



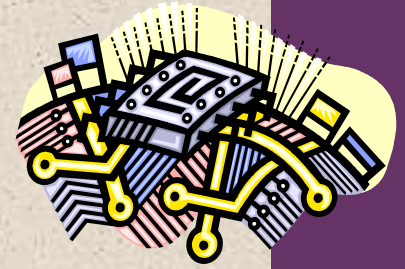
+ Capítulo 4

Estructura y función del procesador

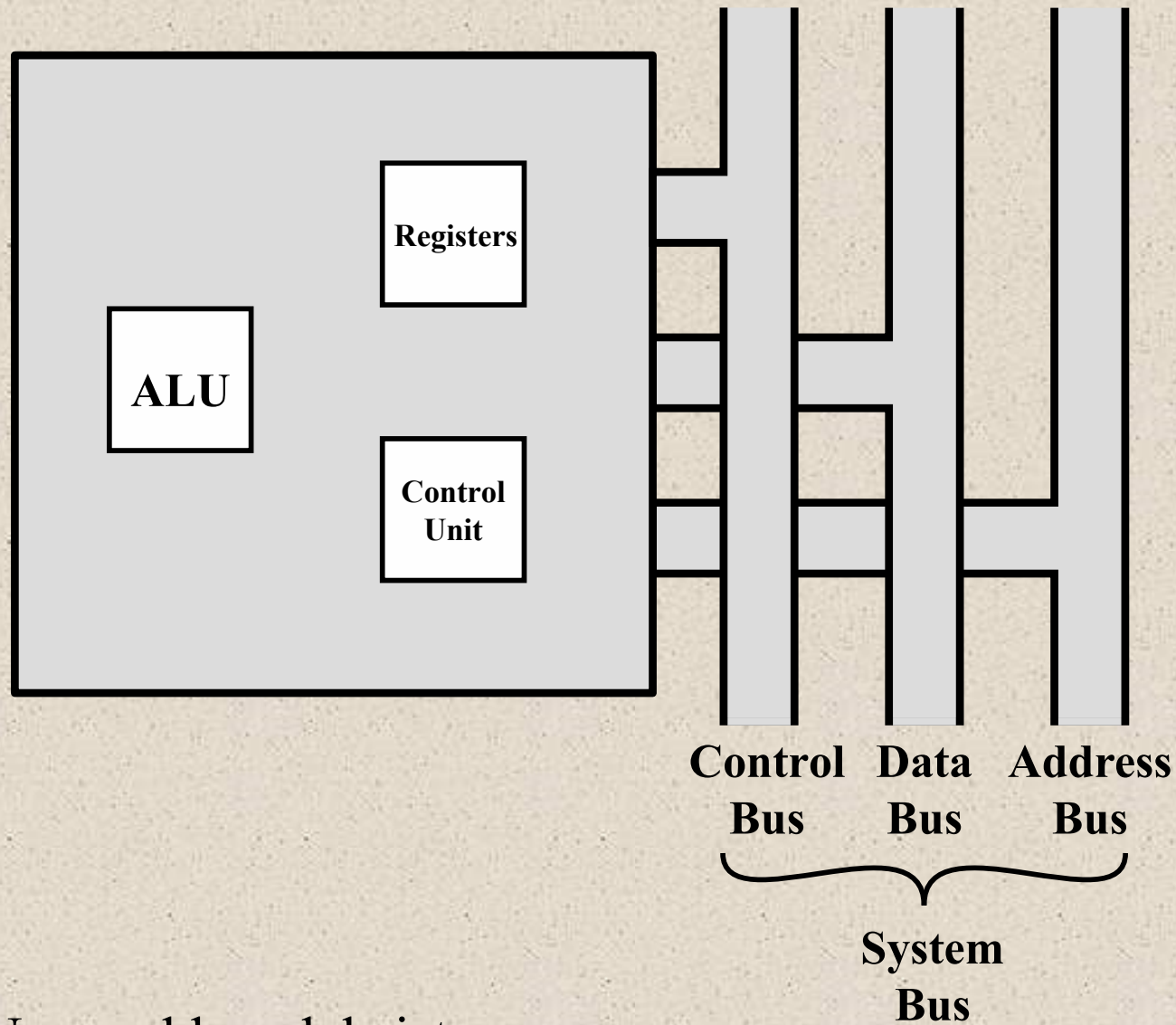


Organizacion del procesador

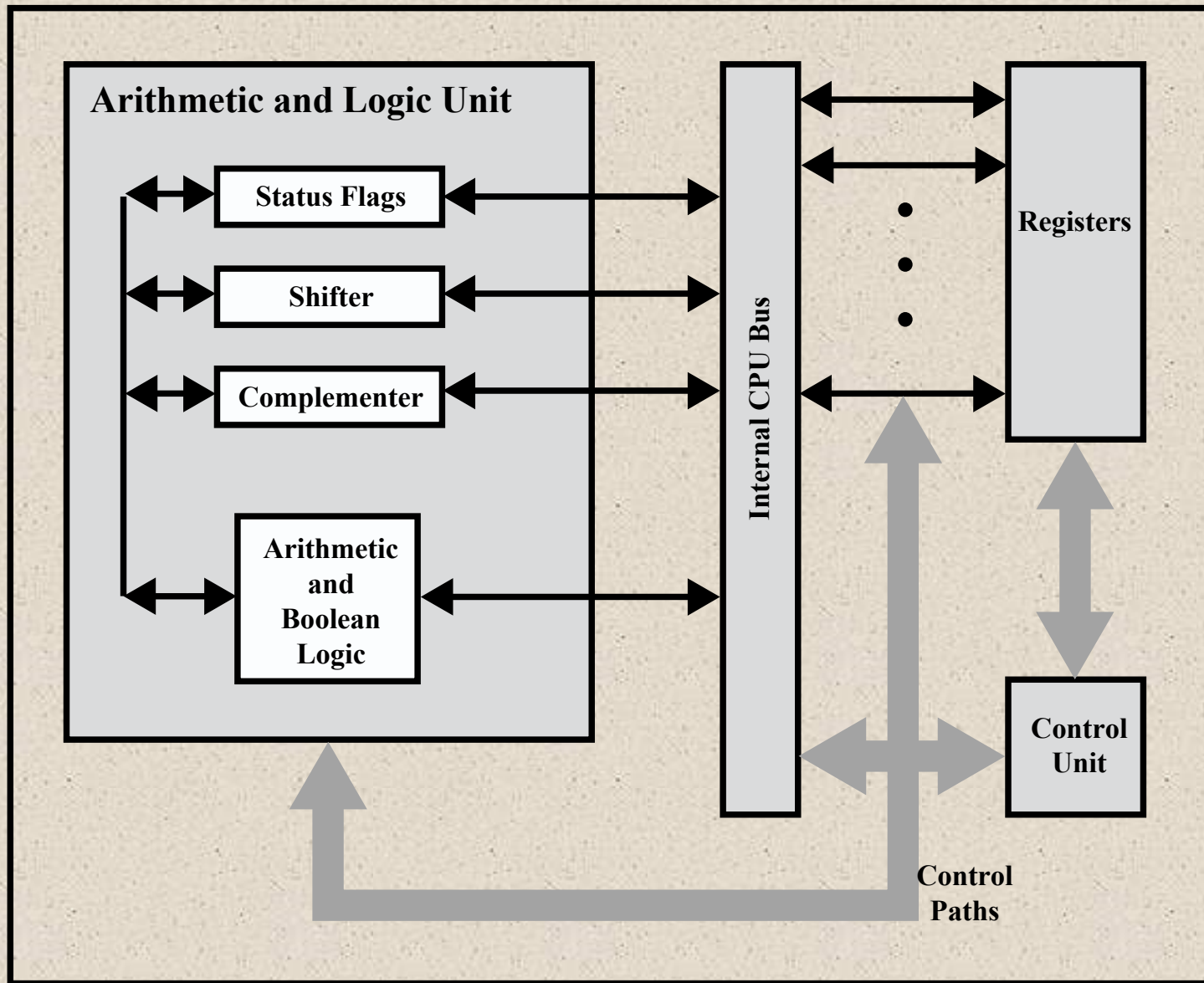
Requisitos del procesador:



- **Fetch instruction**
 - El procesador lee una instrucción de la memoria (registro, caché, memoria principal)
- **Interpret instruction**
 - La instrucción se decodifica para determinar qué acción se requiere
- **Fetch data**
 - La ejecución de una instrucción puede requerir la lectura de datos de la memoria o un módulo de E / S
- **Process data**
 - La ejecución de una instrucción puede requerir realizar alguna operación aritmética o lógica en los datos
- **Write data**
 - Los resultados de una ejecución pueden requerir la escritura de datos en la memoria o un módulo de E / S
- Para hacer estas cosas, el procesador necesita almacenar algunos datos temporalmente y, por lo tanto, necesita una pequeña memoria interna.



El CPU con el bus del sistema



Estructura interna del CPU



Organización de los Registros

- Dentro del procesador hay un conjunto de registros que funcionan como un nivel de memoria por encima de la memoria principal y la memoria caché en la jerarquía
- Los registros en el procesador realizan dos funciones:

Registros visibles por el usuario

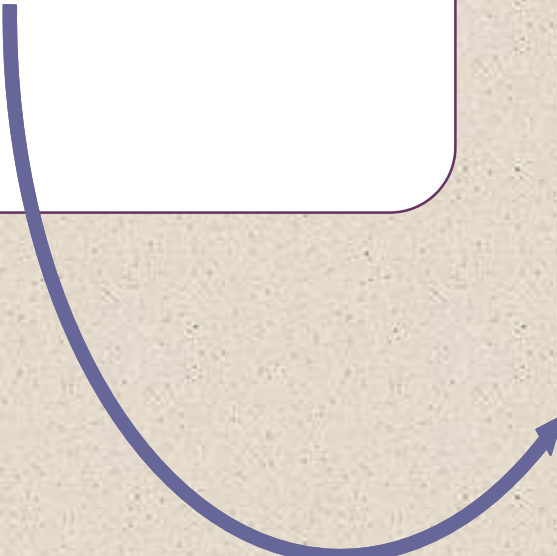
- Permite que el programador de lenguaje máquina o ensamblador minimice las referencias de memoria principal optimizando el uso de los registros

Registros de control y estado

- Utilizado por la unidad de control para controlar el funcionamiento del procesador y por programas privilegiados de sistema operativo para controlar la ejecución de programas

Registros visibles por el usuario

Referenciado por el lenguaje de máquina que ejecuta el procesador.



Categorías:

- **General purpose**
 - Puede ser asignado a una variedad de funciones por el programador
- **Data**
 - Puede usarse solo para guardar datos y no puede emplearse en el cálculo de una dirección de operando
- **Address**
 - Puede ser también de propósito general o estar dedicado a un modo de direccionamiento particular
 - Ejemplos: punteros de segmento, registros de índice, puntero de pila
- **Condition codes**
 - Conocidos como banderas
 - Bits establecidos por el hardware del procesador como resultado de las operaciones.

