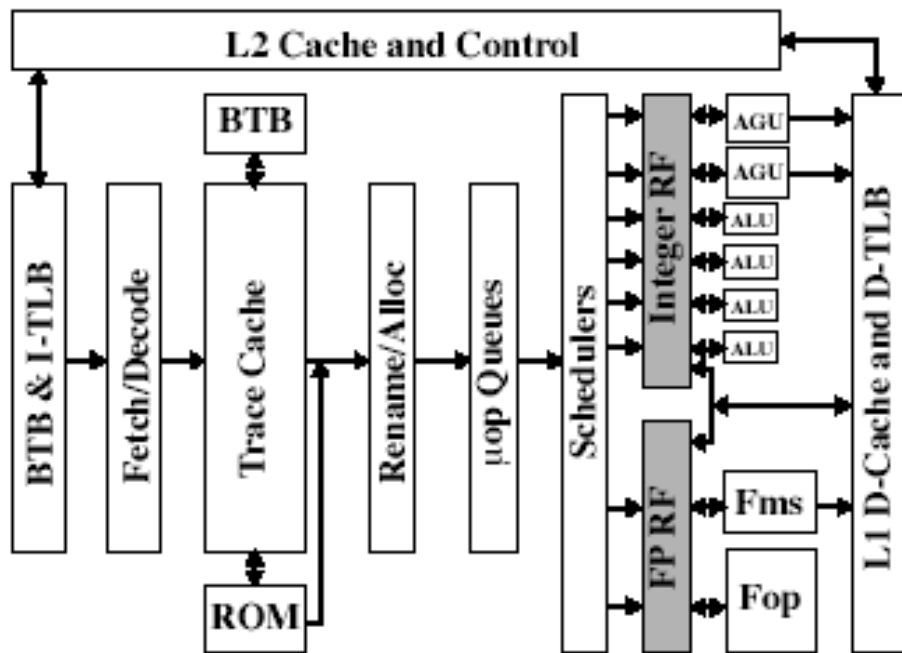


(g) Micro-op scheduling

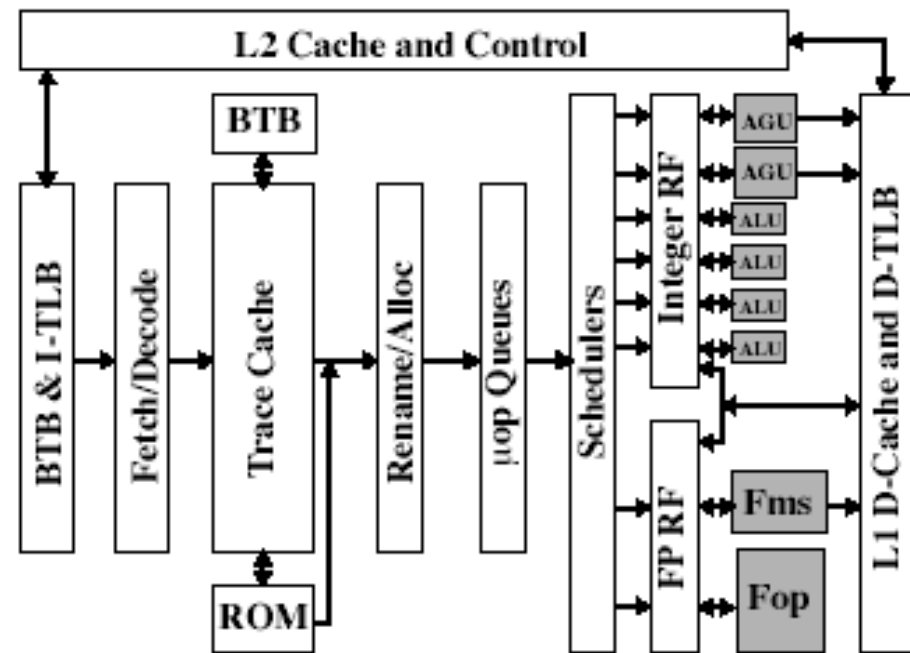


(h) Dispatch

# Operación del cauce (5)

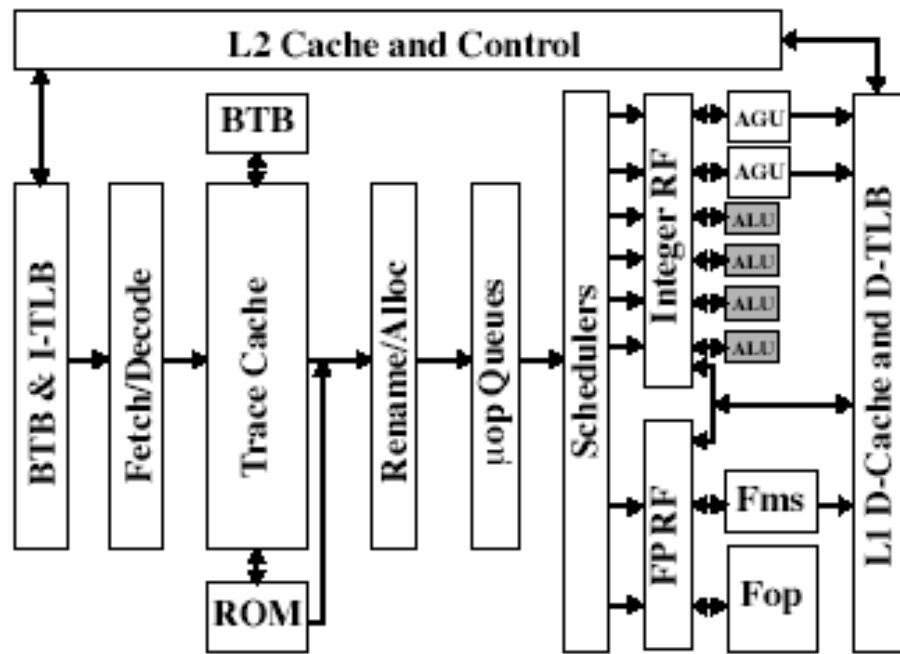


(i) Register file

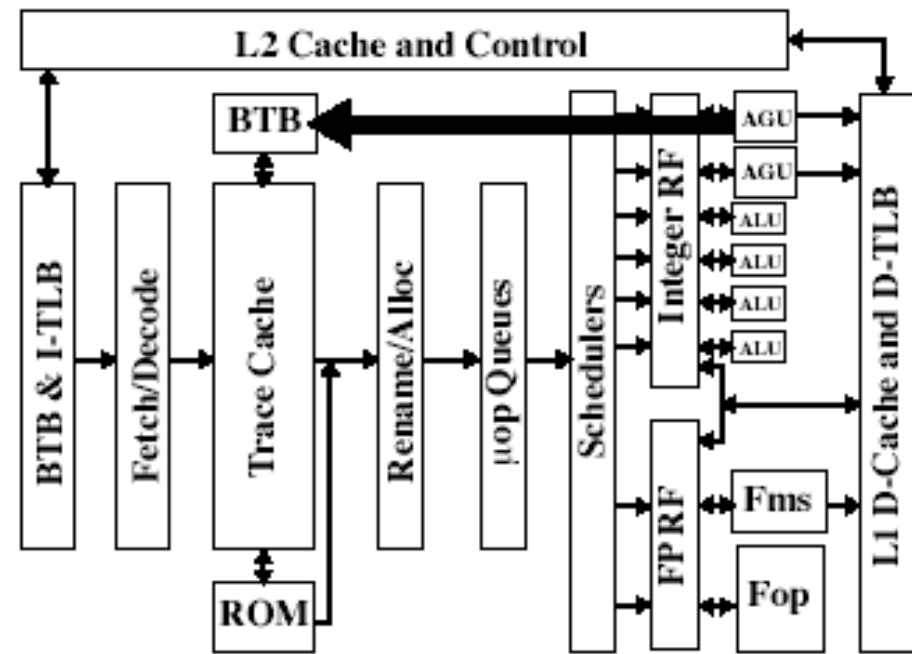


(j) Execute; flags

# Operación del cauce (6)



(k) Branch check



(l) Branch check result

# Hacia IA64

---

- Pentium 4 aparece como el último de la línea x86
- Desarrollo conjunto de Intel y Hewlett-Packard
- Nueva arquitectura
  - 64 bit
  - No es extensión de x86
  - No adapta la arquitectura HP de sus RISC de 64 bits
- Utiliza muchos circuitos a altas velocidades
- Hace uso sistemático del paralelismo
- Base superscalar

# Motivación

---

- Paralelismo a nivel instrucción - ILP
  - Implícito en la instrucción de máquina
  - No determinado por el procesador en tiempo de ejecución
- Palabras de instrucción larga o muy larga (LIW/VLIW)
- Predicación de saltos (no es predicción de saltos)
- Carga especulativa

Intel-HP lo llaman EPIC **E**xplicit **P**arallel **I**nstruction **C**omputing

- Arquitectura IA-64 pensada para implementación EPIC
- Itanium es el primer producto de Intel

