

EL253 - Sistemas Digitales

Semestre 2022-1

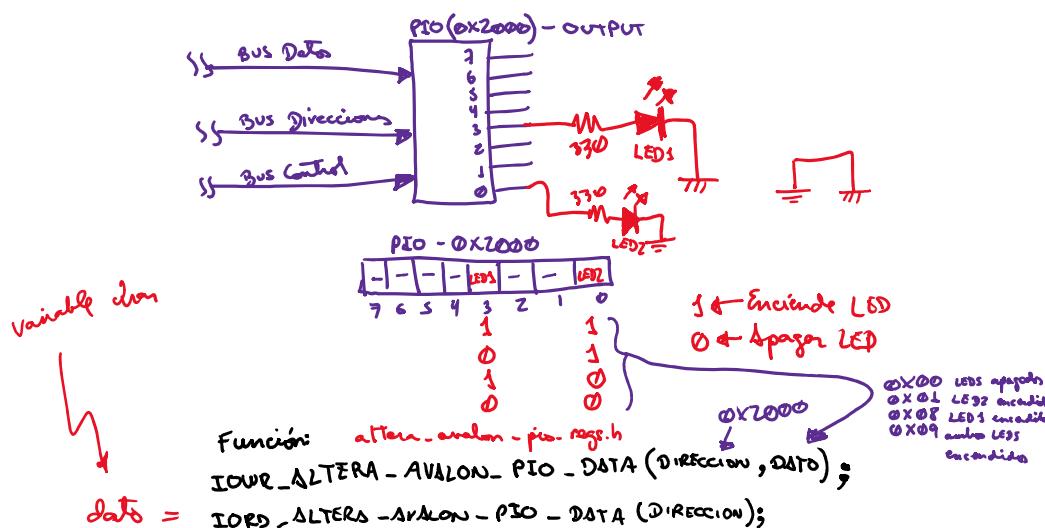
Profesor Kalun José Lau Gan

Sesión de Teoría Semanas 10 y 11

1

Preguntas previas

- ¿Cómo es la estructura de la función para el manejo de los puertos?



2

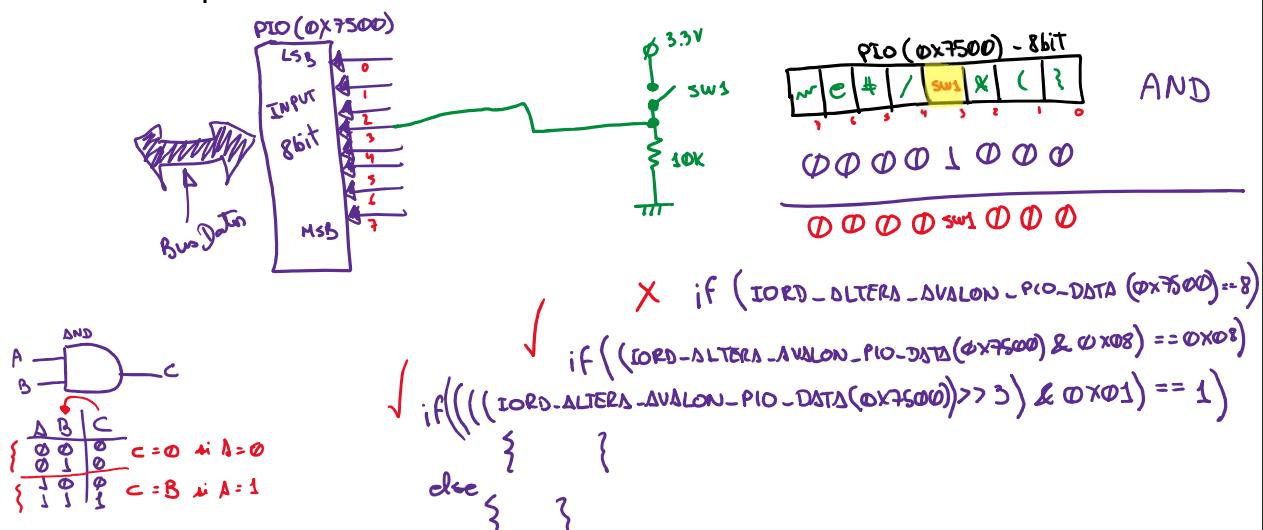
Preguntas previas

- ¿De qué tamaño es el dato en el comando de lectura?
 - Dependiendo de lo declarado en el Qsys pudiendo ser 8, 16 ó 32 bits
 - Tener en cuenta lo detallado en:
<https://www.intel.com/content/dam/support/jp/ja/programmable/support-resources/bulk-container/pdfs/literature/hb/nios2/n2cpu-nii51007.pdf>
- ¿Qué tipo de variable utilizaría para alojar lo leído de un Puerto?
 - Dependiendo del ancho del puerto:
 - 8bits : unsigned char
 - 16bits: unsigned int

3

Preguntas previas

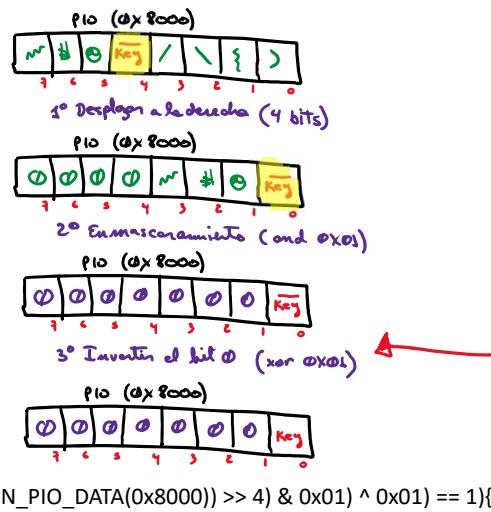
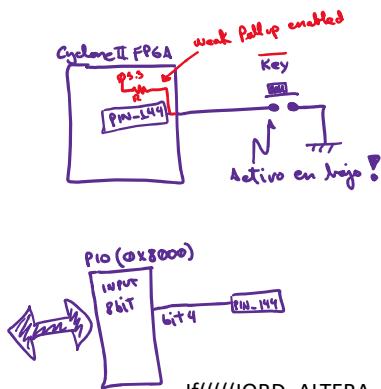
- ¿Qué es el enmascaramiento de un dato y por qué se usa en la lectura de un puerto?



4

Preguntas previas

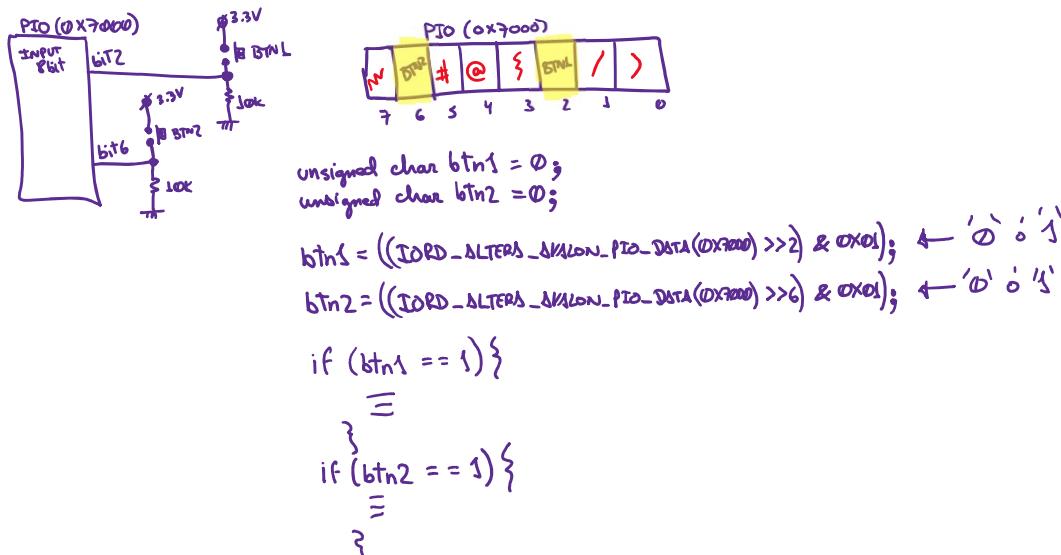
- Sobre el pulsador integrado en la tarjeta. ¿Cómo lo puedo leer en el NIOS II?
 - Pulsador en PIN_144 se encuentra en bit4 de PIO 0x8000
 - Se debe de negar para que pueda evaluarse como activo en alto