Tabla 4.1 Códigos de condición

Ventajas	Desventajas
<ol style="list-style-type: none">1. Debido a que se establecen mediante instrucciones aritméticas y de movimiento de datos normales, reducen el número de instrucciones COMPARE y TEST necesarias.2. Las instrucciones condicionales, como BRANCH, se simplifican en relación con las instrucciones compuestas, como TEST AND BRANCH.3. Los códigos de condición facilitan bifurcaciones de múltiples vías. Ej. una instrucción TEST puede ir seguida de dos saltos, uno en menor o igual a cero y otro en mayor que cero.4. Los códigos de condición se pueden guardar en la pila durante las llamadas de subrutina junto con otra información de registro.	<ol style="list-style-type: none">1. Añaden complejidad, tanto al hardware como al software. Los bits de código de condición a menudo se modifican de diferentes maneras mediante instrucciones diferentes, lo que dificulta la vida tanto para el microprogramador como para el que escribe el compilador.2. Son irregulares; por lo general, no forman parte de la ruta principal de datos, por lo que requieren conexiones de hardware adicionales.3. Con frecuencia, se debe agregar instrucciones especiales para situaciones especiales, como la verificación de bits, el control de bucle y las operaciones de semáforo atómico.4. En una implementación canalizada, los códigos de condición requieren una sincronización especial para evitar conflictos



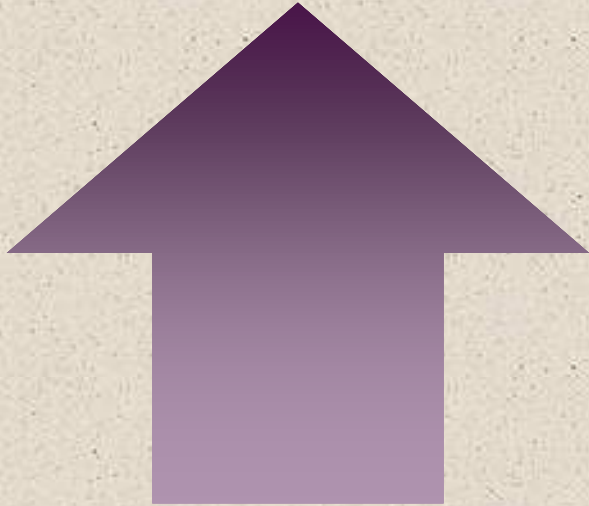
Registros de control y estado

Cuatro registros son esenciales para la ejecución de la instrucción:

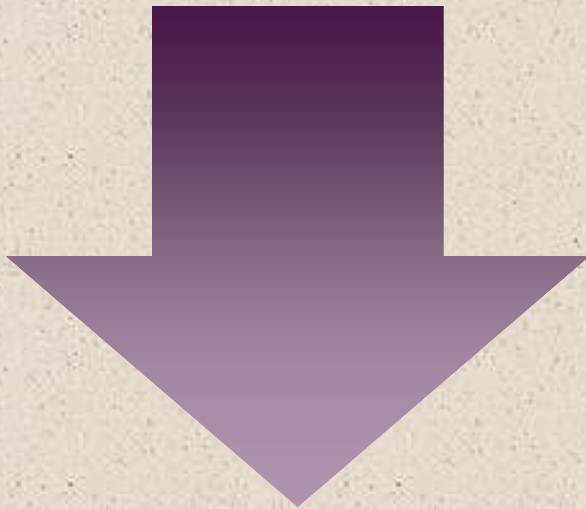
- Program counter (PC)
 - Contiene la dirección de una instrucción para ser buscada.
- Instruction register (IR)
 - Contiene la instrucción más reciente obtenida.
- Memory address register (MAR)
 - Contiene la dirección de una ubicación en memoria.
- Memory buffer register (MBR)
 - Contiene una palabra de datos para escribir en la memoria o la última palabra leída



+ Program Status Word (PSW)



Registro o conjunto de registros que contienen información de estado.



Los campos comunes incluyen:

- Sign
- Zero
- Carry
- Equal
- Overflow
- Interrupt Enable/Disable
- Supervisor

Data registers	
D0	
D1	
D2	
D3	
D4	
D5	
D6	
D7	

Address registers	
A0	
A1	
A2	
A3	
A4	
A5	
A6	
A7'	

Program status	
Program counter	
Status register	

(a) MC68000

General registers

AX	Accumulator
BX	Base
CX	Count
DX	Data

Pointers & index

SP	Stack ptr
BP	Base ptr
SI	Source index
DI	Dest index

Segment

CS	Code
DS	Data
SS	Stack
ES	Extrat

Program status

Flags
Instr ptr

(b) 8086

General Registers

EAX		AX
EBX		BX
ECX		CX
EDX		DX

ESP		SP
EBP		BP
ESI		SI
EDI		DI

Program Status

FLAGS Register
Instruction Pointer

(c) 80386 - Pentium 4

Ejemplo de organizaciones de registros de microprocesadores

Ciclo de instrucción

Incluye las siguientes etapas:

Fetch

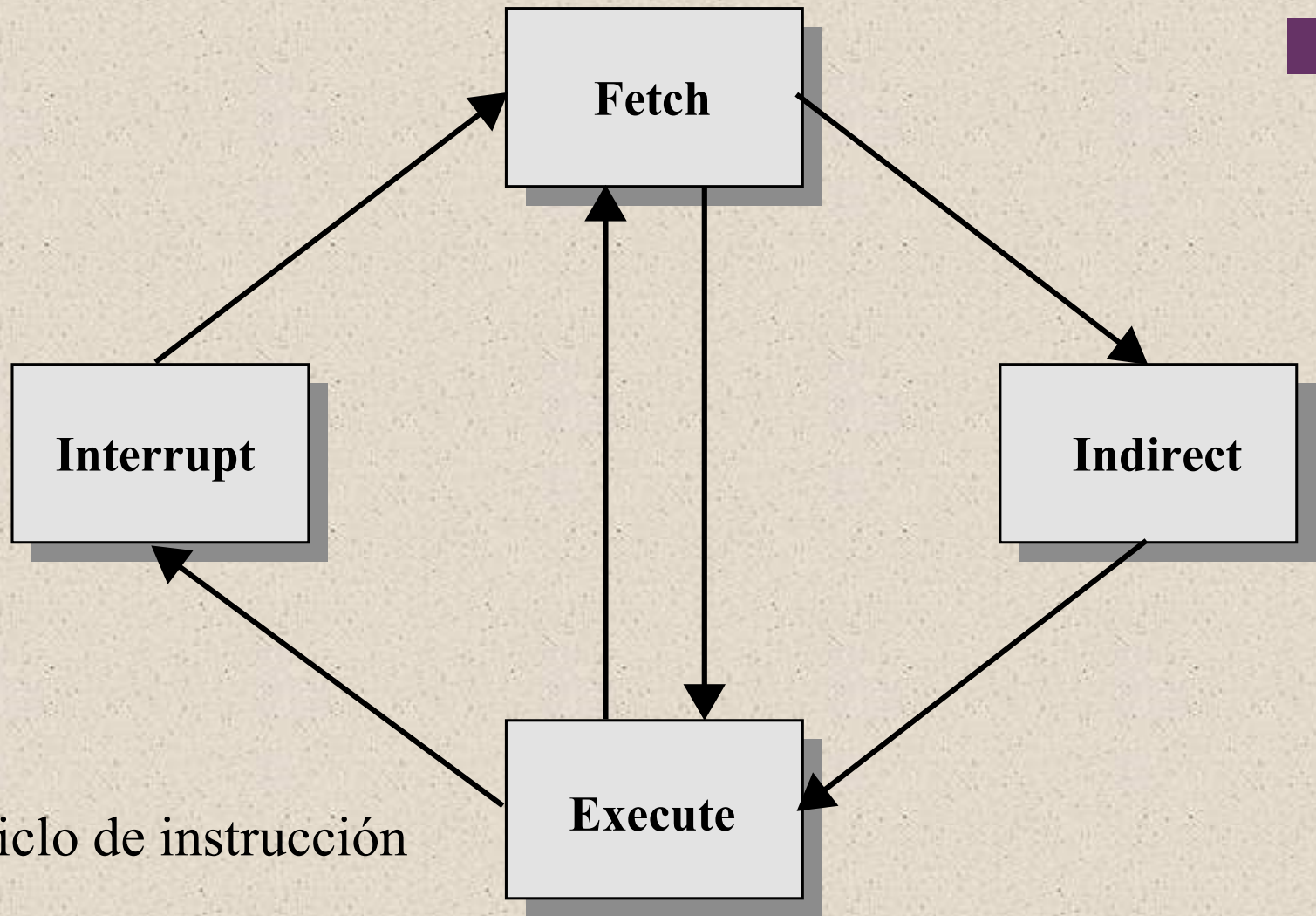
Leer las siguientes instrucciones de la memoria en el procesador

Execute

Interpretar el código de operación y realizar la operación indicada.

Interrupt

Si las interrupciones están habilitadas y se ha producido una interrupción, guarda el estado del proceso actual y atiende la interrupción.