# ¿Porqué nueva arquitectura?

---

- Hardware no compatible con x86 (IA32)
- Muchos millones de transistores disponibles en chip
  - Se pueden armar caches mas grandes
  - Se pueden colocar varios procesadores
  - Se pueden agregar mas unidades de ejecución
    - Aumenta superescalado, procesador mas "ancho"
    - Se necesita mas lógica para dirigir, mejor predicción de saltos, cauces mas largos, mas registros de "renaming"
    - Hay mayores penalidades por mala predicción
    - Retiro de hasta seis instrucciones por ciclo

# Paralelismo explícito

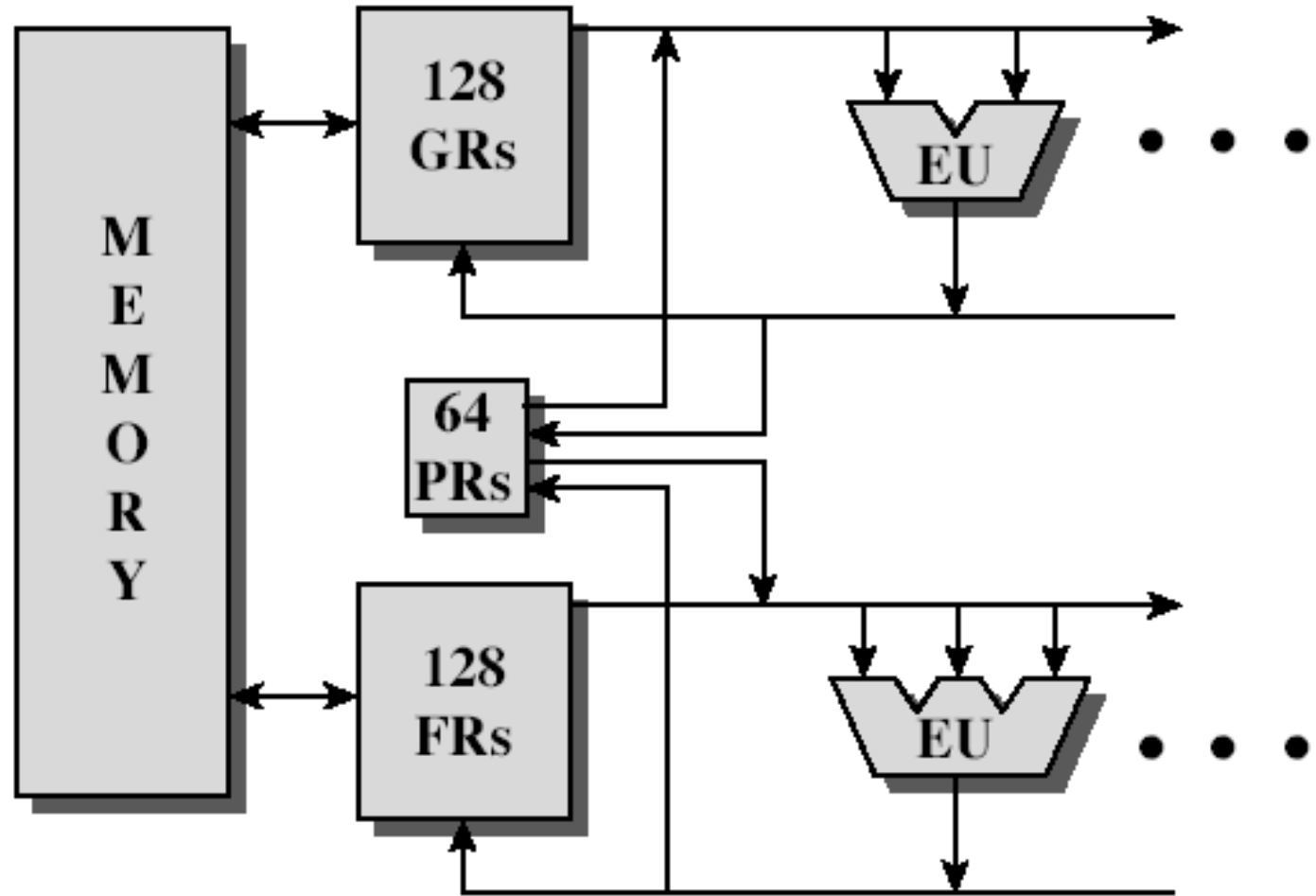
---

- Paralelismo de instrucción programado en tiempo de compilación
  - Incluido con la instrucción de máquina
- El procesador usa esa información para realizar ejecución paralela
- Requiere circuitos menos complejos
- El compilador tiene mas tiempo para determinar posibles operaciones paralelas
- El compilador mira todo el programa



# Organización IA-64

---



GR = General-purpose or integer register  
FR = Floating-point or graphics register  
PR = One-bit predicate register  
EU = Execution unit

