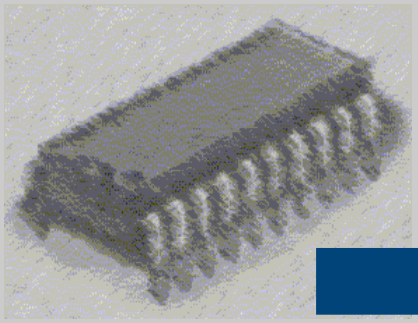
Soluciones a los problemas impares

Tema 4. La Unidad de Control

Arquitectura de Computadores I



I. T. Informática de Sistemas

Curso 2009-2010

Base teórica

La Unidad de Control es el órgano de gobierno del computador. Es el elemento encargado de indicar a los demás componentes qué, cuándo y con qué realizar sus tareas mediante un conjunto de señales de control.

La ejecución de una instrucción está gobernada por un contador de periodos, durante los cuales, la Unidad de Controls genera las señales de control necesarias en función de la información de que dispone en la propia instrucción, en el registro de estado o en señales de E/S .

Operaciones elementales

Todo lo que realiza un computador se lleva a cabo mediante pequeños pasos, llamados operaciones elementales.

Las operaciones elementales pueden ser de dos tipos:

- **Operaciones elementales de transferencia**: mueven datos de un sitio a otro.
- **Operaciones elementales de proceso**: realizan algún tipo de operación con los datos.

Lo que siempre debe ocurrir es que una operación elemental comienza en uno o varios elementos de almacenamiento y termina en uno o varios elementos de almacenamiento.

Temporización de las señales de control

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.

Además habrá que actualizar el Contador de Programa para que apunte a la instrucción siguiente a ejecutar.

Cada una de las fases anteriores se realiza en un número determinado de periodos de reloj, de ahí que las instrucciones tengan diferente duración según la operación y el modo de direccionamiento de los operandos empleados.

Diseño de la Unidad de Control

Uno de los temas más complejos a las que se enfrenta el arquitecto de computadores es el diseño de la Unidad de Control debido al elevado número de estados y señales de control que debe tener en cuenta. Básicamente existen dos técnicas de diseño de la Unidad de Control: la que emplea lógica cableada y la que emplea lógica microprogramaza.

Lógica cableada

La unidad de control es una máquina secuencial en la que las salidas dependen del estado en el que se encuentra y no sólo de las entradas.

Para determinar el número de estados se deberá calcular el 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 puede hacerse de las siguientes formas:

- Método de la tabla de estados
- Método de las células de retardo
- Método del contador secuenciador

Lógica microprogramada

Traslada el concepto de programación al nivel de operaciones elementales.

Para ejecutar una instrucción existirá un conjunto de microinstrucciones encargado de llevar a cabo dicha tarea. Al conjunto de microprogramas que ejecutan las instrucciones se les llama *firmware* o microcódigo

A la hora de diseñar una Unidad de Control microprogramada se debe 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 µIs

Secuenciamiento de las µIs

Secuenciamiento explícito:

Cada µI. incluye la dirección de la siguiente µI e incluye un bit que indica si es la última del microprograma.

El código de operación apunta a la posición de memoria en la que se inicia la secuencia de cada microprograma

Secuenciamiento implícito:

Todas las µIs. correspondientes a un microprograma se encuentran ordenadas secuencialmente

Es necesario:

- Un contador de microprograma (µCP) que apunte a las sucesivas µIs.
- Una ROM que indique la posición de la primera µI. del microprograma

Codificación de µinstrucciones

Por criterios de diseño, de memoria y de rendimiento se debe decidir el nivel de codificación de las microinstrucciones:

- **μprogramación horizontal**, si no se usa codificación
- **μprogramación vertical**, si las μIs. están altamente codificadas

Codificación de las μinstrucciones

Las señales de control agrupan en:

- Acceso al bus de datos
- Acceso al bus de direcciones
- Gobierno de la ALU
- Gobierno del banco de registros
- Gobierno de la memoria
- Gobierno de la unidad de direccionamiento
- Estado o condición
- Gobierno de la E/S

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 microsalto que seleccione la ejecución de un microprograma u otro

El mecanismo dependerá del tipo de secuenciamiento utilizado

1. **Secuenciamiento explícito**: cada microinstrucción lleva la dirección de la siguiente la dirección de la μI siguiente difiere en un bit. El bit será 1 si se cumple la condición y 0 si no se cumple

2. **Secuenciamiento implícito**: para poder elegir entre la secuencia normal u otra μI, la dirección de la μI debe estar contenida en el campo que comprueba la condición. Para ahorrar bits se solapa el campo de dirección con un campo de función excluyente

1. Se dispone de una computadora que consta, entre otros, de los siguientes elementos:

   • ALU que se alimenta de dos entradas y permite realizar 16 operaciones.

   • Contiene un registro acumulador para almacenar resultados temporales.

   • Banco de registros de 32 registros, con una puerta de entrada y dos puertas de salida.

   • Contador de programa conectado al registro de datos/direcciones.

   • Memoria principal de 128 Mbytes, organizada en palabras de 32 bits.

   • Se considerará que las lecturas y escrituras en memoria se realizan en dos periodos de reloj.

   • El bus de datos / direcciones es de 32 bits.

   Se quiere que la CPU ejecute la siguiente instrucción de una palabra:

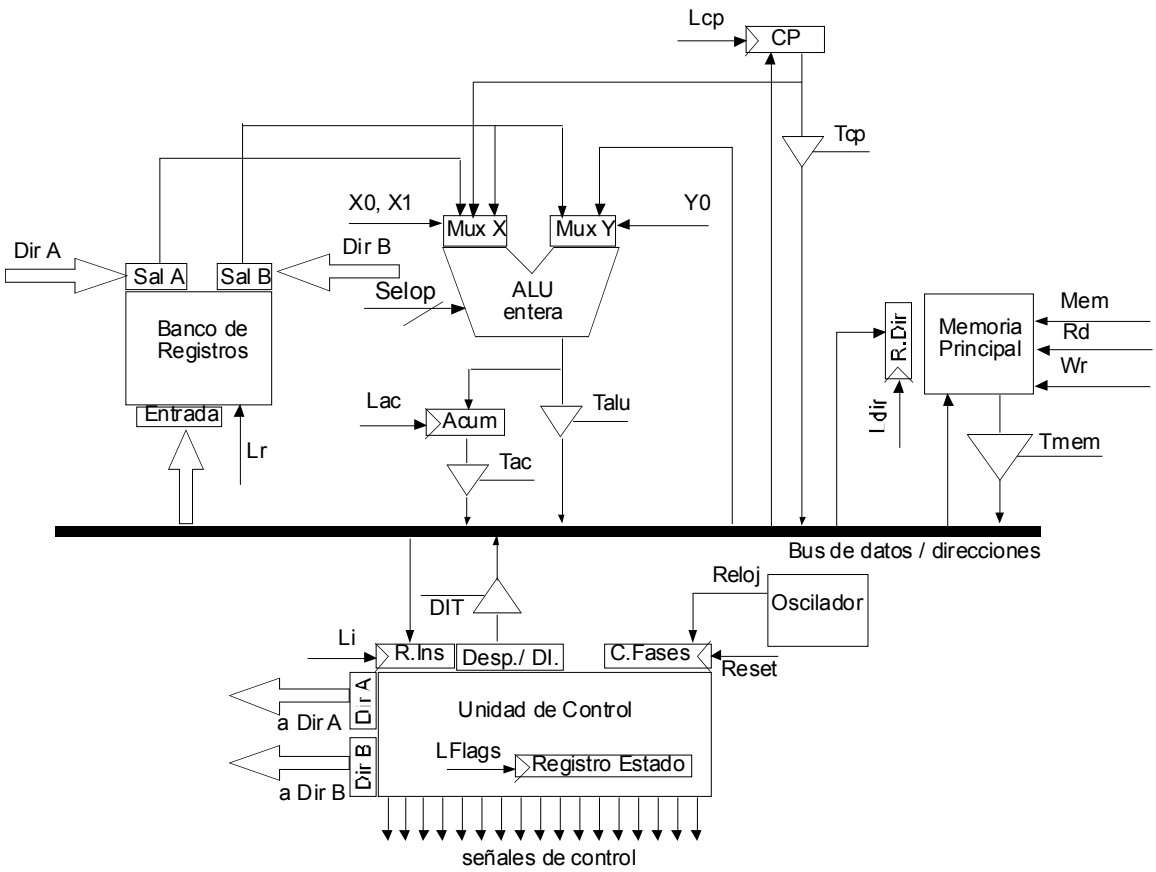
   SHL F, 5

   a. Describir las operaciones elementales que se realizan en cada una de las fases de ejecución de la instrucción.

   b. Realizar el cronograma correspondiente a la anterior secuencia de operaciones elementales.

   c. Formato de microinstrucción. Memoria de control de 64k.

