UNIDADES FUNCIONALES DEL COMPUTADOR

Tema 5

Principios de Computadoras

Temas

Introducción

1. Bus
2. Memoria
3. Entrada/Salida
4. Unidad aritmética-lógica
5. Unidad de Control

5. Unidad de Control

Tema 5

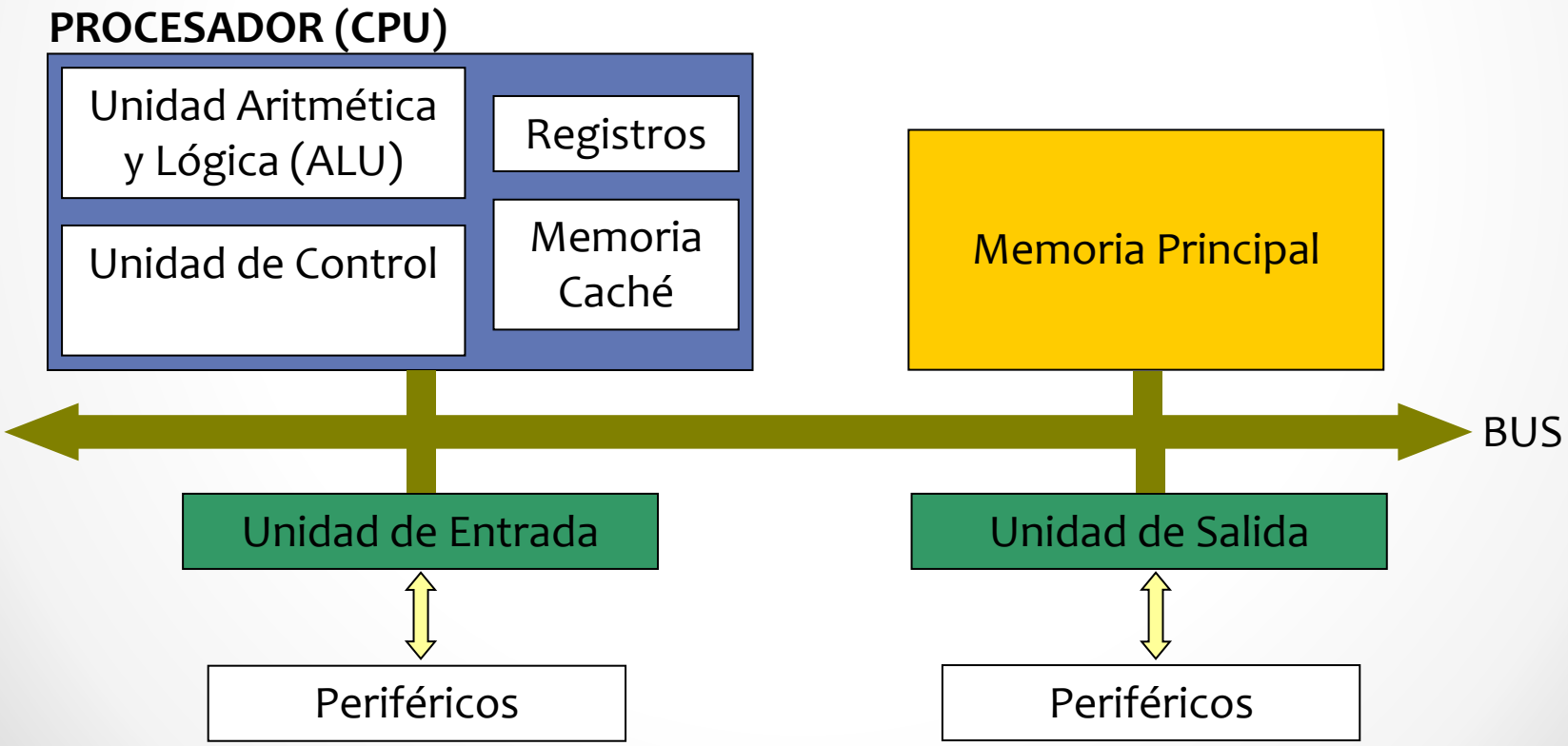
Principios de Computadoras

5. Unidad de Control

- Introducción
- Repaso: Registros del procesador
- Repaso: Búsqueda de instrucciones
- Repaso: Ejecución de una instrucción
- Ciclo de instrucción y microoperaciones
- Camino de datos
- Señales de la Unidad de Control
- Diseño de la Unidad de Control

Introducción

- El procesador (CPU) es el encargado de ejecutar las instrucciones máquina que componen los programas. Está compuesto por: la ALU, el banco de registros y la **UNIDAD DE CONTROL**.