# Características clave

---

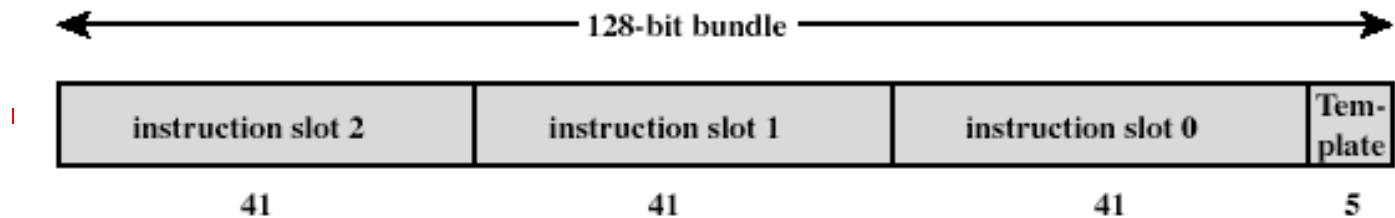
- Gran número de registros
  - Formato de instrucción de IA-64 asume 256 registros
    - 128 de 64 bit - para enteros, lógica y propósito general
    - 128 de 82 bit - para punto flotante y gráficos
  - 64 registros de 1 bit - para ejecución predicada
  - Soportan alto grado de paralelismo
- Múltiples unidades de ejecución
  - Se espera 8 o mas
  - Depende del número de transistores disponible
  - Ejecución paralela depende en hardware disponible
    - 8 instrucciones en paralelo pueden dividirse en 2 lotes de 4 si hay 4

# Unidades de ejecución

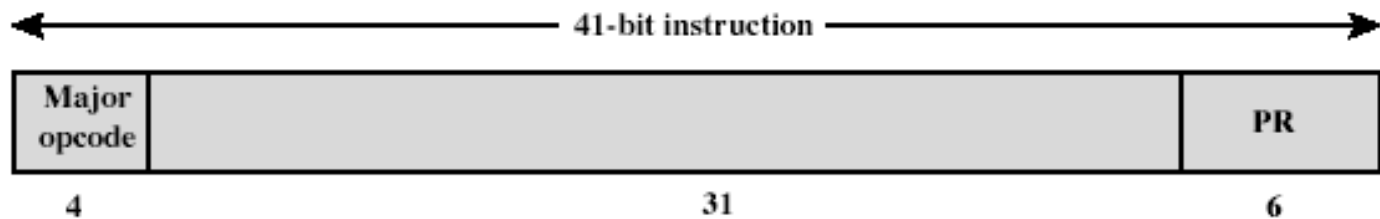
---

- I-Unit
  - Aritmética de enteros
  - Suma y desplazamientos
  - Lógica
  - Comparaciones
  - Operaciones multimedia de enteros
- M-Unit
  - Load y Store
    - entre registro y memoria
  - alguna ALU de enteros
- B-Unit
  - Instrucciones de salto
- F-Unit
  - Instrucciones de punto flotante

