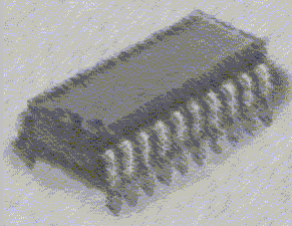


## Tema 4. La Unidad de Control

### Arquitectura de Computadores



I. T. Informática de Gestión

Curso 2009-2010

Tema 4:

Transparencia: 2 / 73

La Unidad de Control

### Índice

- Operaciones elementales
- Computador elemental
  - Memoria principal
  - Banco de registros
  - Unidad aritmético-lógica
  - Unidad de direccionamiento
  - Unidad de control
- Temporización de las señales de control
- Ejecución de instrucciones
- Diseño de la Unidad de Control: cableado vs. Microprogramado
- Microprogramación y nanoprogramación: Motorola 68000
- Excepciones en el computador
- Arranque del computador
- Bibliografía



Departamento de Automática  
Área de Arquitectura y Tecnología de Computadores

Arquitectura de Computadores  
I. T. Informática de Gestión

## Operaciones elementales (I)

- **La unidad de control tiene como función básica la ejecución la secuencia siguiente:**
  - Tomar la instrucción apuntada por el CP de la memoria principal (fase de *fetch*)
  - Decodificar la instrucción leída
  - Ejecutar la instrucción
  - Actualizar debidamente el CP
- **La ejecución de una instrucción está gobernada por un contador de periodos, durante los cuales, la UC genera las señales de control necesarias en función de la información de que dispone en:**
  - La propia instrucción
  - El registro de estado
  - Señales de E/S (interrupciones, DMA, *reset*, etc.)

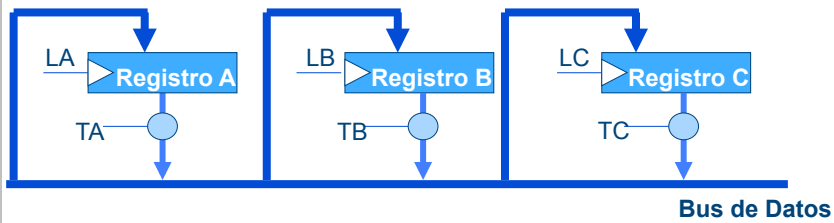


## Operaciones elementales (II)

- La ejecución de cada instrucción requiere realizar una serie de pequeños pasos mediante señales de control; estos pasos se llaman operaciones elementales
- Las operaciones elementales que puede realizar todo sistema computador se clasifican en los grupos siguientes:
  - **Operaciones de transferencia.** Mover información de un elemento a otro
  - **Operaciones de proceso.** La información origen pasa a través de un operador
- Todas las operaciones elementales, ya sean de transferencia o de proceso comienzan en un elemento de almacenamiento y terminan en otro



## Operaciones elementales (III) Transferencia



Operación de transferencia: **MOV RegB, RegA**

- Volcar el contenido del Registro A al Bus de Datos
- Cargar el contenido del Bus de Datos en Registro B



## Operaciones elementales (y IV) Proceso



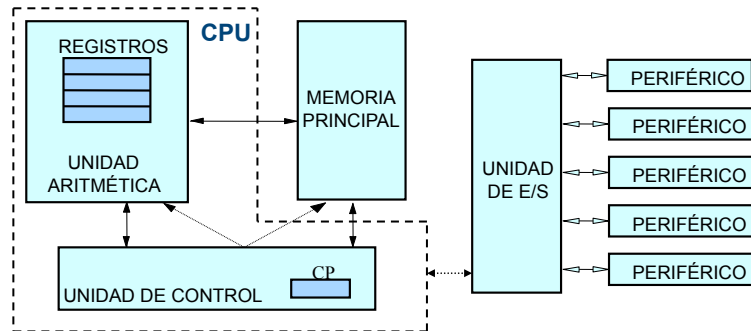
Operación de proceso: **XOR RC, RA, RB**

- Llevar el Registro A a la ALU
- Llevar el Registro B a la ALU
- Escoger la operación XOR
- Cargar el resultado en el Registro C



## Estructura computador elemental (I)

- Estudiaremos las señales de control que genera una Unidad de Control por medio del empleo de una máquina simplificada que siga la arquitectura de Von Neumann

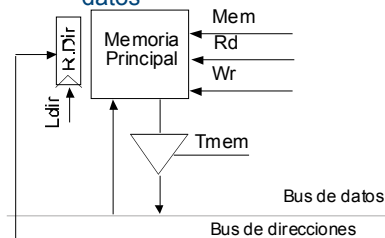


## Estructura computador elemental (II) Memoria Principal (I)

- El bloque de memoria principal consta:
  - Pastilla/s de memoria RAM
  - Registro de direcciones
  - Buffer bidireccional al bus de datos

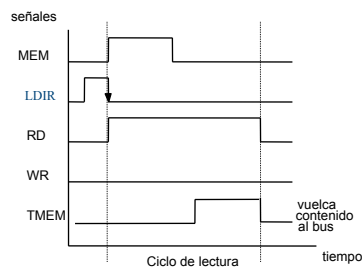
Las señales de control son:

Señal	Activa por	Nombre	Significado
MEM	Nivel	Memory	Iniciar ciclo de memoria
RD	Nivel	Memory read	Ciclo de lectura
WR	Nivel	Memory write	Ciclo de escritura
LDIR	Flanco	Address latch enable	Carga el registro de direcciones
TMEM	Nivel	---	A nivel alto los datos se vuelcan al bus

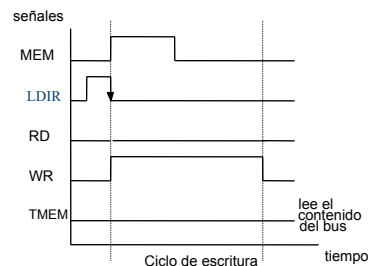


## Estructura computador elemental (III) Memoria Principal (y II)

- La duración de la señales RD y WR depende de la memoria, es decir, de sus tiempos de lectura y escritura
- La duración de MEM será la mínima posible, es decir, la correspondiente al periodo básico de la máquina



Cronograma del ciclo de lectura

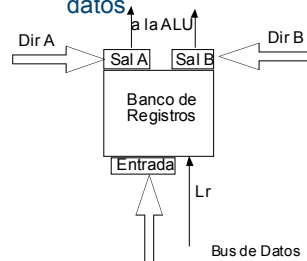


Cronograma del ciclo de escritura



## Estructura computador elemental (IV) Banco de Registros

- El banco de registros consta de los siguientes elementos:
  - 16 registros de propósito general accesibles al usuario desde el ensamblador o lenguaje máquina
  - 2 puertas de salida que permiten leer 2 registros simultáneamente
  - 1 puerta de entrada que permite cargar un registro desde el bus de datos



La señal de control es la siguiente:

Señal	Activa por	Nombre	Significado
LR	Flanco	Load register	Carga el registro indicado por DIRECCIÓN A con un dato del bus de datos

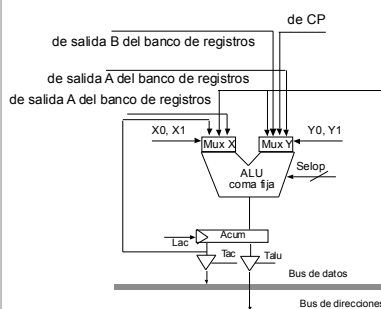


## Estructura computador elemental (V) Unidad Aritmético-Lógica (I)

- La unidad aritmético-lógica consta de los siguientes elementos:
  - Un operador con 4 señales de control para selección de la operación a realizar
  - 2 multiplexores de 4 entradas y 1 salida, que seleccionan entre los posibles operandos
  - Un registro acumulador que permite almacenar resultados intermedios
  - La salida del registro acumulador, puede transferirse al multiplexor X, al bus de datos o al bus de direcciones, según indiquen las señales de control