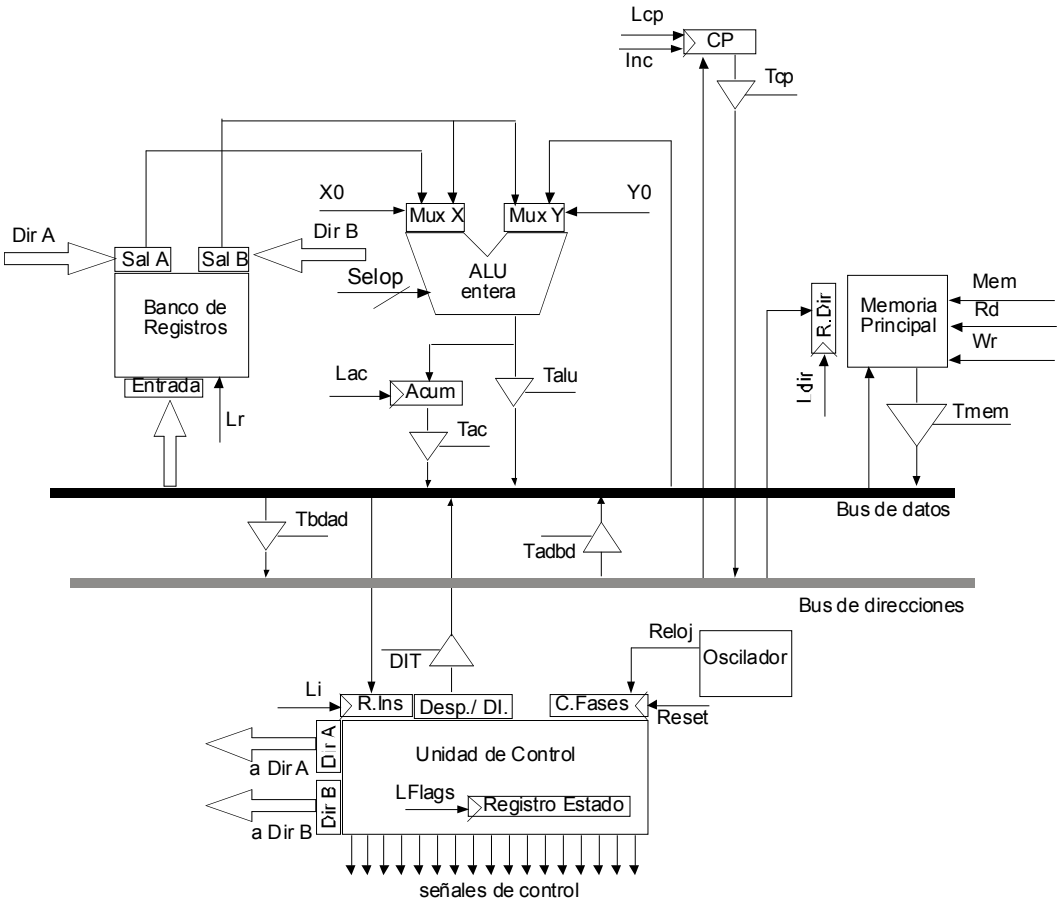
   d. Microprograma para la fase de ejecución de la instrucción.



2. Se dispone de una computadora que consta, entre otros, de los siguientes elementos:

   - ALU que se alimenta de dos entradas y permite realizar 8 operaciones: +, ×, desplazamiento aritmético a la izquierda y a la derecha, AND, OR, XOR y desplazamiento lógico a la derecha.. Cuenta con un registro acumulador para almacenar los resultados temporales
   - Banco de registros de 16 registros, con una puerta de entrada y dos puertas de salida.
   - Contador de programa con posibilidad de autoincremento.
   - Memoria principal de 16 Mbytes, organizada en palabras de 32 bits.
   - Tanto el bus de datos como el bus de direcciones, son de 32 bits.

- El formato de todas las instrucciones de la máquina ocupa cuatro palabras.



Se quiere que la CPU ejecute la instrucción XOR F, [E + 1234h], que realiza la siguiente operación:

F ← F xor M(E + 1234h)

a. Describir las operaciones elementales que se realizan en cada una de las fases de ejecución de la instrucción.

b. Realizar el cronograma correspondiente a la anterior secuencia de operaciones elementales.

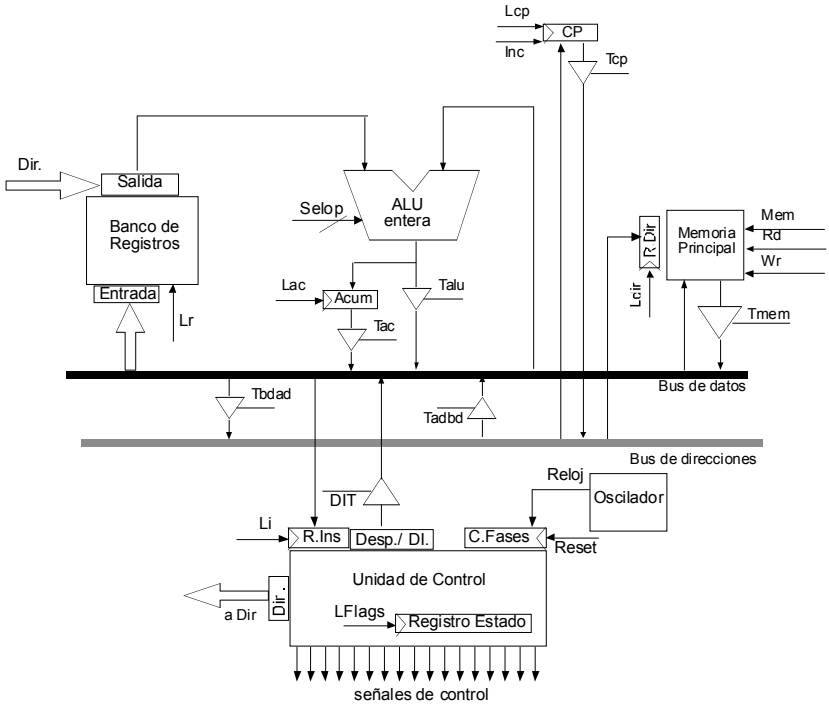
c. Formato de microinstrucción. Memoria de control de 64k.

d. Microprograma para la fase de ejecución de la instrucción.

.

3. Se dispone de una computadora que consta, entre otros, de los siguientes elementos:

- ALU que se alimenta de dos entradas y permite realizar 16 operaciones, entre las que destancan: transferir la entrada de la ALU a la salida, y las operaciones de resta A-B y de B-A, ambas en complemento a 2.
- Contiene un registro acumulador para almacenar resultados temporales.
- Banco de registros de 8 registros, con una puerta de entrada y una puerta de salida.
- Contador de programa con posibilidad de autoincremento y conectado al registro de direcciones.
- Memoria principal de 128 Mbytes, organizada en palabras de 16 bits.
- Se considerará que las lecturas y escrituras en memoria se realizan en dos periodos de reloj.
- Tanto el bus de datos como el bus de direcciones, son de 16 bits.
- Se cuenta con la posibilidad de transferir el contenido del bus de datos, al bus de direcciones



Se quiere que la CPU ejecute la siguiente instrucción de una palabra:

SUB [B++], A

Esta instrucción realiza las siguientes operaciones, y en ese orden:

1. M(B) ← M(B) - A
2. B ← B + 1

a. Describir las operaciones elementales que se realizan en cada una de las fases de ejecución de la instrucción.