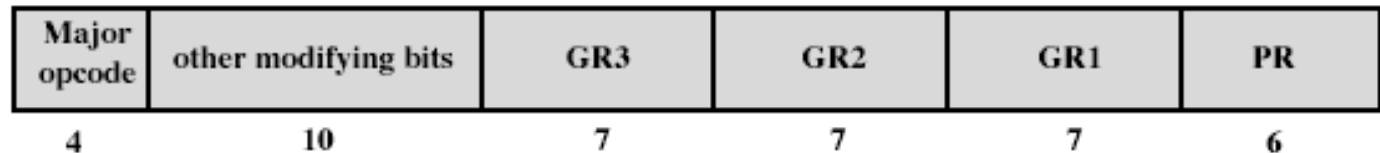
# Formato de instrucción IA-64



(a) IA-64 bundle



(b) General IA-64 instruction format



(c) Typical IA-64 instruction format

PR = Predicate register

GR = General or floating-point register

# Ejecución con predicados

---

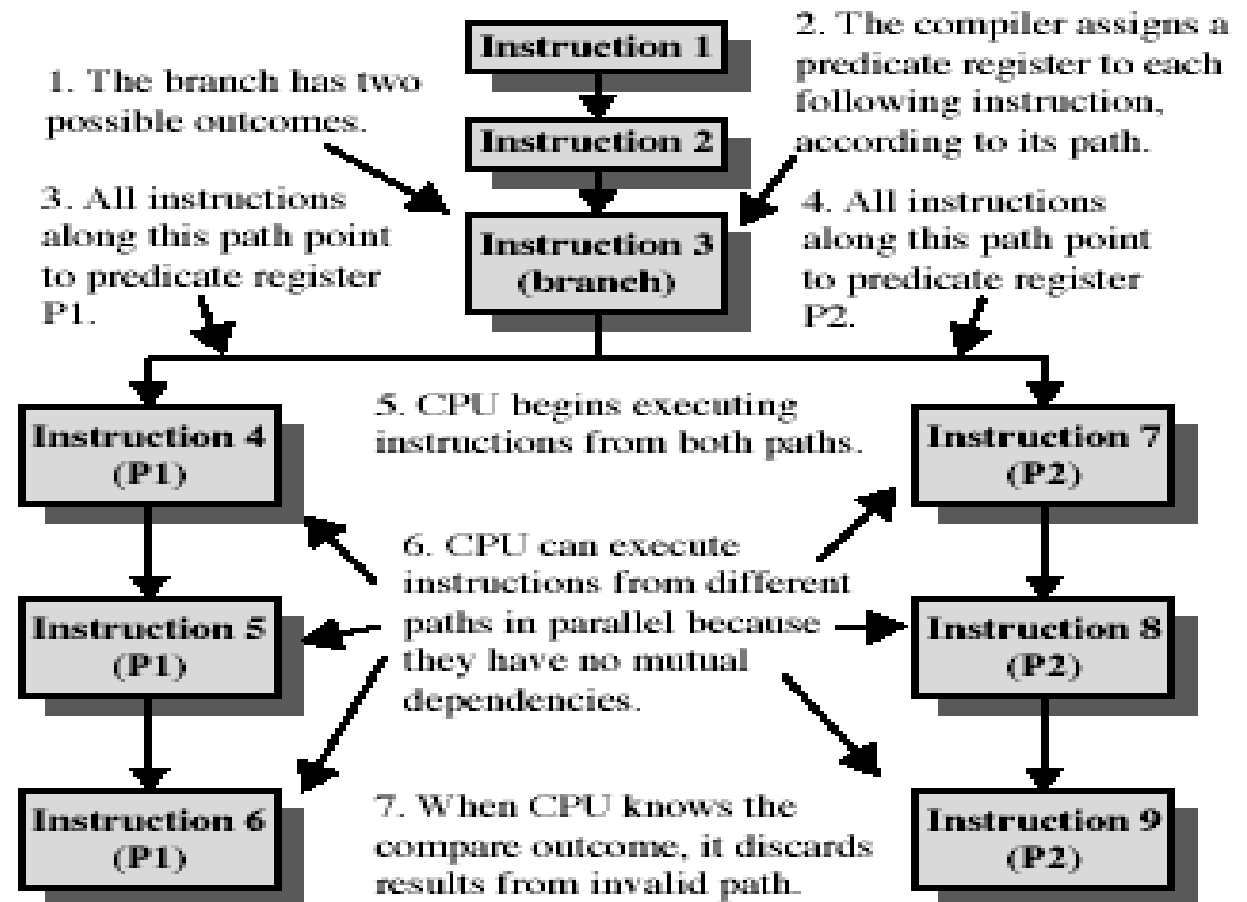
- Cualquier operación puede referirse a un registro de predicado
  - $\langle \text{PRi} \rangle$  operación
- La operación se retirará (resultados visibles) sólo cuando el valor del predicado sea verdad ( $\text{PRi}=1$ )
  - Si el valor se conoce cuando la instrucción se emite, la operación es ejecutada sólo si ese valor es verdadero
  - Si el valor no se conoce, la operación se inicia; si el valor se convierte en falso, la operación se descarta
- Si no se menciona registro de predicado, la operación es ejecutada y retirada incondicionalmente

# Saltos predicados

---

- Es una técnica de compilación
  - Para generar código con alto grado de ILP
- Se basa en la ejecución con predicados de la IA-64
- Se deja que ambas ramas de un salto condicional se ejecuten en paralelo
  - Para explotar todo el potencial de paralelismo
- Se eliminan los saltos y reemplazan por ejecución condicional
  - Se requiere soporte de hardware

# Saltos predicados



The compiler might rearrange instructions in this order, pairing instructions 4 and 7, 5 and 8, and 6 and 9 for parallel execution.