5

Preguntas previas:

- Dos pulsadores como entradas en un PIO



6

Uso de “for” en el Eclipse

```
int main(){
    unsigned char x = 0;
    for(x=0;x<100;x++){
        IOWR_ALTERA_AVALON_PIO_DATA(0x9000, x);
        usleep(100000);
    }
}

unsigned tabla[]={0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80};

int main(){
    unsigned char x = 0;
    for(x=0;x<8;x++){
        IOWR_ALTERA_AVALON_PIO_DATA(0x9000, tabla[x]);
        usleep(100000);
    }
}
```

7

Agenda:

- Neumann vs Harvard
- Estructura de un procesador
 - Unidad de proceso
 - ALU
 - Registros internos
 - Unidad de control
 - Decod Instr.
- Código de máquina
 - Soporte de lenguaje Assembler
- Sistema operativo

8

Neumann vs. Harvard

- Tipos de arquitecturas presentes en procesadores
- Harvard viene a ser una evolución de Neumann
 - Neumann -> Lo antiguo
 - Harvard -> Lo nuevo
- Neumann -> Tu PC
- Harvard -> Tu disp. Móvil

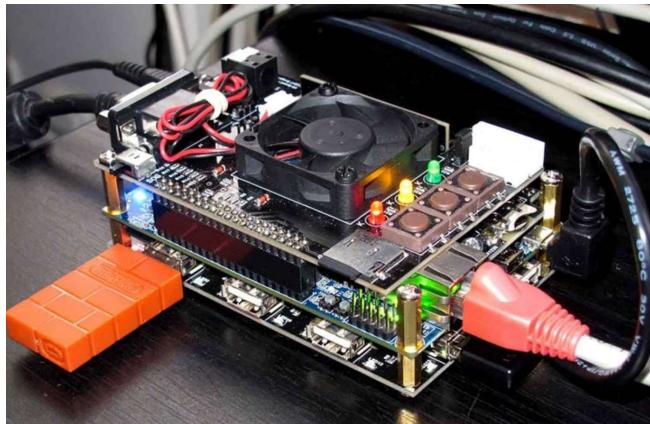
9

¿Qué es “emulación”?

- Imitar una plataforma de hardware para que pueda ejecutarse una aplicación desarrollada en esa plataforma de hardware (distinta)
- También denominada “máquina virtual”.
- Por ejemplo: Emular el Play Station en el computador usando VMWare para jugar Final Fantasy VII
- “cross-platform”
- Siempre tendrá menos desempeño lo “emulado” frente a un hardware original.

10

Caso Mister FPGA



- Emulación por hardware de las plataformas de los juegos de arcada.



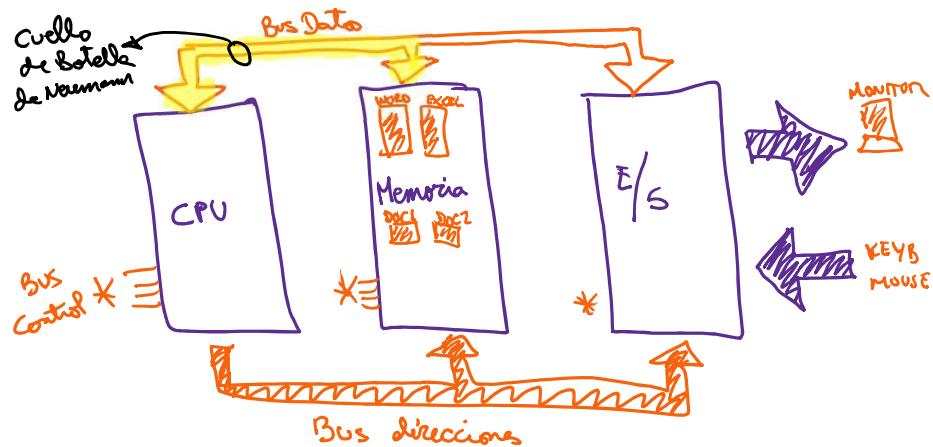
11

Otros casos prácticos de emulación

- Notarías.
- Servicios registrales públicos.
- Servicios de identificación pública.
- Servicios de administración tributaria.

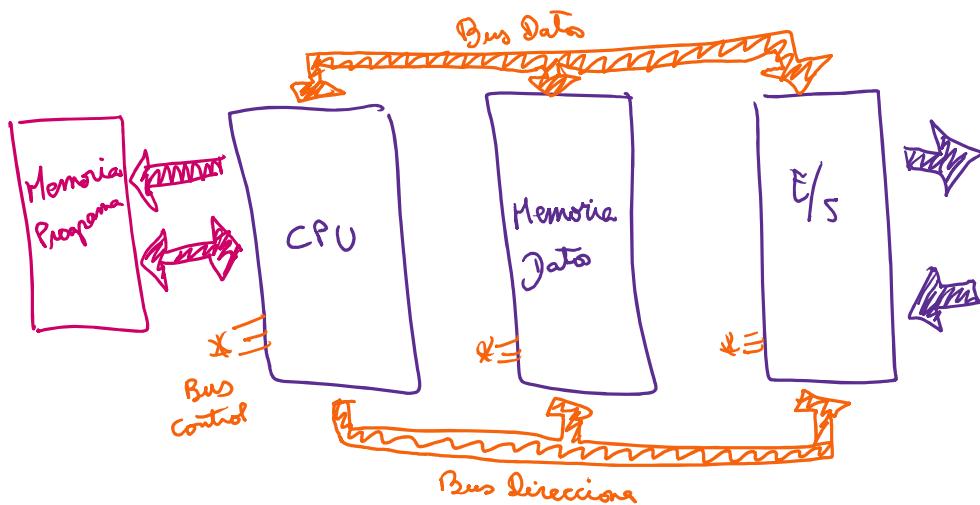
12

Arquitectura Von Neumann



13

Arquitectura Harvard



14

Persistencia de Neumann hasta hoy

- ¿Por qué se sigue usando Neumann en las PCs sabiendo que Harvard es mejor?
 - Precio y fácil de fabricar?
 - Uso mas eficiente de la memoria?
 - Compatibilidad?
 - ¿Qué es compatibilidad?
 - ¿A qué se debe la compatibilidad? (Causas y efectos)
 - ¿Sigue habiendo compatibilidad?
 - Motivos económicos por parte del fabricante

15

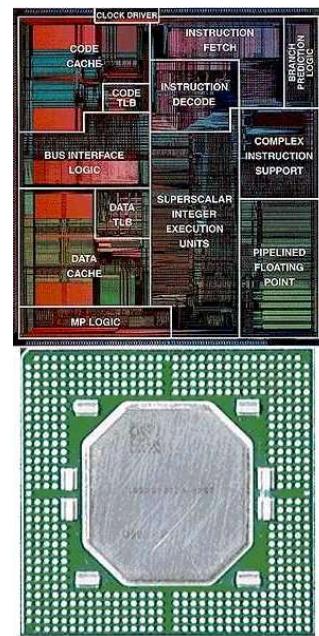
Compatibilidad en computadores

- Fundamental para que los usuarios intercambien información
- Compatibilidad tanto en software como en hardware:
 - Ej. Las computadoras tanto PC como Apple emplean USB
 - Ej. Tanto Windows como MacOS tienen Microsoft Office
 - Todos los computadores y dispositivos móviles emplean navegadores Web que soportan HTML
 - Hay compatibilidad también en el sistema de archivos.
- Gracias a la emulación la compatibilidad se extiende aún mas.