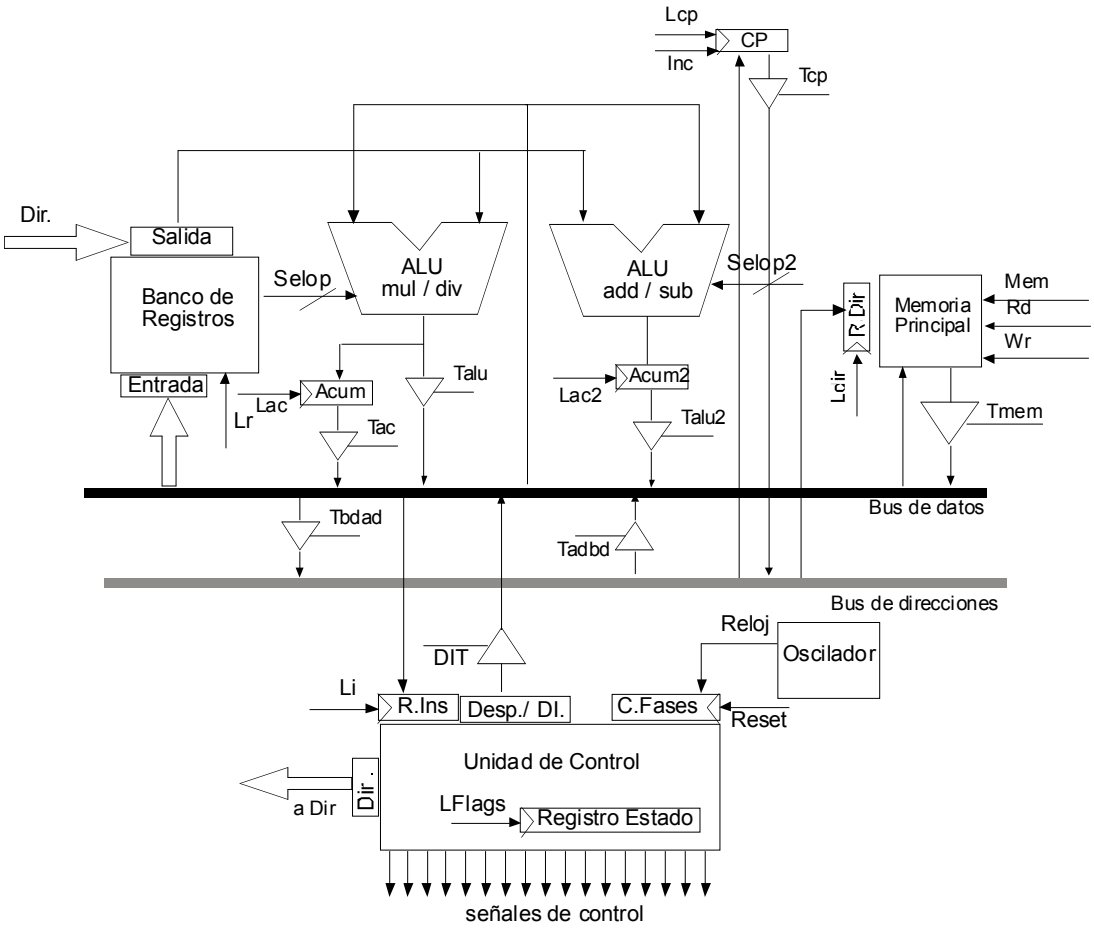
b. Realizar el cronograma correspondiente a la anterior secuencia de operaciones elementales

c. Formato de microinstrucción. Memoria de control de 32k

d. Microprograma para la fase de ejecución de la instrucción.

4. Se dispone de un computador que consta, entre otros, de los siguientes elementos:

    - Dos ALUS una especilizada en operaciones de multiplicación y división con y sin signo y otra especializada en operaciones de suma-resta
    - Ambas ALUs contienen un registro acumulador para almacenar resultados temporales.
    - Banco de 32 registros con una salida y una puerta de entrada.
    - Contador de programa con posibilidad de autoincremento y de carga del contenido del bus de datos.
    - Memoria principal, organizada en palabras de 32 bits.
    - Se cuenta con la posibilidad de transferir el contenido del bus de datos, al bus de direcciones.
    - Los buses son de 32 bits.
    - El formato de las instrucciones se ha diseñado de forma que todas ocupan 32 bits.



Se quiere que la CPU ejecute la siguiente instrucción de una palabra:

Div C, D

Esta instrucción realiza la siguiente operación: C ← C / D

a. Describir las operaciones elementales que se realizan en cada una de las fases de ejecución de la instrucción.

b. Realizar el cronograma correspondiente a la anterior secuencia de operaciones elementales

c. Formato de microinstrucción. Memoria de control de 32k.

d. Microprograma para la fase de ejecución de la instrucción.

5. Se dispone de una computadora que consta, entre otros, de los siguientes elementos:

    - ALU que se alimenta a través de dos multiplexores, que permiten seleccionar el origen de los operandos, y es capaz de realizar 32 operaciones entre ellas la de transferir la entrada de la ALU a la salida. Además, contiene un registro acumulador y otro temporal, transparentes al usuario, para almacenar resultados temporales.
    - Banco de 16 registros con dos salidas A y B, y una puerta de entrada.
    - Registro puntero de pila conectado al bus de direcciones
    - Memoria principal de 16 Mbytes.
    - Se cuenta con la posibilidad de transferir el contenido del bus de datos, al bus de direcciones.
    - Los buses son de 32 bits, y la memoria se organiza en palabras de 32 bits.
    - El formato de las instrucciones se ha diseñado de forma que todas ocupan 32 bits.

Se quiere que la CPU ejecute la siguiente instrucción de una palabra:

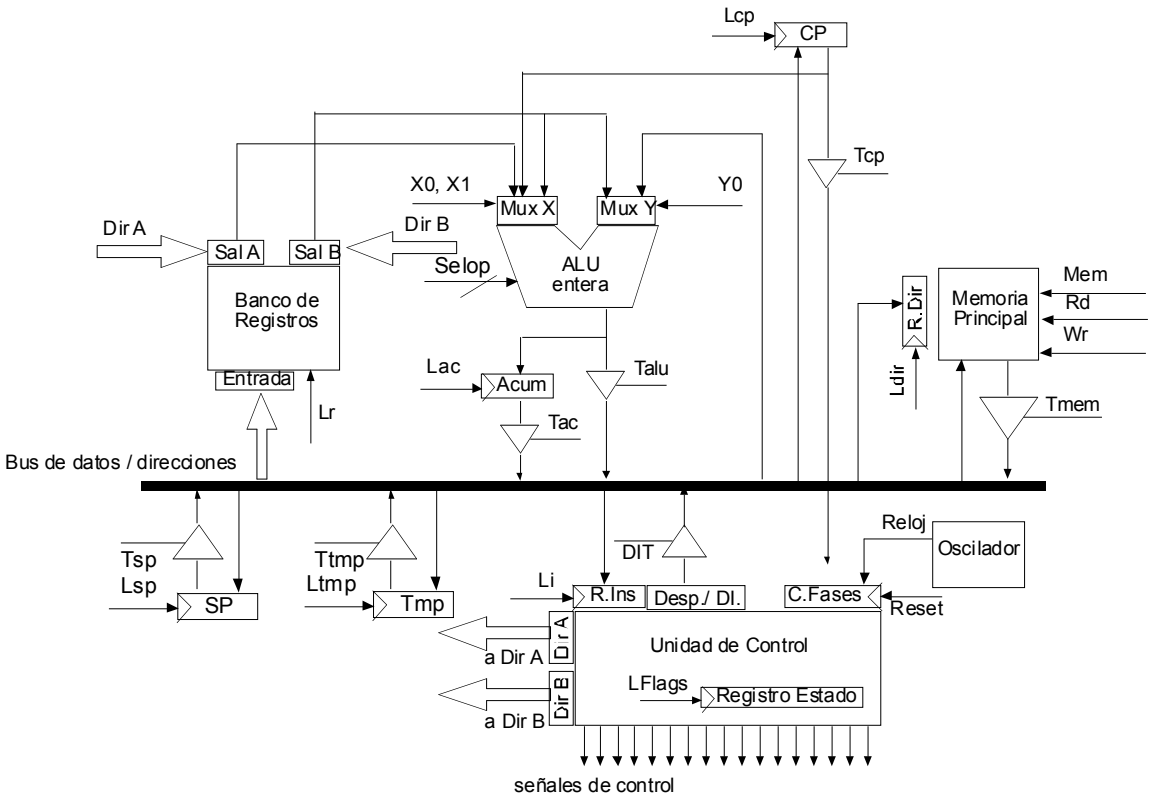
ADD A, B

Esta instrucción realiza la operación: A ← A + B.