## Estructura computador elemental (VI) Unidad Aritmético-Lógica (y II)



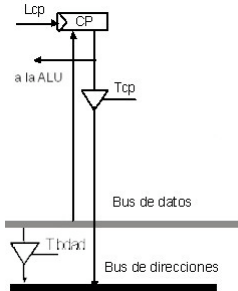
Las señales de control son

Señal	Activa por	Nombre	Significado
Selop	Nivel	Selection	Selecciona la operación a realizar
X0, X1	Nivel	---	Selecciona uno entre 3 operandos
Y0, Y1	Nivel	---	Selecciona uno entre 4 operandos
LAC	Flanco	Load acumulador	Carga el acumulador con la salida del operador
TAC	Nivel	Data transfer	Transfiere el acumulador al bus de datos
TALU	Nivel	Address transfer	Transfiere el contenido del acumulador al bus de direcciones



## Estructura computador elemental (VII) Unidad de Direccionamiento

- La unidad de direccionamiento se encarga de generar las direcciones de memoria
- El CP debe actualizarse cada vez que se ejecuta una instrucción, incrementándose según la dirección de la siguiente instrucción
- El incremento vendrá dado por la información disponible en el bus de datos



Señal	Activa por	Nombre	Significado
LPC	Flanco	<i>Load program counter</i>	Carga una dirección nueva en el CP desde el bus de datos
TBDAD	Nivel	<i>Data bus transfer</i>	Transfiere el contenido del bus de datos al bus de direcciones
TCP	Nivel	<i>Program counter transfer</i>	Transfiere el contenido del registro CP al bus de direcciones

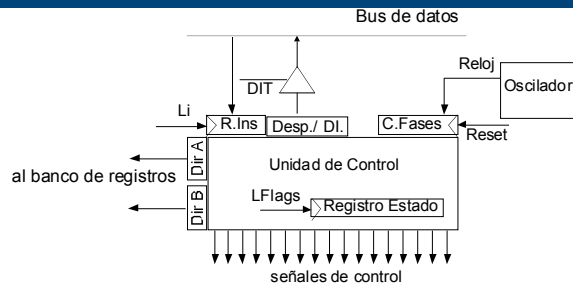


## Estructura computador elemental (VIII) Unidad de Control

- La UC genera todas las señales que forman el bus de control
- Además, necesita de:
  - **Registro de instrucciones**, donde almacena varios bytes de código máquina correspondiente a una instrucción
  - **Puerto de salida**, que vuelca en el bus de datos los datos inmediatos o desplazamientos que llevan asociados algunas instrucciones
  - **Registro de estado**, que contiene los flags. Este registro se carga selectivamente con varias señales de control activas por flanco y provenientes de la ALU
  - **Registro contador de fases**, gobernado por el reloj del sistema y que se puede poner a 0 con la señal RESET
  - **Dos buses de 4 bits**, que sirven para generar las señales de selección de registro



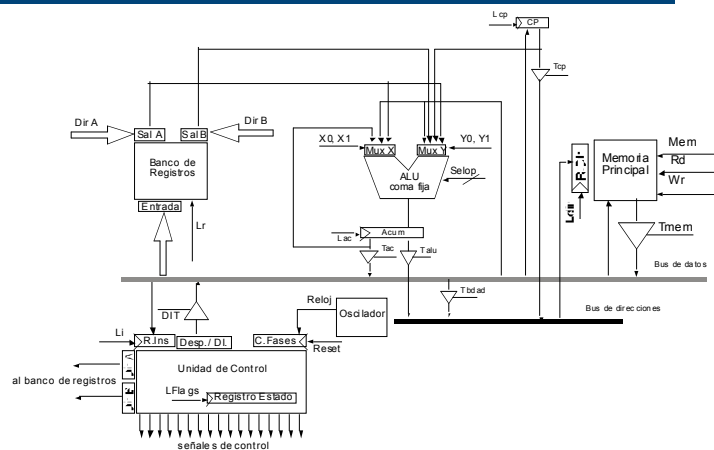
## Estructura computador elemental (VIII) Unidad de Control



Señal	Activa por	Nombre	Significado
LI	Flanco	<i>Load instruction</i>	Carga una instrucción desde el bus de datos
DIT	Nivel	<i>Data inst. transfer</i>	Transfiere un dato inmediato o un desplazamiento al bus de datos
LFlags	Flanco	<i>Load flags</i>	Carga los <i>flags</i> después de una instrucción aritmética o lógica
RESET	Flanco	<i>Reset</i>	Pone a 0 el contador de fases



## Estructura computador elemental (y IX)





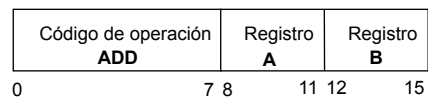
## Temporalización

- El comportamiento de un computador es síncrono. Está gobernado por un reloj
- La ejecución de una instrucción se divide en:
  - 1. Fase de búsqueda de la instrucción o fase de fetch
  - 2. Decodificar la instrucción
  - 3. Ejecución de la instrucción y en caso necesario:
    - 3.1 Leer los operandos
    - 3.2 Realizar la operación
    - 3.3 Almacenar el resultado
    - 3.4 Actualizar el registro de estado
- Habrá que actualizar el CP para que apunte a la instrucción siguiente a ejecutar
- Cada una de las fases se realiza en un número determinado de periodos de reloj



## Ejecución de instrucciones (I) ADD A, B (I)

- Instrucción de suma con direccionamiento a registro
- El formato de la instrucción es el siguiente:



- La instrucción ocupa dos bytes
- El significado de la instrucción es  $A := A + B$ ;



## Ejecución de instrucciones (II)

### ADD A, B (II)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
<b>Traer instrucción:</b>		
⊙ Direccionar	CP ⊗ bus direcciones Cargar registro de direcciones	TCP LDIR
⊙ Leer instrucción	Ciclo de memoria Lectura Abrir buffer al bus de datos Cargar instrucción en la Unidad de Control (UC)	MEM RD TMEM LI
<b>Decodificación:</b>		
Retardo (1 periodo)		
<b>Incrementar CP:</b>		
⊙ Sumar CP+2	La UC entrega el 2 al bus de datos Selección de operandos y de operación (simultáneamente)	DIT X0, X1 (B.Datos), Y0, Y1 (CP), Selop (suma)
	Carga del acumulador	LAC
⊙ Cargar nuevo CP	Transferencia al bus de datos Carga del nuevo CP	TAC LPC



## Ejecución de instrucciones (III)

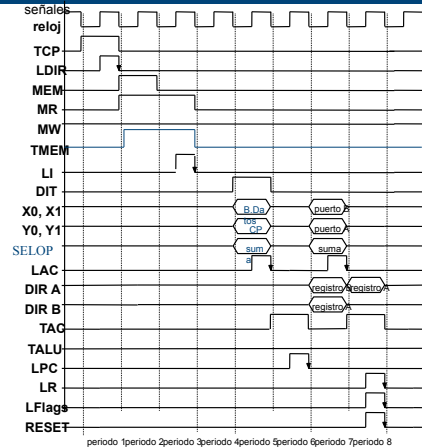
### ADD A, B (III)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
<b>Realizar suma:</b>		
⊙ Obtener direcciones registros	La UC genera las direcciones de los registros A y B	DIR A (registro B), DIR B (registro A)
⊙ Realizar la suma	Selección de operandos y de operación (simultáneamente) Carga del acumulador	X0, X1 (salida B), Y0, Y1 (salida A), Selop (suma) LAC
<b>Guardar resultado:</b>		
⊙ Transferir dato del acumulador	Transferencia al bus de datos	TAC
⊙ Cargar registro A	La UC genera dirección del registro A Señal de carga	DIR A (registro A) LR
<b>Actualizar registro de estado:</b>		
		LFlags
<b>Poner a 0 el contador de fases:</b>		
		RESET