16

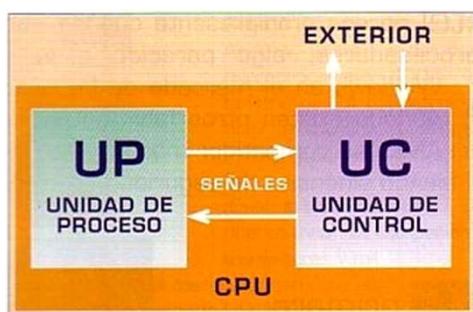
Secciones internas del μP

- Como podemos apreciar en la imagen, el interior del microprocesador está conformado por muchos elementos.
- Es como una pequeña “ciudad”.
- Para confeccionar un uP se emplea tecnología microelectrónica basado en fotolitografía.



17

El interior del CPU



- Lo conforman dos componentes lógicos con funciones específicas.
- Al igual que en un automóvil, hay un motor y un conductor que lo controla.
 - El motor es la UP
 - El conductor es la UC

18

La Unidad de Proceso

- Se encarga de interpretar y calcular las instrucciones y los datos necesarios para obtener el resultado deseado.
- Dentro de la unidad de proceso podemos encontrar:
 - **La Unidad Aritmética Lógica (ALU)**, lleva a cabo las funciones de procesamiento de datos.
 - **La Unidad de Cálculo**, se encarga de operaciones matemáticas específicas (sumar, multiplicar, dividir por números enteros, etc), aliviando y facilitando el trabajo de la ALU.
 - **Los Registros**, pequeñas memorias que almacenan datos a procesar.

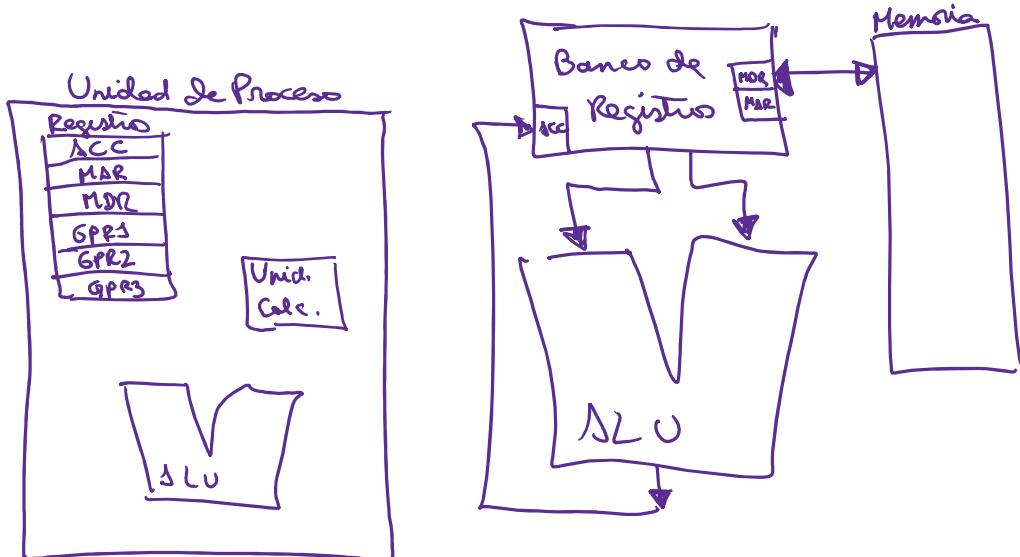
19

Los Registros

- **ACC – El Acumulador**, almacena resultados intermedios y finales de un cálculo. Es el área principal de trabajo de la ALU.
- **MAR – Registro de la dirección de memoria**, almacena la dirección del dato a acceder.
- **MDR – Registro del dato de memoria**, almacena el dato a ser almacenada/leída de memoria.
- **Registro de Propósito General**, puede ser usado para almacenar alguna información temporal durante la ejecución de una instrucción.

20

Unidad de Proceso



21

La Unidad de Control

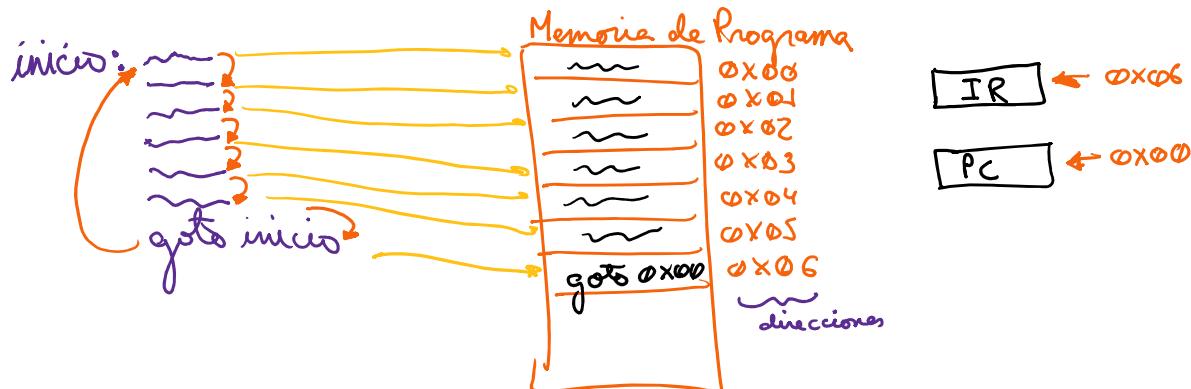


- De ella depende la correcta gestión y funcionamiento de la Unidad de Proceso, la comunicación con el resto del sistema, la obtención de los datos y las instrucciones que se van a procesar en el justo momento en que son necesarios, así como el envío del resultado de este procesamiento.
- Para esto, la UC envía y recibe señales de control de los periféricos.
- Dentro de la unidad de control podemos encontrar:
 - **El Contador de Programa (PC – Program Counter)**, contiene dirección de la siguiente instrucción a ejecutar.
 - **El Decodificador de Instrucciones (Instruction Decoder)**, es el dispositivo que interpreta la instrucción a ser ejecutada.
 - **El Registro de Instrucción (IR – Instruction Register)**, contiene la instrucción que se está ejecutando.
 - **El Registro de Estado (STATUS Register)**, se encuentran los “flags” del estado de la ALU al procesar una operación.
 - **AD – Decodificador de Direcciones**, interpreta la dirección de la MAR y selecciona la celda apropiada de la memoria principal.

22

¿Qué es un programa?