Además, se considerará que las lecturas y escrituras en memoria se realizan en dos periodos de reloj.

Se pide:

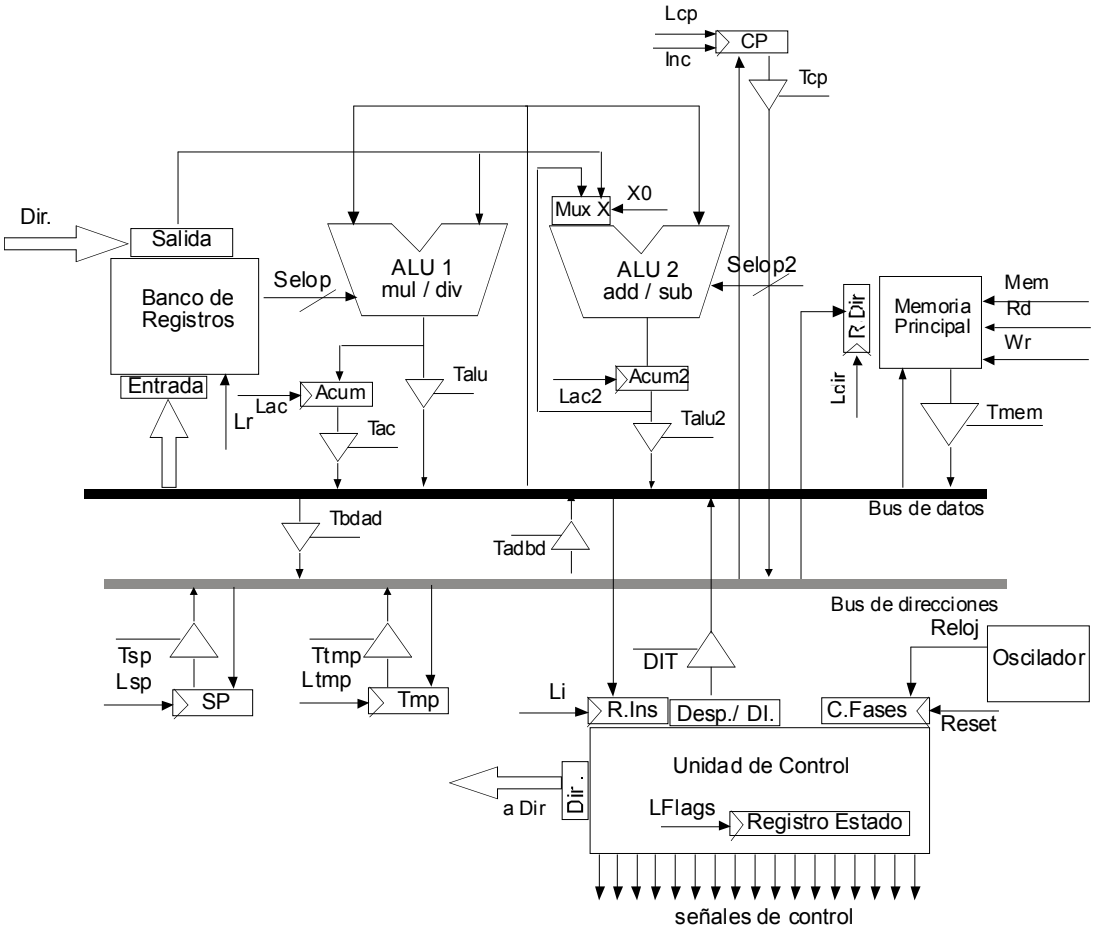
a. Describir las operaciones elementales que se realizan en cada una de las fases de ejecución de la instrucción.
b. Realizar el cronograma correspondiente a la anterior secuencia de operaciones elementales



6. Se dispone de un computador que consta, entre otros, de los siguientes elementos:

   - Dos ALU una especilizada en operaciones de multiplicación y división con y sin signo y otra especializada en operaciones de suma-resta
   - Ambas ALUs contienen un registro acumulador para almacenar resultados temporales. Existe también un registro temporal transparente al usuario
   - Banco de 32 registros con una salida y una puerta de entrada.
   - Contador de programa con posibilidad de autoincremento y de carga del contenido del bus de datos.

- Registro puntero de pila conectado al bus de direcciones
- Memoria principal, organizada en palabras de 32 bits.
- Se cuenta con la posibilidad de transferir el contenido del bus de datos, al bus de direcciones.
- Los buses son de 32 bits.
- El formato de las instrucciones se ha diseñado de forma que todas ocupan 32 bits.



Se quiere que la CPU ejecute la siguiente instrucción de una palabra:

ADD [[B + 1000h]], [C + 1234h]

La instrucción realiza la siguiente operación:

$$M(M(B + 1000h)) \leftarrow M(M(B + 1000h)) + M(C\ 1234h)$$

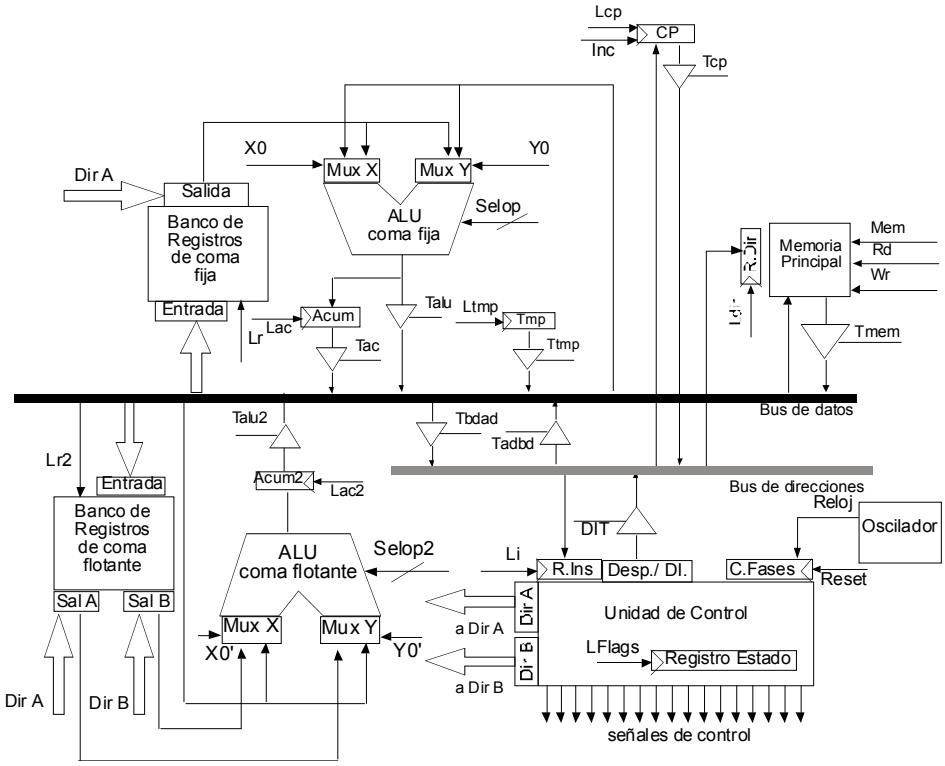
a. Describir las operaciones elementales que se realizan en cada una de las fases de ejecución de la instrucción.

b. Realizar el cronograma correspondiente a la anterior secuencia de operaciones elementales

---

7. Se dispone de un computador que consta, entre otros, de los siguientes elementos:

   - Dos ALU una especilizada en operaciones de coma fija y otra de coma flotante
   - Ambas ALUs contienen un registro acumulador para almacenar resultados temporales. Existe también un registro temporal transparente al usuario
   - Dos Banco de 32 registros con una puerta de entrada y una de salida. Uno de los bancos contiene registros de coma fija y otro de coma flotante
   - Contador de programa con posibilidad de autoincremento y de carga del contenido del bus de datos.
   - Memoria principal, organizada en palabras de 32 bits.
   - Se cuenta con la posibilidad de transferir el contenido del bus de datos, al bus de direcciones.
   - Los buses son de 32 bits.
   - El formato de las instrucciones se ha diseñado de forma que todas ocupan 32 bits.



