

# ARQUITECTURAS AVANZADAS DE PROCESADORES

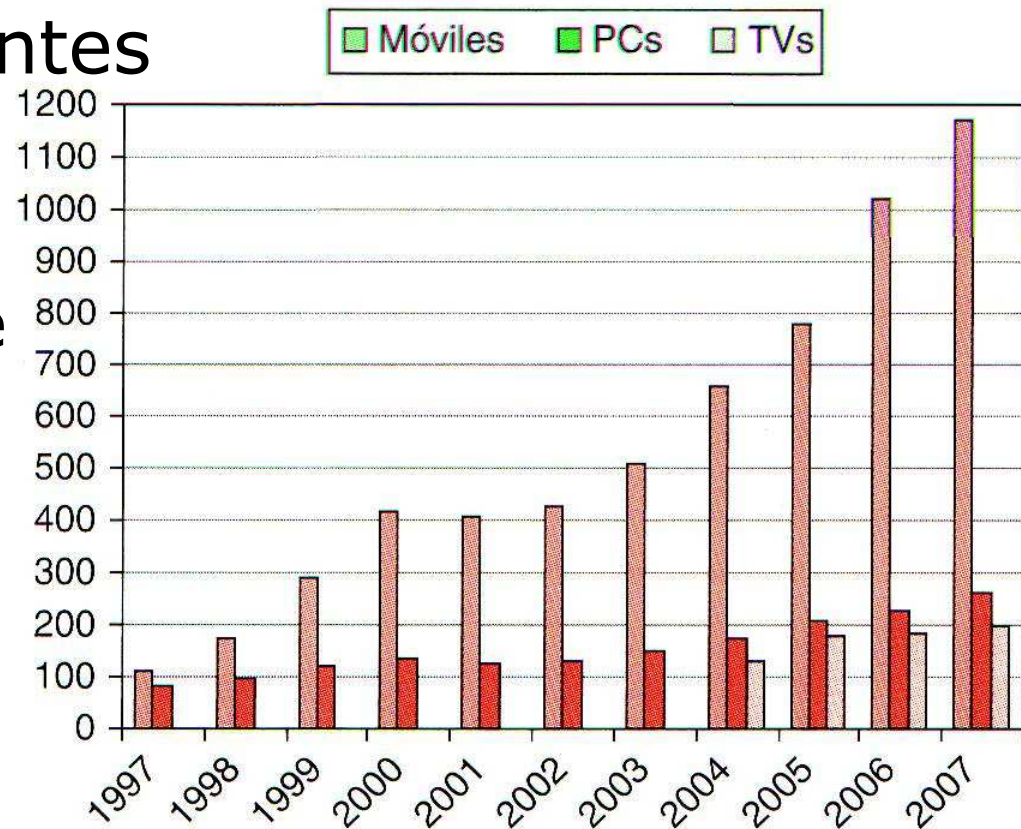


UNIDAD II. ABSTRACCIONES Y  
TECNOLOGÍA DE LOS COMPUTADORES.  
RENDIMIENTO DE UN COMPUTADOR

# ABSTRACCIONES Y TECNOLOGÍA DE LOS COMPUTADORES

- Los computadores se pueden clasificar en tres clases diferentes según sus aplicaciones:

- Computadores de sobremesa
- Servidores
- Computadores empotrados



# ABSTRACCIONES Y TECNOLOGÍA DE LOS COMPUTADORES

---

- ❑ Los programadores interesados en las prestaciones deben centrar sus estudios en la jerarquía de la memoria y la naturaleza paralela de los procesadores
- ❑ Los programadores que buscan construir versiones competitivas de los compiladores, los sistemas operativos, las bases de datos, e incluso de aplicaciones, tendrán que incrementar su conocimiento de la organización de los computadores.
- ❑ Intentaremos contestar a las siguientes preguntas cuando acabemos esta asignatura:
  - ¿Cómo se escriben los programas en un lenguaje de alto nivel, cómo se traducen al lenguaje del hardware, y cómo ejecuta el hardware el programa resultante?
  - ¿Cuál es la interfaz entre el software y el hardware, y cómo el software instruye al hardware para realizar las funciones necesarias?
  - ¿Qué determina las prestaciones de un programa, y cómo un programador puede mejorarlo?
  - ¿Qué técnicas pueden usar los diseñadores de hardware para mejorar las prestaciones?
  - ¿Cuáles son las razones y consecuencias del reciente paso del procesamiento secuencial al paralelo?