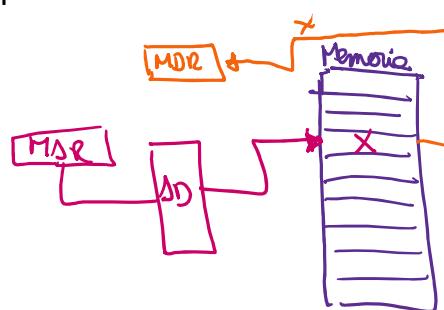
- Es un conjunto de instrucciones en el cual el procesador ejecuta una a una de manera secuencial



23

¿Cómo se leen datos desde la memoria?

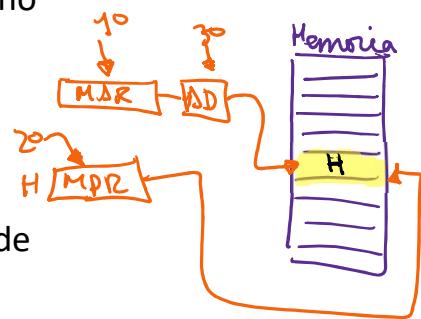
- La Unidad de Control envía una señal de lectura a la memoria.
- El MAR obtiene la dirección del dato requerido.
- El AD selecciona la celda apropiada
- El dato requerido es enviado al MDR, para luego ser procesado.



24

¿Cómo se escriben datos en la memoria?

- La Unidad de Control envía una señal de escritura a la memoria.
 - EL MAR obtiene la dirección de la celda destino en la memoria.
 - El MDR obtiene el dato a ser escrito
 - El AD selecciona la celda requerida en la memoria
 - El dato en MDR es transferido hacia la celda de memoria seleccionada.



25

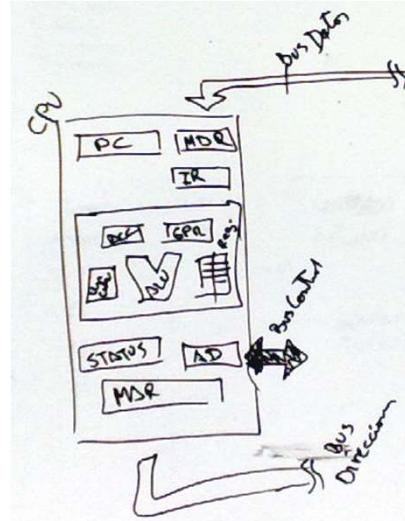
¿Cómo se ejecuta una instrucción?

- En general, un ciclo de Instrucción posee tres etapas:
 - Fetch (Captura)
 - La dirección de la instrucción es enviada a la memoria.
 - La instrucción es recibida desde la memoria y almacenada en IR.
 - El PC es incrementado en 1.
 - Decode (Decodificación)
 - La instrucción en IP es interpretado por el Decodificador de Instrucciones.
 - Execute (Ejecución)
 - La instrucción es ejecutada.

26

¿Cómo se ejecuta una instrucción?

- Si has aprendido un lenguaje de alto nivel (C++, Java, etc), podrías pensar que la ejecución de las instrucciones es una cosa sencilla, esto no es cierto!
- Actualmente, una sentencia en un lenguaje de alto nivel puede conllevar a realizar cientos de operaciones en la CPU.



Ej:
printf ("Hola mundo");

27

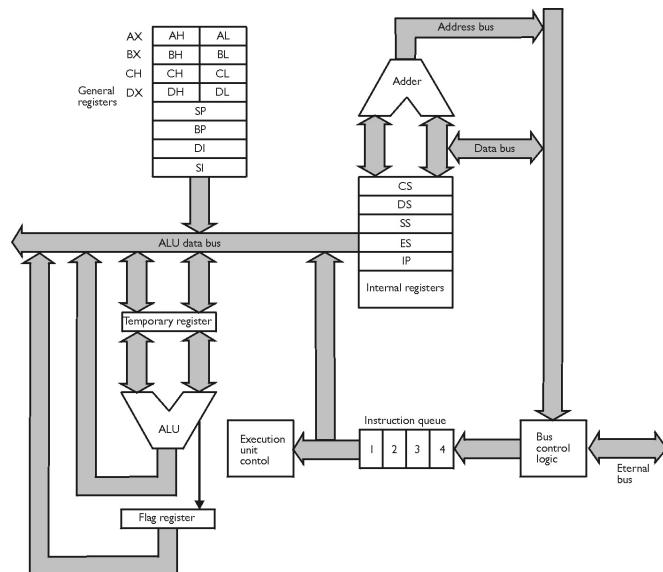
¿Cómo se ejecuta una instrucción?

- Tomamos “ $a = b + c$ ”, una simple operación de adición como ejemplo:
 - La CPU necesita hacer lo siguiente:
 - Carga el contenido de la celda de la memoria asociado con B hacia el Acumulador
 - Adiciona el contenido de la celda de la memoria asociado con C con el Acumulador
 - Almacena el contenido del Acumulador a la celda de memoria asociada con A
 - Se podría pensar (nuevamente) que estas tres operaciones son algo simples, actualmente el sólo cargar el contenido de B en el Acumulador involucra:
 - Transferir el contenido del PC hacia MAR
 - Decodificar la dirección obtenida en MAR
 - Transferir la instrucción hacia MDR
 - La instrucción es conducida a IR por el MDR
 - Se incrementa en uno el PC
 - Se decodifica la instrucción en IR por el Decodificador de Instrucciones
 - La parte del operando de la instrucción es transferido hacia MAR
 - Se decodifica la dirección obtenida en MAR
 - Se transfiere el contenido hacia MDR
 - El MDR conduce el contenido hacia el Acumulador
- Ahora puedes darte cuenta lo complicado que es!

} Código de
maquina
(Assembler)

28

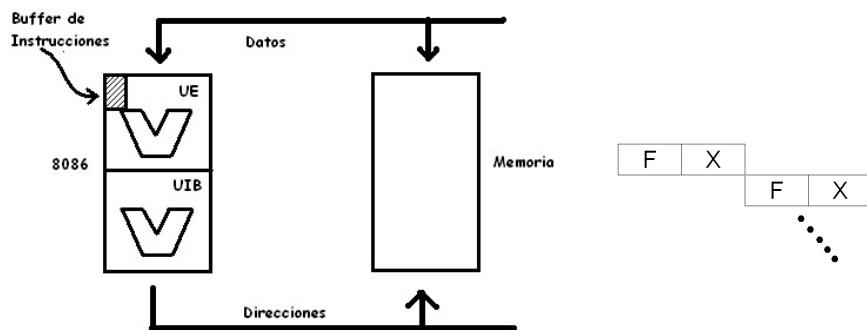
Arquitectura Intel 8086



29

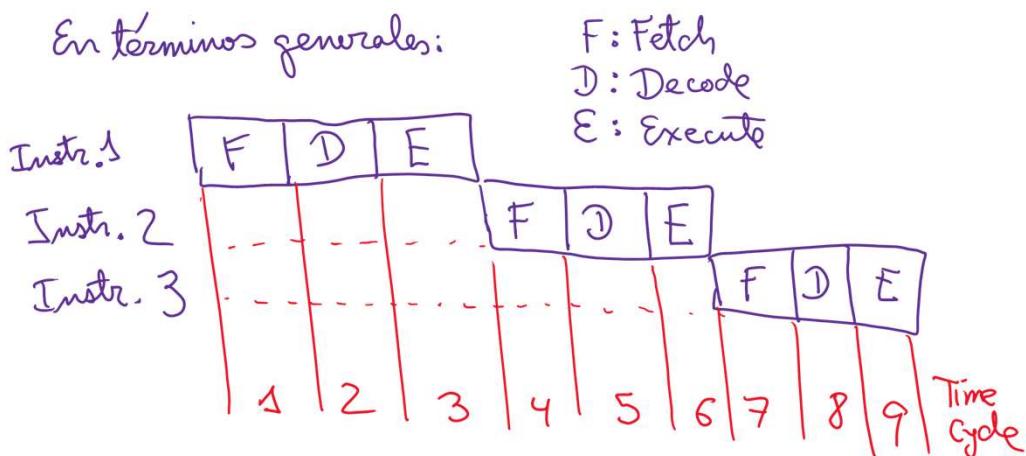
Intel 8086

- Procesador Escalar Segmentado de dos niveles:
 - UIB: Accede a memoria por instrucción / dato
 - UE: Ejecuta las instrucciones



