



Estructura de Computadores

Tema 3: Procesador

- Introducción
- Operaciones elementales
- Estructura de un computador elemental y sus señales de control
- Cronogramas
- Diseño de la Unidad de Control
- Unidad de Control microprogramada
- Control de excepciones

1



Objetivos

- Visión dinámica del computador:
 - mostrar cómo se ejecutan las instrucciones
 - describir el órgano encargado de que esto se lleve a cabo (la unidad de control)
 - entender la ejecución de instrucciones mediante su representación en el tiempo por cronogramas
- Comprender el diseño de la unidad de control
 - Ser capaz de desglosar las instrucciones en operaciones básicas según la estructura del computador
 - Diseño de la unidad de control microprogramada
- Comprender y analizar aspectos de diseño que pueden mejorar el rendimiento a través de mejoras en la estructura del computador o de la UC

2



Bibliografía

- de Miguel, P. "*Fundamentos de los computadores*", Paraninfo, 2004. 9ª edición
- Patterson, D. A.; Hennessy, J. L. *Computer Organization and Design*. Morgan-Kaufmann. 2009. 4ª edición
- Stallings, W. "*Organización y arquitectura de computadores*", Prentice Hall, 2006, 7ª Edición

3



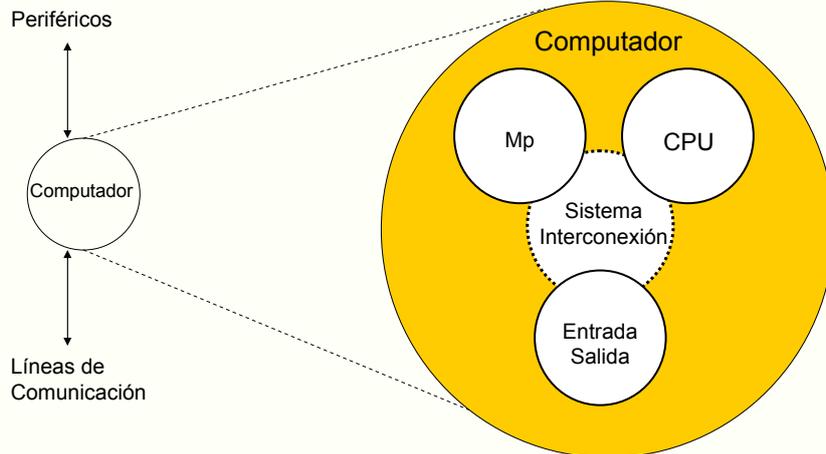
Introducción

- Estudiaremos:
 - **Unidad de control:**
 - Encargada de interpretar las instrucciones del programa y gobernar la ejecución de las mismas
 - **Camino de datos:**  
 - Parte de la CPU donde se transfieren los datos procedentes de la memoria o registros internos, para obtener los resultados
 - Debe soportar el conjunto de operaciones de las instrucciones del repertorio capaz de interpretar la unidad de control
- Organización de procesadores:
 - desarrollo tecnológico
 - necesidad de obtener altas prestaciones

4

Introducción

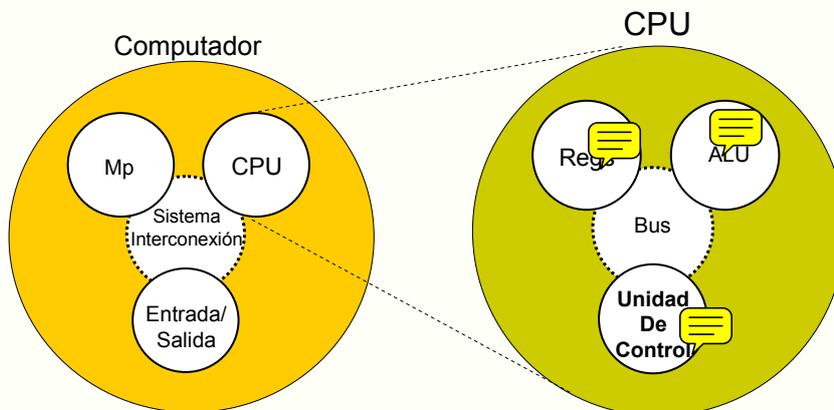
Visión global del computador y de la CPU



5

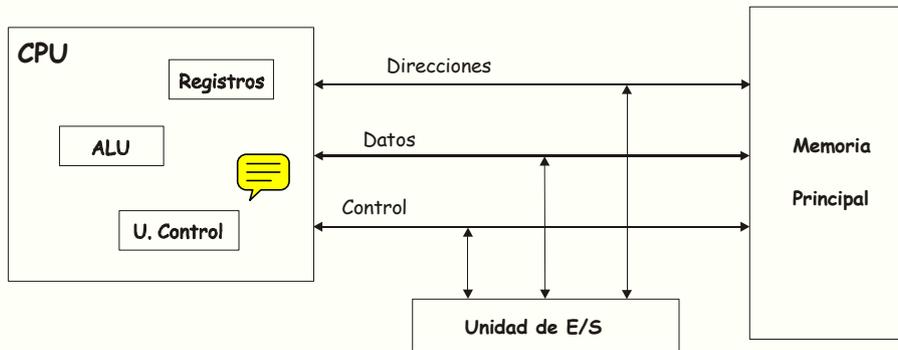
Introducción

Visión global del computador y de la CPU



6

Esquema básico del computador Von Neumann. Componentes



7

Introducción

- Función básica de C:

ejecutar instrucciones

1. Leer la instrucción en memoria
2. Decodificar la instrucción
para determinar qué acciones se han de realizar
3. Ejecución: como máximo
 - Búsqueda de operandos
 - cálculo de operando
 - leer operando en memoria
 - Operación
 - Almacenar resultado
4. Preparar la siguiente instrucción (ir a paso 1)

Fase de ejecución

Ejecución de la instrucción

8

Introducción

- Funciones de la :
 1. Ejecuta instrucciones (función básica)
 - Lectura, decodificación, e interpretación de las instrucciones
 - Generación de órdenes para la ejecución
 - Secuenciamiento de las instrucciones
(decidir cuál es la siguiente a ejecutar)
 2. Resuelve situaciones anómalas
(desbordamiento, operación no válida, error de paridad, etc)
 3. Controla la comunicación con periféricos