# ABSTRACCIONES Y TECNOLOGÍA DE LOS COMPUTADORES

---

- ▣ Las prestaciones de un programa dependen de una combinación de efectividad de los algoritmos usados en el programa, de los sistemas del software usados para crear y traducir el programa en instrucciones máquina, y de la efectividad del computador al ejecutar esas instrucciones

Componente Hardware o software	Cómo afecta este componente a las prestaciones	Dónde se cubre este tema
Algoritmo	Determina el número de sentencias de alto nivel y el número de operaciones de E/S que se ejecutarán	¡Otros libros!
Lenguaje de programación, compilador y arquitectura	Determina el número de instrucciones máquina que se ejecutarán por cada sentencia de alto nivel	capítulos 2 y 3
Procesador y sistema de memoria	Determina cuán rápido se pueden ejecutar las instrucciones	capítulos 4, 5 y 7
Sistema de E/S (hardware y sistema operativo)	Determina cuán rápido se pueden ejecutar las operaciones de E/S	capítulo 6

---

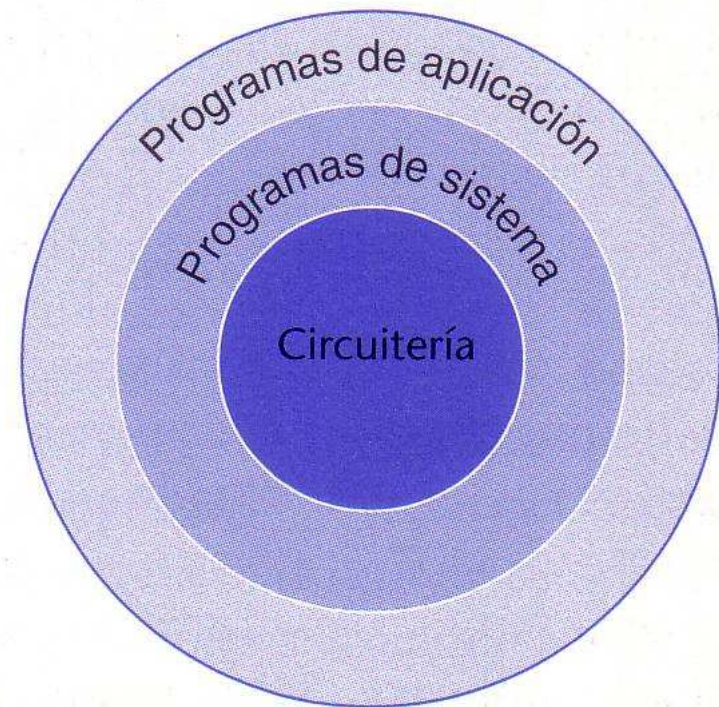


# ABSTRACCIONES Y TECNOLOGÍA DE LOS COMPUTADORES

## Bajo los programas

---

- ❑ Organización jerárquica de las capas software y hardware para el estudio del funcionamiento de un computador
- ❑ En la capa software destacar dos fundamentalmente:
  - El sistema operativo: manejo de operaciones básicas de E/S, asignación de espacio de almacenamiento y memoria, facilitar la multitarea
  - El compilador: traducción de un programa de alto nivel a instrucciones que el hardware puede ejecutar, tarea hoy en día muy compleja