## Ejecución de instrucciones (IV)

### ADD A, B (IV)



## Ejecución de instrucciones (V)

### ADD A, B (y V)

- **Comentarios y mejoras:**

- La fase de fetch solamente trae 1 byte si el tamaño del bus de datos es de 1 byte; en ese caso tendríamos que realizar dos accesos a memoria para traer la instrucción entera
- El periodo 4 se "pierde" en decodificar la instrucción; es parte del tiempo de ejecución de la instrucción
- Los periodos 5 y 6 de actualización del CP se podrían ahorrar con un sumador aparte para la Unidad de direccionamiento
- Igualmente, la actualización de CP llevaría un periodo menos si los resultados de la ALU se volcaran directamente al bus de datos o direcciones sin pasar por el acumulador
- Se puede optimizar el funcionamiento si hacemos que antes de terminar de ejecutar la instrucción se empiece la siguiente fase de fetch
- Hay que tener cuidado con los posibles conflictos en los buses



## Ejecución de instrucciones (VI)

### MOV A, [B + 1234h] (I)

- Instrucción de transferencia con direccionamiento relativo a registro
- El formato de la instrucción es el siguiente:

Código de operación <b>MOV</b>	Registro <b>A</b>	Registro <b>B</b>	1234h
0	7 8	11 12	15 16
			31

- La instrucción ocupa cuatro bytes
- El significado de la instrucción es A:= Memoria (B + 1234h);



## Ejecución de instrucciones (VII)

### MOV A, [B + 1234h] (II)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
<b>Traer la instrucción</b> Direccionar	CP → bus de direcciones Cargar registro de direcciones	TCP LDIR
Leer la instrucción	Inicio ciclo de memoria Lectura Buffer de memoria al bus de datos Cargar registro de instrucciones	MEM RD TMEM LI
<b>Decodificación</b>		1 periodo de retardo



## Ejecución de instrucciones (VIII)

### MOV A, [B + 1234h] (III)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
Incrementar CP Sumar CP + 4	UC pone en el bus de datos el 4 Selección de operandos y operación de suma  Cargar el acumulador	DIT <X0, X1 = B. Dat> <Y0, Y1 = CP> <Selop = sumar> LAC
Cargar nuevo valor en CP	Acumulador → Bus de datos Cargar el registro CP	TAC LCP



## Ejecución de instrucciones (IX)

### MOV A, [B + 1234h] (IV)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
Calcular dirección del operando fuente Selección de B + desplazamiento	UC pone 1234h. → B.datos UC genera dirección B	DIT <DIR A = B>
Sumar	Selección de operandos y operación  Cargar el acumulador	<X0, X1 = b.dat> <Y0, Y1 = Sal A> <Selop = suma> LAC

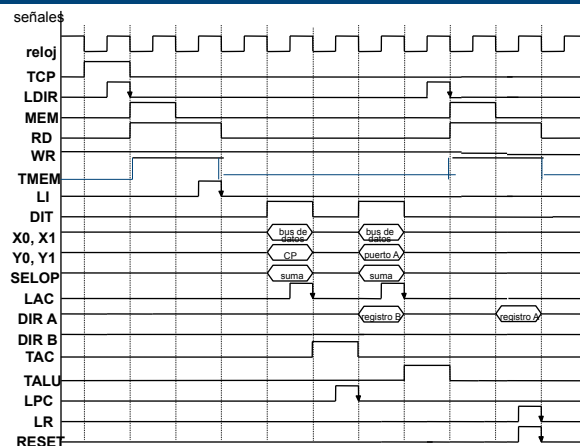


## Ejecución de instrucciones (X) MOV A, [B + 1234h] (V)

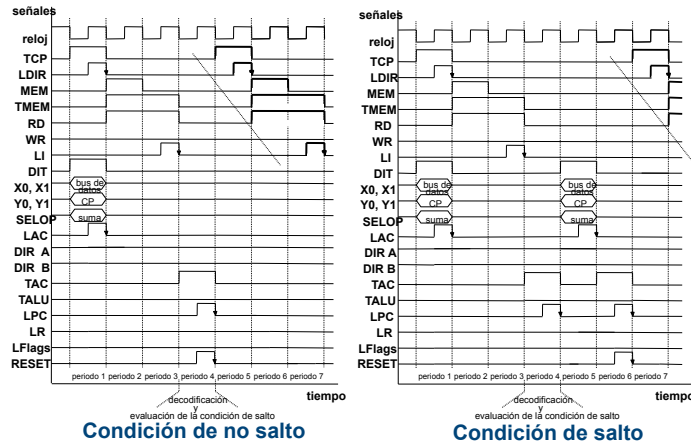
ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
<b>Guardar el resultado</b> Volcar al b. datos el operando fuente	Transferencia del acumulador al bus de direcciones Cargar registro direcciones Inicio ciclo de memoria Lectura Buffer de memoria al bus de datos UC genera dirección A Cargar banco de registros	TALU LDIR MEM RD TMEM <DIR A = A> LR
<b>Poner a 0 el contador de fases</b>		RESET



## Ejecución de instrucciones (XI) MOV A, [B + 1234h] (y VI)



## Ejecución de instrucciones (y XII) JZ EsCero

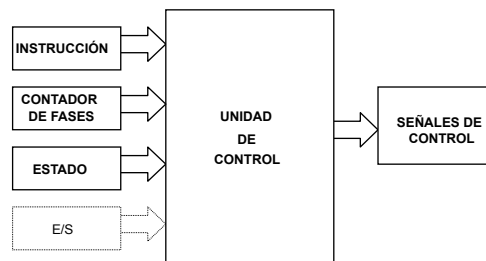


Departamento de Automática  
Área de Arquitectura y Tecnología de Computadores

Arquitectura de Computadores  
I. T. Informática de Gestión

## Diseño de la Unidad de Control (I)

- La UC necesita del código de operación de la instrucción, el estado del sistema y las señales de E/S, al ritmo del contador de fases
- La UC es un circuito que genera las señales de control necesarias para ejecutar una instrucción
- El diseño de la UC se realiza definiendo todas las señales de control en función de todas las posibles entradas



Departamento de Automática  
Área de Arquitectura y Tecnología de Computadores

Arquitectura de Computadores  
I. T. Informática de Gestión

## Diseño de la Unidad de Control (II)

- **UC cableada** (lógica cableada)
- Método tradicional de diseño lógico
- **Ejemplos:**
- Alpha 21164, 21264, AMD K6, Pentium
- Computadores con lógica sencilla
- **VENTAJAS:**
  - Circuito más rápido que el de lógica almacenada
- **INCONVENIENTES:**
  - Muy laborioso de diseñar
  - Muy difícil de modificar

**UC microprogramada**(lógica almacenada)

- Almacena, en memoria de control, las palabras de control correspondientes a cada periodo de ejecución de una instrucción

**Ejemplos:**

- Ordenadores medios i80x86

**VENTAJAS:**

- Modificación del juego de instrucciones
- Un computador puede admitir varios juegos de instrucciones