9

Introducción

- Entradas y salidas de la UC:
 - Entradas:
 - Registro de Instrucción:   CO, MDS
 - Reloj: registro contador de fases
 - Registro de estado 
 - Señales de control externas (E/S, Memoria,)
 - Salidas
 - Todas las señales de control que permiten realizar cada una de las instrucciones máquina

10

Introducción

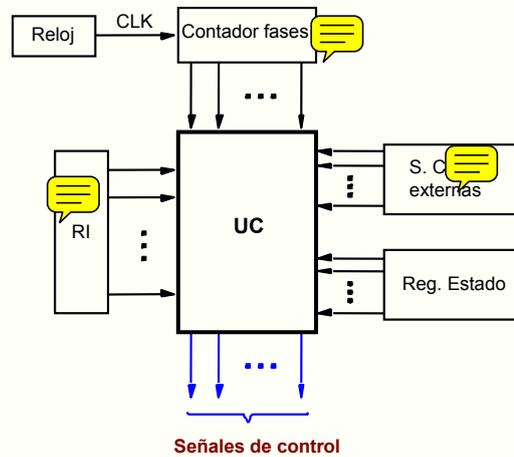
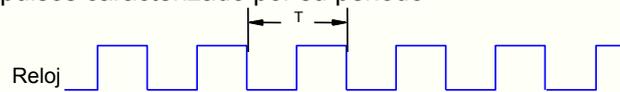


Figura Entradas y salidas de la UC.

11

Introducción

- **Reloj:** tren de pulsos caracterizado por su periodo

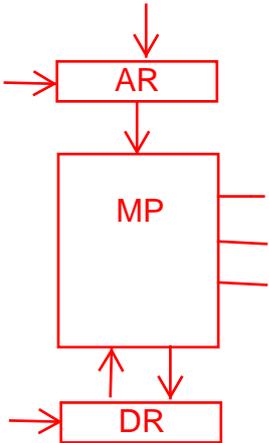


- **señales de control:** siempre sincronizadas con el reloj
- Define el tiempo de cada operación.
 - Memoria: tiempo de lectura y escritura
 - ALU: tiempo de operación
- **CAMINO CRÍTICO:** camino de máximo retardo entre un origen y un destino. Depende de los dispositivos que tengan que atravesar las señales.

Ej: si $f=100$ MHz

$T=10$ ns y conviene que una operación aritmética se realice en un tiempo algo menor que 10 ns y la Memoria?

12



Introducción

Ciclo de lectura

Se supone AR siempre disponible en el bus de direcciones de memoria

1. dirección → AR
2. M(AR) → DR

Señales de control:
FAR, MEMRQ, R, FDR

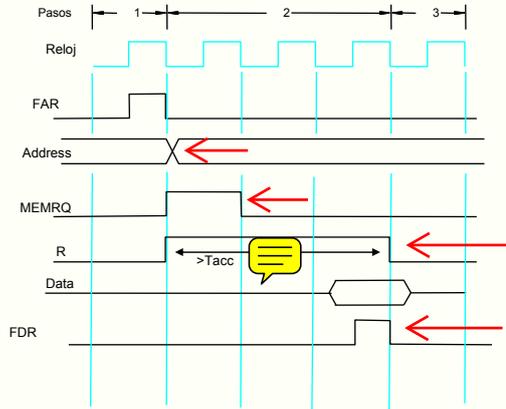


Figura . Temporización de la lectura en memoria

13



Introducción

Ciclo de escritura

AR y DR disponibles en buses de dirección y datos

1. dirección → AR
- dato → DR
3. DR → M(AR) ; escritura

Señales de control:
FAR, FDR, MEMRQ, W

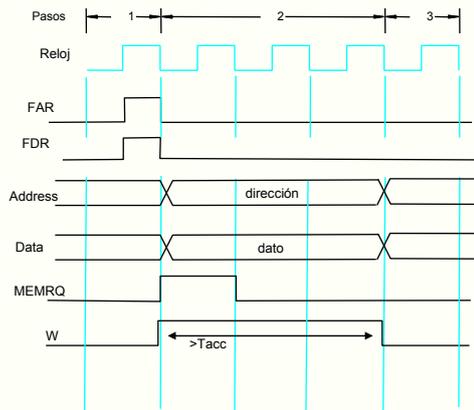


Figura . Temporización de la escritura en memoria

14

Introducción

Ciclos de lectura y escritura: ciclo de BUS

- implica un acceso al exterior a la CPU
 - durante ese tiempo sólo la CPU puede acceder a los buses
 - (ningún periférico)
- suele durar más de 1 ciclo de reloj (determinado por el camino crítico) interno a la CPU

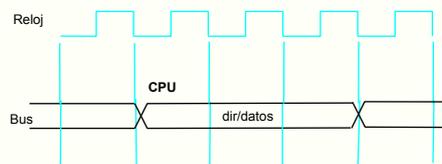


Figura . Ciclo de bus

15

Introducción

Evaluación del rendimiento

- Supóngase los tiempos de ejecución:
 - Acceso a memoria: 8 ns
 - ALU y sumadores: 2 ns
 - Acceso a registros: 1 ns
- ¿Cuál de las siguientes realizaciones será más rápida?
 - Una en la que cada instrucción se ejecuta en un ciclo de tamaño fijo (cada instrucción tarda lo que tardaría la más lenta).
 - Una realización donde cada instrucción se ejecuta en un ciclo de longitud variable (cada instrucción tarda únicamente lo necesario)
- ¿De qué dependería la duración del periodo de reloj?