# ABSTRACCIONES Y TECNOLOGIA DE LOS COMPUTADORES

## Del lenguaje de alto nivel al lenguaje hardware

- ❑ Las órdenes a los computadores se les da mediante instrucciones, es decir, agrupaciones de 0 y 1
- ❑ En vez de utilizar combinaciones de 0 y 1, se utiliza el lenguaje ensamblador, que mediante un programa lo que hace es traducir la versión simbólica de un programa (ensamblador) a la versión binaria
- ❑ Los lenguajes de alto nivel lo que hacen es traducir y simplificar el trabajo de los programadores; y los compiladores, crean el código (instrucciones del lenguaje ensamblador) para poder ser ejecutado en un computador
- ❑  $A+B$  (lenguaje de alto nivel) el compilador lo traduce a *add A, B* (lenguaje ensamblador) el programa ensamblador lo traduce a 1000110010100000 (lenguaje máquina)

Programa  
en lenguaje  
de alto nivel  
(en C)

```
swap(int v[], int k)
{
    int temp;
    temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```

Compilador de C

Programa  
en lenguaje  
ensamblador  
(para MIPS)

```
swap:
    muli $2, $5, 4
    add $2, $4, $2
    lw $15, 0($2)
    lw $16, 4($2)
    sw $16, 0($2)
    sw $15, 4($2)
    jr $31
```

Ensamblador

Programa  
en lenguaje  
máquina  
binario  
(para MIPS)