El ciclo de instrucción

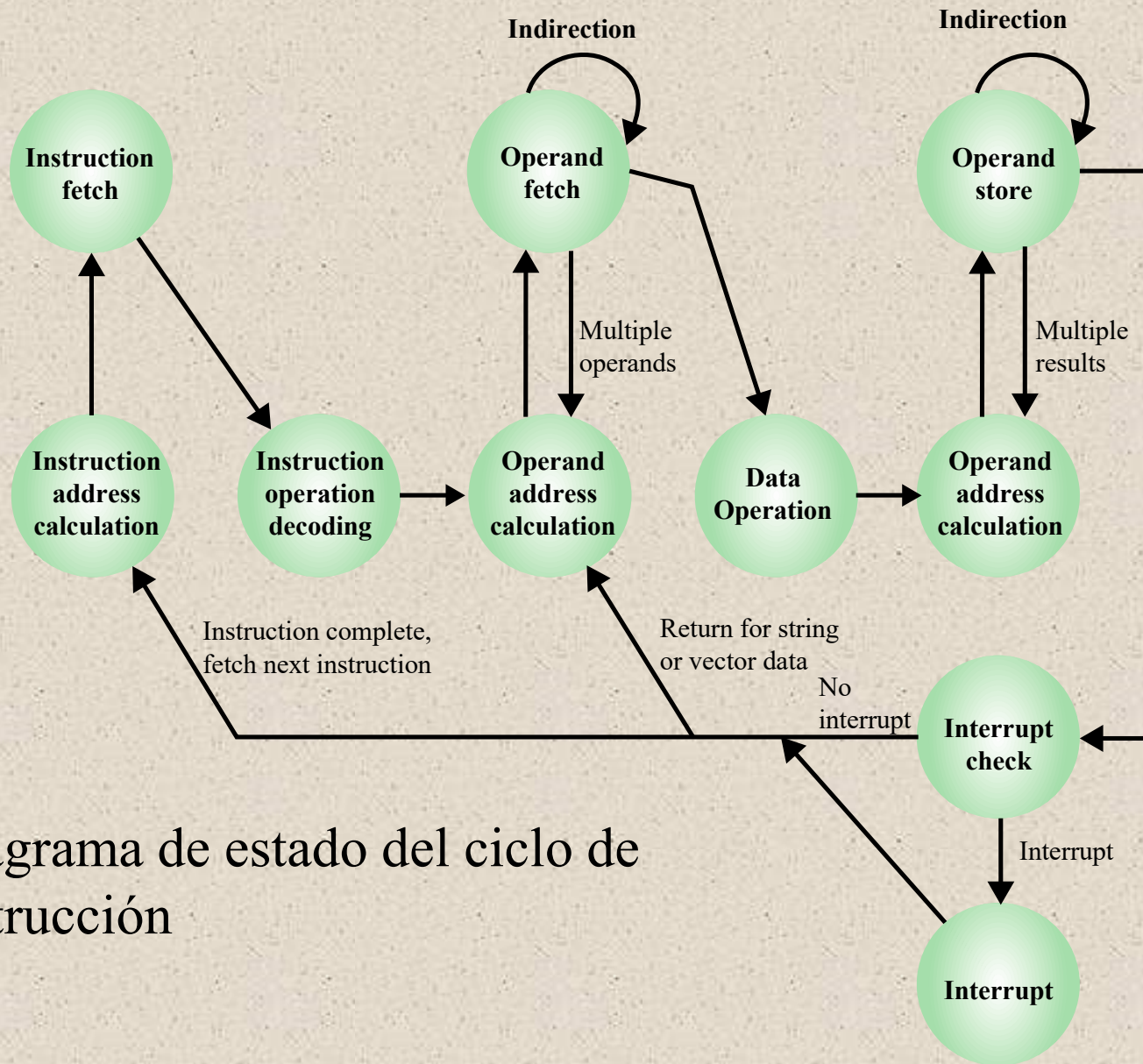
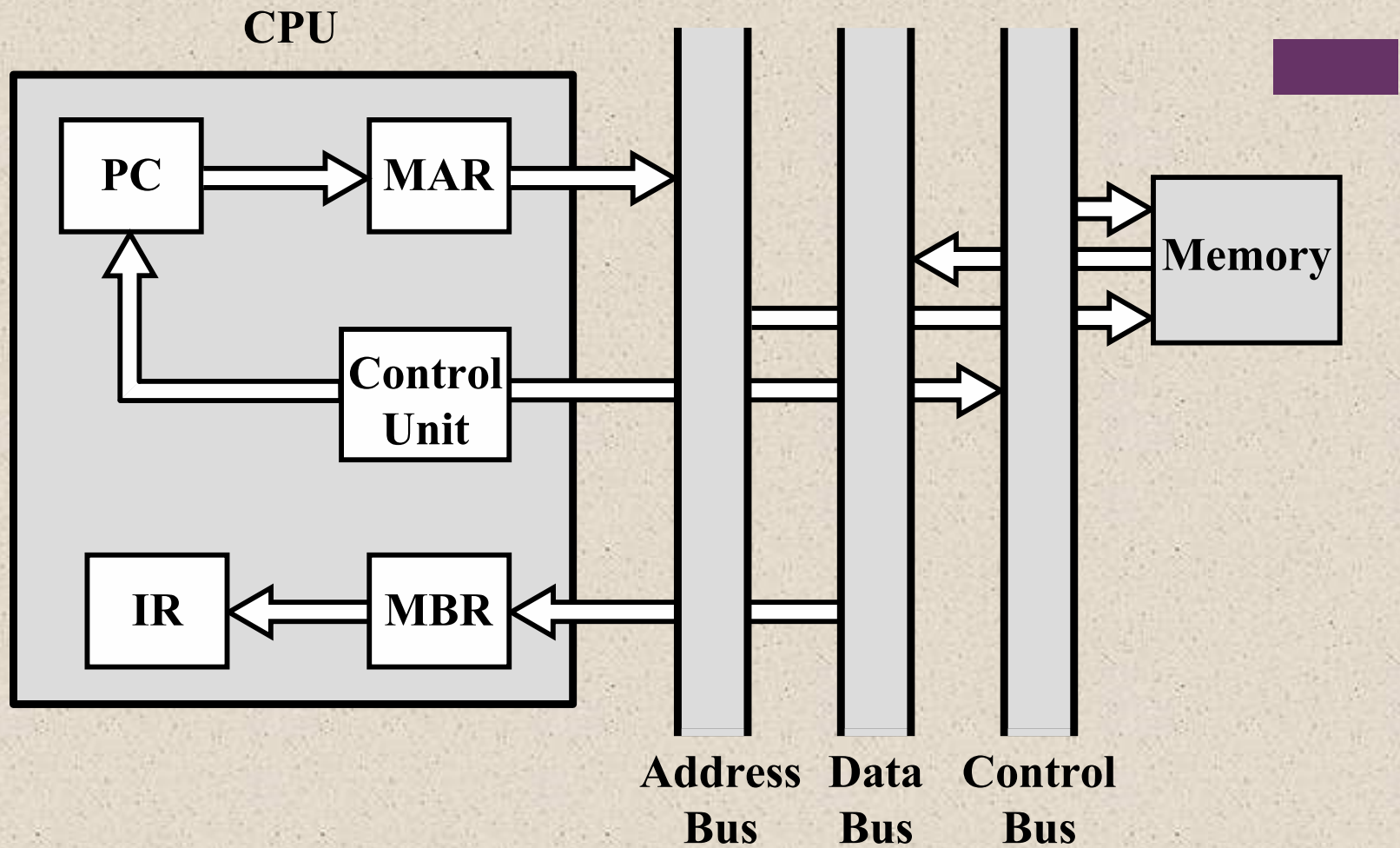
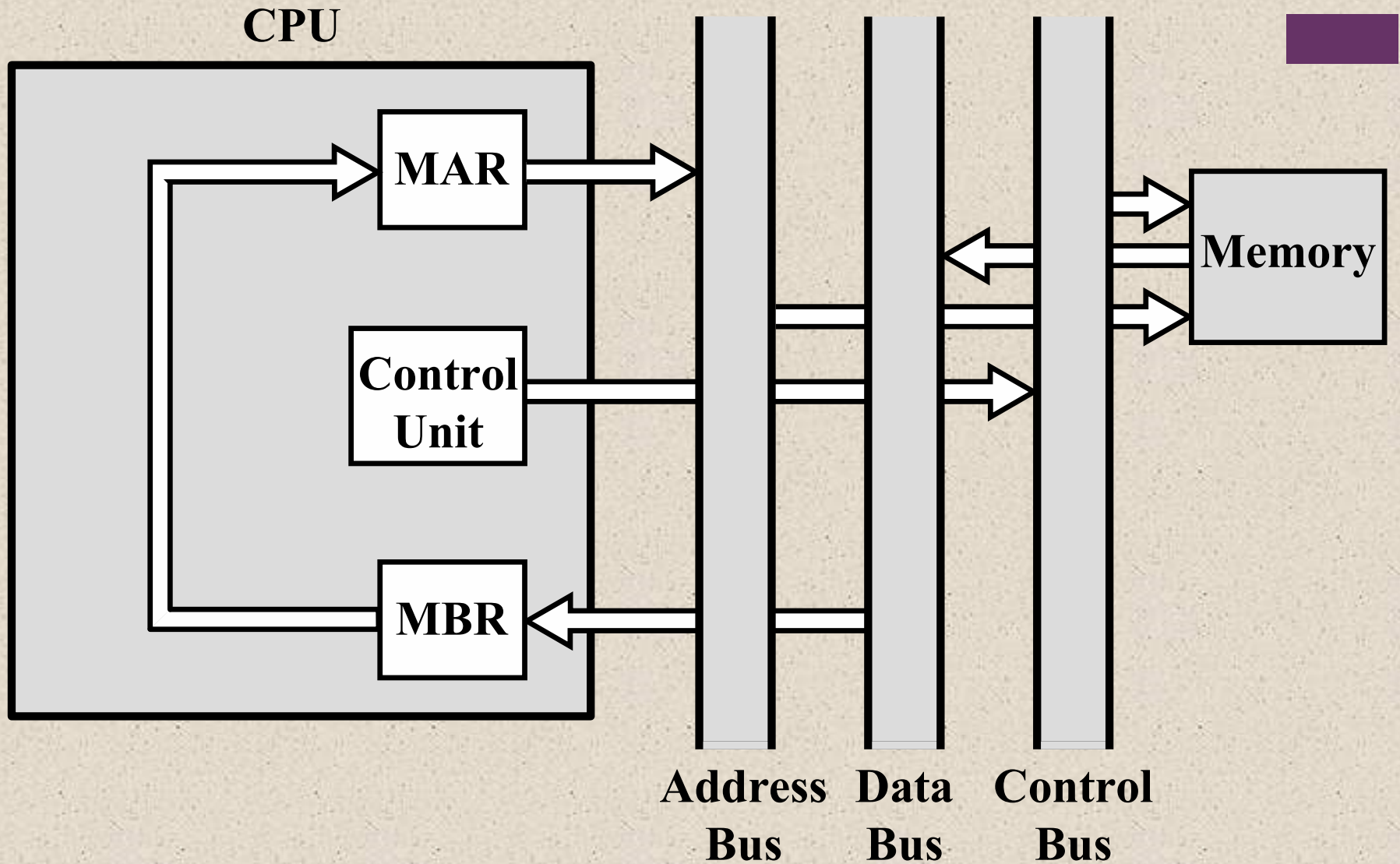