16



Operaciones Elementales

- Ciclo de instrucción: conjunto de acciones que requiere la ejecución de una instrucción
 1. Leer la instrucción en memoria
 2. Decodificar la instrucción
 3. Ejecución: como máximo
 - Búsqueda de operandos, que puede conllevar
 - Cálculo de operando
 - Leer operando en memoria
 - Operación
 - Almacenar resultado
 4. Preparar la siguiente instrucción (ir a paso 1)

17



Operaciones Elementales

- Las **fases de ejecución** ayudan a simplificar el diseño de la UC.
 - Ej: fetch común a todas las instrucciones
- Funcionamiento del computador durante la ejecución de un programa consiste en una **sucesión de ciclos de instrucción**

18



Operaciones Elementales

- **OPERACIONES ELEMENTALES:** operaciones (realizables directamente por el hardware) en que la UC divide cada una de las fases de ejecución de una instrucción
- Las operaciones elementales se realizan por la UC mediante la activación de señales de control
- Cada operación elemental dura un ciclo de reloj (excepto las referidas a los accesos a Memoria)
- Se pueden simultanear en el tiempo, cuidando:
 - Orden preestablecido
 - Conflictos en las rutas de datos

19



Operaciones Elementales

- Tipos:
 - **De transferencia**
Llevan información de un origen a un destino
 - **De proceso**
Llevan información de un origen a un destino, pero ésta pasa por un operador en su camino al destino

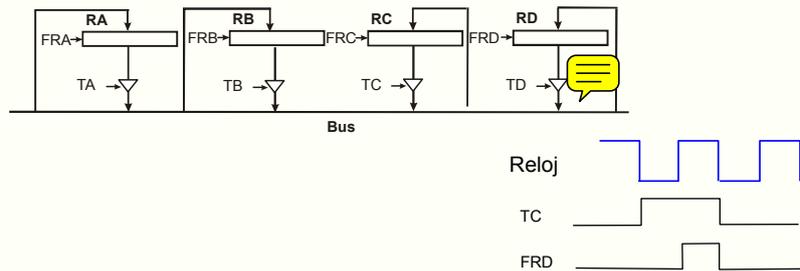
20

Operaciones Elementales

OE de transferencia

- Llevar información de un ORIGEN a un DESTINO
 - Establecer camino físico entre salida de origen y entrada de destino
 - Enviar señal al destino para que tome la información

Ej: Transferencia entre dos registros, a través de un bus: **C → D**



21

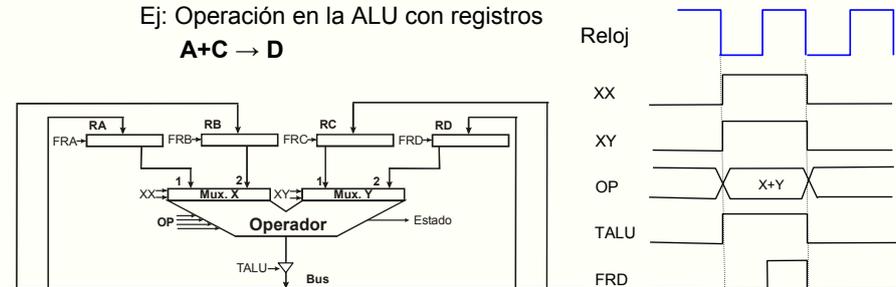
Operaciones Elementales

OE de proceso

- Funcionamiento similar a la transferencia, pero la información de ORIGEN se pasa por un operador que la procesa en su camino al DESTINO

Ej: Operación en la ALU con registros

$$A + C \rightarrow D$$

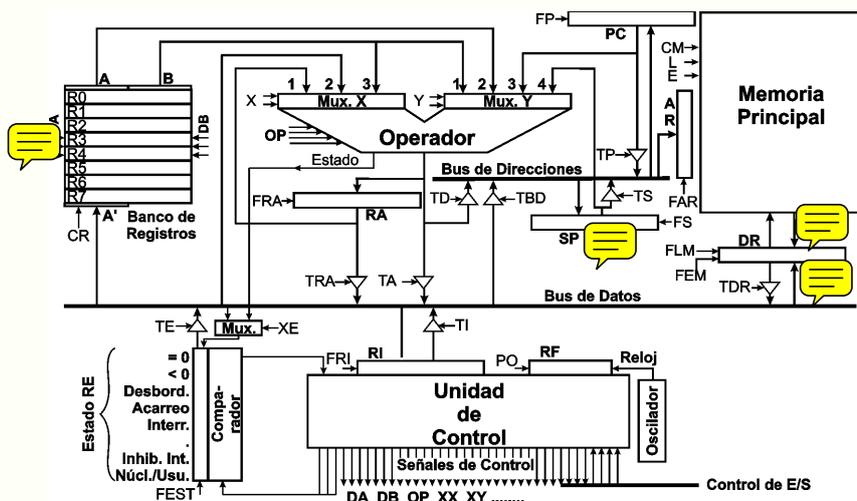


22

Operaciones Elementales

- **Reglas de agrupación de operaciones elementales**
 - Respetar orden preestablecido de algunas acciones
 Ej: antes de leer, guardar dirección en AR
 - Evitar conflictos en el mismo recurso físico: un dispositivo no puede estar en dos estados diferentes al mismo tiempo
 Ej: conflicto en bus (dos op. elem. intentan usar el mismo bus);
 memoria no puede leer y escribir al mismo tiempo
 - La información debe guardarse en algún sitio: ORIGEN y DESTINO deben poder almacenar información (registro o Memoria)
 Ej: no se deben simultanear con el mismo destino si la información es distinta

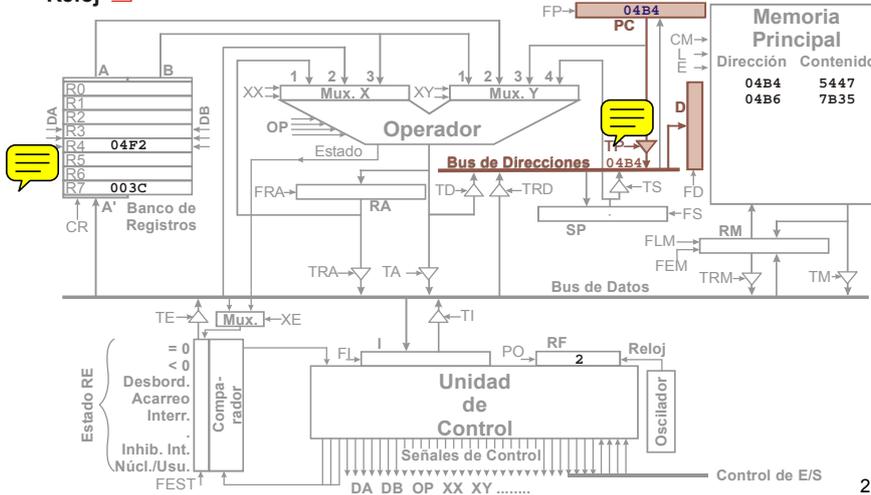
Estructura de Computador Elemental y sus señales de control