Introducción

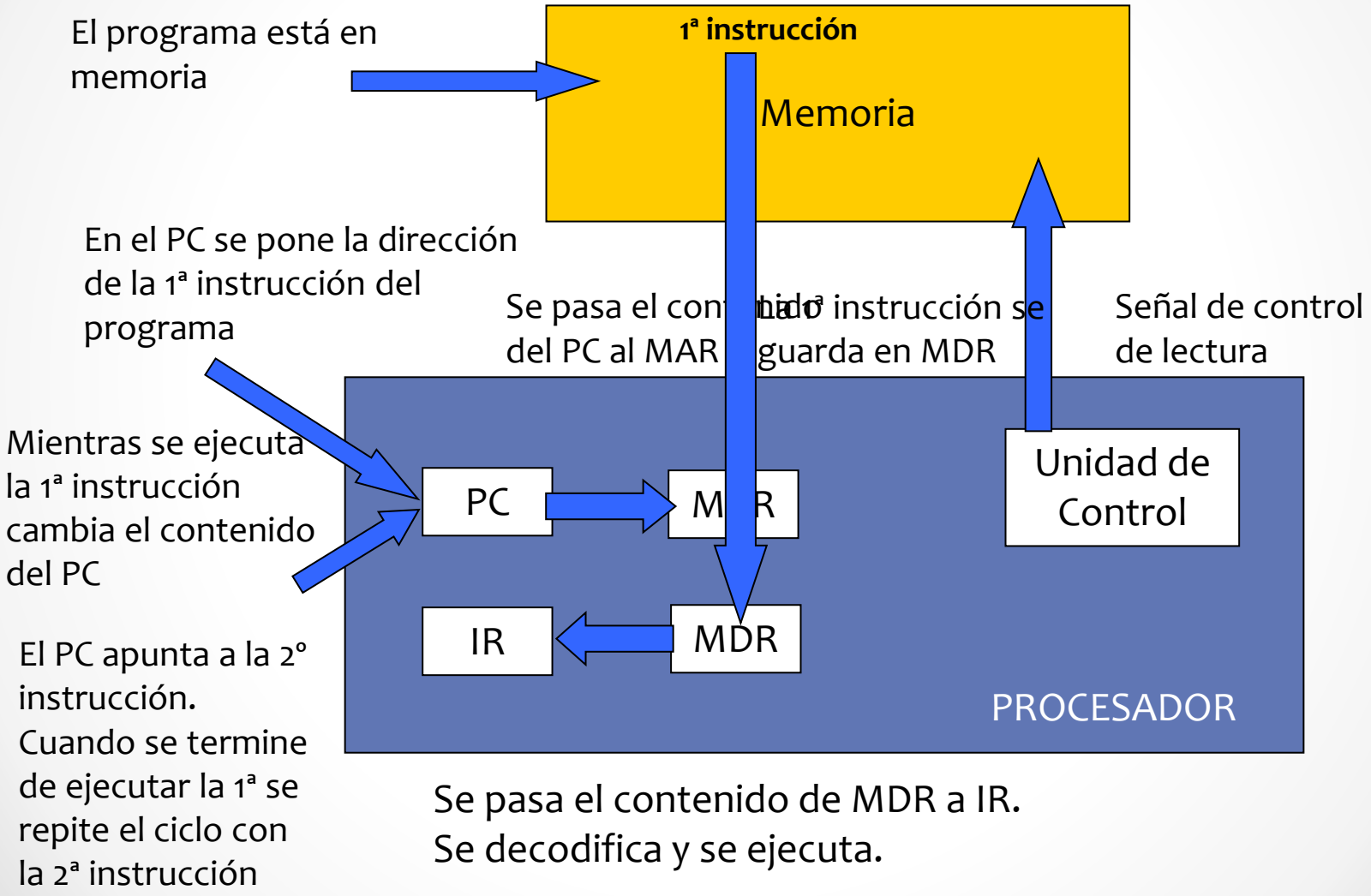
- La **ALU** es la encargada de realizar las operaciones lógicas y aritméticas.
- Los **registros** sirven para almacenar temporalmente datos e instrucciones que residen en la memoria principal del computador (tiempo de acceso a registros menor que a memoria principal).
- Los computadores actuales incluyen dentro del procesador **memorias cachés** para mejorar el rendimiento en el acceso de memoria.
- La **UNIDAD DE CONTROL** es la que controla el funcionamiento de todos los elementos del procesador y del computador.