Se quiere que la CPU ejecute la siguiente instrucción de una palabra:

MULF D, C, 3.27

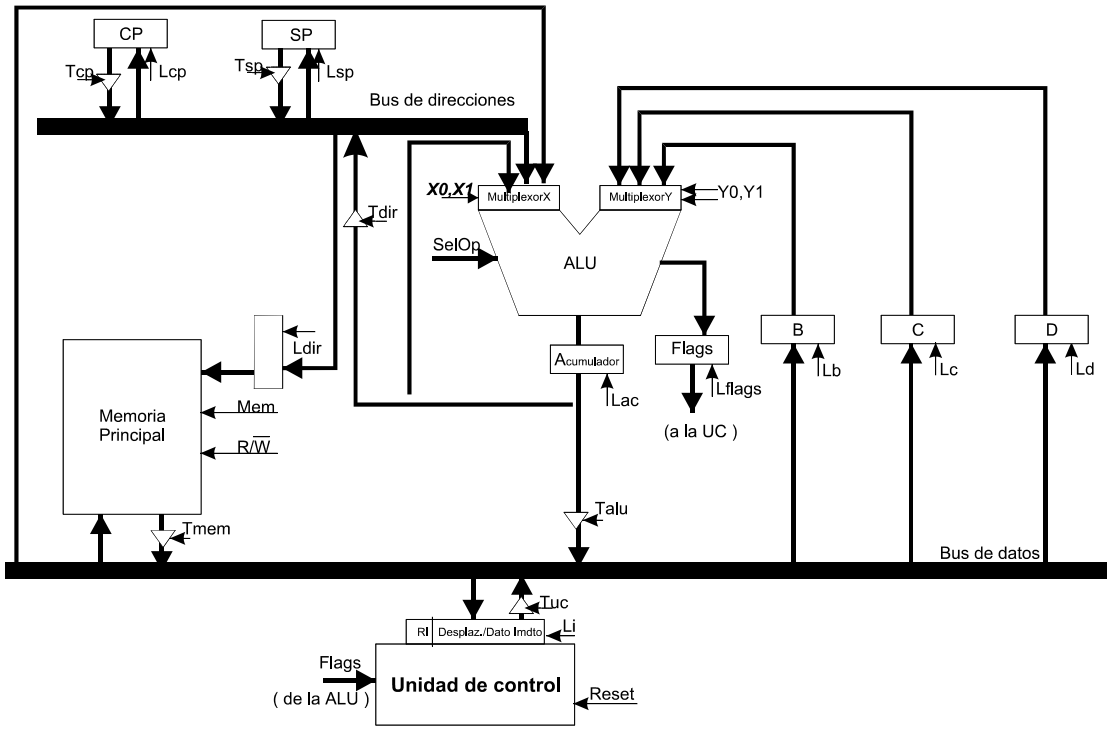
La instrucción realiza la siguiente operación: D ← C x 3,27

a. Describir las operaciones elementales que se realizan en cada una de las fases de ejecución de la instrucción.

b. Realizar el cronograma correspondiente a la anterior secuencia de operaciones elementales

8. Se dispone de un computador que consta, entre otros, de los siguientes elementos:

   - ALU que se alimenta de dos entradas y permite realizar 16 operaciones, entre ellas la de traspasar la entrada del multiplexor Y al registro Acumulador.
   - La máquina consta de un registro contador de programa CP, un registro puntero de pila SP y un registro Acumulador para almacenar los resultados.
   - Memoria principal de 32 Kbytes, organizada en palabras de 8 bits.
   - Se considera que las lecturas y escrituras en memoria se realizan en dos periodos de reloj.
   - El bus de datos es de 8 bits.
   - El bus de direcciones es de 16 bits.
   - No todas las instrucciones son d el mismo tamaño



Se quiere que la CPU ejecute la siguiente instrucción de una palabra:

POP B

Esta instrucción realiza las siguientes operaciones:

1. B ← M(SP)
2. SP ← SP + 1

a. Describir las operaciones elementales que se realizan en cada una de las fases de ejecución de la instrucción.

b. Realizar el cronograma correspondiente a la anterior secuencia de operaciones elementales

---

9. Se dispone de un computador que consta, entre otros, de los siguientes elementos:

   - ALU que se alimenta de dos entradas y permite realizar 16 operaciones. Entre ellas, la de incrementar en una unidad la entrada que le llega por el multiplexor X
   - Banco de registros de 3 registros independientes B, C y D.
   - Memoria principal de 640 Kbytes, organizada en palabras de 32 bits.
   - Se considerará que las lecturas y escrituras en memoria se realizan en dos periodos de reloj.
   - Tanto el bus de datos como el bus de direcciones, son de 32 bits.

Se quiere que la CPU ejecute la siguiente instrucción de una palabra:

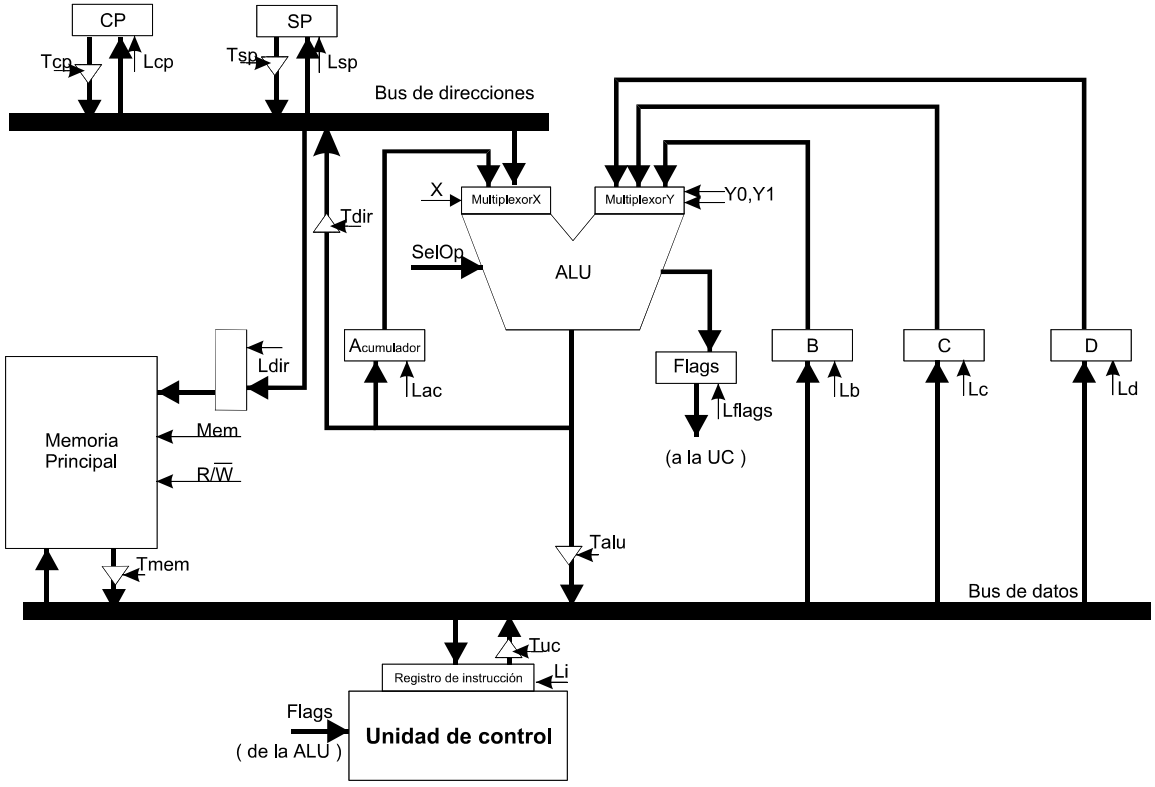
SUB B, [C + 1000h]