**INCONVENIENTES:**

- Más lentas



## Diseño de la Unidad de Control (III) Lógica cableada (I)

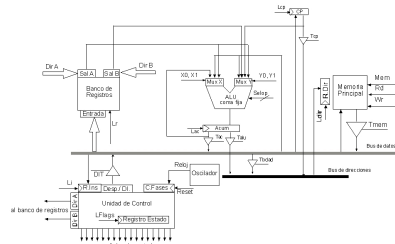
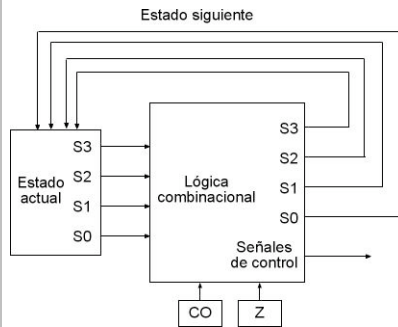
- La unidad de control es una máquina secuencial del tipo Moore, en la que las salidas dependen del estado en el que se encuentra y no de las entradas únicamente
- El número de estados depende del número de operaciones elementales a realizar, lo que dará el número de biestables para realizarlos
- Inicialmente la unidad de control está en estado de espera y necesita una señal externa para cambiar de estado
- La implementación de este circuito puede hacerse mediante:
  - **Método de la tabla de estados.** El control pasa por una serie de estados que dependiendo de las entradas determinan el estado siguiente y qué señales de control se activan
  - **Método de las células de retardo.** Al diagrama de control de estados se le sustituyen los estados por células de retardo
  - **Método del contador secuenciador.** Se emplea un contador de ciclos que indica las señales a activar dependiendo de la instrucción





## Diseño de la Unidad de Control (IV) Lógica cableada (II). Máquina Estados (I)

- El diseño de la Unidad de Control será por tanto:



## Diseño de la Unidad de Control (V) Lógica cableada (III). Máquina Estados (II)

- Camino de datos anterior en la que se supone un repertorio RISC, siendo las instrucciones del tipo registro-registro.
- El repertorio de instrucciones será
  - Instrucciones de memoria:**
    - LD R, Memoria ( $R \leftarrow \text{Contenido de la Memoria}$ )
    - ST Memoria, R ( $\text{Contenido de la Memoria} \leftarrow R$ )
  - Instrucciones de proceso:**
    - ADDI / SUBI R1, R2, Inmediato ( $R1 = R2 \pm \text{Dato inmediato}$ )
    - ADD / SUB R1, R2, R3 ( $R1 = R2 \pm R3$ )
  - Instrucciones de control:**
    - JZ Dirección de salto ( $CP \leftarrow CP + \text{Desplazamiento}$ )
    - JMP Dirección de salto ( $CP \leftarrow \text{Dirección de salto}$ )
- Modos de direccionamiento permitidos:
  - Inmediato**, para las instrucciones de proceso
  - Relativo a registro**, para las instrucciones de acceso a memoria y la de salto condicional
  - Directo a memoria**, para la instrucción de salto incondicional



## Diseño de la Unidad de Control (VI) Lógica cableada (IV). Máquina Estados (III)

- Formato de instrucción **todas** de 32 bits.
- Los códigos de operación serán:

Instrucción	Código de operación		
	O3	O2	O1
LD	0	0	0
ST	0	0	1
ADD R1, R2, R3	0	1	0
SUB R1, R2, R3	0	1	1
ADDI R1, R2, Inm	1	0	0
SUBI R1, R2, Inm	1	0	1
JZ Dirección	1	1	0
JMP Dirección	1	1	1

CO	R	R	Desplazamiento
(3)	(4)	(4)	(21)

CO	R1	R2	Inmediato
(3)	(4)	(4)	(21)

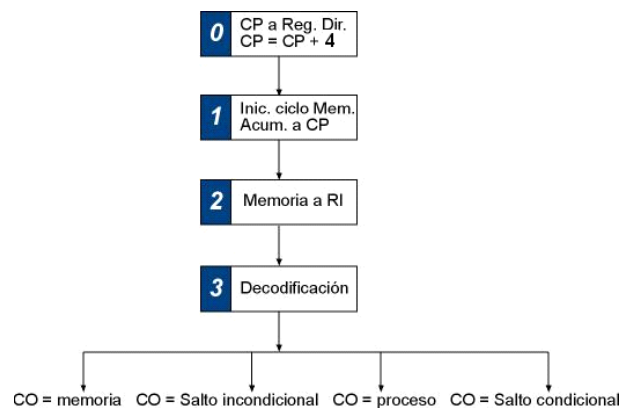
CO	R1	R2	R3	No usado
(3)	(4)	(4)	(4)	(17)

CO	Desplazamiento / Dirección directa de memoria
(3)	(29)



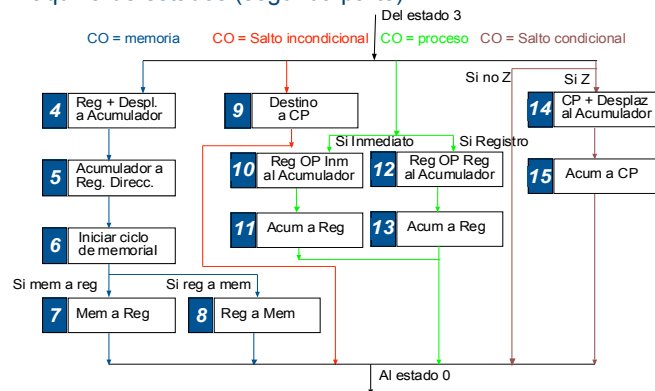
## Diseño de la Unidad de Control (VII) Lógica cableada (V). Máquina Estados (IV)

- Máquina de estados (primera parte)



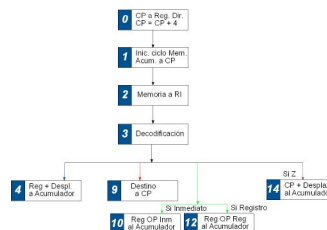
## Diseño de la Unidad de Control (VIII) Lógica cableada (VI). Máquina Estados (V)

- Máquina de estados (segunda parte)



## Diseño de la Unidad de Control (IX) Lógica cableada (VII). Máquina Estados (VI)

- Hace falta definir a partir del diagrama de estados la función de transición de estados
- Como son 15 estados harán falta 4 bits para numerarlos
- Una opción es usar el número del estado en binario, por ejemplo, el estado 0 (0000), el estado 1 (0001), el estado 2 (0010), etc
- Las ecuaciones del cambio de estado:

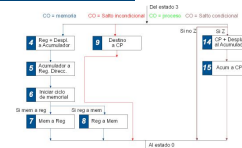


Estado	Estado nuevo				Condición	Ecuación
	S3	S2	S1	S0		
1	0	0	0	1	Estado Anterior 0	$S0 = \overline{S3} \cdot \overline{S2} \cdot \overline{S1} \cdot S0$
2	0	0	1	0	Estado Anterior 1	$S1 = \overline{S3} \cdot \overline{S2} \cdot S1 \cdot \overline{S0}$
3	0	0	1	1	Estado Anterior 2	$S1 = \overline{S3} \cdot \overline{S2} \cdot S1 \cdot S0$ y $S0 = \overline{S3} \cdot \overline{S2} \cdot S1 \cdot \overline{S0}$
4	0	1	0	0	Estado Anterior 3 Y CO=memoria	$S2 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot \overline{S0} (\overline{S2} \cdot \overline{S1} \cdot \overline{S0} + \overline{S2} \cdot \overline{S1} \cdot S0)$



## Diseño de la Unidad de Control (X) Lógica cableada (VIII). Máquina Estados (VII)

- Ecuaciones del cambio de estado (cont):

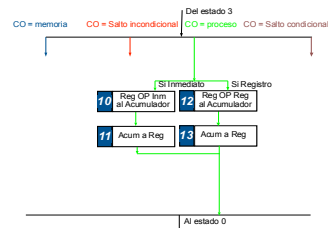


Estado	Estado nuevo				Condición	Ecuación
	S3	S2	S1	S0		
5	0	1	0	1	Estado Anterior 4	$S2 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot \overline{S0}$ y $S0 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot \overline{S0}$
6	0	1	1	0	Estado Anterior 5	$S2 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0$ y $S1 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0$
7	0	1	1	1	Estado Anterior 6 Y CO = ST	$S2 = S1 = S0 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0 \cdot \overline{CO} \cdot \overline{ST} \cdot 1 \cdot 0$
8	1	0	0	0	Estado Anterior 6 Y CO = LD	$S3 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot \overline{S0} \cdot \overline{CO} \cdot \overline{LD} \cdot 1 \cdot 0$
9	1	0	0	1	Estado Anterior 3 Y CO = Salto incondicional	$S3 = S0 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0 \cdot \overline{CO} \cdot 1 \cdot 0$
14	1	1	1	0	Estado Anterior 3 Y CO = Salto condicional Y Z	$S3 = S2 = S1 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0 \cdot \overline{CO} \cdot \overline{Y} \cdot Z$
15	1	1	1	1	Estado Anterior 14	$S3 = S2 = S1 = S0 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0$