Repaso...

Repaso: Registros del procesador

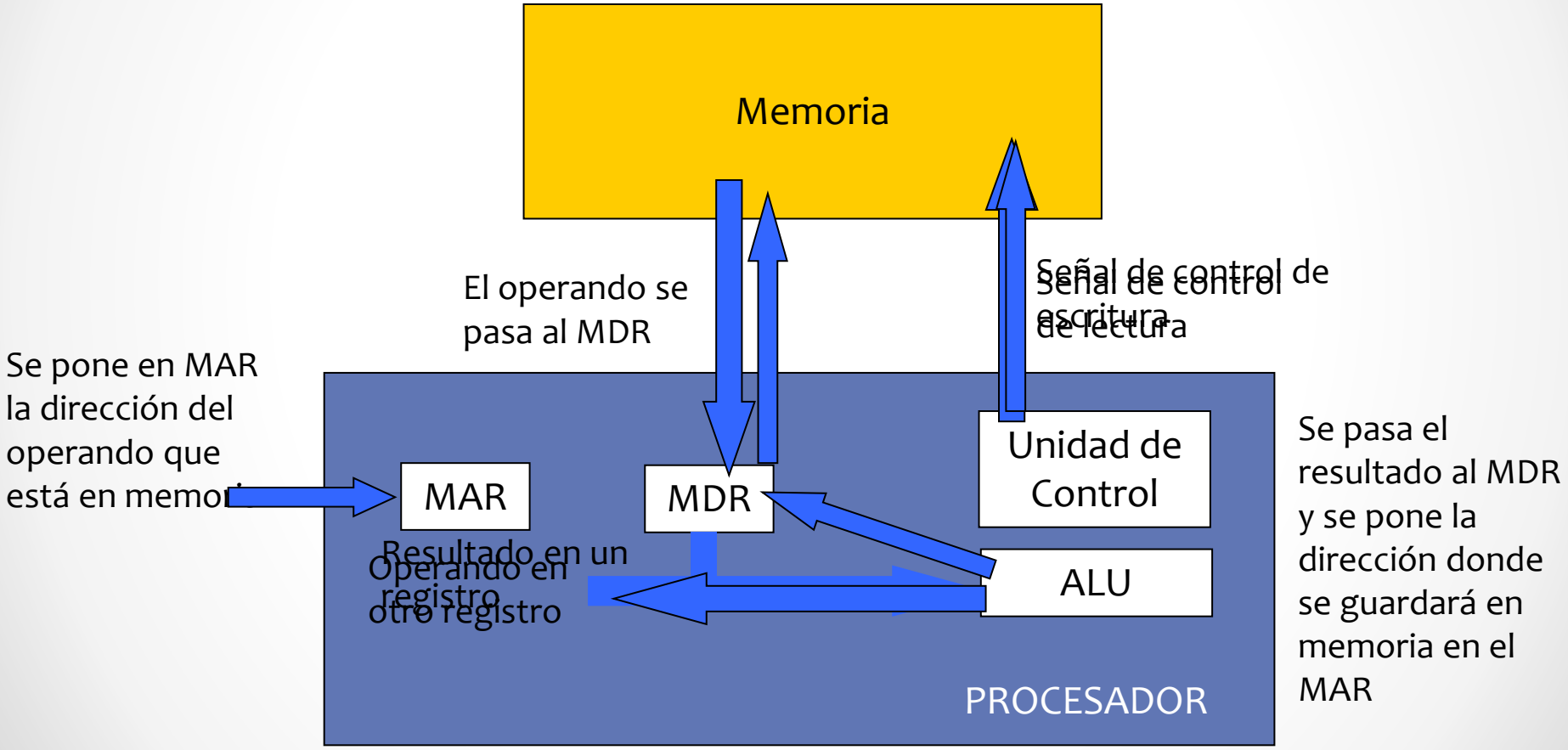
- **Registro de instrucción (IR = *Instruction Register*)**: contiene la instrucción que se está ejecutando en ese momento.
- **Contador de programa (PC = *Program Counter*)**: contiene la dirección de memoria donde está la siguiente instrucción que se ejecutará (PC “apunta” a la siguiente instrucción).
- **Registro de datos de memoria (MDR = Memory Data Register)**: contiene el dato que se ha leído o que se va a escribir en una posición de memoria determinada.
- **Registro de dirección de memoria (MAR = Memory Address Register)**: contiene esa dirección de memoria.
- **Registros de uso general**: n registros (R0 .. Rn-1).