Esta instrucción realiza la siguiente operación: B ← B - M(C + 1000 h)

a. Modificar la arquitectura, si fuese necesario, para realizar dicha instrucción. Justificando la respuesta

b. Describir las operaciones elementales que se realizan en cada una de las fases de ejecución de la instrucción.

c. Realizar el cronograma correspondiente a la anterior secuencia de operaciones elementales

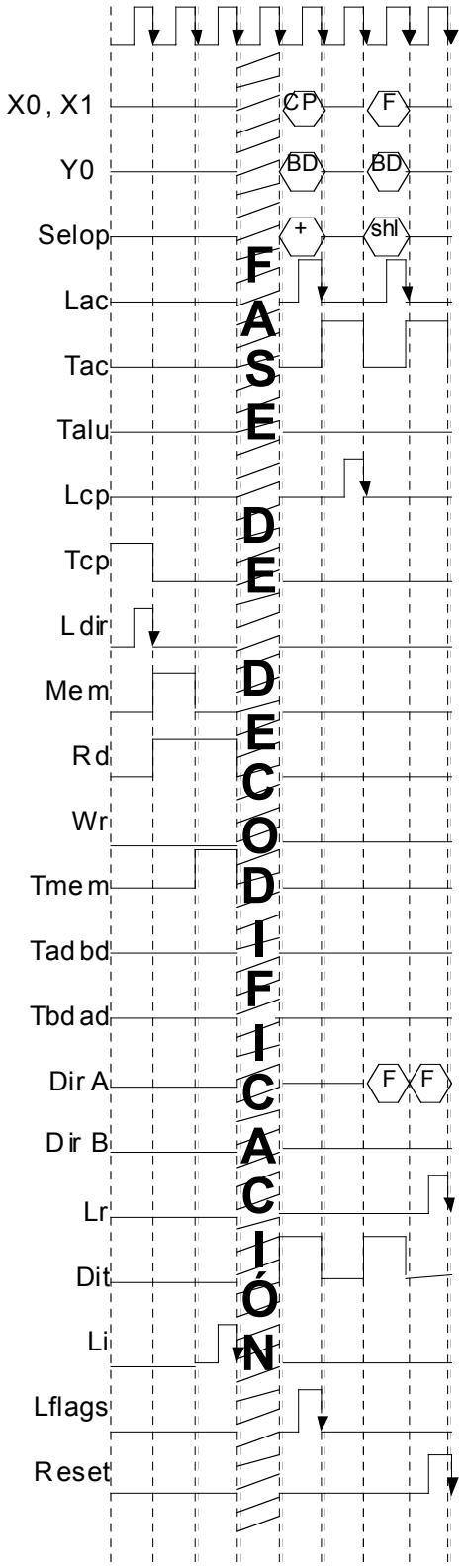


Solución ejercicio 1

Apartado a)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
Fase de fetch		
Direccionar	CP → bus de datos-direcciones	(TCP)
	Bus de datos-direcc. → Reg.Dir	(LDIR)
Traer la instrucción	Inicio ciclo de memoria	(MEM)
	Leer	(RD) dos períodos
	Contenido memoria → B. Datos	(TMEM)
	Cargar registro de instrucción	(LI)
Decodificación		1 período
Actualización de CP		
Sumar CP + 1	Tamaño → Bus de datos	(DIT)
	Bus de dat-dir → Multiplexor Y	< Y0 = b. datdir>
	CP → Multiplexor X	< X0, X1 = CP >
	Sumar	< Selop = + >
	Cargar el acumulador	(LAC)
Cargar nuevo valor en CP	Acumulador → Bus de datdir	(TAC)
	Cargar el CP	(LCP)
Ejecución de la instrucción		
Realizar el desplazamiento	Poner F en salida A	< DIR A = F>
	Salida A → Multiplexor X	< X0, X1 = SAL A >
	Núm. Desplaz → B. Dat-dir	(DIT)
	B. dat-dir→ Multiplexor Y	< Y0 = B. DatDir >
	Desplazar	< Selop = Desplazar >
	Cargar el acumulador	(LAC)
	Modificar los flags de estado	(LFLAGS)
Cargar nuevo valor en F		
	Acumulador → Bus de dat-dir	(TAC)
	Elegir F como registro destino	< DIR A = F >
	Cargar el banco de registros	(LR)
Poner contador de fases a 0	Activar el Reset	(RESET)

Apartado b)


X0 , X1
Y0
Selop
Lac
Tac
Talu
Lcp
Tcp
L dir
Me m
R d
Wr
Tme m
Tad bd
Tbd ad
Dir A
D ir B
Lr
Dit
Li
Lflags
R eset
CP
F
BD
BD
+
shl
F
F
FASE DE DECODIFICACIÓN

Apartado c)

Señales de carga de registros y Reset	
B0	***Reset***
B1	***Lflags***
B2	***LI***
B3	***LR***
B4	***Lac***
B5	***Ldir***

Codificación de las señales al bus de datos / direcciones

Acceso al bus de datos / direcciones			
B8	B7	B6	
0	0	0	***Tac***
0	0	1	***Tmem***
0	1	0	***Tcp***
0	1	1	***Dit***
1	0	0	***Talu***
1	0	1	***nop***
1	1	X	

Codificación de la ALU: multiplexores Y y X y Selop

Y0	
B9	
0	***Salida B***
1	***Bus datos / direcciones***

X1	***X0***	
B11	B10	
0	0	***Salida A***
0	1	***Salida B***
1	0	***CP***
1	1	***nop***

Señales de selección de operación				
B15	B14	B13	B12	
0	0	0	0	***ADD***
0	0	0	1	***SUB***
0	0	1	0	***MUL***
0	0	1	1	***DIV***
0	1	0	0	***ADC***
0	1	0	1	***SBB***
0	1	1	0	***SHL***
…	…	…	…	
1	1	1	1	***ROR***

Señales de acceso a la memoria

Mem	***R/W***	
B17	B16	
0	X	***nop***
1	0	***E***
1	1	***L***

Señales de la unidad de direccionamiento

Lcp	
B18	
0	***nop***
1	***Cargar valor en CP***

Banco de registros

Banco de registros (salida A)					
B23	B22	B21	B20	B19	
0	0	0	0	0	***A***
0	0	0	0	1	***B***
0	0	0	1	0	***C***
0	0	0	1	1	***D***
0	0	1	0	0	***E***
0	0	1	0	1	***F***
…	…	…	…	…	***…***
1	1	1	1	1	***…***

Banco de registros (salida B)					
B28	B27	B26	B25	B24	
0	0	0	0	0	***A***
0	0	0	0	1	***B***
0	0	0	1	0	***C***
0	0	0	1	1	***D***
0	0	1	0	0	***E***
0	0	1	0	1	***F***
…	…	…	…	…	***…***
1	1	1	1	1	***…***

Cumple la condición de microsalto

Cond	
B29	
0	***No cumple condición***
1	***Si cumple condición***

La instrucción lleva microsalto

μSalto	
B30	
0	***No lleva microsalto***
1	***Si lleva microsalot***

Bit de secuenciamiento: final de microprograma

Fin µprg	
B31	
0	***No es la última***
1	***Última microinstrucción***

La dirección de microsalto, suponiendo una memoria de control de 64k requeriría un total de 16 bits para especificarla. Tomamos entonces los bits b15-b0 solapando los campos de la ALU, del bus de datos / direcciones y las señales de carga de registros y reset.