## Diseño de la Unidad de Control (XI) Lógica cableada (IX). Máquina Estados (VIII)

- Ecuaciones del cambio de estado (final):



Estado	Estado nuevo				Condición	Ecuación
	S3	S2	S1	S0		
10	1	0	1	0	Estado Anterior 3 Y CO = ADDI o SUBI	$S3 = S1 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0 \cdot (\overline{CO} \cdot 1 \cdot 0 + \overline{CO} \cdot 1 \cdot 0)$
11	1	0	1	1	Estado Anterior 10	$S3 = S1 = S0 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0$
12	1	1	0	0	Estado Anterior 3 Y CO = ADD o SUB	$S3 = S2 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0 \cdot (\overline{CO} \cdot 1 \cdot 0 + \overline{CO} \cdot 1 \cdot 0)$
13	1	1	0	1	Estado Anterior 12	$S3 = S2 = S0 = \overline{S3} \cdot S2 \cdot \overline{S1} \cdot S0$



## Diseño de la Unidad de Control (XII) Lógica cableada (X). Máquina Estados (IX)

- Quedan por definir las señales que se deben generar en cada uno de los estados

Estado	Op. Elmtales.	Señales
0	CP a reg dir	TCP, LDIR
	CP = CP + 4	MUX Y, DIT, MUX X, SELOP, LAC
1	Inicio ciclo mem	MEM, RD
	Acum. A CP	TAC, LCP
2	Memoria a RI	RD, TMEM, LI
3	Decodificación	NADA solamente se miran los flag
4	Reg + desp al acumulador	DIR A, DIT, MUX X, MUX Y, SELOP, LAC
5	Acum a Reg Dir	TALU, LDIR
6	Inicio ciclo mem	MEM
7	Memoria a Reg	RD, TMEM, DIRA, LR, RESET
8	Reg a memoria	DIR A, DIR B, MUX X, MUX Y, SELOP, TAC, WR, RESET



## Diseño de la Unidad de Control (XIII) Lógica cableada (XI). Máquina Estados (X)

- Señales que se generan en los estados

Estado	Op. Elmtales.	Señales
9	Destino a CP	DIT, LCP, RESET
10	Reg OP Inm al acumulador	DIR A, DIT, MUX Y, MUX X, SELOP, LAC, LFLAGS
	Acum a Reg	TAC, DIR A, LR, RESET
12	Reg OP Reg al acumulador	DIR A, DIR B, MUX Y, MUX X, SELOP, LAC, LFLAGS
	Acum a Reg	TAC, DIR A, LR, RESET
14	CP + Desp al acumulador	MUX Y, DIT, MUX X, SELOP, LAC
15	Acum a CP	TAC, LCP, RESET



## Diseño de la Unidad de Control (XIV) Lógica cableada (XII). Máquina Estados (XI)

- Una vez definida la función de transición de estados, la unidad de control se mueve por dicho diagrama
- Como se ha visto en varios estados se generan las mismas señales de control
- Por ejemplo la señal LCP se activa en los estados 1, 9 y 15 y la señal de LAC en los estados 0, 4, 10, 12 y 14 con lo que la ecuación para estas señales serían:

Señal	Estados en los que se activa	Ecuación para la señal de salida
LCP	1, 9 y 15	$LCP = \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + S_3 \overline{S_2} \overline{S_1} \overline{S_0} + S_3 S_2 S_1 \overline{S_0}$
LAC	0, 4, 10, 12 y 14	$LAC = \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + \overline{S_3} S_2 \overline{S_1} \overline{S_0} + S_3 \overline{S_2} S_1 \overline{S_0} + S_3 S_2 \overline{S_1} \overline{S_0} + S_3 S_2 S_1 \overline{S_0}$
....	....	....



## Diseño de la Unidad de Control (XV) Lógica cableada (XIII). Máquina Estados (XII)

- La generación de las señales de control sería:

Señal	Estados en los que se activa	Ecuación para la señal de salida
LCP	1, 9 y 15	$LCP = \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + S_3 \overline{S_2} \overline{S_1} \overline{S_0} + S_3 S_2 S_1 \overline{S_0}$
TCP	0	$TCP = \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0}$
DIR A	4, 7, 8, 10, 11, 12 y 13	$DIRA = \overline{S_3} S_2 S_1 \overline{S_0} + \overline{S_3} S_2 S_1 S_0 + S_3 \overline{S_2} S_1 \overline{S_0} + S_3 \overline{S_2} S_1 S_0 + S_3 S_2 \overline{S_1} \overline{S_0} + S_3 S_2 \overline{S_1} S_0$
DIR B	8 y 12	$DIRB = S_3 \overline{S_2} \overline{S_1} \overline{S_0} + S_3 S_2 \overline{S_1} \overline{S_0}$
LR	7, 11 y 13	$LR = \overline{S_3} S_2 S_1 \overline{S_0} + S_3 \overline{S_2} S_1 \overline{S_0} + S_3 S_2 \overline{S_1} \overline{S_0}$
MUX X	0, 4, 8, 10, 12 y 14	$MUXX = \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + \overline{S_3} S_2 \overline{S_1} \overline{S_0} + S_3 \overline{S_2} \overline{S_1} \overline{S_0} + S_3 \overline{S_2} S_1 \overline{S_0} + S_3 S_2 \overline{S_1} \overline{S_0} + S_3 S_2 S_1 \overline{S_0}$
MUX Y	0, 4, 8, 10, 12 y 14	$MUXY = \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + \overline{S_3} S_2 \overline{S_1} \overline{S_0} + S_3 \overline{S_2} \overline{S_1} \overline{S_0} + S_3 \overline{S_2} S_1 \overline{S_0} + S_3 S_2 \overline{S_1} \overline{S_0} + S_3 S_2 S_1 \overline{S_0}$
SELOP	0, 4, 8, 10, 12 y 14	$SELOP = \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + \overline{S_3} S_2 \overline{S_1} \overline{S_0} + S_3 \overline{S_2} \overline{S_1} \overline{S_0} + S_3 \overline{S_2} S_1 \overline{S_0} + S_3 S_2 \overline{S_1} \overline{S_0} + S_3 S_2 S_1 \overline{S_0}$
LAC	0, 4, 10, 12 y 14	$LAC = \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + \overline{S_3} S_2 \overline{S_1} \overline{S_0} + S_3 \overline{S_2} S_1 \overline{S_0} + S_3 S_2 \overline{S_1} \overline{S_0} + S_3 S_2 S_1 \overline{S_0}$
TAC	1, 8, 11, 13 y 15	$TAC = \overline{S_3} \overline{S_2} \overline{S_1} S_0 + S_3 \overline{S_2} \overline{S_1} S_0 + S_3 \overline{S_2} S_1 S_0 + S_3 S_2 \overline{S_1} S_0 + S_3 S_2 S_1 S_0$
TALU	5	$TALU = S_3 S_2 \overline{S_1} S_0$



## Diseño de la Unidad de Control (XVI) Lógica cableada (XIV). Máquina Estados (XIII)

- La generación de las señales de control sería (final):