30

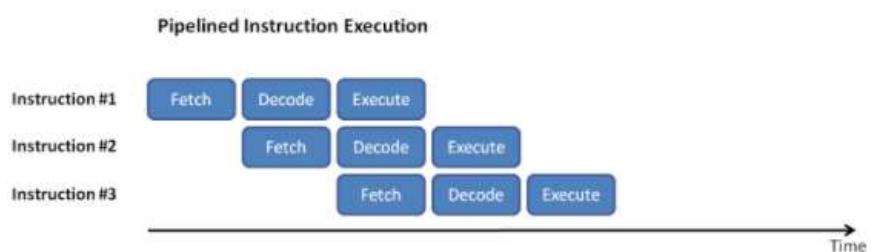
Fases del procesamiento de instrucciones



31

Instruction pipeline

- Técnica empleada en el diseño de microprocesadores para aumentar la eficiencia en el procesado de las instrucciones.
- En un ciclo de reloj puede estarse ejecutando varias instrucciones en diferentes etapas de su procesamiento.



32

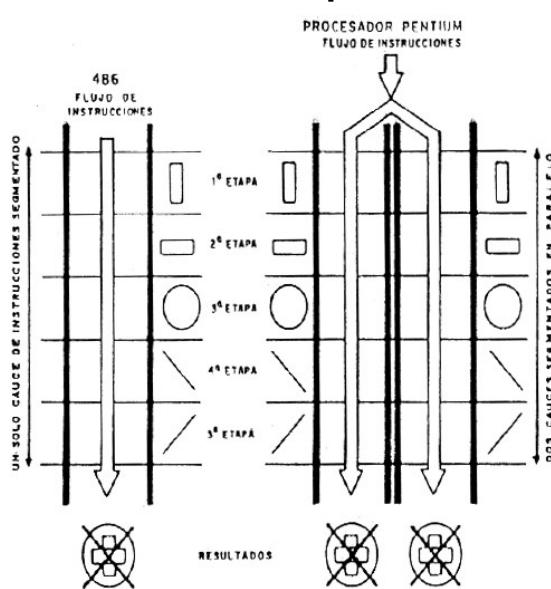
¿Procesador Super-Escalares?

- Lo esencial del enfoque superescalar es su habilidad para ejecutar instrucciones de manera independiente en diferentes cauces. El concepto puede llevarse más lejos, permitiendo que las instrucciones se ejecuten en un orden diferente al del programa.
- Hay múltiples unidades funcionales, cada una de las cuales está implementada como un cauce segmentado, que admiten la ejecución en paralelo de varias instrucciones

Fuente: <http://informatica.utm.cl/~hlorre/Arq-Comp/Introduccion/Paralelo-Segment.pdf>

33

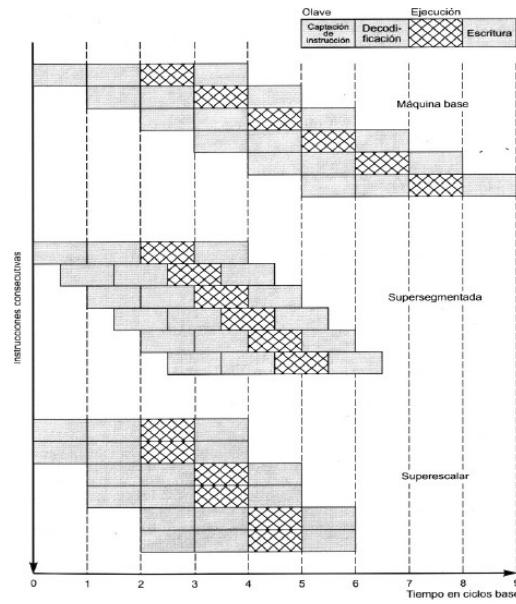
¿Procesador Super-Escalares?



34

Super-Escalar vs Super-Segmentado

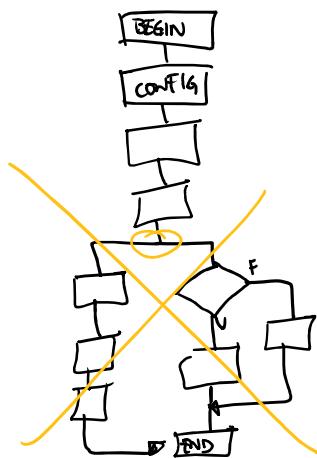
- La supersegmentación aprovecha el hecho de que muchas etapas del cauce realizan tareas que requieren menos de la mitad de un ciclo de reloj
- La implementación superescalar es capaz de ejecutar en paralelo dos instrucciones en cada etapa



35

¿Romper el cauce en un diagrama de flujo?

- Romper la secuencialidad del programa



36

Limitaciones

- La aproximación superescalar depende de la habilidad para ejecutar múltiples instrucciones en paralelo. La expresión **parallelismo a nivel de instrucciones** se refiere al grado en que, en promedio, las instrucciones de un programa se puede ejecutar en paralelo. Las limitaciones fundamentales del parallelismo son:
 - Dependencia de datos verdadera.
 - Dependencia relativa al procedimiento.
 - Conflictos en los recursos.
 - Dependencia de salida.
 - Antidependencia.

37

Dependencia de datos verdadera

Consideremos la siguiente secuencia:

```

add    r1, r2 ; cargar el registro r1 con el contenido de r2 más
                el contenido de r1
move   r3, r1 ; cargar el registro r3 con el contenido de r1
  
```

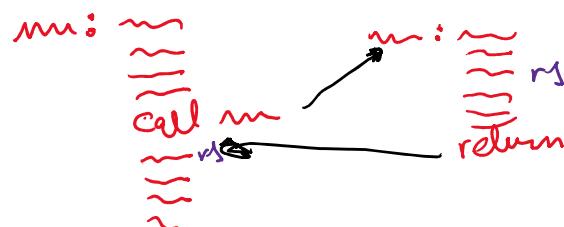
La segunda instrucción se puede captar y decodificar, pero no se puede ejecutar hasta que finalice la ejecución de la primera instrucción. Ejemplo de dependencia de datos verdadera.

$\text{add } r1, r3$
 $\text{move } r2, r4$

38

Dependencias relativas al procedimiento

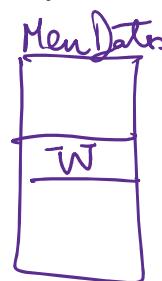
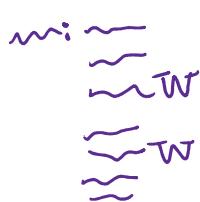
- Las instrucciones que siguen a una bifurcación tienen una dependencia relativa al procedimiento en esa bifurcación y no pueden ejecutarse hasta que ésta lo haga



39

Conflicto en los recursos

- Un conflicto en un recurso es una pugna de dos o más instrucciones por el mismo recurso al mismo tiempo.
- Desde el punto de vista del cauce, un conflicto en los recursos presenta el mismo comportamiento que una dependencia de datos.