Formato de microinstrucción **sin** salto condicional

Fin µprg	µSalto	Cond	Bango de registros salidas A y B											CP	Memoria		Alu: multiplexores X, Y y selección de operación							Bus datos/direc			Carga de Registros y Reset					
b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	B1	b0

Formato de microinstrucción **con** salto condicional

Fin µprg	µSalto	Cond	Bits no utilizados														Dirección de µsalto condicional o incondicional según valor de b29 o b30															
b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	B1	b0

Apartado c)

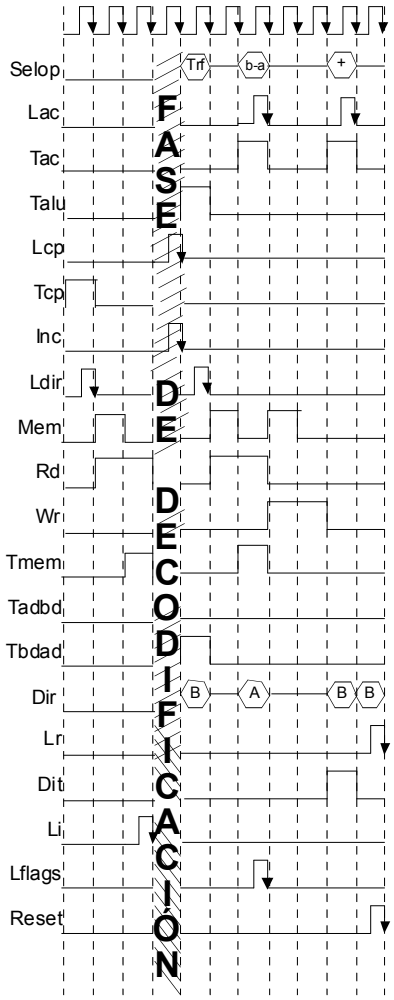
0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	1	0	0	1	0
1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1

Solución ejercicio 3

Apartado a)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
Fase de fetch		
Direccionar	CP → bus de direcciones	(TCP)
	Bus dir.→ Reg. Direcciones	(LDIR)
Traer la instrucción	Inicio ciclo de memoria	(MEM)
	Leer	(RD) dos períodos
	Contenido memoria → B. Datos	(TMEM)
	Cargar registro de instrucción	(LI)
Decodificación y actualizar CP		
	Incrementar CP	(INC)
	Cargar nuevo valor en CP	(LCP)
Elecución de la instrucción		
Calcular dirección y direccionar	Elegir B	< DIR = B>
	Traspasar B por la ALU	< Selop = traspasar >
	Salida ALU → Bus de datos	(TALU)
	Bus de datos → B. direcciones	(TBDAD)
	Cargar el registro de direcciones	(LDIR)
Traer el operando y operar	Inicio ciclo de memoria	(MEM)
	Leer	(RD) 2 períodos
	Memoria → Bus de datos	(TMEM)
	Elegir A	< DIR = A >
	Restar B - A	< Selop = B - A >
	Cargar el acumulador	(LAC)
	Modificar los flags de estado	(LFLAGS)
Cargar el nuevo valor	Acumulador → Bus de datos	(TAC)
	Inicio ciclo de memoria	(MEM)
	Escribir	(WR)
Actualizar B = B + 1	Elegir B	< DIR = B >
	Incremento → Bus de datos	(DIT)
	Sumar	< Selop = + >
	Cargar acumulador	(LAC)
Cargar el nuevo valor en B	Acumulador → Bus de datos	(TAC)
	Elegir F como registro destino	< DIR = B >
	Cargar el banco de registros	(LR)
Poner contador de fases a 0	Activar el Reset	(RESET)

Apartado b)



Apartado c)

Señales de carga de registros y Reset	
B0	***Reset***
B1	***Lflags***
B2	***LI***
B3	***LR***
B4	***Lac***
B5	***Ldir***

Codificación de las señales al bus de datos

Acceso al bus de datos			
B8	B7	B6	
0	0	0	***Tac***
0	0	1	***Talu***
0	1	0	***Tmem***
0	1	1	***Dit***
1	0	0	***Tadbd***
1	X	X	***Nop***

Codificación de las señales al bus de direcciones

Acceso al bus de direcciones		
B10	B9	
0	0	***Tbdad***
0	1	***Tcp***
1	X	***nop***

Señales de la ALU

Selección operación ALU				
B14	B13	B12	B11	
0	0	0	0	***ADD***
0	0	0	1	***SUB***
0	0	1	0	***MUL***

Selección operación ALU				
B14	B13	B12	B11	
0	0	1	1	***DIV***
0	1	0	0	***INC***
0	1	0	1	***Transf***
…	…	…	…	***…***
1	1	1	1	***NOT***

Señales de acceso a la memoria

Mem	***R/W***	
B16	B15	
0	X	***nop***
1	0	***E***
1	1	***L***

Señales de la unidad de direccionamiento

Inc	***Lcp***	
B18	B17	
0	0	***Incrementar CP***
0	1	***CP ← B. Datos***
1	0	***CP ← CP + B.dat***
1	1	***nop***

Banco de registros

Banco de registros			
B21	B20	B19	
0	0	0	***A***
0	0	0	***B***
0	0	1	***C***
0	0	1	***D***
...	...	...	***...***
0	1	1	***H***

Cumple la condición de microsalto

Cond	
B22	
0	***No cumple condición***
1	***Si cumple condición***

La instrucción lleva microsalto

μSalto	
B23	
0	***No lleva microsalto***
1	***Si lleva microsalot***

Bit de secuenciamiento: final de microprograma

Fin µprg	
B24	
0	***No es la última***
1	***Última microinstrucción***

La dirección de microsalto, suponiendo una memoria de control de 32k requeriría un total de 15 bits para especificarla. Tomamos entonces los bits b14-b0 solapando los campos de la ALU, del bus de datos / direcciones y las señales de carga de registros y reset.

Formato de microinstrucción sin salto

Fin µprg	µSalto	Cond	Bango de registros			CP		Memoria		ALU: selección de operación				Bus direcciones		Bus datos			Carga de Registros y Reset					
b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0

Formato de microinstrucción con salto

Fin µprg	µSalto	Cond	Bits no utilizados							Dirección de µsalto condicional o incondicional según valor de b22 o b23														
b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0

Apartado c)

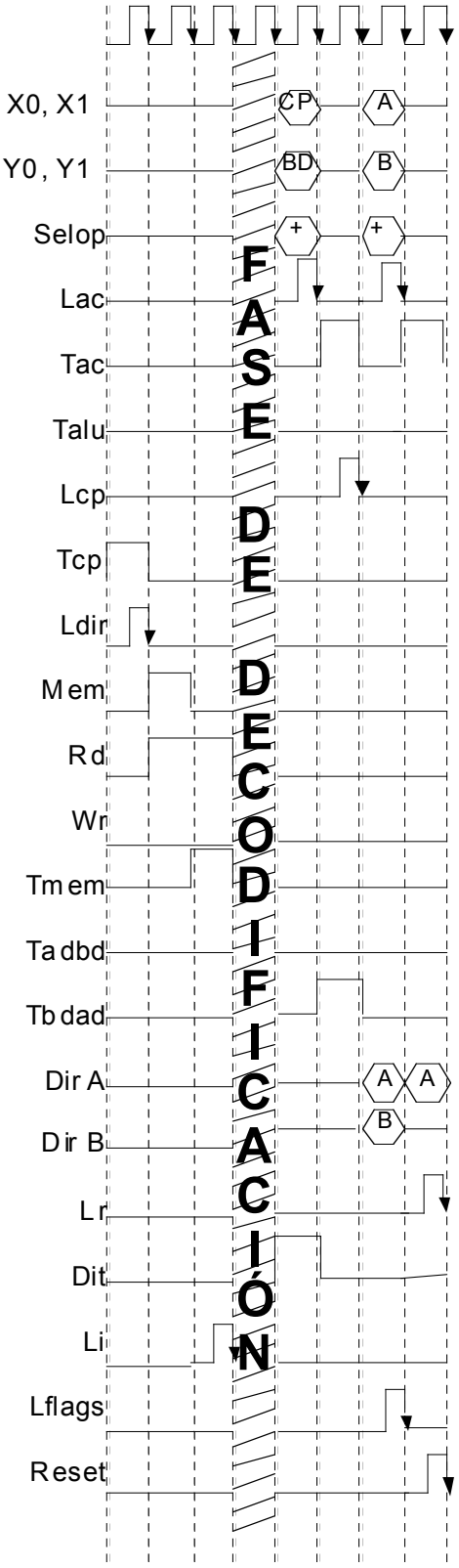
0	0	0	0	0	1	1	1	0	0	0	1	0	1	0	1	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	0	1	0	0	1	0	0	1	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	1	0	1	1	0	1	0	0	0	0
1	0	0	0	0	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	1