Prepara la lectura instrucción
 Se prepara: $D \leftarrow PC$

ADD .R4,.R7

Reloj



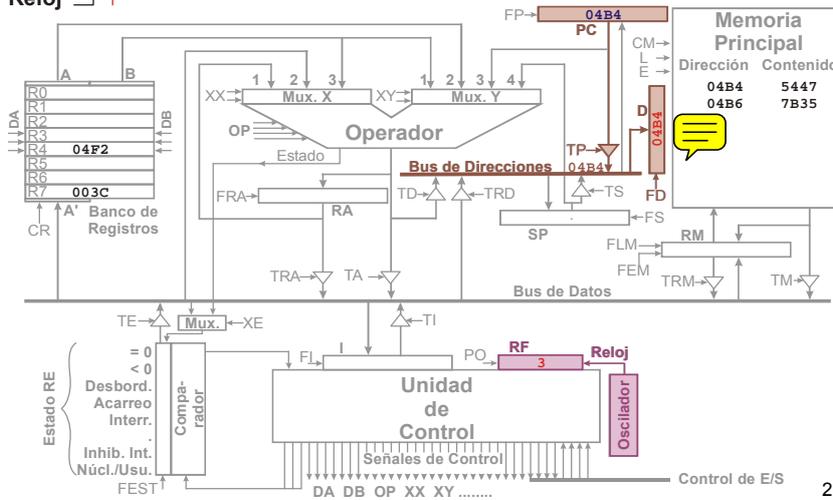
27

Prepara la lectura instrucción

Se realiza: $D \leftarrow PC$
 Se incrementa RF: $RF \leftarrow RF + 1$

ADD .R4,.R7

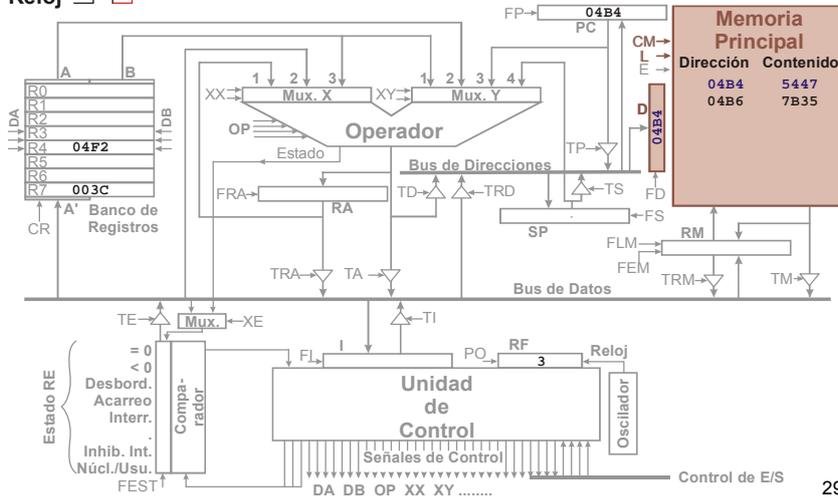
Reloj



28

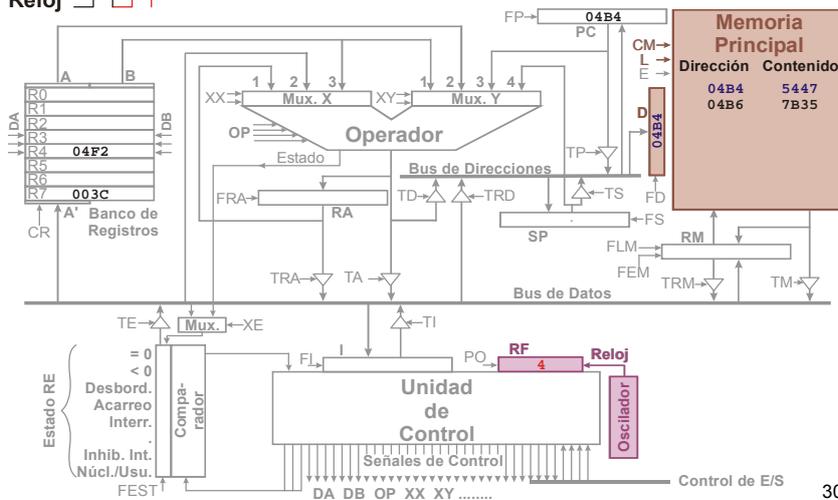
Lectura instrucción
Primer ciclo de memoria

Relej 



Lectura instrucción
Primer ciclo de memoria
Se incrementa RF: $RF \leftarrow RF + 1$