40

Paralelismo a nivel de instrucciones y paralelismo de la máquina

- El paralelismo a nivel de instrucciones es cuando las instrucciones de una secuencia son independientes y, por tanto, pueden ejecutarse en paralelo solapándose.
- El paralelismo de la máquina es una medida de la capacidad del procesador para sacar partido al paralelismo a nivel de instrucciones. El paralelismo de la máquina depende del número de instrucciones que pueden captarse y ejecutarse al mismo tiempo (número de cauces paralelos).

41

Políticas de emisión de instrucciones

- Esencialmente el microprocesador intenta localizar instrucciones más allá del punto de ejecución en curso, que puedan introducirse en el cauce y ejecutarse. Formas:
 - El orden en que se captan las instrucciones.
 - El orden en que se ejecutan las instrucciones.
 - El orden en que las instrucciones actualizan los contenidos de los registros y de las posiciones de memoria.
- La única restricción que tiene el procesador es que el resultado debe ser correcto.
- En términos generales, podemos agrupar las políticas de emisión de instrucciones de los procesadores superescalares en las siguientes categorías:
 - Emisión en orden y finalización en orden.
 - Emisión en orden y finalización desordenada.
 - Emisión desordenada y finalización desordenada.

42

Emisión en orden y finalización en orden

- La política de emisión de instrucciones más sencilla es emitir instrucciones en el orden exacto en que lo haría una ejecución secuencial, y escribir los resultados en ese mismo orden

43

Emisión en orden y finalización desordenada

- Con la finalización desordenada puede haber cualquier número de instrucciones en la etapa de ejecución en un momento dado, hasta alcanzar el máximo grado de paralelismo de la máquina.
- La finalización desordenada necesita una lógica de emisión de instrucciones más compleja que la finalización en orden. Además, es mas difícil ocuparse de las interrupciones y excepciones.

44

Emisión desordenada y finalización desordenada

- Para permitir la emisión desordenada, es necesario desacoplar las etapas del cauce de decodificación y ejecución. Esto se hace mediante un buffer llamado **ventana de instrucciones**.
- Con esta organización, cuando un procesador termina de decodificar una instrucción, coloca ésta en la ventana de instrucciones. Mientras el buffer no se llene, el procesador puede continuar captando y decodificando nuevas instrucciones.
- Cualquier instrucción puede emitirse, siempre que (1) necesite la unidad funcional particular que está disponible y (2) ningún conflicto ni dependencia la bloquee.
- El resultado de esta organización es que el procesador tiene capacidad de anticipación, lo que le permite identificar instrucciones independientes que pueden introducirse en la etapa de ejecución. Las instrucciones se emiten desde la ventana de instrucciones, sin que se tenga muy en cuenta su orden original en el programa.
- Recordemos que la única restricción es que el programa funcione correctamente.

45

Decodificación	Ejecución	Escritura	Ciclo
I1 I2	I1 I2		1
I3 I4	I1		2
I3 I4		I3	3
I4		I4	4
I5 I6		I5	5
I6		I6	6
			7
			8

(a) Emisión en orden y finalización en orden

Decodificación	Ejecución	Escritura	Ciclo
I1 I1	I1 I2		1
I3 I4	I1		2
I4		I3	3
I5 I6		I4	4
I6		I5	5
		I6	6
			7

(b) Emisión en orden y finalización desordenada

Decodificación	Ventana	Ejecución	Escritura	Ciclo
I1 I2	I1, I2	I1 I2		1
I3 I4	I3, I4	I1		2
I5 I6	I4, I5, I6	I6 I4		3
	I5	I5		4
			I2	5
			I1 I3	6
			I4 I6	7
			I5	8

(c) Emisión desordenada y finalización desordenada

Ejemplo

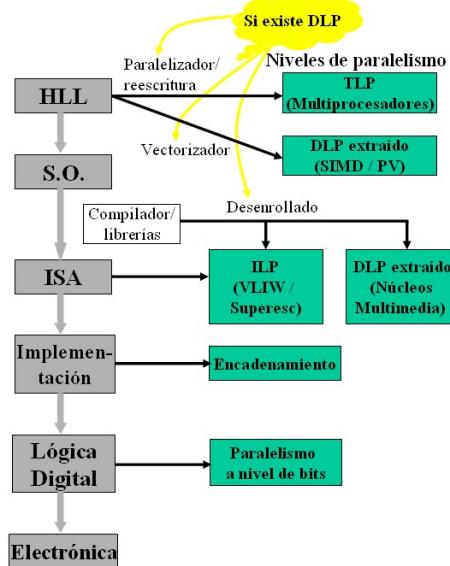
- I1 necesita dos ciclos para ejecutarse.
- I3 e I4 compiten por la misma unidad funcional.
- I5 depende de un valor producido por I4.
- I5 e I6 compiten por una unidad funcional.

46

Sistemas Multi-procesadores

En los sistemas actuales la CPU:

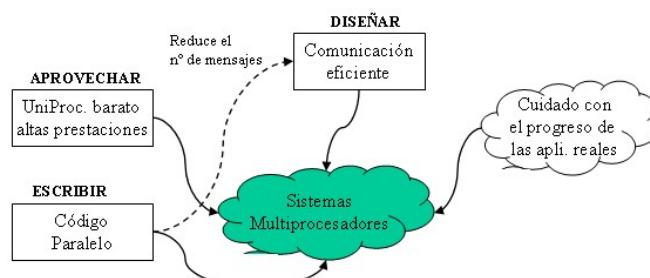
- es un todo monolítico
- está separada de la memoria principal y de la E/S
- Alcanza grandes velocidades



47

Sistemas Multi-procesadores

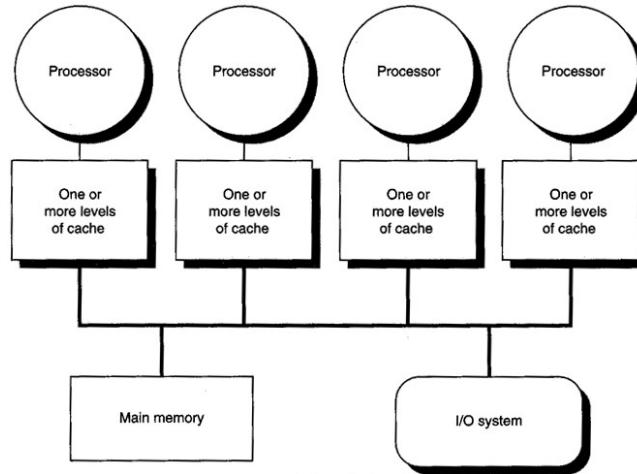
- Vamos a romper con esta concepción para diseñar los multiprocesadores



48

Sistemas Multi-procesadores

- Lo que vamos a construir se parece a:



49

Sistemas Multi-procesadores

- La red de interconexión juega un papel importante
 - Comunicación entre procesadores
 - Comunicación entre procesadores y memoria
- La memoria está separada de los procesadores, pero no totalmente ⇒ Cada procesador tiene su caché local. La forma de organizar la jerarquía de memoria es vital

50

Video:

<https://www.youtube.com/watch?v=IOU6ZMeVrB4>

- ¿En qué se usa este procesador Xeon Phi? Algoritmos, redes neuronales, AI, simulación avanzada
- Intel Xeon Phi
- No es para usar en una desktop con Windows
- 64 núcleos, 256 hilos

51

Argumentos a favor de los Multiprocesadores

- En 15 ó 20 años, el rendimiento de los Uniprocesadores llegará a su fin, debido a tres razones:
 - Incrementar el rendimiento conectando uniprocesadores en paralelo.
 - Menos costoso que diseñar un nuevo procesador.
 - Más flexible para todo tipo de soft que un nuevo diseño.
 - El ritmo de innovación en la endoarquitectura es difícil de mantener indefinidamente
 - Actualmente el incremento en transistores es superior al de rendimiento ($n^{\text{trans}} \times 4$; rend. $\times 3$ cada 3 años).
 - Algunos estudios indican el inicio de la saturación en prestaciones de los uniprocesadores.
 - Parece que los MPr's son la única forma de aumentar considerablemente el rendimiento a un precio razonable.
 - Parece que los MPr's son la única forma de aumentar considerablemente el rendimiento a un precio razonable.