Diagrama de estado del ciclo de instrucción

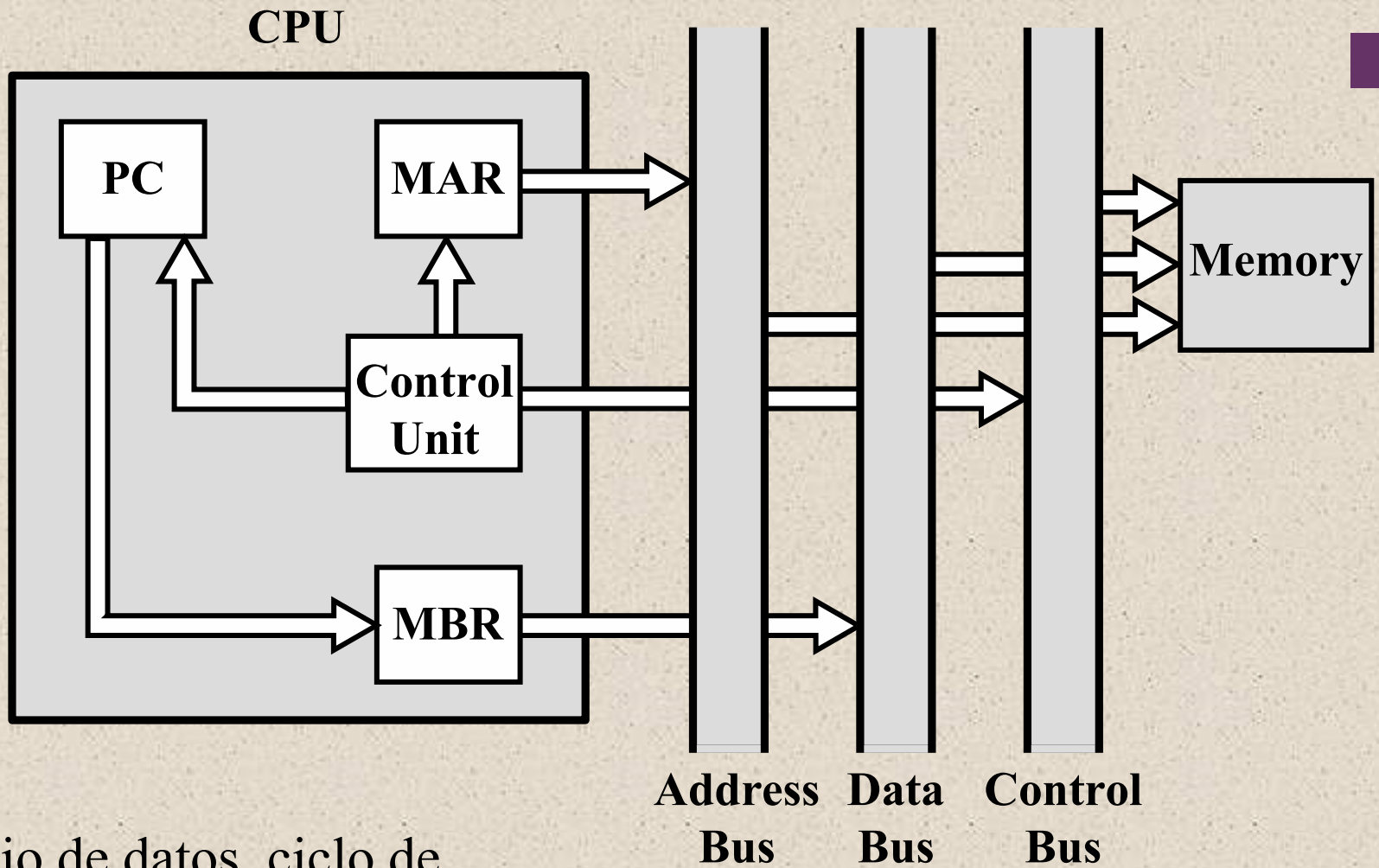


MBR = Memory buffer register
MAR = Memory address register
IR = Instruction register
PC = Program counter

Flujo de datos, ciclo de recuperación



Flujo de datos, ciclo indirecto



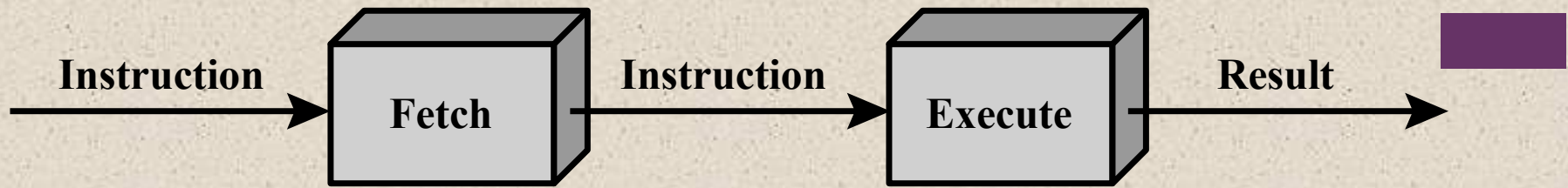
Flujo de datos, ciclo de interrupción

Estrategia de canalizacion (pipelining - entubamiento)

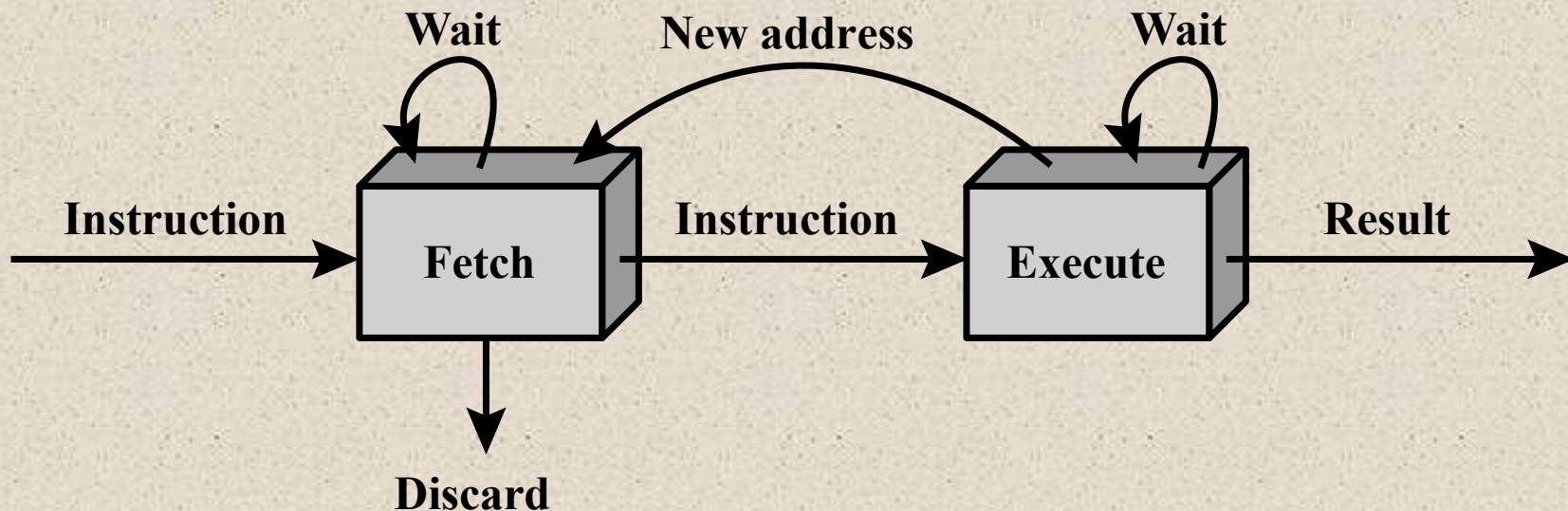
Similar al uso de una
línea de ensamblaje en
una planta de
fabricación.

Para aplicar este
concepto a la
ejecución de
instrucciones,
debemos reconocer
que una instrucción
tiene varias etapas.

Las nuevas entradas se
aceptan en un extremo
antes de que las
entradas previamente
aceptadas aparezcan
como salidas en el otro
extremo



(a) Simplified view



(b) Expanded view

Canalización de instrucciones de dos etapas

+ Etapas adicionales

■ Fetch instruction (FI)

- Lee la siguiente instrucción esperada en un búfer

■ Decode instruction (DI)

- Determina el opcode y los especificadores de operandos.

■ Calculate operands (CO)

- Calcula la dirección efectiva de cada operando fuente
- Esto puede implicar desplazamiento, registro indirecto, indirecto u otras formas de cálculo de direcciones.

■ Fetch operands (FO)

- Obtiene cada operando de la memoria
- Los operandos en los registros no necesitan ser recuperados

■ Execute instruction (EI)

- Realiza la operación indicada y almacena el resultado, si lo hubiera, en la ubicación de operando de destino especificada

■ Write operand (WO)

- Almacena el resultado en la memoria.