Relej 



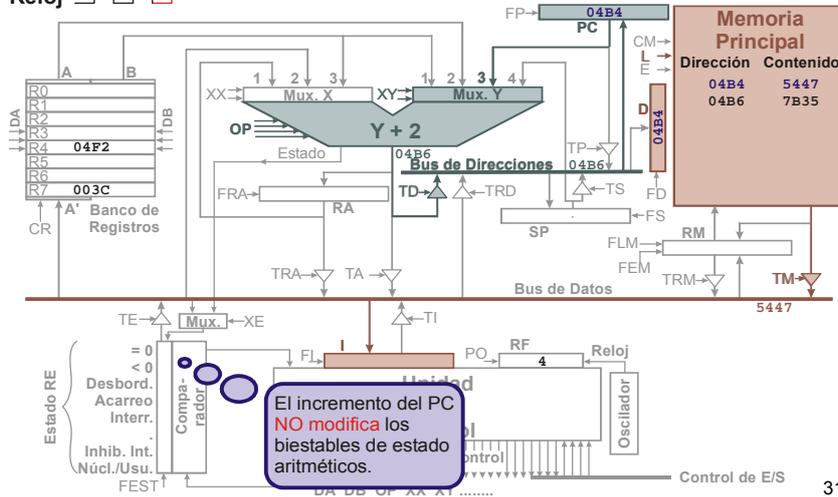
ADD .R4,.R7

Lectura instrucción

Segundo ciclo de memoria, se prepara: $I \leftarrow M(D)$

Se prepara incremento de PC: $PC \leftarrow PC + 2$

Reloj 



31

ADD .R4,.R7

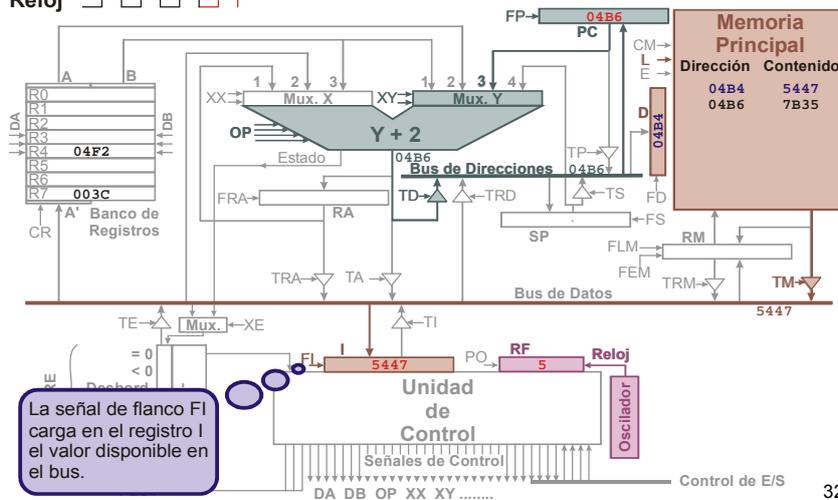
Lectura instrucción

Segundo ciclo de memoria, se realiza: $I \leftarrow M(D)$

Se realiza incremento de PC: $PC \leftarrow PC + 2$

Se incrementa RF: $RF \leftarrow RF + 1$

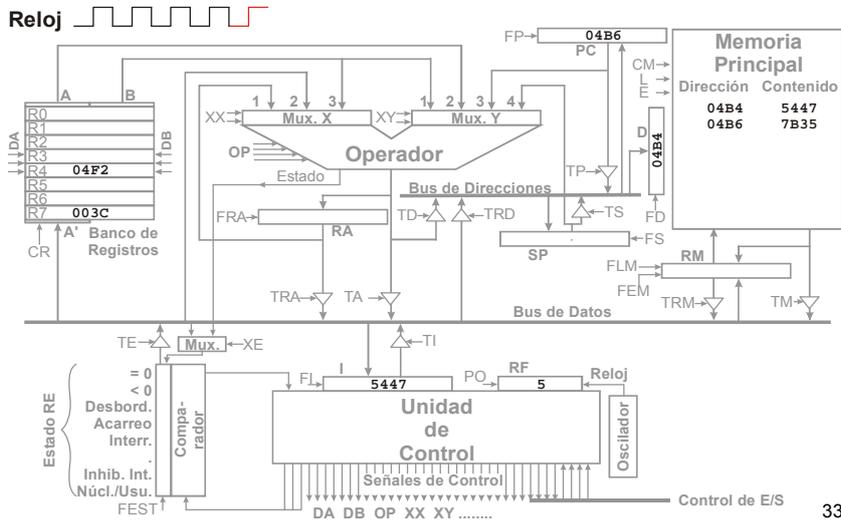
Reloj 



32

Decodificación instrucción

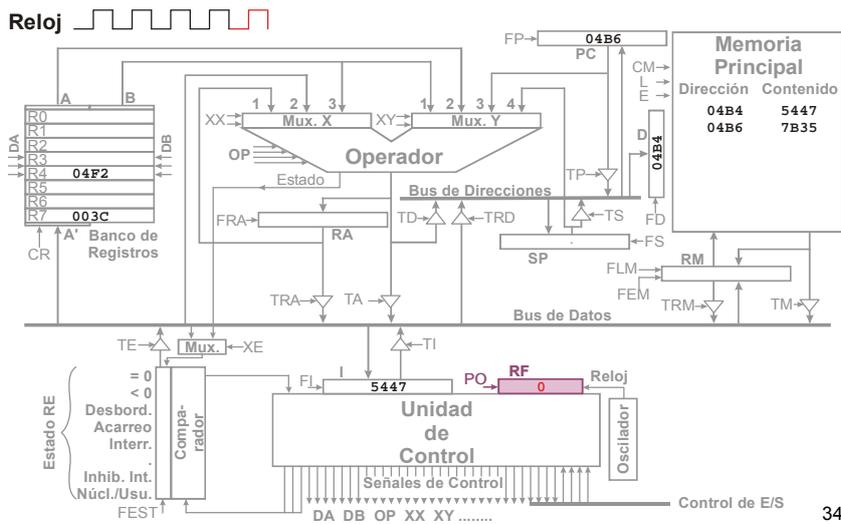
ADD .R4,.R7



Decodificación instrucción

ADD .R4,.R7

Se pone RF a 0: RF ← 0

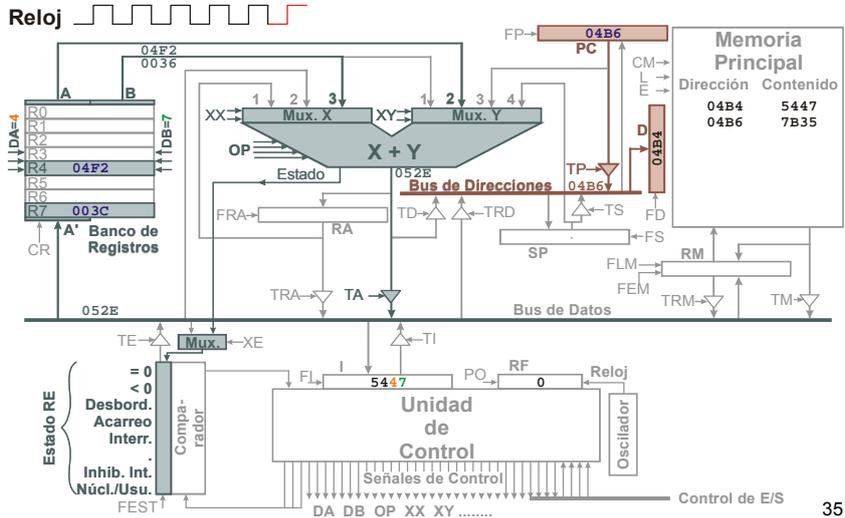