Instruction 1	Instruction 2	Instruction 3
Instruction 4	Instruction 7	Instruction 5
Instruction 8	Instruction 6	Instruction 9

(a) Predication

# Ubicación de operaciones de carga

---

- Una instrucción LOAD (carga desde memoria) puede ser reubicada de modo de evitar la latencia de memoria
  - el valor debe estar cuando se necesite
- ¿si se mueve de dentro de un salto?
  - La carga se ejecuta para ambas ramas
    - Si hay recursos hardware disponibles, no habría problemas
    - Si no se necesita la rama, será tiempo perdido
  - Se debe monitorear si provoca una excepción (ej. fallo de cache)
    - Interesa cuando la rama es útil por los recursos y el tiempo que la gestión de la misma necesita

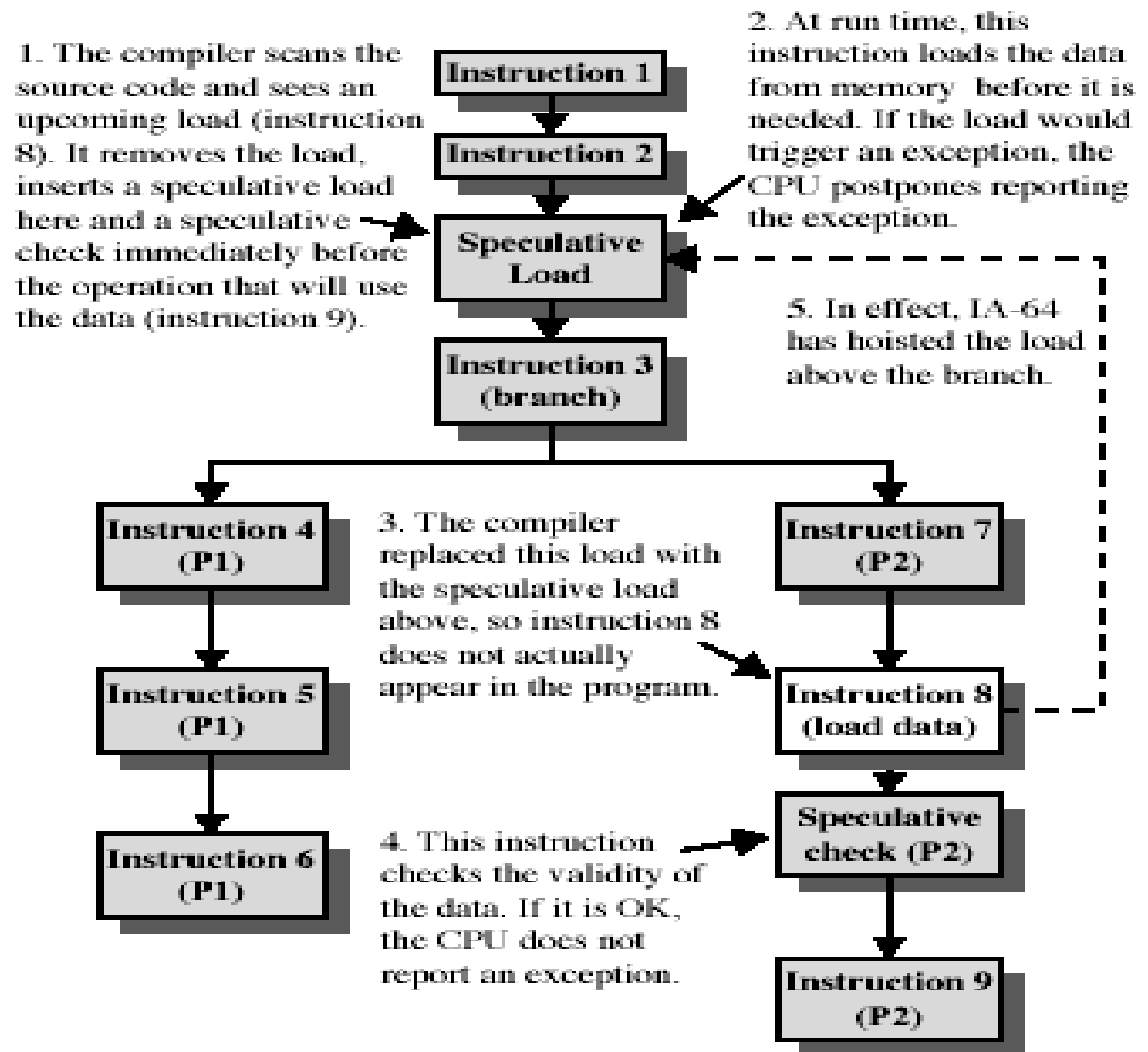


# Carga especulativa

---

- La instrucción LOAD en el programa original puede ser reemplazada por 2 instrucciones
  - Carga especulativa (ej. LOAD.S)
    - Búsqueda en memoria
    - Detección de excepción generada. No se reporta al S.O.
    - Ésta instrucción es la que cambia de lugar
  - Chequeo (ej. CHK.S)
    - Reporta la excepción si la carga especulativa la detectó
    - Ésta instrucción queda en su lugar
    - Si no se detectó excepción, no pasa nada
- Puedo manejar excepciones