52

Argumentos en contra de los Multiprocesadores

- Cada 2 ó 3 años, alguien anuncia el fin de los UniPr
 - A los 2 ó 3 años se disparan las prestaciones por una nueva idea arquitectónica. Esto es debido a que hay mucho mercado \Rightarrow mucha inversión \Rightarrow mucho avance
- Lentitud en las comunicaciones
 - Latencia a memoria local es del orden de pocos ns
 - En MPr, si un Pr necesita un dato remoto \Rightarrow alta latencia, del orden de microsegundos (Ejemplo)
- Limitación del paralelismo de las aplicaciones
 - La ley de Amdahl nos indica el efecto del paralelismo de las aplicaciones en la aceleración

53

¿Threading (Hilo de ejecución)?

- Un hilo de ejecución, en sistemas operativos, es similar a un proceso en que ambos hilos representan una secuencia simple de instrucciones ejecutada en paralelo con otras secuencias.
- Los hilos permiten dividir un programa en dos o más tareas que corren simultáneamente, por medio de la multiprogramación.
- Una tarea es básicamente cualquier programa realizado en una computadora, pero una tarea con hilos de ejecución de multithreading podrá realizar dos o más programas a la misma vez. ¿Por qué es esto tan importante? Al separar tareas entre diferentes hilos de ejecución, la computadora puede hacer más cosas a la misma vez.
- Esto aumenta la productividad y permite conseguir que más trabajos sea terminados en menos tiempo.

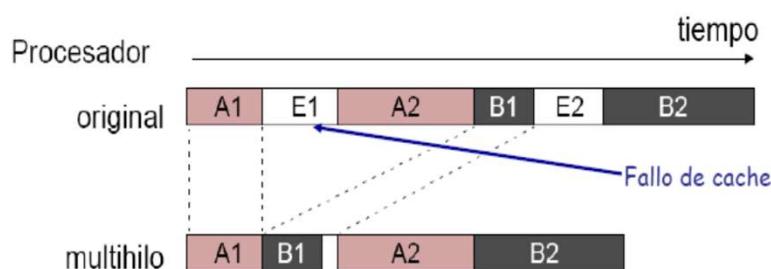
54

¿Qué es Multithreading?

- Manera de programar software que permite compartir el mismo estado de memoria entre varios hilos de ejecución
- Requiere crear diferentes secciones de códigos que corren a diferentes tiempos y controlar como el procesador procesa datos e instrucciones de la aplicación.

55

■ Multithreading

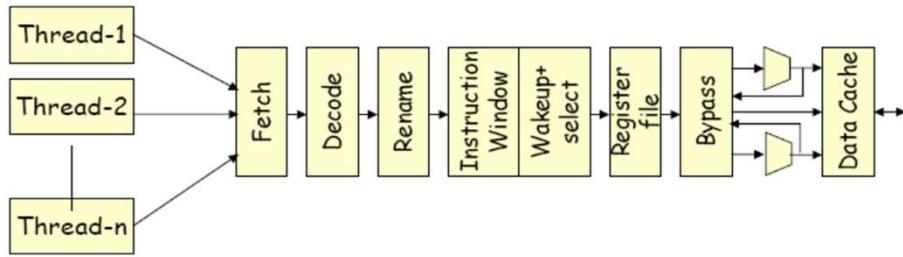


- Incrementar el trabajo procesado por unidad de tiempo
- Si los hilos son del mismo trabajo se reduce el tiempo de ejecución

La técnica multihilo no es ideal y se producen pérdidas de rendimiento.
Por ejemplo, un programa puede ver incrementado su tiempo de ejecución aunque el número de programas ejecutados por unidad de tiempo sea mayor cuando se utiliza multihilo.