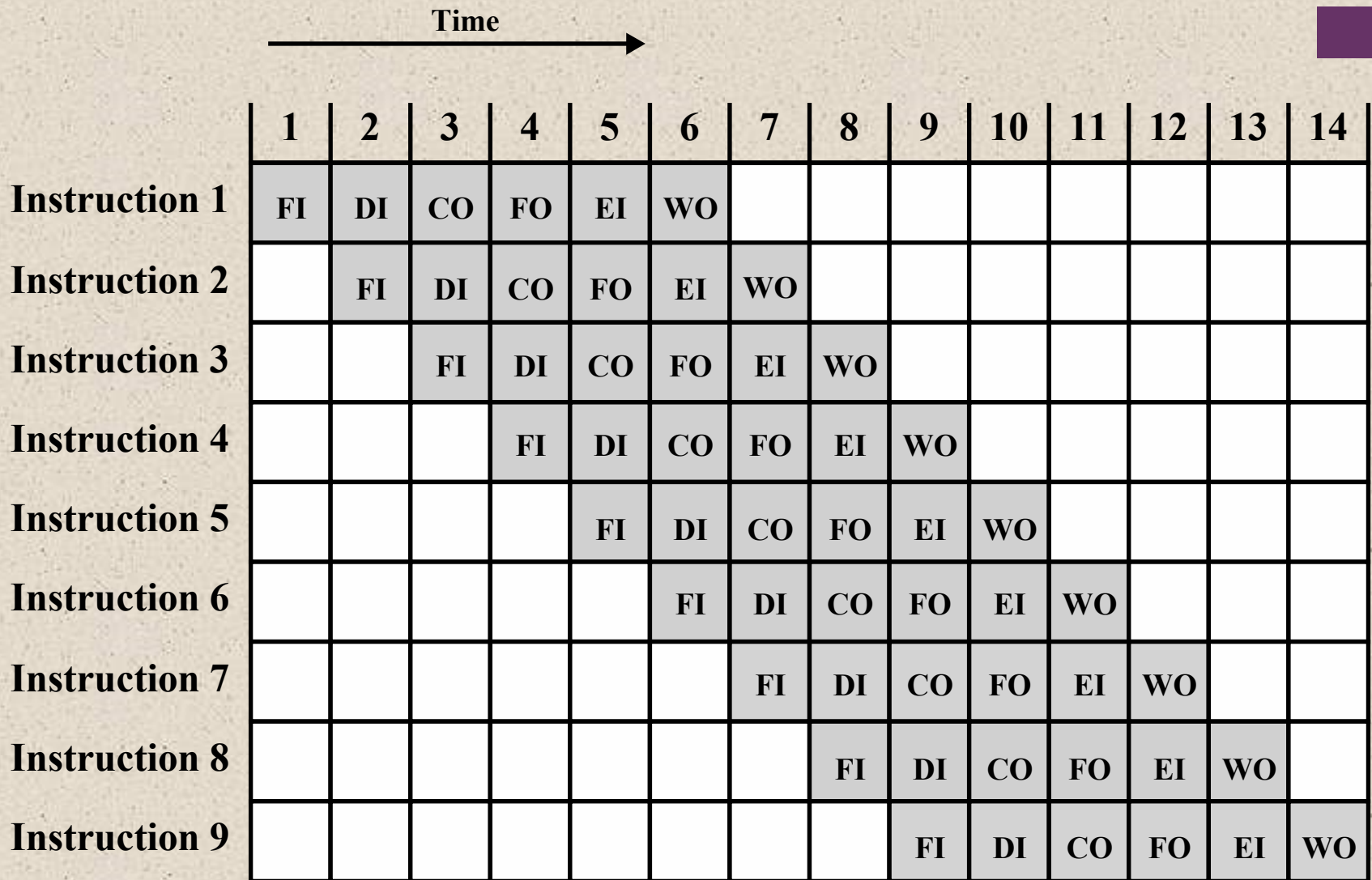
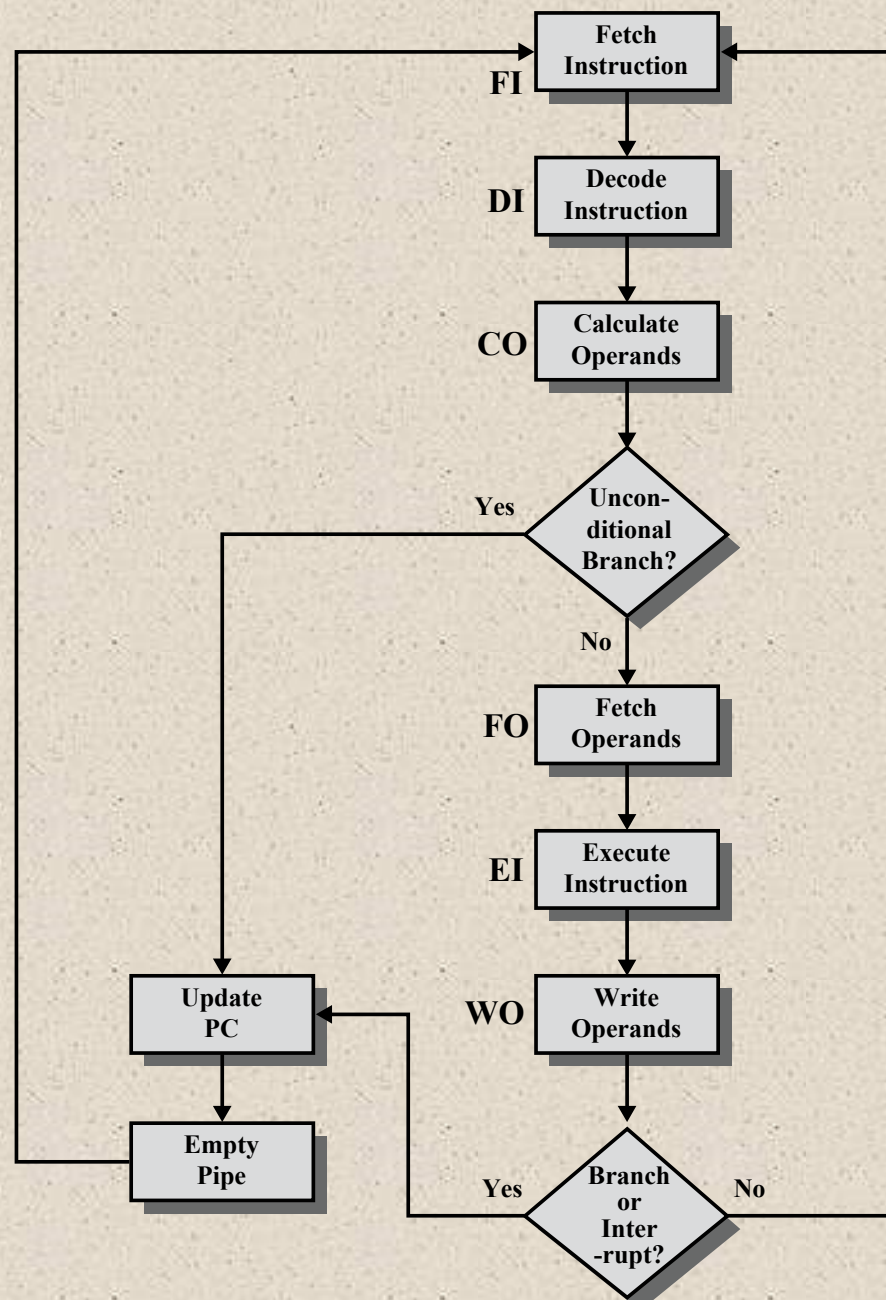


Diagrama de tiempo para la operación de instrucciones canalizadas

	Time →							← Branch Penalty						
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Instruction 1	FI	DI	CO	FO	EI	WO								
Instruction 2		FI	DI	CO	FO	EI	WO							
Instruction 3			FI	DI	CO	FO	EI	WO						
Instruction 4				FI	DI	CO	FO							
Instruction 5					FI	DI	CO							
Instruction 6						FI	DI							
Instruction 7							FI							
Instruction 15								FI	DI	CO	FO	EI	WO	
Instruction 16									FI	DI	CO	FO	EI	WO

Efecto de una rama condicional en la operación de instrucciones canalizadas

Canalización de instrucciones de CPU en seis etapas



Time
↓

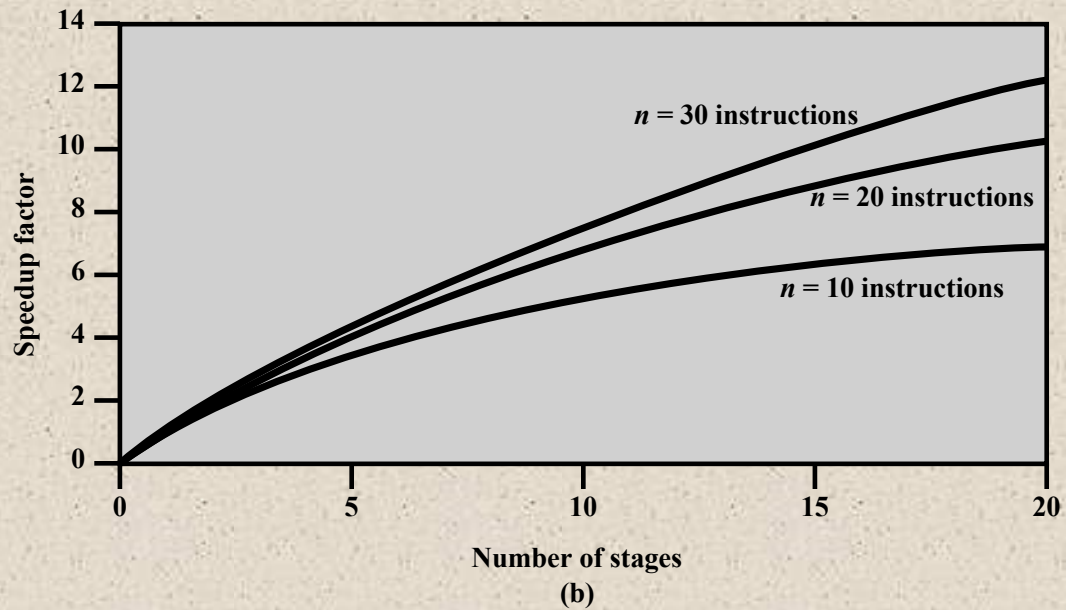
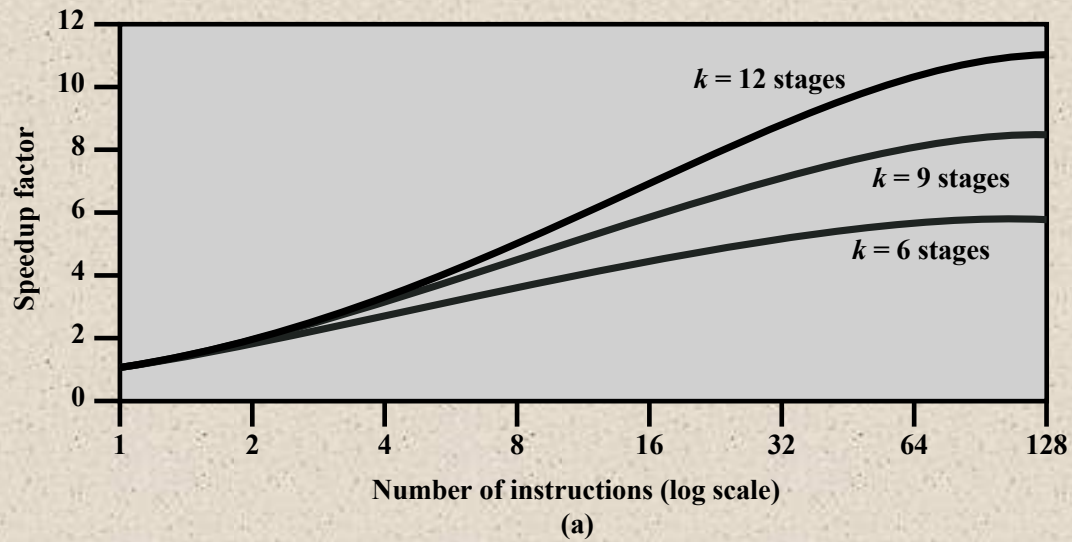
	FI	DI	CO	FO	EI	WO
1	I1					
2	I2	I1				
3	I3	I2	I1			
4	I4	I3	I2	I1		
5	I5	I4	I3	I2	I1	
6	I6	I5	I4	I3	I2	I1
7	I7	I6	I5	I4	I3	I2
8	I8	I7	I6	I5	I4	I3
9	I9	I8	I7	I6	I5	I4
10		I9	I8	I7	I6	I5
11			I9	I8	I7	I6
12				I9	I8	I7
13					I9	I8
14						I9

(a) No branches

	FI	DI	CO	FO	EI	WO
1	I1					
2	I2	I1				
3	I3	I2	I1			
4	I4	I3	I2	I1		
5	I5	I4	I3	I2	I1	
6	I6	I5	I4	I3	I2	I1
7	I7	I6	I5	I4	I3	I2
8	I15					I3
9	I16	I15				
10		I16	I15			
11			I16	I15		
12				I16	I15	
13					I16	I15
14						I16

(b) With conditional branch

Una representación de canalización
alternativa



Factores de aceleración con la canalización de instrucciones

Peligros de la canalización

17/06/2019

Ocurre cuando la tubería, o alguna parte de la tubería, debe detenerse porque las condiciones no permiten la ejecución continua