ADD .R4,.R7

Ejecuta instrucción y prepara lectura instrucción siguiente

Se prepara suma: $R4 \leftarrow R4 + R7$; $RE \leftarrow$ Estado ALU

Se prepara: $D \leftarrow PC$



35

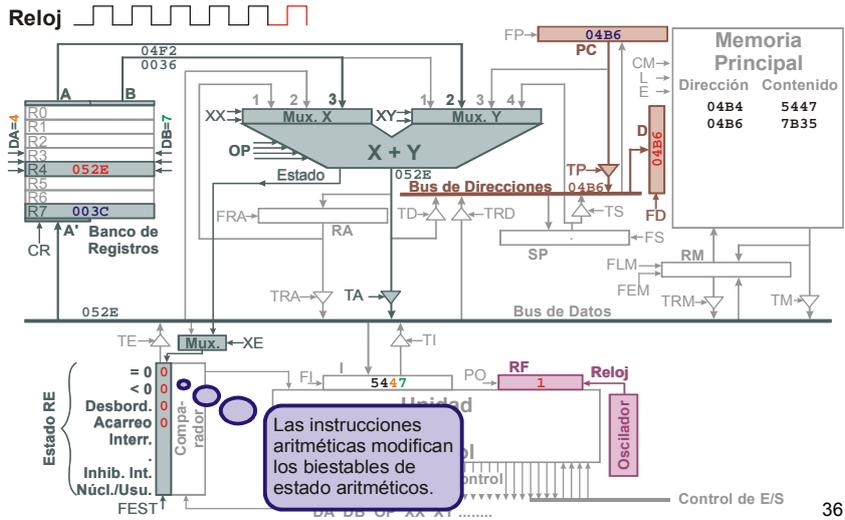
ADD .R4,.R7

Ejecuta instrucción y prepara lectura instrucción siguiente

Se realiza suma: $R4 \leftarrow R4 + R7$; $RE \leftarrow$ Estado ALU

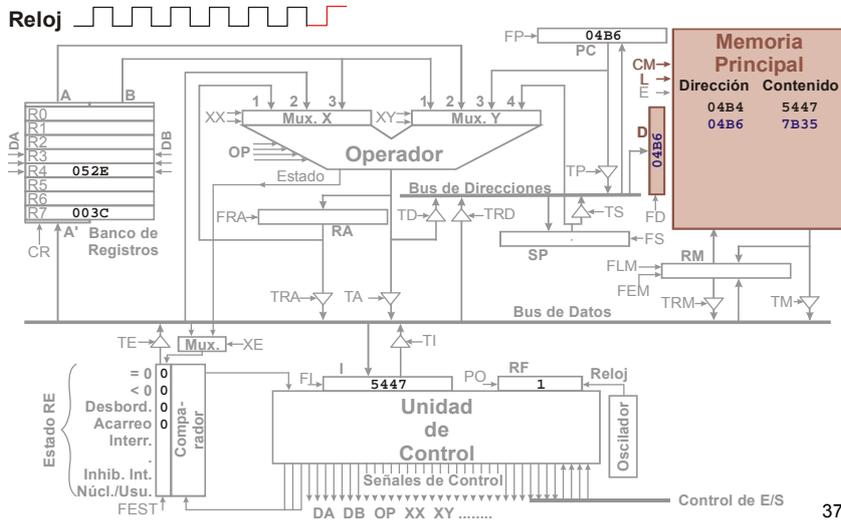
Se realiza: $D \leftarrow PC$

Se incrementa RF: $RF \leftarrow RF + 1$

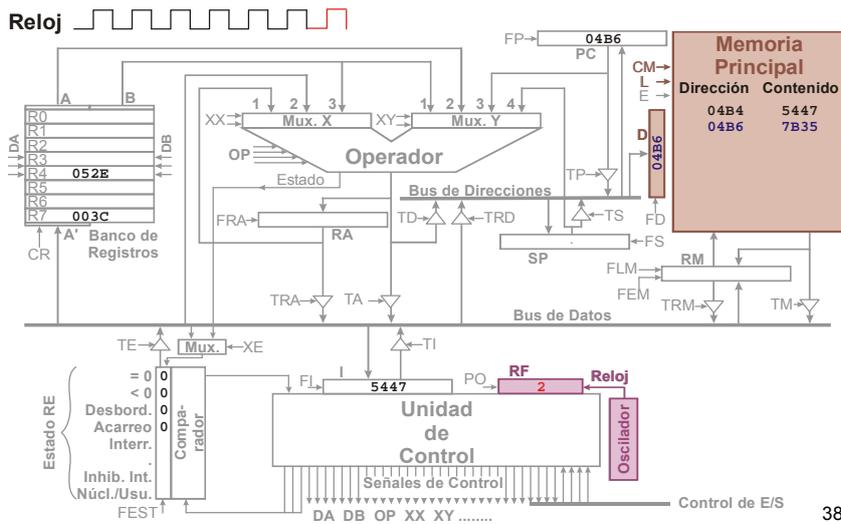


36

Lectura instrucción siguiente
Primer ciclo de memoria



Lectura instrucción siguiente
Primer ciclo de memoria
Se incrementa RF: $RF \leftarrow RF + 1$