```
000000001010000100000000000011000
00000000000110000001100000100001
10001100011000100000000000000000
100011001111001000000000000000100
101011001111001000000000000000000
101011000110001000000000000000100
00000011111000000000000000001000
```

# ABSTRACCIONES Y TECNOLOGIA DE LOS COMPUTADORES

## Bajo la Cubierta

---

- Bajo la cubierta es lo que denominamos el hardware.
- **Tarea 1: Componentes básicos de un sistema computador**
- **Tarea 2: Tecnología para construir procesadores y memorias. ¿Qué es? ¿En qué se basa hoy la tecnología de fabricación de los microprocesadores?**

# RENDIMIENTO DE LOS COMPUTADORES

## El tiempo

---

- ❑ Complejidad de definir claramente el rendimiento de un computador.
- ❑ El tiempo como una medida del Rendimiento.
- ❑ A menor tiempo mayor rendimiento.
- ❑ ¿Qué tiempo es el que se tiene en cuenta?
  - Tiempo de ejecución de un programa medido en segundos
  - Tiempo de respuesta. Incluye:
    - ❑ Acceso a discos.
    - ❑ Acceso a memoria.
    - ❑ Gasto del Sistema Operativo.
  - En entornos multiprogramación se utiliza el tiempo de CPU (Unidad Central de Proceso).
    - ❑ No incluye E/S.
    - ❑ No incluye tareas del Sistema.
  - El tiempo de CPU se puede dividir en tiempo de CPU del usuario (ejecución del programa), y tiempo de CPU de sistema (tiempo empleado por el sistema operativo en realizar tareas requeridas por el programa)



# RENDIMIENTO DE LOS COMPUTADORES

## El tiempo de CPU

---

$$C_{\text{CPU}} = \text{NI} * \text{CPI}$$

$$t_{\text{CPU}} = \text{NI} * \text{CPI} * 1/\text{fr}$$

- ▣  $C_{\text{CPU}}$  = ciclos de reloj de la CPU
- ▣  $\text{NI}$  = n° de instrucciones
- ▣  $\text{CPI}$  = n° de ciclos por instrucción
- ▣  $\text{fr}$  = frecuencia de reloj

# RENDIMIENTO DE LOS COMPUTADORES

## El tiempo de CPU

---

- El aumento del rendimiento depende de tres factores.
  - fr: Tecnología hardware y organización.
  - CPI: organización, arquitectura a nivel de lenguaje máquina.
  - NI: arquitectura a nivel de lenguaje máquina y tecnología de compiladores.
- Otra forma de definir los ciclos de CPU es teniendo en cuenta los CPI de cada instrucción.

$$C_{CPU} = \sum_{i=1}^n (CPI_i * I_i)$$

- $I_i$  : n° de veces de la instrucción  $i$
- $CPI_i$ : ciclos por instrucción de la instrucción  $i$

$$t_{cpu} = \sum_{i=1}^n (CPI_i * I_i) * T \quad T \text{ es el periodo del reloj}$$

# RENDIMIENTO DE LOS COMPUTADORES

## El tiempo de CPU

---

- Ahora el CPI puede obtenerse (CPI medio):

$$CPI = \sum_{i=1}^n (CPI_i * \frac{I_i}{NI}) \quad NI \text{ n}^{\circ} \text{ de instrucciones}$$

- El cálculo de CPIi debe incluir fallos de caché y demás ineficiencias del sistema de memoria.
- Dilema:
  - Disminuir el n° de instrucciones: puede conducir a una organización con un T mayor.
  - Disminuir el CPI: puede conducir a un aumento del NI.
  - Aumentar fr: problemas de tecnología.
  - Hay que tener en cuenta la frecuencia de las Instrucciones.

# RENDIMIENTO DE LOS COMPUTADORES

## El tiempo de CPU

---

Ejemplo I:

- CPU-A: Una instrucción de comparación y una instrucción de salto condicional
  - CPIsalto = 2 ciclos
  - CPIresto = 1 ciclo
- CPU-B: Una comparación que incluye el salto (DBEQ).
  - CPIsalto = 2 ciclos
- El ciclo de reloj de la CPU-A es un 25% más rápido que el de la CPU-B.
  - CPU-A 20% de Salto => 20% de comparación.
  - CPU-B 20 de 80 de Salto => 25% de Salto => 80% de instrucción respecto a CPU-A

$$CPI_A = (0,20 * 2) + (0,80 * 1) = 1,2$$

$$t_{CPU_A} = NI_A * 1,2 * T_A = 1,2 * NI_A * T_A$$

$$CPI_B = (0,25 * 2) + (0,75 * 1) = 1,25$$

$$t_{CPU_B} = NI_B * 1,25 * T_B = 0,8 * NI_A * 1,25 * 1,25 * T_A = 1,25 * NI_A * T_A$$

- La CPU-A es más rápida a pesar de tener la CPU-B menos instrucciones.