Solución ejercicio 5

Apartado a)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
Fase de fetch		
Direccionar	CP → bus de datdir	(TCP)
	Bus datdir→ Reg. Direcciones	(LDIR)
Traer la instrucción	Inicio ciclo de memoria	(MEM)
	Leer	(RD) dos períodos
	Contenido memoria → B. Datos	(TMEM)
	Cargar registro de instrucción	(LI)
Decodificación		1 período
Actualización de CP		
Sumar CP + 1	Tamaño → Bus de datos	(DIT)
	Bus de datos → Multiplexor X	< X0, X1 = CP>
	CP → Multiplexor Y	< Y0 = B.datos >
	Sumar	< Selop = + >
	Cargar el acumulador	(LAC)
Cargar nuevo valor en CP	Acumulador → Bus de datdir	(TAC)
	Cargar el CP	(LCP)
Ejecución de la instrucción		
Realizar el desplazamiento	Elegir registro A en salida A	< DIR A = A>
	Elegir registro B en salida B	< DIR B = B>
	Salida A → Multiplexor X	< X0, X1 = SAL A >
	Salida B → Multiplexor Y	< Y0, Y1 = SAL B >
	Sumar	< Selop = + >
	Cargar el acumulador	(LAC)
	Modificar los flags de estado	(LFLAGS)
Cargar nuevo valor en A	Acumulador → Bus de datos	(TAC)
	Elegir A como registro destino	< DIR A = A >
	Cargar el banco de registros	(LR)
Poner contador de fases a 0	Activar el Reset	(RESET)

Apartado b)

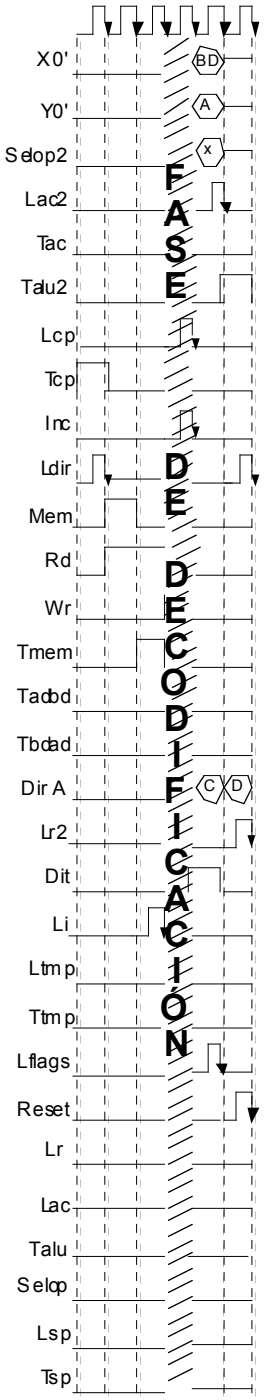

X0, X1
Y0 , Y1
Selop
Lac
Tac
Talu
Lcp
Tcp
Ldir
M em
R d
Wr
Tm em
Ta dbd
Tb dad
Dir A
D ir B
L r
Dit
Li
Lflags
R eset
CP
A
BD
B
+
+
A
A
B
FASE DE DECODIFICACIÓN

Solución ejercicio 7

Apartado a)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
Fase de fetch		
Direccionar	CP → bus de direcciones	(TCP)
	Bus dir.→ Reg. Direcciones	(LDIR)
Traer la instrucción	Inicio ciclo de memoria	(MEM)
	Leer	(RD) dos períodos
	Contenido memoria → B. Datos	(TMEM)
	Cargar registro de instrucción	(LI)
Decodificación y actualizar CP	Incrementar CP	(INC)
	Cargar nuevo valor en CP	(LCP)
Elecución de la instrucción		
Búsqueda de operandos	3.27 → Bus de datos	(DIT)
	B. datos → Multiplexor X0'	< X0' = B. datos >
	Elegir registro C	< DIR A = C>
	Salida A → Multiplexor Y0'	< Y0' = SAL A >
	Multiplicar	< Selop = multiplicar >
	Cargar acumulador 2	(LAC2)
	Actualizar los flags de estado	(LFLAGS)
Cargar nuevo valor en D	Acumulador 2 → Bus de datos	
	Elegir D como registro destino	
	Cargar el banco de registros	(TALU2)
		< DIR A = D >
		(LR)
Poner contador de fases a 0	Activar el Reset	(RESET)

Apartado b)

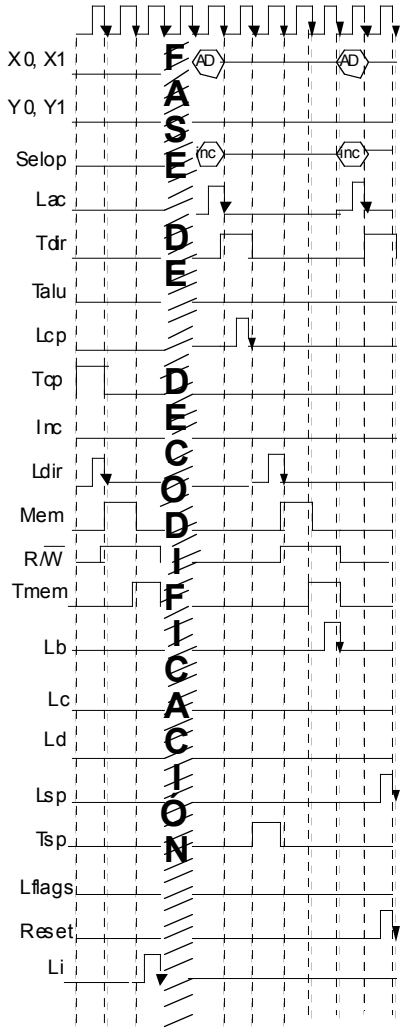

X0'
Y0'
Selop2
Lac2
Tac
Talu2
Lcp
Tcp
Inc
Ldir
Mem
Rd
Wr
Tmem
Tadbd
Tbdad
Dir A
Lr2
Dit
Li
Ltmp
Ttmp
Lflags
Reset
Lr
Lac
Talu
Selop
Lsp
Tsp
BD
A
x
C
D
FASE DECODIFICACIÓN

Solución ejercicio 9

Apartado a)

ACCIÓN	OPERACIÓN ELEMENTAL	SEÑAL DE CONTROL
Fase de fetch		
Direccionar	CP → bus de direcciones	(TCP)
	Bus dir. → Reg. Direcciones	(LDIR)
Traer la instrucción	Inicio ciclo de memoria	(MEM)
	Leer	(R/W) dos períodos
	Contenido memoria → B. Datos	(TMEM)
	Cargar registro de instrucción	(LI)
Decodificación		1 periodo
Actualización de CP	CP → Bus de direcciones	(TCP)
Incrementar CP en 1	Bus dir.→ Multiplexor X	< X0, X1 = B. dir >
	Incrementar	< Selop = INC >
	Cargar acumulador	(LAC)
Cargar nuevo valor en CP	Acumulador → Bus direcciones	(TDIR)
	Bus direcciones → CP	(LCP)
Elecución de la instrucción		
Direccionar SP	SP → bus de direcciones	(TSP)
	Bus dir. → Reg. Direcciones	(LDIR)
Cargar nuevo valor en B	Inicio ciclo de memoria	(MEM)
	Leer	(R/W) dos períodos
	Contenido memoria → B. Datos	(TMEM)
	Cargar registro B	(LB)
Incrementar SP en 1	SP → Bus de direcciones	(TSP)
	Bus dir. → Multiplexor X	< X0, X1 = B. dir >
	Incrementar	< Selop = INC >
	Cargar acumulador	(LAC)
Cargar nuevo valor en SP	Acumulador → Bus direcciones	(TDIR)
	Bus direcciones → SP	(LSP)
Poner contador de fases a 0	Activar el Reset	(RESET)

Apartado b)


X0, X1
Y0, Y1
Selop
Lac
Tdir
Talu
Lcp
Tcp
Inc
Ldir
Mem
R/W
Tmem
Lb
Lc
Ld
Lsp
Tsp
Lflags
Reset
Li
FASE DE DECODIFICACIÓN
AD
AD
inc
inc