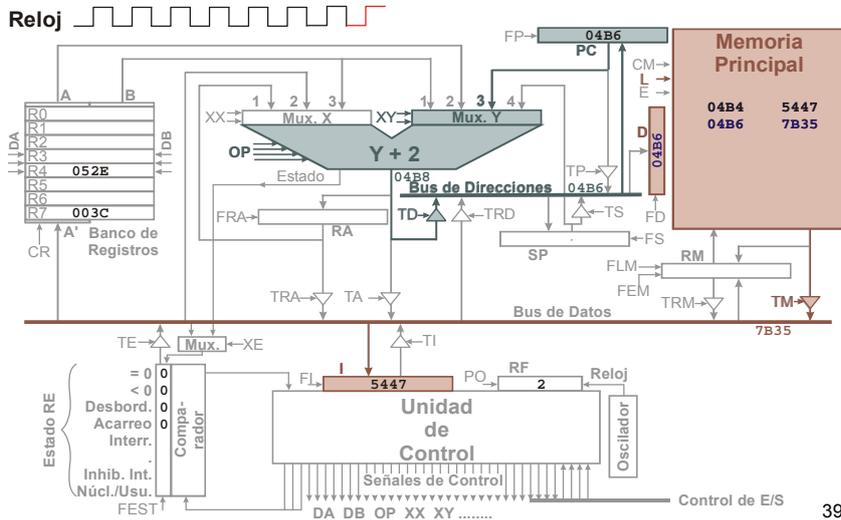


ADD .R4,.R7

Lectura instrucción

Segundo ciclo de memoria, se prepara: $I \leftarrow M(D)$

Se prepara incremento de PC: $PC \leftarrow PC + 2$



39

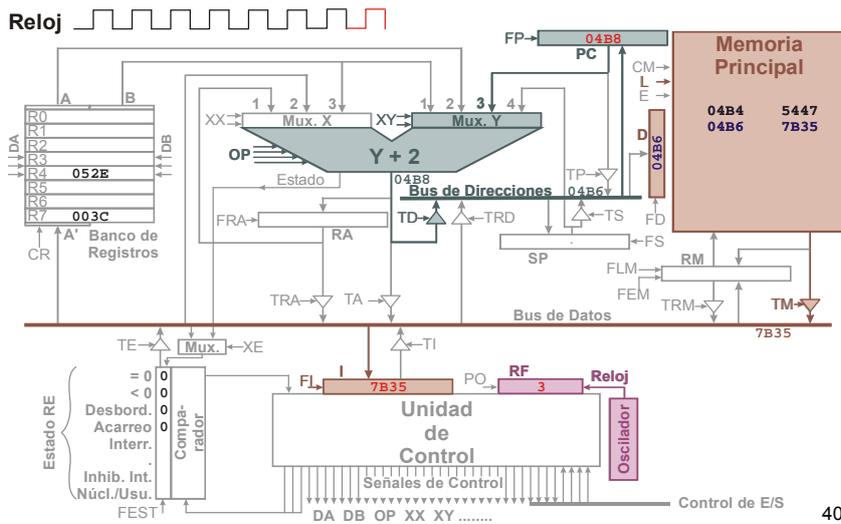
ADD .R4,.R7

Lectura instrucción

Segundo ciclo de memoria, se realiza: $I \leftarrow M(D)$

Se realiza incremento de PC: $PC \leftarrow PC + 2$

Se incrementa RF: $RF \leftarrow RF + 1$

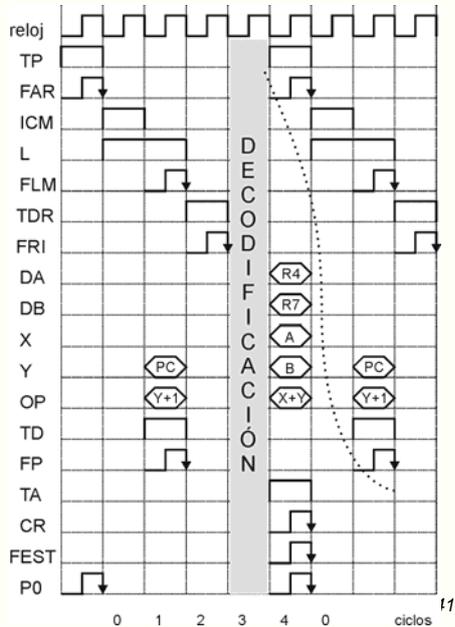


40



Cronogramas

- **ADD .R4, .R7**
- **Lectura de la instrucción**
 - PC → AR (1 ciclo)
 - M(AR) → DR, PC + 1 → PC (2 ciclos)
- DR → RI (1 ciclo)
- **Decodificación (1 ciclo)**
- **Ejecución y fetch sig. Instrucción**
 - R4+R7 → R4 y FEST;
 - PC → AR (1 ciclo)
 - M(AR) → DR, PC + 1 → PC (2 ciclos)
 - DR → RI (1 ciclo)



Diseño de la Unidad de Control

- El diseño de la UC exige haber definido previamente
 - El repertorio de instrucciones (formatos y direccionamientos)
 - A nivel de cronogramas o de operaciones elementales
 - La estructura del computador
- La UC funciona como un TRADUCTOR
RI(CO, MD), Reloj, Estado, señ ext → señales de control

Ej: CO: 8 bits, 1 instr=16 ciclos → RF: 5 bits, ...
traductor de unas 20-30 señales de entrada y 150 señales salida



Diseño de la Unidad de Control

- Formas de diseño de la UC
 - UC CABLEADA
 - Utilizando exclusivamente puertas lógicas
 - UC MICROPROGRAMADA
 - Utilizando una memoria de control

43



Diseño de la Unidad de Control

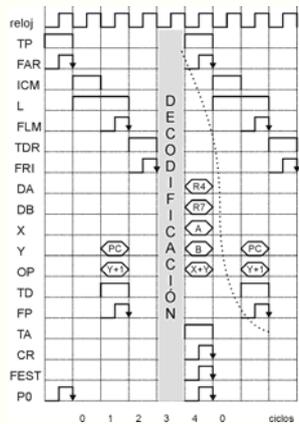
- UC CABLEADA
 - Se construye mediante métodos generales del diseño lógico (circuito secuencial o combinacional)
 - Más rápidos
 - Diseño y construcción complejo y costoso
 - Las técnicas actuales (CAD, compilación de silicio) permiten que sea solución viable y válida cara a construir máquinas rápidas

44

Diseño de la Unidad de Control

- **UC MICROPROGRAMADA**
 - Utiliza una memoria para almacenar el estado de las señales de control en cada periodo de la ejecución de cada instrucción.
 - Generar el cronograma de cada instrucción (ejecutar la Instrucción) es ir leyendo de esta memoria (Memoria de Control).

Diseño de la Unidad de Control

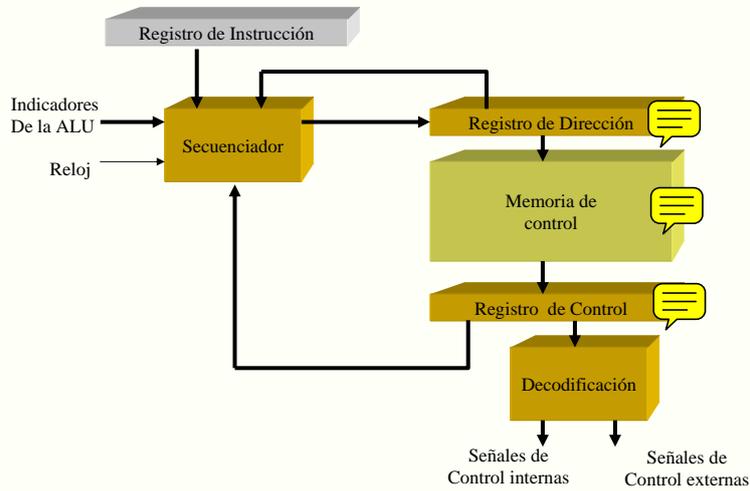


Memoria de Control

	FP	FAR	ICM	L	FLM	TDR	DB	FR	X	Y	DB	TD	FP	TA	FEST	CR	...
0	x	x	x	x	x	0	0	0	0	0	0	0	1	1	0	0	0
1	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0
6	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
7	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0
8	x	x	x	x	x	0	0	1	1	0	0	0	0	0	0	0	0
9	x	x	x	x	x	0	1	0	0	0	0	0	0	0	0	0	0
10	x	x	x	x	x	0	1	0	0	0	0	0	0	0	0	0	0
11	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0
12	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0