# RENDIMIENTO DE LOS COMPUTADORES

## El tiempo de CPU

---

Ejemplo II:

- El mismo caso pero con unas diferencias en las frecuencias de reloj de solo el 10%

$$t_{CPU_A} = 1,20 * NI_A * T_A$$

$$t_{CPU_B} = (0,8 * NI_A) * 1,25 * (1,10 * T_A) = 1,10 * NI_A * T_A$$

- En este caso la máquina B es más rápida.



# RENDIMIENTO DE LOS COMPUTADORES

## El tiempo de CPU

Ejemplo III:

- ❑ Máquina A de carga/almacenamiento cuya frecuencia y ciclos de las instrucciones se da en la figura:

Operación	Frecuencia	Cuenta de ciclos de reloj
ops ALU	43 %	1
Cargas	21 %	2
Almacenamientos	12 %	2
Salto	24 %	2

- ❑ Máquina B: el 25% de las operaciones en la ALU utiliza un dato cargado que no se utiliza de nuevo.
- ❑ Se añaden instrucciones a la ALU con un operando en memoria de duración 2 ciclos.
- ❑ La duración de un salto es de 1 ciclo de reloj más.
- ❑ No se afecta la frecuencia del reloj.

# RENDIMIENTO DE LOS COMPUTADORES

## El tiempo de CPU

$$CPI_A = (0,43 * 1 + 0,21 * 2 + 0,12 * 2 + 0,24 * 2) = 1,57$$

$$t_{CPU_A} = NI_A * 1,57 * T_A$$

$$CPI_B = \frac{(0,43 - 0,25 * 0,43) * 1 + (0,21 - 0,25 * 0,43) * 2 + \dots}{\dots \frac{(0,25 * 0,43) * 2 + 0,12 * 2 + 0,24 * 3}{1 - (0,25 * 0,43)}} = 1,9076$$

$$t_{CPU_B} = (0,8925 * NI_A) * 1,9076 * T_A = 1,703 * NI_A * T_A$$

- Conclusión: añadir más instrucciones empeora el rendimiento

# RENDIMIENTO DE LOS COMPUTADORES

## Otras medidas de rendimiento MIPS

---

- ❑ *Medidas del rendimiento.*
- ❑ • MIPS: Millones de Instrucciones por segundo.
- ❑ " Una forma de intentar indicar el rendimiento de un computador.

$$MIPS = \frac{NI}{t_{ejecucion} * 10^6} = \frac{Freloj}{CPI * 10^6}$$

- ❑ Su relación con el tiempo de ejecución:

$$t_{ejecucion} = \frac{NI}{MIPS * 10^6}$$

- ❑ Ventajas de la métrica de MIPS.
  - Fáciles de entender.
  - Mayor MIPS mayor rapidez de ejecución.

# RENDIMIENTO DE LOS COMPUTADORES

## Otras medidas de rendimiento MIPS

---

### ❑ Inconvenientes:

- Los MIPS dependen del repertorio de Instrucciones. Difícil comparar máquinas con diferente conjunto de instrucciones.
- Los MIPS varían entre programas en un mismo computador
- Los MIPS pueden variar inversamente al rendimiento. Ejemplo de máquinas con hardware en punto flotante.
  - ❑ Si dispone baja los MIPS
  - ❑ Si no dispone suben los MIPS.
- Los MIPS no reflejan el tiempo de computación y pueden fallar al dar una visión del rendimiento del computador.
- Se puede dar como medida los MIPS relativos a una máquina

$$MIPS_{relativos} = \frac{\text{tiempo referencia}}{\text{tiempo no estimado}} * MIPS_{referencia}$$

tiempo referencia: tiempo del programa en la maquina de referencia

tiempo no estimado: tiempo de ejecución del programa en la máquina a medir

MIPSreferencia: MIPS de la máquina de referencia

# RENDIMIENTO DE LOS COMPUTADORES

## Otras medidas de rendimiento MIPS

---

### ❑ Inconvenientes:

- Problema de encontrar una máquina de referencia (VAX-11/780 en los años ochenta)
- A medida que el software se desarrolla (modernos computadores) hay dificultades de acoplarlos a máquinas de referencia.
- Los MIPS relativos no aportan demasiadas ventajas ya que hay que conocer el tiempo de ejecución, el programa y la entrada.



# RENDIMIENTO DE LOS COMPUTADORES

## Otras medidas de rendimiento MFLOPS

---

- MFLOPS: Millones de operaciones en punto flotante por segundo.

$$MFLOPS = \frac{\text{nº de operaciones en p. flotante del programa}}{\text{tiempo de ejecucion} * 10^6}$$

- Son aplicables a problemas que impliquen cálculo en P.F. (p. ej. para un compilador no sirve).
- No todas las operaciones en P.F. tienen la misma importancia.

Operaciones reales PF	Operaciones normalizadas PF
ADD, SUB, COMPARE, MULT	1
DIVIDE, SQRT	4
EXP, SIN, ...	8

# RENDIMIENTO DE LOS COMPUTADORES

## Otras medidas de rendimiento MFLOPS

Ejemplo:

- Programa Spice ejecutándose en una DECStation 3100 en 94 segundos.
- El nº de instrucciones en P.F. se muestra en la figura

ADDD	25 999 440
SUBD	18 266 439
MULD	33 880 810
DIVD	15 682 333
COMPARED	9 745 930
NEGD	2 617 846
ABSD	2 195 930
CONVERTD	1 581 450
TOTAL	109 970 178

- La estimación de MFLOPS para un programa no puede generalizarse para todo programa por lo que no son una medida útil para todos los programas.

$$\text{MFLOPS normalizados} = \frac{\text{nº instrucciones P.F. normalizadas}}{\text{ejecución} * 10^6} \approx \frac{157M}{94 * 10^6} \approx 1,7$$

nº instrucciones en P.F. = 110M  
MFLOPS reales = 121M

# RENDIMIENTO DE LOS COMPUTADORES

## Elección de programas para evaluar el Rendimiento

---

- ❑ El usuario de un computador que ejecute día tras día un conjunto de programas es el ideal para evaluar.
- ❑ Hay diferentes niveles de programas usados para evaluar el rendimiento.
- ❑ Programas reales. Estos tienen entradas, salidas y opciones que actúan mientras se ejecute el programa:
  - Compiladores C.
  - Software de tratamiento de textos.
  - Herramientas CAD (Spice).
- ❑ Núcleos (Kernels).
  - Se extrae piezas claves de programas para evaluar el rendimiento.
    - ❑ "Livermore Loops" y "Linpack".
  - Estos núcleos sólo se utilizan para evaluar rendimiento.
- ❑ Benchmarks reducidos. Programas reducidos, entre 10 y 100 líneas, con resultados conocidos antes de ejecutar.
  - Criba de Eratóstenes.
  - Puzzle.
  - Clasificación rápida (Quick sort).

# RENDIMIENTO DE LOS COMPUTADORES

## Elección de programas para evaluar el Rendimiento

---

- ❑ Benchmarks Sintéticos.
  - No son parte de programas reales y no los ejecutan los usuarios porque no dan nada.
  - Intentar determinar la frecuencia media de operaciones y operandos de un gran conjunto de programas.
    - ❑ Whetstone.
    - ❑ Dhrystone.
  - Peligro de dirigir la optimización de los computadores a dar buenos valores de Benchmarks sintéticos.
- ❑ La utilización de “Benchmarks” y “Núcleos” puede ser atractiva cuando se empieza con un diseño.
  - Son pequeños y fácilmente simulables.
  - Se pueden hacer a mano.
  - Pueden ejecutarse aunque no exista todavía compilador.
  - Son fáciles de estandarizar.
- ❑ No puede extenderse para evaluar el rendimiento de sistemas computadores en funcionamiento.
  - Standarización de Sistemas Operativos: UNIX, DOS.
  - Existencia de software de distribución libre.
  - Poder ejecutar todo programa real en cualquier máquina.

# RENDIMIENTO DE LOS COMPUTADORES

## Información sobre los resultados del rendimiento

---

- Reproducibilidad:
  - Todos los datos necesarios para que todo usuario pueda repetir las medidas.
- Decir que el programa Spice emplea en ejecutarse 94 seg. en una DECStation 3100 no es suficiente:
  - ¿Hay entradas?
  - ¿Qué compilador se ha usado?
  - ¿Cuanta memoria principal?
  - ¿Cuántos discos y tipo usados?



# RENDIMIENTO DE LOS COMPUTADORES

## Comparación y resumen de rendimientos

- Cuando se ejecutan programas se puede llegar a cierta confusión en comparación de un solo programa.

	Computador A	Computador B	Computador C
Programa 1 (s)	1	10	20
Programa 2 (s)	1000	100	20
Tiempo total (s)	1001	110	40

- Observando la tabla se puede llegar a las siguientes conclusiones
  - A es 900 por 100 más rápido que B para el programa 1
  - B es 900 por 100 más rápido que A para el programa 1
  - A es 1900 por 100 más rápido que C para el programa 1
  - C es 4900 por 100 más rápido que A para el programa 2
  - B es 100 por 100 más rápido que C para el programa 1
  - C es 400 por 100 más rápido que B para el programa 2
- Cada una de estas afirmaciones puede ser útil de forma individual, pero colectivamente, puede inducir a confusión