Señal	Estados en los que se activa	Ecuación para la señal de salida
DIT	0, 4, 9, 10 y 14	$DIT = \overline{S_3} \overline{S_2} S_1 \overline{S_0} + \overline{S_3} S_2 \overline{S_1} \overline{S_0} + S_3 \overline{S_2} \overline{S_1} \overline{S_0} + S_3 \overline{S_2} S_1 \overline{S_0} + S_3 S_2 S_1 \overline{S_0}$
LI	2	$LI = \overline{S_3} \overline{S_2} S_1 \overline{S_0}$
RESET	7, 8, 9, 11, 13 y 15	$RESET = \overline{S_3} S_2 S_1 S_0 + S_3 \overline{S_2} S_1 \overline{S_0} + S_3 S_2 \overline{S_1} S_0 + S_3 \overline{S_2} \overline{S_1} S_0 + S_3 S_2 S_1 S_0$
LFLAGS	10 y 12	$LFLAGS = \overline{S_3} \overline{S_2} S_1 \overline{S_0} + S_3 S_2 \overline{S_1} \overline{S_0}$
MEM	1 y 6	$MEM = \overline{S_3} \overline{S_2} S_1 \overline{S_0} + S_3 S_2 S_1 \overline{S_0}$
RD	2 y 7	$RD = \overline{S_3} \overline{S_2} S_1 \overline{S_0} + \overline{S_3} S_2 S_1 S_0$
WR	8	$WR = S_3 \overline{S_2} \overline{S_1} \overline{S_0}$
LDIR	0 y 5	$LDIR = \overline{S_3} \overline{S_2} \overline{S_1} \overline{S_0} + \overline{S_3} S_2 \overline{S_1} \overline{S_0}$
TMEM	2 y 7	$TMEM = \overline{S_3} \overline{S_2} S_1 \overline{S_0} + \overline{S_3} S_2 S_1 S_0$



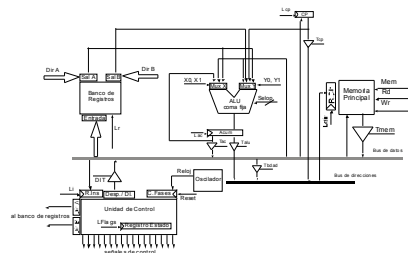
## Diseño de la Unidad de Control (XVII) Lógica cableada (XV). Células de retardo (I)

- El diseño se basa en implementar el diagrama de estados mediante unos elementos que únicamente proporcionan un retardo llamados células de retardo
- Las líneas de control se conectan a las salidas de las células de retardo que representan el estado en el que se deben activar
- La función de las células de retardo es la de sincronizar el secuenciamiento de las señales de control
- Una vez realizadas las conexiones se introduce un único pulso por el circuito
- El pulso circula por el circuito activando las señales adecuadas en cada momento

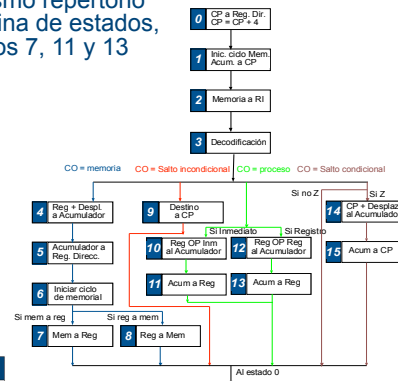


## Diseño de la Unidad de Control (XVIII) Lógica cableada (XVI). Células de retardo (II)

- Para la misma máquina con el mismo repertorio de instrucciones y la misma máquina de estados, la señal LR se activa en los estados 7, 11 y 13

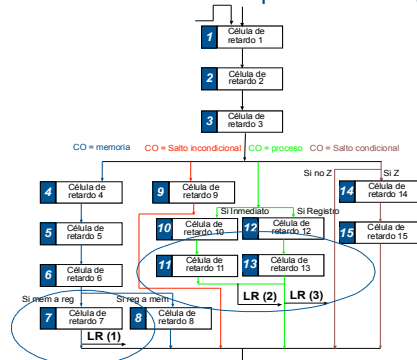


Señal	Estados en los que se activa	Ecuación para la señal de salida
LR	7, 11 y 13	$LR = \overline{LR} \cdot \overline{LR} \cdot \overline{LR} + \overline{LR} \cdot \overline{LR} \cdot \overline{LR} + \overline{LR} \cdot \overline{LR} \cdot \overline{LR}$



## Diseño de la Unidad de Control (XIX) Lógica cableada (XVII). Células de retardo (III)

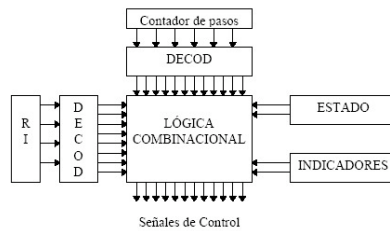
- Introduciendo un único pulso en el diagrama





## Diseño de la Unidad de Control (XX) Lógica cableada (XVIII). Contador secuenciador

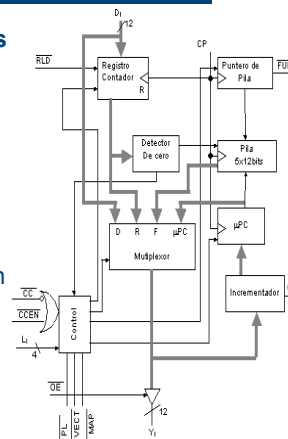
- Emplea un contador que marca los instantes en que se deben activar las señales
- La unidad de control debe conocer además:
  - La instrucción que se está ejecutando
  - Información del estado de la máquina
- Con esa información las instrucciones recibirán para ejecutarse las señales apropiadas en el momento preciso



## Diseño de la Unidad de Control (XXI) Lógica cableada (XIX). Contador secuenciador (II)

### Secuenciador AMD 2910. Características

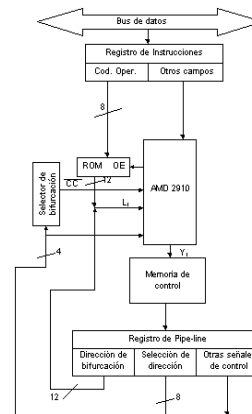
- Contador de microinstrucciones con autoincremento
- Micropila de cinco niveles
- Contador con detector de cero para la realización de microbucles
- Multiplexor para determinar la dirección de la siguiente microinstrucción
- Señales para recoger la salida de un comparador de condición de bifurcación para producir microbifurcaciones condicionales



## Diseño de la Unidad de Control (XXII) Lógica cableada (y XX) Contador secuenciador (III)

### Unidad de control del IBM 370 / 45 empleando un secuenciador AMD2910

- Procesador basado en el secuenciador AMD 2910
- Tiene un formato de instrucción de 32 bits
- Código de operación es de 8 bits que entra a la ROM de instrucciones



## Diseño de la Unidad de Control (XXIII) Lógica microprogramada (I)

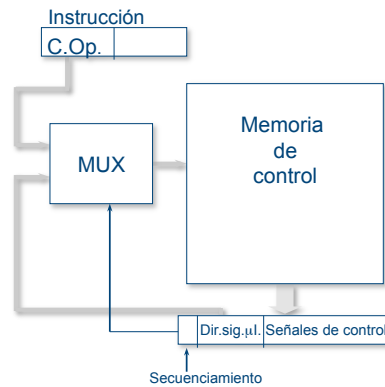
- Un conjunto de microinstrucciones es un microprograma encargado de ejecutar una determinada instrucción
- El conjunto de microprogramas que ejecutan las instrucciones se llama firmware o micro código
- Requisitos a tener en cuenta:
  - Limitación del tamaño de la memoria de control a utilizar
  - Establecer una correspondencia entre cada instrucción máquina y su microprograma correspondiente
  - Control del secuenciamiento de las  $\mu$ ls