# Carga especulativa



(b) Speculative loading

# Comparación

---

## Superescalar

- Instrucciones de línea RISC, una por palabra
- Múltiples unidades de ejecución paralela
- Reordena y optimiza flujo de instr en tiempo de ejecución
- Predicción de salto con ejecución especulativa de un camino
- Carga datos de memoria sólo cuando necesita e intenta en cache primero

## IA-64

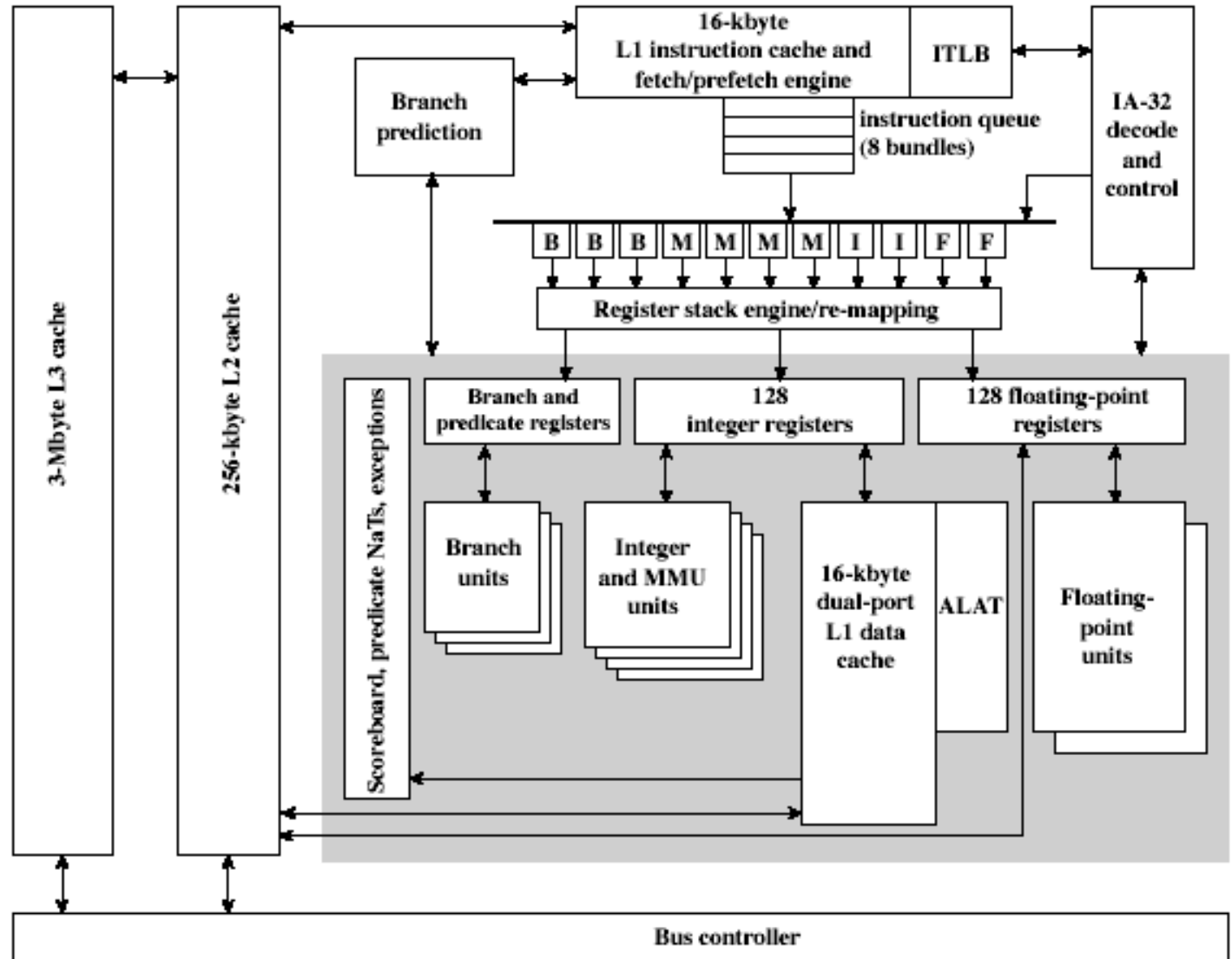
- Instrucciones de línea RISC, empaquetadas en grupos de a 3
- Múltiples unidades de ejecución paralela
- Reordena y optimiza flujo de instr en tiempo de compilación
- Ejecución especulativa a lo largo de ambos caminos de un salto
- Carga especulativa de datos antes de necesitarlos e intenta en cache primero