Hay tres tipos de peligros:

Recurso
Datos
Control

También se conoce como una burbuja de tubería



		Clock cycle								
		1	2	3	4	5	6	7	8	9
Instrucción	I1	FI	DI	FO	EI	WO				
	I2		FI	DI	FO	EI	WO			
	I3			FI	DI	FO	EI	WO		
	I4				FI	DI	FO	EI	WO	

(a) Five-stage pipeline, ideal case

		Clock cycle								
		1	2	3	4	5	6	7	8	9
Instrucción	I1	FI	DI	FO	EI	WO				
	I2		FI	DI	FO	EI	WO			
	I3			Idle	FI	DI	FO	EI	WO	
	I4					FI	DI	FO	EI	WO

(b) I1 source operand in memory

Ejemplo de riesgo de recursos



		Clock cycle									
		1	2	3	4	5	6	7	8	9	10
ADD EAX, EBX		FI	DI	FO	EI	WO					
SUB ECX, EAX			FI	DI	Idle		FO	EI	WO		
I3			FI				DI	FO	EI	WO	
I4							FI	DI	FO	EI	WO

Ejemplo de peligro de datos



Tipos de riesgo de datos



- Leer después de escribir (RAW), o verdadera dependencia (Read after write)
 - Una instrucción modifica un registro o ubicación de memoria.
 - La siguiente instrucción lee los datos en la memoria o la ubicación del registro
 - El peligro ocurre si la lectura tiene lugar antes de que se complete la operación de escritura
- Escribir después de leer (WAR), o antidependencia. (Write after read)
 - Una instrucción lee un registro o ubicación de memoria
 - La instrucción sucesiva escribe a la ubicación
 - El peligro se produce si la operación de escritura se completa antes de que tenga lugar la operación de lectura
- Escritura tras escritura (WAW), o dependencia de salida (Write after write)
 - Dos instrucciones ambas escriben en el mismo lugar.
 - El peligro ocurre si las operaciones de escritura tienen lugar en el orden inverso a la secuencia deseada



Peligro de control

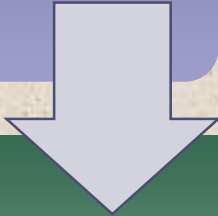


- También conocido como un riesgo de rama
- Ocurre cuando la canalización toma la decisión equivocada sobre una predicción de ramificación
 - Coloca instrucciones en la canalización que debe ser descartada posteriormente
- Tratamiento de las ramificaciones:
 - Múltiples flujos
 - Tomar previamente el objetivo de la ramificación
 - Búfer de bucle
 - Predicción de ramificación
 - Ramificación retrasada



Múltiples flujos

Una canalización simple sufre una penalización por una instrucción de bifurcación porque debe elegir una de las dos instrucciones a continuación y puede tomar la decisión equivocada.



Enfoque de fuerza bruta: replicar las partes iniciales de la canalización y permitir que la canalización obtenga ambas instrucciones, haciendo uso de dos flujos



Inconvenientes:

- Con múltiples tuberías hay demoras de contención para el acceso a los registros y a la memoria
- Las instrucciones adicionales de la bifurcación pueden ingresar a la canalización antes de que se resuelva la decisión original

Tomar previamente el objetivo de la ramificación

- Cuando se reconoce una rama condicional, el destino de la rama se recupera previamente, además de la instrucción que sigue a la rama
- El destino se guarda hasta que se ejecuta la instrucción de bifurcación.
- Si se toma la bifurcación, el objetivo ya ha sido pre-recuperado
- La IBM 360/91 utiliza este enfoque





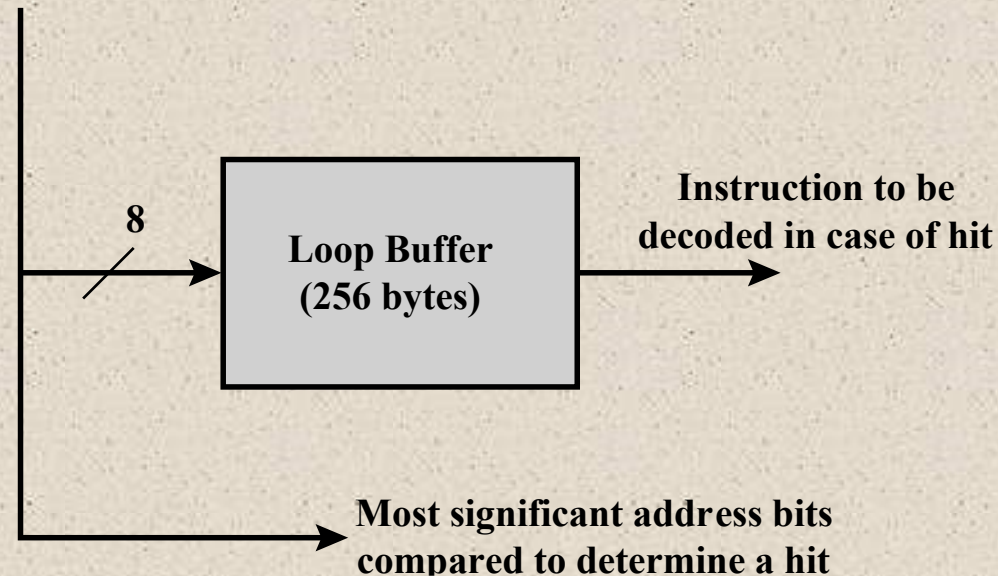
Buffer de bucle

- Pequeña memoria de velocidad muy alta, mantenida por la etapa de obtención de instrucciones de canalización. Contiene las n instrucciones más recientes recuperadas, en secuencia

- **Beneficios:**

- Las instrucciones estarán disponibles sin el tiempo habitual de acceso a la memoria.
- Si se produce una derivación a un objetivo solo unas pocas ubicaciones antes de la dirección de la instrucción de derivación, el objetivo ya estará en el búfer
- Esta estrategia es particularmente adecuada para tratar con bucles.
- En principio similar a un caché dedicado a las instrucciones.
 - Diferencias
 - El búfer de bucle solo retiene instrucciones en secuencia
 - Es mucho más pequeño en tamaño y por lo tanto más bajo en costo

Branch address





Predicción de ramificación



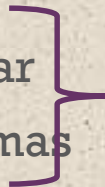
- Se pueden usar varias técnicas para predecir si se tomará una rama. Predecir:

1. Nunca se tomará
2. Siempre se tomará
3. Mediante opcode

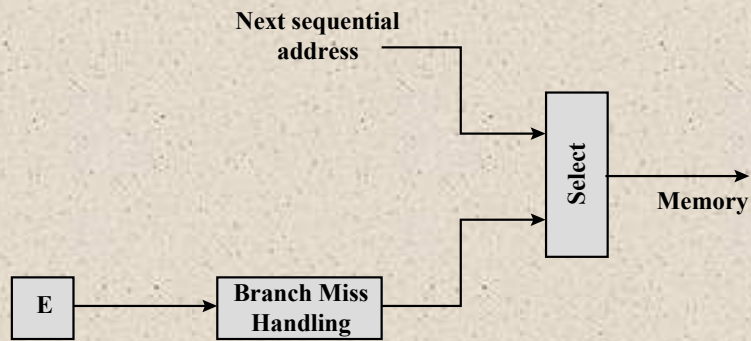


- Estos enfoques son estáticos.
- No dependen del historial de ejecución hasta el momento de la instrucción de rama condicional

1. Switch Toma/no tomar
2. Tabla histórica de ramas

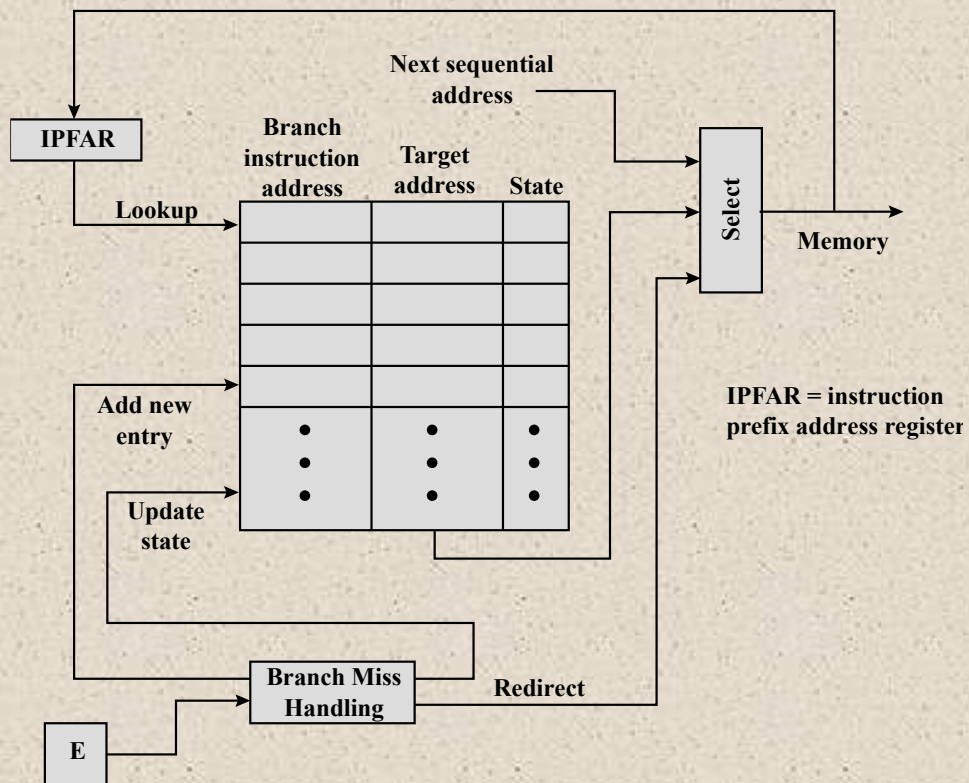


- Estos enfoques son dinámicos.
- Dependen del historial de ejecución.