## Diseño de la Unidad de Control (XXIV) Lógica microprogramada (II)

### Secuenciamiento explícito:

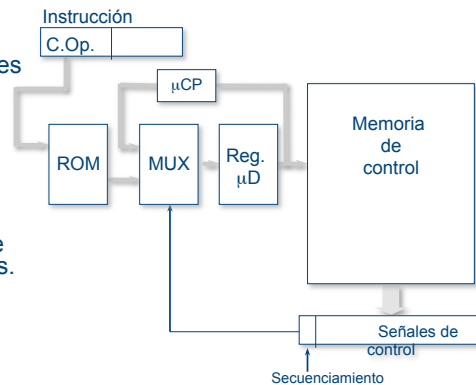
- Cada  $\mu l.$  incluye la dirección de la siguiente  $\mu l.$
- El código de operación apunta a la posición de memoria en la que se inicia la secuencia de cada microprograma
- Cada  $\mu l.$  incluye un bit que indica si es la última
- **Ventaja:** microprogramas diferentes, pueden utilizar secuencias  $\mu l.$  comunes
- **Inconveniente:** Se necesita memoria de control para almacenar la dirección de la siguiente  $\mu l.$



## Diseño de la Unidad de Control (XXV) Lógica microprogramada (III)

### Secuenciamiento implícito:

- Todas las  $\mu l.$  correspondientes a un microprograma se encuentran ordenadas secuencialmente
- Siendo necesario:
  - Un contador de microprograma ( $\mu CP$ ) que apunte a las sucesivas  $\mu l.$
  - Una ROM que indique la posición de la primera  $\mu l.$  del microprograma



## Diseño de la Unidad de Control (XXVI) Lógica microprogramada (IV)

### Codificación de $\mu$ instrucciones

- $\mu$ programación horizontal, si no se usa codificación
- $\mu$ programación vertical, si las  $\mu$ ls. están altamente codificadas

#### $\mu$ programación horizontal

- Las  $\mu$ ls. sólo realizan una operación elemental por periodo
- Cada uno de los  $n$  bits controla directamente cada pieza de hardware, evitando otros niveles de decodificación
- Las  $\mu$ ls. presentan formatos largos
- Las  $\mu$ ls. son rápidas
- Se desperdicia memoria

#### $\mu$ programación vertical

- Se reduce el tamaño de la memoria de control
- Las  $\mu$ ls. son lentas, ya que necesitan decodificación



## Diseño de la Unidad de Control (XXVII) Lógica microprogramada (V)

### Codificación de las $\mu$ instrucciones

Las señales de control agrupan en:

- |   |   |
|---|---|
| <ul style="list-style-type: none"> <li>- Acceso al bus de datos</li> <li>- Acceso al bus de direcciones</li> <li>- Gobierno de la ALU</li> <li>- Gobierno del banco de registros</li> <li>- Gobierno de la memoria</li> </ul> | <ul style="list-style-type: none"> <li>- Gobierno de la unidad de direccionamiento</li> <li>- Estado o condición</li> <li>- Gobierno de la E/S</li> </ul> |
|---|---|

Bus de datos	Bus de direcciones	ALU	Banco de registros	Memoria	Unidad de direccion.
TMEM	TALU	SELOP	LR	MEM	LPC
TAC	TCP	X0, X1	DIR. A	RD	
DIT	TBDAD	Y0, Y1	DIR. B	WR	
		LAC		LDIR	



## Diseño de la Unidad de Control (XXVIII) Lógica microprogramada (VI)

### Microbifurcaciones condicionales

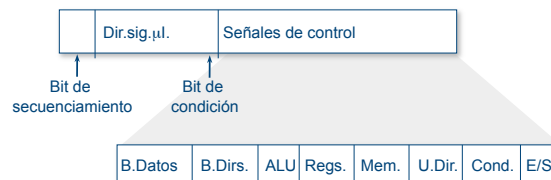
- Las instrucciones de salto condicional tienen dos cronogramas posibles, es decir, poseen dos microprogramas diferentes que se ejecutan dependiendo de la condición
- Se necesita un mecanismo de micro salto que seleccione la ejecución de un microprograma u otro
- El mecanismo dependerá del tipo de secuenciamiento utilizado



## Diseño de la Unidad de Control (XXIX) Lógica microprogramada (VII)

### Microbifurcaciones condicionales. Secuenciamiento explícito:

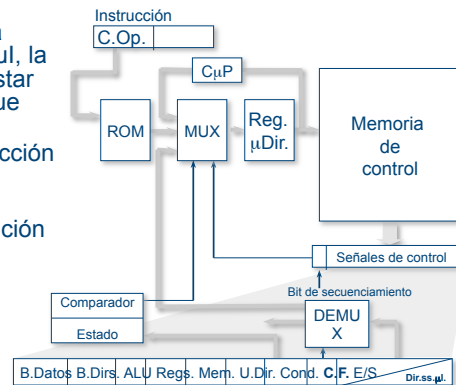
- Cada microinstrucción lleva la dirección de la siguiente
- Incluir dos direcciones significa agrandar demasiado el formato de la microinstrucción por lo que se hace que difieran en un solo bit
- El bit en será: 1 si se cumple la condición y 0 si no se cumple



## Diseño de la Unidad de Control (XXX) Lógica microprogramada (VIII)

### Microbifurcaciones condicionales. Secuenciamiento implícito:

- Para poder elegir entre la secuencia normal u otra  $\mu$ , la dirección de la  $\mu$  debe estar contenida en el campo que comprueba la condición
- Solapar el campo de dirección con un campo de función excluyente
- C.F.: Bit de control de función



## Diseño de la Unidad de Control (XXXI) Lógica microprogramada (IX)

- Una memoria de control contiene muchas microinstrucciones
- Si muchas están repetidas se pueden eliminar mediante la nanoprogramación, descomponiendo las microinstrucciones de tal manera que en la segunda memoria todas sean distintas
- A las palabras de control del segundo nivel se les denominan nanoinstrucciones
- Se pueden agrupar en microrrutinas
- El diseño merece la pena si se cumple que:

$$nm < (n \log_2 d) + (d \cdot m)$$

En donde :

$n$  es el número de microinstrucciones de la memoria de control

$m$  es el número de bits de cada microinstrucción

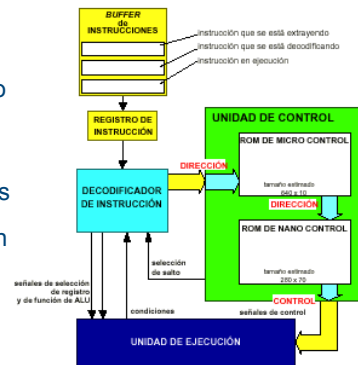
$d$  es el número de microinstrucciones no repetidas que hay



## Diseño de la Unidad de Control (XXXII) Lógica microprogramada (y X)

### Nanoprogramación M68000

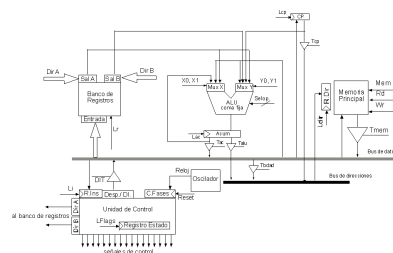
- Emplea un sistema de nanoprogramación en dos niveles:
  - El primer nivel las palabras de control son punteros al segundo nivel
  - El segundo nivel son palabras de control no duplicadas
  - En el segundo nivel las palabras de control se traducen horizontalmente en la activación (desactivación) de las señales
- El tamaño de la palabra de control de segundo nivel es de 70 bits



## Diseño de la Unidad de Control (XXXIII) Diseño microprogramado del computador (I)

### Señales de carga de registro

- Reset
- Lflags
- Li
- Lr
- Lac
- Ldir



Señales de carga de registro					
5	4	3	2	1	0
Ldir	Lac	Lr	Li	Lflags	Reset



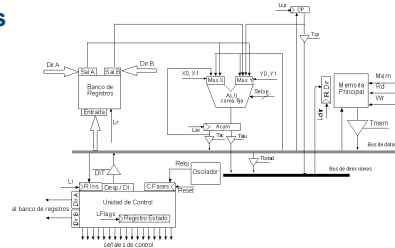
## Diseño de la Unidad de Control (XXXIV) Diseño microprogramado del computador (II)