# Organización del Itanium

---

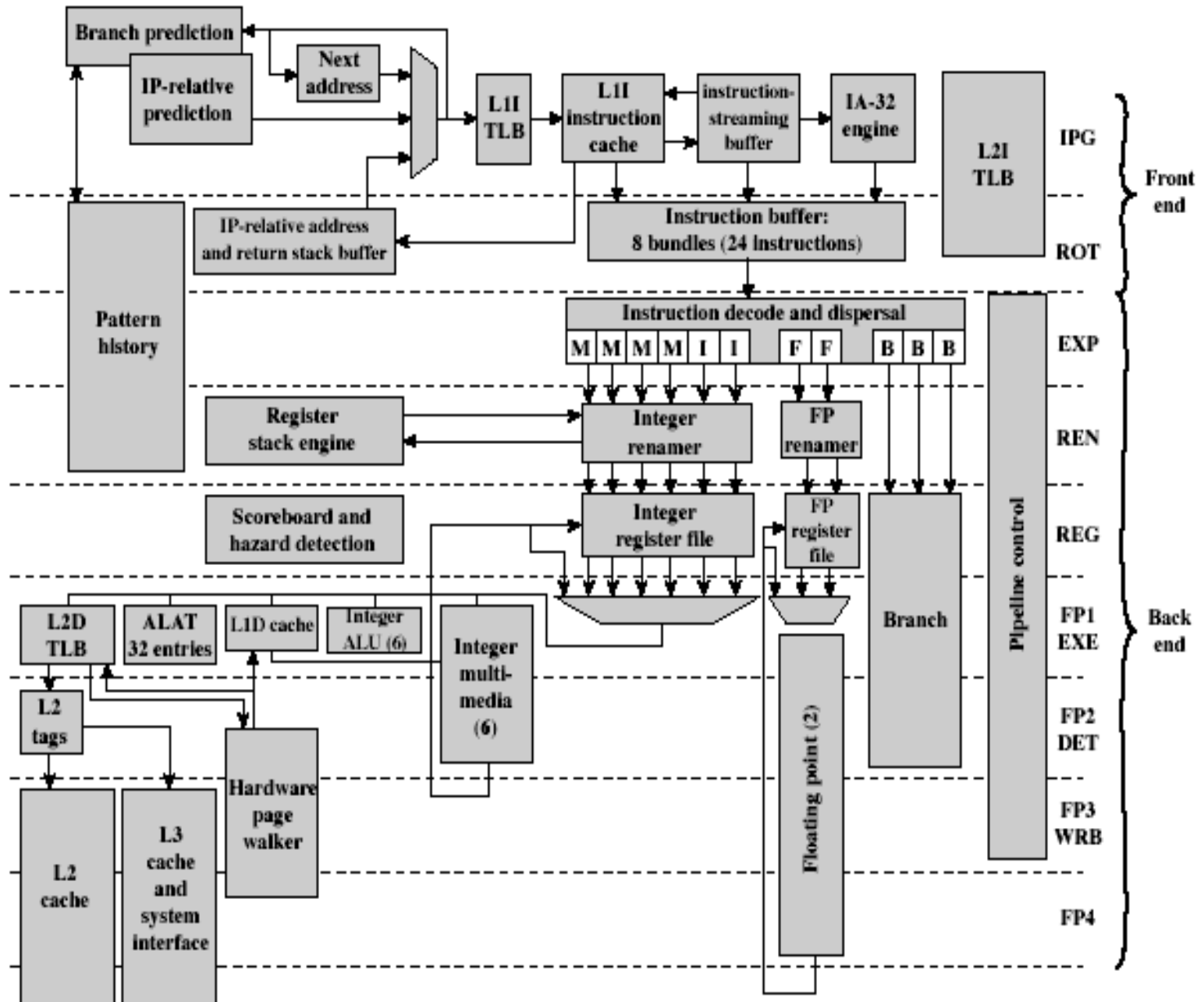
- Características Superscalares
  - Ancho 6, 10 etapas en el cauce segmentado
  - Prebúsqueda dinámica
  - Predicción de saltos
  - Registros marcadores (scoreboard) para optimización en tiempo de compilación no determinística
- Características EPIC
  - Soporte Hardware para ejecución predicada
  - Control y datos especulativos
  - Segmentación por software (desenrollado de bucles)

# Itanium2: Organización



# Itanium2

## cauce



# Lecturas recomendadas

---

- *Organización y Arquitectura de Computadoras*, W.Stallings, Capítulo 13 en 5º ed. ó Capítulos 14 y 15 en 7º ed.
- *Diseño y evaluación de arquitecturas de computadores*, M. Pardo y A. Guzmán, Capítulo 3.3. 1º ed.