(a) Predict never taken strategy

Tratamiento de bifurcaciones

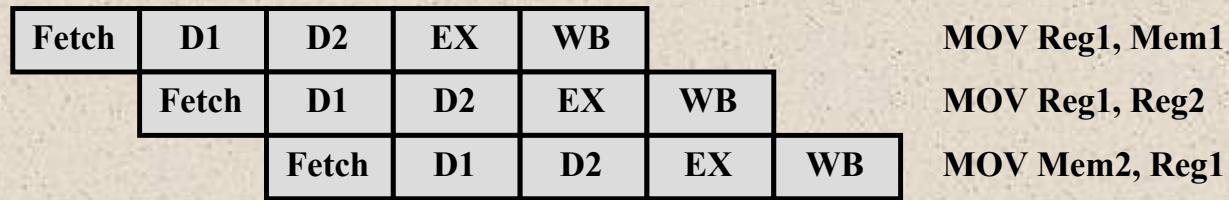


(b) Branch history table strategy

Canalización en Intel 80486



Ejemplo de canalización de instrucción 80486



(a) No Data Load Delay in the Pipeline



(b) Pointer Load Delay



(c) Branch Instruction Timing

(a) Integer Unit in 32-bit Mode

Type	Number	Length (bits)	Purpose
General	8	32	General-purpose user registers
Segment	6	16	Contain segment selectors
EFLAGS	1	32	Status and control bits
Instruction Pointer	1	32	Instruction pointer

(b) Integer Unit in 64-bit Mode

Type	Number	Length (bits)	Purpose
General	16	32	General-purpose user registers
Segment	6	16	Contain segment selectors
RFLAGS	1	64	Status and control bits
Instruction Pointer	1	64	Instruction pointer

(c) Floating-Point Unit

Type	Number	Length (bits)	Purpose
Numeric	8	80	Hold floating-point numbers
Control	1	16	Control bits
Status	1	16	Status bits
Tag Word	1	16	Specifies contents of numeric registers
Instruction Pointer	1	48	Points to instruction interrupted by exception
Data Pointer	1	48	Points to operand interrupted by exception

Registros del procesador X86

8 General: se pueden usar para todos los tipos de instrucciones x86

6 de Segmento: contiene selectores de segmento, que se indexan en tablas de segmentos

1 Bandera: EFLAGS contiene códigos de condición y varios bits de modo.

Puntero de instrucción: contiene la dirección de la instrucción actual.

Otros para punto flotante



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	I D	V I P	V I F	A C	V M	R F	0	N T	I O P L	O F	D F	I F	T F	S F	Z F	0	A F	0	P F	1	C F	

X ID = Identification flag
X VIP = Virtual interrupt pending
X VIF = Virtual interrupt flag
X AC = Alignment check
X VM = Virtual 8086 mode
X RF = Resume flag
X NT = Nested task flag
X IOPL = I/O privilege level
S OF = Overflow flag

C DF = Direction flag
X IF = Interrupt enable flag
X TF = Trap flag
S SF = Sign flag
S ZF = Zero flag
S AF = Auxiliary carry flag
S PF = Parity flag
S CF = Carry flag

S Indicates a Status Flag
C Indicates a Control Flag
X Indicates a System Flag
Shaded bits are reserved

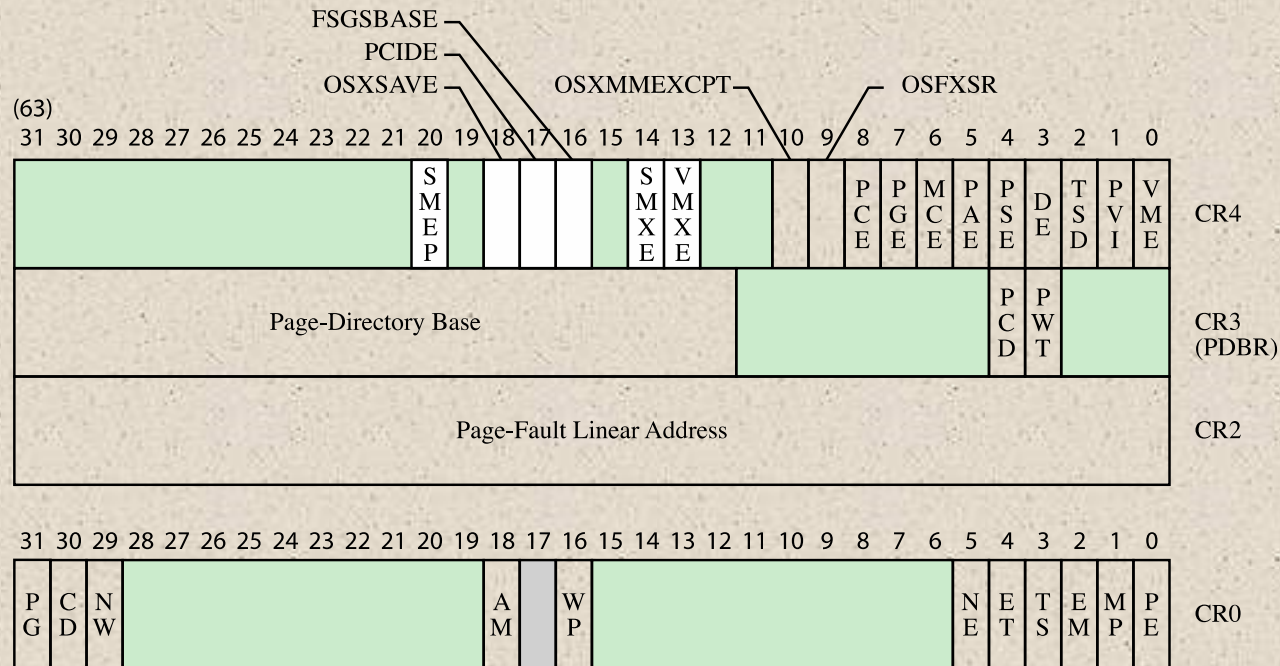
Registro EFLAGS x86

Indica el estado del procesador y ayuda a controlar su funcionamiento.

Incluye los seis códigos de condición (acarreo, paridad, auxiliar, cero, signo, desbordamiento), que informan los resultados de una operación de entero.

Además, hay bits en el registro que pueden denominarse bits de control:

Los 4 registros de control x86 .



shaded area indicates reserved bits

OSXSAVE	=	XSAVE enable bit	VME	=	Virtual 8086 Mode Extensions
PCIDE	=	Enables process-context identifiers	PCD	=	Page-level Cache Disable
FSGSBASE	=	Enables segment base instructions	PWT	=	Page-level Writes Transparent
SMXE	=	Enable Safer mode extensions	PG	=	Paging
VMXE	=	Enable virtual machine extensions	CD	=	Cache Disable
OSXMMEXCPT	=	Support unmasked SIMD FP exceptions	NW	=	Not Write Through
OSFXSR	=	Support FXSAVE, FXSTOR	AM	=	Alignment Mask
PCE	=	Performance Counter Enable	WP	=	Write Protect
PGE	=	Page Global Enable	NE	=	Numeric Error
MCE	=	Machine Check Enable	ET	=	Extension Type
PAE	=	Physical Address Extension	TS	=	Task Switched
PSE	=	Page Size Extensions	EM	=	Emulation
DE	=	Debug Extensions	MP	=	Monitor Coprocessor
TSD	=	Time Stamp Disable	PE	=	Protection Enable
PVI	=	Protected Mode Virtual Interrupt			



Procesamiento de interrupciones

Interrupciones y excepciones

■ Interrupciones

- Generado por una señal del hardware y puede ocurrir en momentos aleatorios durante la ejecución de un programa
- Enmascarable
- No enmascarable

■ Excepciones

- Generado a partir de software y es provocado por la ejecución de una instrucción.
- Procesador detectado
- Programado

■ Tabla de vectores de interrupción

- A cada tipo de interrupción se le asigna un número.
- El número se usa para indexar en la tabla de vectores de interrupción

Vector Number	Description
0	Divide error; division overflow or division by zero
1	Debug exception; includes various faults and traps related to debugging
2	NMI pin interrupt; signal on NMI pin
3	Breakpoint; caused by INT 3 instruction, which is a 1-byte instruction useful for debugging
4	INTO-detected overflow; occurs when the processor executes INTO with the OF flag set
5	BOUND range exceeded; the BOUND instruction compares a register with boundaries stored in memory and generates an interrupt if the contents of the register is out of bounds.
6	Undefined opcode
7	Device not available; attempt to use ESC or WAIT instruction fails due to lack of external device
8	Double fault; two interrupts occur during the same instruction and cannot be handled serially
9	Reserved
10	Invalid task state segment; segment describing a requested task is not initialized or not valid
11	Segment not present; required segment not present
12	Stack fault; limit of stack segment exceeded or stack segment not present
13	General protection; protection violation that does not cause another exception (e.g., writing to a read-only segment)
14	Page fault
15	Reserved
16	Floating-point error; generated by a floating-point arithmetic instruction
17	Alignment check; access to a word stored at an odd byte address or a doubleword stored at an address not a multiple of 4
18	Machine check; model specific
19-31	Reserved
32-255	User interrupt vectors; provided when INTR signal is activated

Tabla de vectores de interrupción y excepción x86

Sin sombrear: excepciones

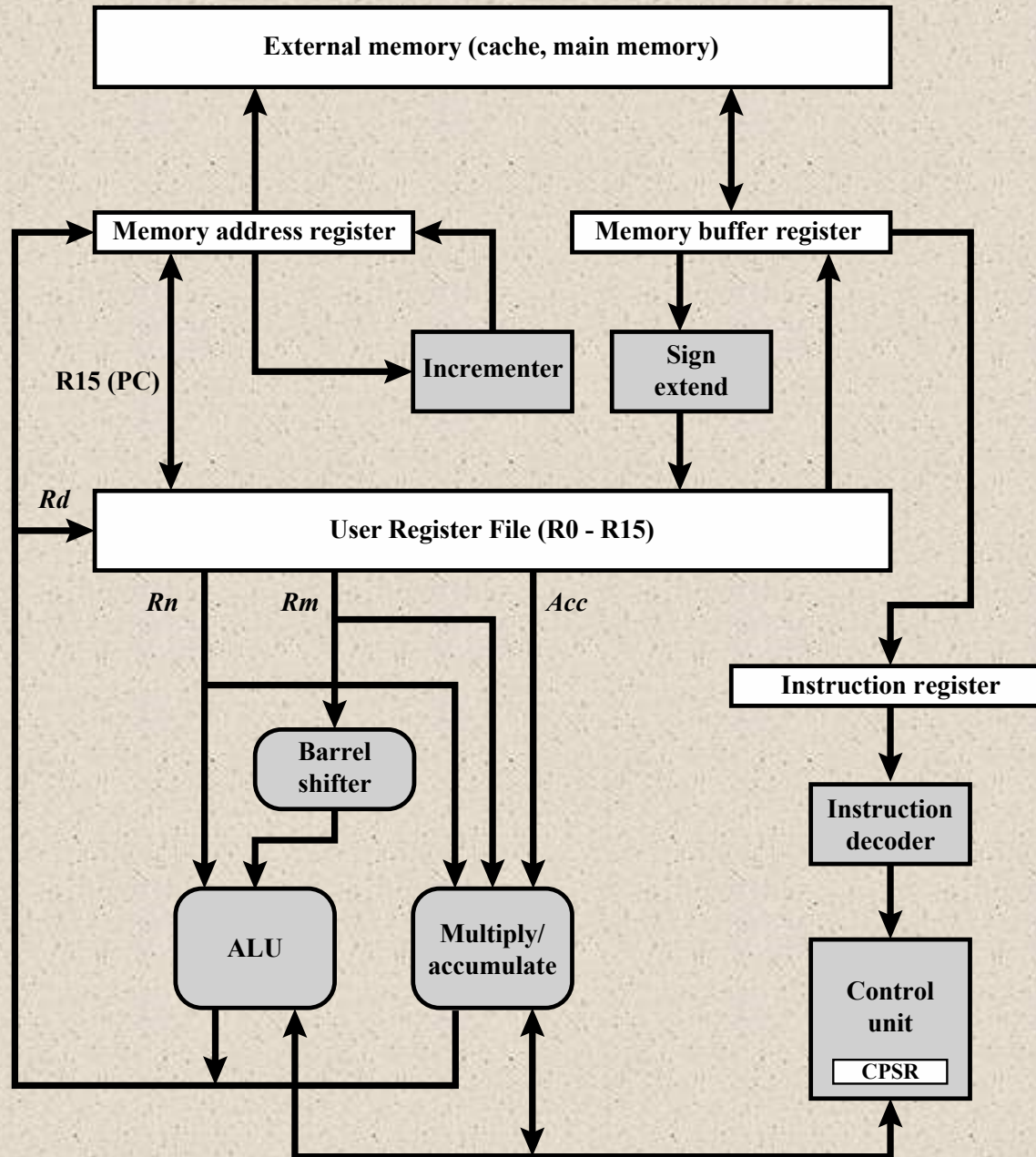
Sombreado: interrupciones

+ El procesador ARM

ARM es un sistema RISC con los siguientes atributos:

- Arreglo moderado de registros uniformes.
- Un modelo de carga / almacenamiento de procesamiento de datos en el que las operaciones solo se realizan en operandos en registros
- Una instrucción de longitud fija uniforme de 32 bits para el conjunto estándar y 16 bits para un conjunto pequeño de instrucciones
- Unidad lógica aritmética separada (ALU) y unidades de desplazamiento
- Un pequeño número de modos de direccionamiento con todas las direcciones de carga / almacenamiento determinadas a partir de registros y campos de instrucción
- Los modos de direccionamiento de incremento automático y decremento automático se utilizan para mejorar el funcionamiento de los bucles de programa
- La ejecución condicional de las instrucciones minimiza la necesidad de instrucciones de derivación condicionales, lo que mejora la eficiencia de la canalización, ya que se reduce el desalojo de la canalización.

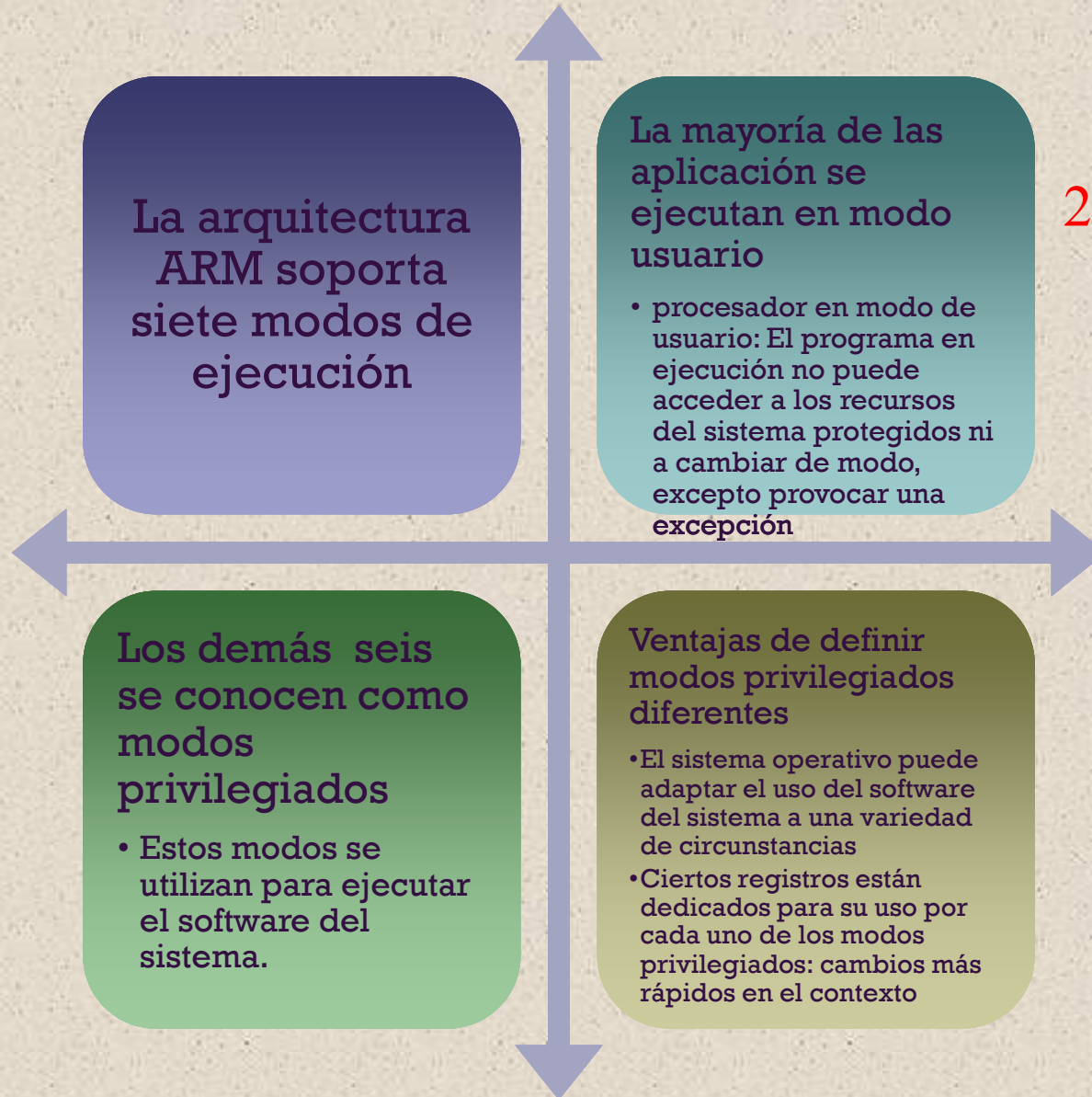
Organización ARM simplificada



Modos de procesador



24-06-2019



Modos de excepción





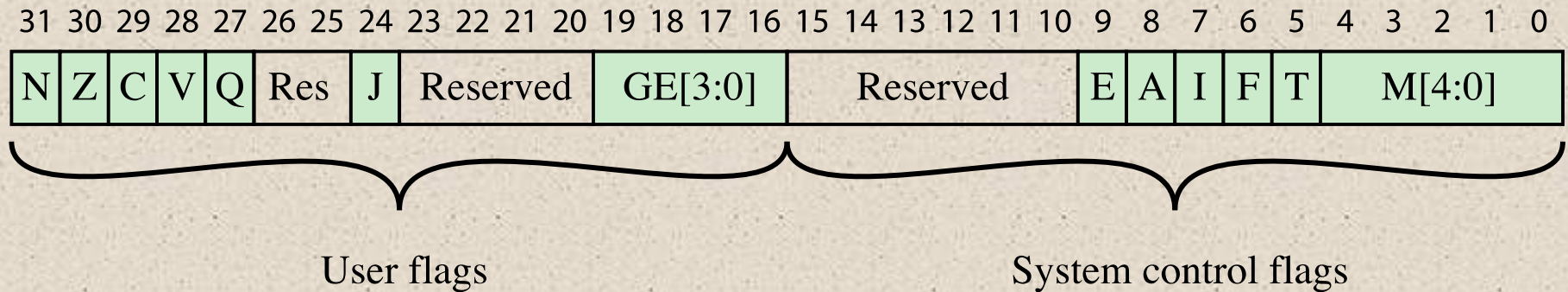
Modes						
		Privileged modes				
		Exception modes				
User	System	Supervisor	Abort	Undefined	Interrupt	Fast Interrupt
R0	R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7	R7
R8	R8	R8	R8	R8	R8	R8_fiq
R9	R9	R9	R9	R9	R9	R9_fiq
R10	R10	R10	R10	R10	R10	R10_fiq
R11	R11	R11	R11	R11	R11	R11_fiq
R12	R12	R12	R12	R12	R12	R12_fiq
R13 (SP)	R13 (SP)	R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
R14 (LR)	R14 (LR)	R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)

registros
visibles por el
usuario para
ARM

CPSR	CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
© 2016 Pearson Education, Inc..		SPSR_svc	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

SP = puntero de pila
CPSR = registro de estado del
programa actual
LR = registro de enlace
SPSR = registro de estado del
programa guardado
PC = contador de programa

Formato de CPSR y SPSR ARM



SP = puntero de pila

CPSR = registro de estado del programa actual

LR = registro de enlace

SPSR = registro de estado del programa guardado

PC = contador de programa



Vector de interrupciones ARM

Exception type	Mode	Normal entry address	Description
Reset	Supervisor	0x00000000	Occurs when the system is initialized.
Data abort	Abort	0x00000010	Occurs when an invalid memory address has been accessed, such as if there is no physical memory for an address or the correct access permission is lacking.
FIQ (fast interrupt)	FIQ	0x0000001C	Occurs when an external device asserts the FIQ pin on the processor. An interrupt cannot be interrupted except by an FIQ. FIQ is designed to support a data transfer or channel process, and has sufficient private registers to remove the need for register saving in such applications, therefore minimizing the overhead of context switching. A fast interrupt cannot be interrupted.
IRQ (interrupt)	IRQ	0x00000018	Occurs when an external device asserts the IRQ pin on the processor. An interrupt cannot be interrupted except by an FIQ.
Prefetch abort	Abort	0x0000000C	Occurs when an attempt to fetch an instruction results in a memory fault. The exception is raised when the instruction enters the execute stage of the pipeline.
Undefined instructions	Undefined	0x00000004	Occurs when an instruction not in the instruction set reaches the execute stage of the pipeline.
Software interrupt	Supervisor	0x00000008	Generally used to allow user mode programs to call the OS. The user program executes a SWI instruction with an argument that identifies the function the user wishes to perform.

+ Resumen

Capítulo 4

- Organización del procesador
- Organización de registro
 - Registros visibles por el usuario
 - Registros de control y estado.
- Ciclo de instrucción
 - Ciclo indirecto
 - Flujo de datos
- Familia de procesadores x86.
 - Organización de registro
 - Procesamiento de interrupciones

Estructura y función del procesador

- Tubería de instrucciones
 - Estrategia de canalización
 - Rendimiento de la tubería
 - Peligros de la tubería
 - Tratar con las ramas
 - Tubería Intel 80486
- El procesador ARM
 - Organización del procesador
 - Modos de procesador
 - Organización de registro
 - Procesamiento de interrupciones