Diseño de la Unidad de Control



Diseño de la Unidad de Control

UC CABLEADA	UC MICROPROGRAMADA
Ventajas <ul style="list-style-type: none"> Alta velocidad de funcionamiento Implementaciones más pequeñas (reducido número de componentes) 	Ventajas <ul style="list-style-type: none"> Son sistemáticas con un formato bien definido Pueden ser fácilmente modificables durante el proceso de diseño
Inconvenientes <ul style="list-style-type: none"> Difícil realizar modificaciones en el diseño: modificación → rediseño No tienen estructura común 	Inconvenientes <ul style="list-style-type: none"> Requieren más componentes para su implementación Tienden a ser más lentas que las unidades de control cableadas debido a que tienen que realizar operaciones de lectura de una memoria para obtener las señales de control
<ul style="list-style-type: none"> Técnica de diseño favorita en las arquitecturas RISC 	<ul style="list-style-type: none"> Emulación <ul style="list-style-type: none"> Se puede utilizar un microprograma para que la UC interprete otro lenguaje máquina distinto (una máquina a emular) sin necesidad de realizar modificaciones en el hardware de la unidad de control -> cambiando sólo el microprograma!



Diseño de la Unidad de Control

- Conceptos básicos
 - UC MICROPROGRAMADA
 - MICROINSTRUCCIÓN
 - CICLO DE MÁQUINA
 - MICROPROGRAMA

49



Conceptos básicos

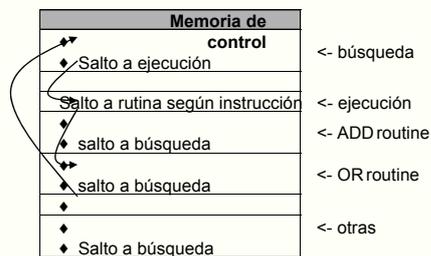
- **Microinstrucción**
 - **Cadena de ceros** y unos que
 - representa la activación o no del conjunto de señales de control durante un ciclo de reloj
 - y la posibilidad de decidir condicionalmente qué microinstrucción se debe ejecutar a continuación
 - **Cada palabra** de la memoria de control es una **microinstrucción**.

50

Conceptos básicos

Microprograma

- Secuencia de μI 's cuya ejecución permite interpretar una instrucción
- Ejecutar una microinstrucción \approx Leer una μI de la memoria
- Una vez diseñado el camino de datos y la memoria de μI 's: el control es tarea de (micro)programación
- Cambiando microprogramas se puede ejecutar otros repertorios de instrucc (simular otras archit) -> **EMULACIÓN!**



51

Conceptos básicos

Memoria de Control

- Memoria que almacena los microprogramas o (conjuntos de μI 's) a partir de los que se obtienen las señales de control necesarias para la ejecución del repertorio de instrucciones
- Suelen ser ROM

52



Diseño de la Unidad de Control

UC MICROPROGRAMADA

- Condiciones básicas:
 1. Capacidad suficiente de MC
 2. Procedimiento de correspondencia
Inst → dir MC comienzo su μ programa
 3. Secuenciamiento de μ instrucciones

53



Conceptos básicos

- **Formato de microinstrucciones**
 - **Señales de control:** Señales para el camino de datos
 - **Condiciones:** Bits para seleccionar la condición que se desea utilizar para en función de si es cierta o no ejecutar la siguiente microinstrucción o saltar a otra
 - **Siguiente μ instrucción:** Campo que indica la siguiente μ instrucción a ejecutar

Señales de control	Condiciones	Siguiente μ instrucción
--------------------	-------------	-----------------------------

54

Control de excepciones

- Además de ejecutar la secuencia de instrucciones del programa de usuario, la UC debe controlar situaciones excepcionales
 - Qué ocurre si el **CV** no es válido?
 - Y si hay overflow?
 - Cómo y cuándo comunicar con algún periférico?

59

Control de excepciones

- Excepción [Overflow...]
 - Evento no planificado que para la ejecución de un programa. **(interno)**
 - Salvar el PC y Estado actual en pila
 - Saltar a rutina de SO que maneje excepciones internas
 - SO retornará después al programa, en su anterior estado
- Interrupción [E/S]
 - Un evento **externo** inesperado rompe la ejecución del programa

60



Control de excepciones

- **Objetivos**
 - *Detección de excepciones* – Cuándo y cómo verlas?
 - *Manejo de excepciones* – Qué hacer?

61



Control de excepciones

Tipos de excepciones

- Interrupciones:
 - Causadas por eventos externos (peticiones de periféricos).
 - Asíncronos a la ejecución del programa (cualquier instante) 
 - Se atienden entre Instrucciones (fetch)
 - Mantienen sin ejecutarse un tiempo el programa de usuario

62

Control de excepciones

Tipos de excepciones

Traps:

- Causadas por eventos internos
 - Condiciones de excepción (ej: overflow)
 - Errores (ej: error de paridad)
 - Fallos (ej: Fallo de pagina)
 - Llamadas al sistema
- Síncronos a la ejecución del programa (se puede saber cuándo)
- Puede que el programa de usuario se retome tras tratar la excepción, o bien que se aborte

63

Control de excepciones



Manejo de excepciones

- Excepción = transferencia de control no programada
- El sistema maneja la excepción:
 - Guarda la dir de la Instrucción que causó la excepción
 - Salvaguarda el estado del programa de usuario
 - Retorna el control a usuario (salvo que se aborte)

64



Control de excepciones

- ◆ Interrupciones Vectorizadas:
 1. Cada excepción tiene una dirección diferente asociada
 2. Detectada la excepción => la CPU calcula para esa **excepción** la dirección a escribir en PC (depende del periférico, que pone un VECTOR en bus).
- Cuando atenderlas?
 - Entre Instrucciones: antes de comenzar el FETCH

*FETCH***: Si INT e INT_no_inhibidas ir a
Microrrut_proceso_excepcion
Si no ir a FETCH