### Señales de acceso al bus de datos

- Tmem
- Tac
- Dit

B.Datos	
7	6

B7	B6	
0	0	Tmem
0	1	Tac
1	0	Dit
1	1	Nop



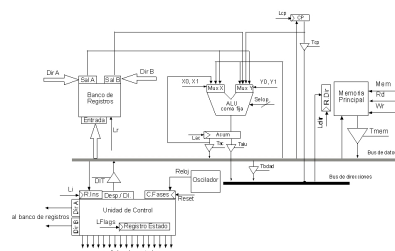
## Diseño de la Unidad de Control (XXXV) Diseño microprogramado del computador (III)

### Señales de acceso al bus de direcciones

- Tcp
- Talu
- Tbdad

B.Direcciones	
9	8

B9	B8	
0	0	Tcp
0	1	Talu
1	0	Tbdad
1	1	Nop





## Diseño de la Unidad de Control (XXXVI) Diseño microprogramado del computador (IV)

### Señales de control de la ALU

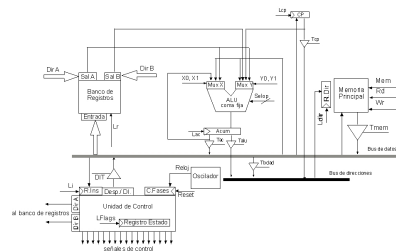
- X1, X0
- Y1, Y0
- Selop

Selop				Y1	Y0	X1	X0
17	16	15	14	13	12	11	10

B11	B10	
0	0	Acumulador
0	1	B. datos
1	0	Salida A
1	1	Nop

B13	B12	
0	0	CF
0	1	B. datos
1	0	Salida A
1	1	Salida B

B17	B16	B15	B14	
0	0	0	0	ADD
0	0	0	1	SUB
0	0	1	0	MUL
0	0	1	1	DIV
...	...	...	...	...
1	1	1	1	XOR



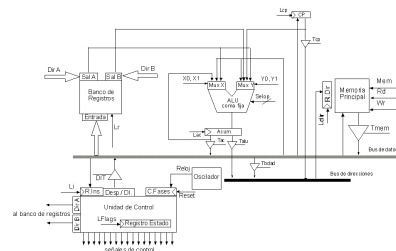
## Diseño de la Unidad de Control (XXXVII) Diseño microprogramado del computador (V)

### Señales de control de la memoria

- Mem
- Rd
- Wr

Mem	Rd/Wr
19	18

B19	B18	
0	0	Nop
0	1	Nop
1	0	Escribir
1	1	Leer

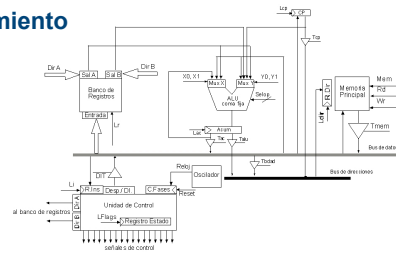


## Diseño de la Unidad de Control (XXXVIII) Diseño microprogramado del computador (VI)

### Señales de la unidad de direccionamiento

- Lcp

Unidad Direccionamiento	
20	
B20	
0	Nop
1	Cargar CP



## Diseño de la Unidad de Control (XL) Diseño microprogramado del computador (VIII)

### Bits que quedan:

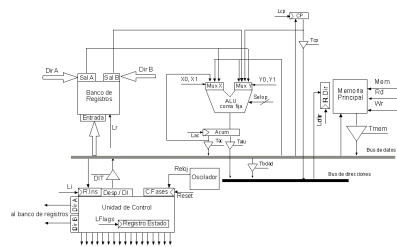
- Bit de condición
- Bit de micros salto
- Bit de secuenciamiento o fin de microprograma

Fin microprograma	Microsalto	Condición
31	30	29

B29	
0	No cumple la condición
1	Si cumple la condición

B30	
0	No lleva micros salto
1	Si lleva micros salto

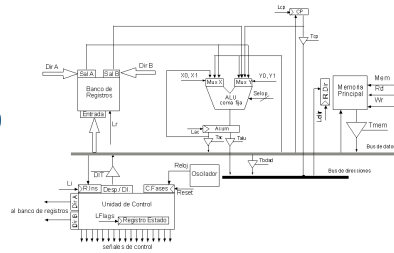
B31	
0	No termina microprg
1	Fin de microprograma



## Diseño de la Unidad de Control (XLI) Diseño microprogramado del computador (IX)

### Bits que quedan:

- Suponiendo una memoria de control de 1K, se necesitarían 10 bits para direccionar
- Empleando el solapamiento de campos emplearemos los bits 0 a 9 que dependiendo del valor de los bits 29 y 30, contendrán una dirección de microsalto válida o las señales de carga de registro y acceso al bus de datos



## Diseño de la Unidad de Control (y XLII) Diseño microprogramado del computador (y X)

### Microinstrucción sin microsalto

Fin microprograma	Microsalto	Condición	Banco de registros				Unidad. Direcccionamiento	Memoria		ALU	Bus de direcciones		Bus de datos		Carga de registros			
31	30	29	28	...	21	20	19	18	17	...	10	9	8	7	6	5	...	0

### Microinstrucción con microsalto

Fin microprograma	Microsalto	Condición	Banco de registros				Unidad. Direcccionamiento	Memoria		ALU	Dirección de microsalto dependiendo del valor que tengan B30 y B29									
31	30	29	28	...	21	20	19	18	17	...	10	9	.....							



## Excepciones

- Las excepciones son eventos inesperados en el procesador de naturaleza asíncrona (división por cero, error de paridad de memoria, desbordamiento, llamadas al sistema operativo (trap), peticiones de entrada salida, etc.)
- Las excepciones deben ser detectadas por la Unidad de Control y tratadas correctamente por el sistema operativo
- Para tratar las interrupciones se debe:
  - Parar el programa en ejecución
  - Enviar la causa de la excepción al sistema operativo
  - Entregar el control al sistema operativo
- Las causas de las excepciones se pueden entregar:
  - **Registro de estado para las excepciones.** La Unidad de Control coloca el código de la excepción en el registro
  - **Excepciones vectorizadas.** Se asignan diferentes direcciones a cada tipo de excepción. El Sistema Operativo ejecutará la rutina de tratamiento de la excepción que encuentre en dicha dirección



## Arranque del computador

- Al arrancar el computador los componentes hardware (registros, flags, etc) toman un estado aleatorio y desconocido. Si no se tiene en cuenta este hecho, la máquina empezará a ejecutar un "programa" desconocido a partir de una posición de memoria tomada al azar
- La solución consiste en forzar en el RESET todos los registros a un estado determinado (p. ej. todo a cero) y apuntar el CP a un programa cargador situado en memoria ROM
- Este programa se encargará de las tareas siguientes:
  - Rutinas de comprobación del computador
  - Rutinas de configuración (dispositivos periféricos, puertos asignados, interrupciones, etc)
  - Carga de vectores de interrupción
  - Carga del sistema operativo
  - Carga de controladores de dispositivos



## Bibliografía

- Estructura y diseño de computadores  
David A. Patterson y John L. Hennessy. Reverté, 2000  
Capítulo 5 y Apéndice C
- Estructura y diseño de computadores  
Jose M<sup>a</sup> Angulo. Paraninfo, 1996  
Capítulo 5
- Arquitectura de computadores. Un enfoque cuantitativo  
John L. Hennessy y David A. Patterson. Mc Graw Hill, 3<sup>a</sup> ed, 2002  
Capítulo 5
- Arquitectura de computadores  
José A. de Frutos y Rafael Rico. Servicio de Publicaciones de la  
Universidad de Alcalá, 1995  
Capítulo 4.