56

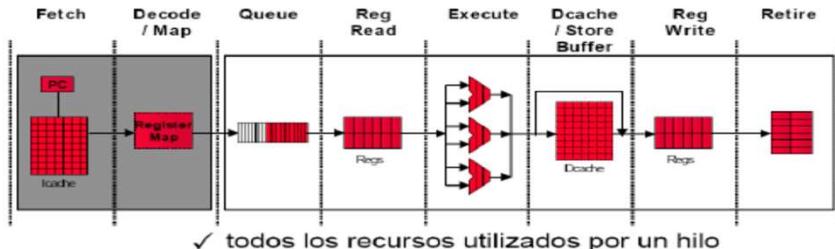
Multithreading



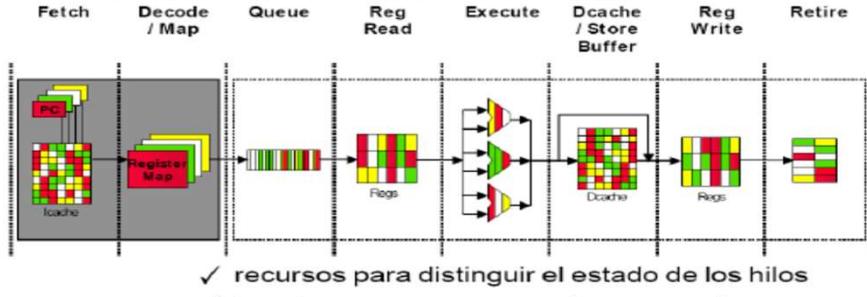
57

□ Simultaneous multithreading

- Un solo hilo: un flujo de instrucciones

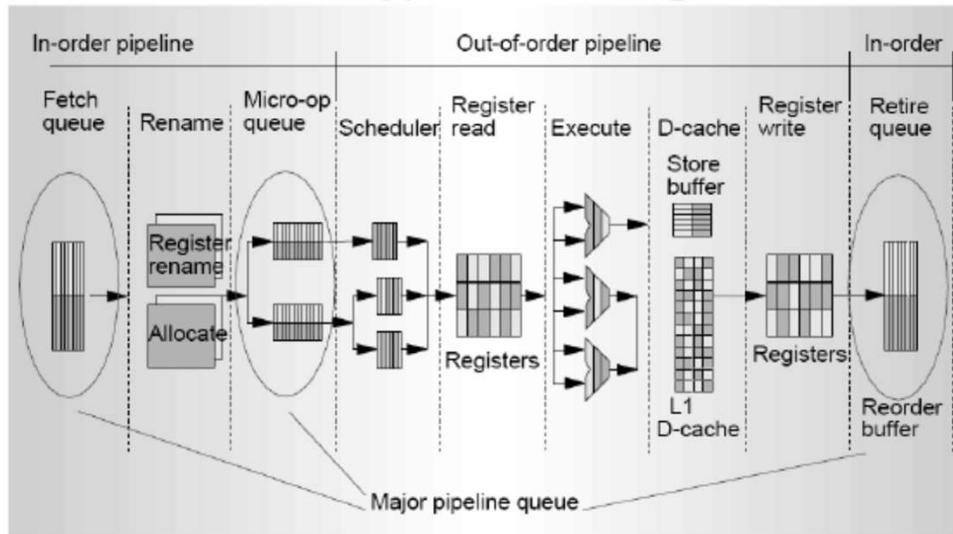


- Multihilo: varios flujos de instrucciones

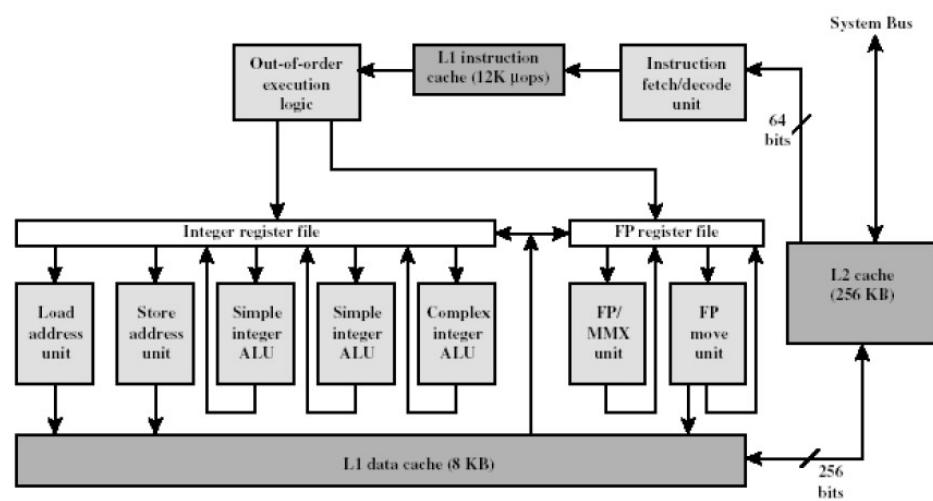


58

□ Pentium 4 HyperThreading

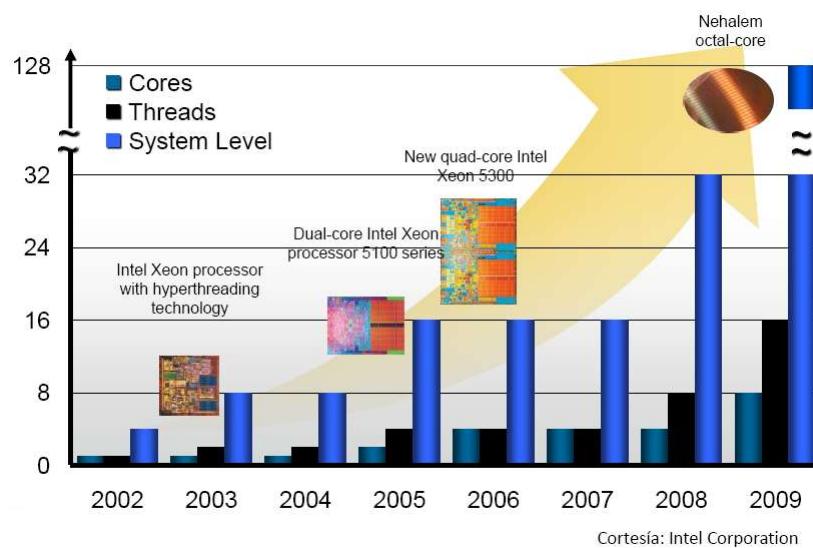


59



60

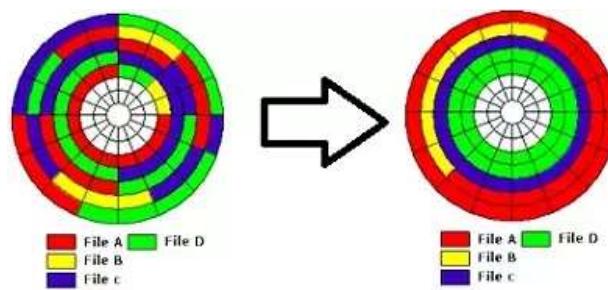
Avances de la Tecnología Multinúcleo



Cortesía: Intel Corporation

61

Temas de discusión:
fragmentación en unidades
de disco



62

Preguntas adicionales

- ¿Cuál es la ventaja de tener una longitud fija en la memoria de instrucciones?
- ¿De qué se trata el procesador MIPS?
- ¿Qué procesador tuvo la primera consola de juegos ATARI?
- Dirección del puerto serial en COM2
- ¿Lógica invertida en el puerto serial?
- De qué se trata el PIC (programmable interrupt controller)

63

Preguntas finales!

- ¿Cómo se construye una Unidad de Control?
- ¿Qué hay en el registro STATUS?
- ¿Cuántas líneas de código tuvo el DOS?
- ¿De qué está constituido la ALU?
- ¿Cómo ejecuta las instrucciones el microprocesador si es que posee dos hilos?
- ¿Qué es el BER?

64

Preguntas finales!

- ¿Porqué el “core voltage” es menor que el “I/O voltage” en un microprocesador?
- ¿Cuál es el “op-code” de la instrucción DEC del procesador x86?
- ¿Qué es lo que controla el “north-bridge”?
- ¿Qué es lo que controla el “south-bridge”?
- ¿Qué es el FSB?
- ¿En qué microprocesador de Intel se implementa el “pipeline”?

65

Preguntas adicionales:

- Detallar las bandas de frecuencia LTE por operador en el Perú
- ¿En qué se diferencia un operador móvil virtual (OMV) de uno regular?
- ¿Cuáles son las implicancias en el cliente al tener NAT3 en lugar de NAT2 en el servicio de Internet?

66

Preguntas adicionales:

- Explica el problema de “Neumann bottleneck”
- ¿Cuál es el rango de valores de resistencia recomendado para el sistema de puesta a tierra para equipos de TI?
- Mostrar una solución de respaldo energético dual (UPS+Grupo Electrógeno) para ser instalado en un datacenter que consume 6000W.

67

Preguntas adicionales:

- Se ha determinado con mediciones eléctricas que un datacenter tiene un consumo energético de 2500W como máximo. Seleccionar el valor mas recomendado de la llave térmica e interruptor diferencial para la parte de seguridad eléctrica sabiendo que el voltaje de la red eléctrica es 220V AC (Tip: emplear ecuación de potencia eléctrica y ley de Ohm).

68

Preguntas adicionales:

- Tecnología en memorias SD (HC, XC, UHS-1, A1, A2, V30)
- Especificaciones de un NAS
- Categorías de cable para redes Ethernet, estándar mínimo en la actualidad.
- UPS Online vs UPS Standby

69

Preguntas adicionales:

- ¿Persistencia visual?
- Relojes “QUARTZ”
- Compuesto térmico de transferencia de calor entre circuito integrado y disipador
- Cooler (ventilador del CPU), ¿Por qué tiene un conector de 3 hilos?

70

Preguntas adicionales:

- Tasa de transferencia del USB versión 1.1 y USB versión 2.0, USB 3.0
- Cantidad de dispositivos que se pueden conectar a un host USB
- Tasa de transferencia de e-SATA
- Diferencias entre generaciones de Bluetooth (1.2, 2.0, 3.0, 4.0, 4.1, 5)

71

Preguntas finales:

- ¿Computadoras en cluster?
- ¿Supercomputación?
- ¿Proyecto SETI @ home?
- ¿Cómo construyo una “malla” (computer grid) de computadores?
- Aplicaciones en donde se usan supercomputadoras

72

Fin de la sesión de teoría