Repaso: Búsqueda de instrucciones



Repaso: Ejecución de una instrucción

Suma de dos operandos: uno en memoria y otro en un registro del procesador



Ciclo de instrucción y microoperaciones

Ciclo de instrucción y microoperaciones

- Es el tiempo que tarda la CPU en ejecutar una instrucción. Se compone de:

  1) **Ciclo de búsqueda de la instrucción**: la instrucción se va a buscar a memoria y se carga en registros.

  2) **Ciclo de ejecución de la instrucción**: se interpreta el código de operación (opcode) y se generan las señales necesarias para realizar la operación que indica la instrucción.

- Cada uno de estos pasos implica la realización de muchas operaciones más simples en el interior de la CPU. Estas operaciones simples son las **MICROOPERACIONES** o **MICROÓRDENES**.

Ciclo de instrucción y microoperaciones

- Las microoperaciones son las operaciones más pequeñas que puede hacer el procesador. Pueden llevar a cabo una de las siguientes acciones:

  1) Transferencia de datos entre registros.

  2) Transferencia de datos entre registros y dispositivos E/S en ambas direcciones.

  3) Realización de una operación aritmética o lógica.

Ciclo de instrucción y microoperaciones

Esquema completo del CICLO DE INSTRUCCIÓN:

1) **Cálculo de la dirección de la siguiente instrucción a ejecutar**: incrementar el contenido del PC (+1 (8 bits), +4 (32 bits), etc). Si hay un salto en el programa, la cantidad a sumar es distinta.

2) **Búsqueda de la instrucción (Instruction Fetch)**: la CPU lee la instrucción desde memoria y la guarda en el IR.

3) **Decodificación de la instrucción (Instruction Decode)**: se extrae el opcode (indica el tipo de operación) y se decodifican las direcciones de los operandos (puede que obtener esta dirección requiera algún tipo de cálculo)

Ciclo de instrucción y microoperaciones

Esquema completo del CICLO DE INSTRUCCIÓN:

4) **Cálculo de la dirección de los operandos**: si alguna instrucción hace referencia a operandos que se encuentran en memoria o en algún dispositivo E/S es necesario calcular la dirección efectiva de los mismos.

5) **Búsqueda de operandos**: se leen los operandos de memoria o de E/S y se guardan en registros de la CPU.

6) **Ejecución de la instrucción**: se lleva a cabo la operación especificada en el opcode con los operandos leídos.

7) **Almacenamiento del operando**: guarda el resultado en memoria o en E/S.

Ciclo de instrucción y microoperaciones

- En una instrucción es posible que algunas etapas no aparezcan (ej.: instrucciones sin resultado que guardar) o que algunas etapas ocurran más de una vez (ej.: operación aritmética con dos operandos, hay que buscar los dos operandos de memoria).

- Falta añadir al esquema anterior una etapa de **comprobación de interrupción**. Una vez ejecutada la instrucción actual la CPU comprueba si hay alguna solicitud de interrupción. Si es así, atiende la interrupción y luego continúa con la búsqueda de la siguiente instrucción.

Ciclo de instrucción y microoperaciones

Esquema completo del CICLO DE INSTRUCCIÓN desglosado en microoperaciones:

1. Ciclo de búsqueda de la instrucción:

Etapa	Microoperación
1	PC → MAR
2	Memoria → MDR
	PC → PC+n
3	MDR → IR

Ciclo de instrucción y microoperaciones

Esquema completo del CICLO DE INSTRUCCIÓN desglosado en microoperaciones:

2. Ciclo de decodificación de la instrucción:

Etapa	Modo directo	Modo indirecto
1	IR (dirección) → MAR	IR (dirección) → MAR
2	Memoria → MDR	Memoria → MDR
3	MDR → RX	MDR → IR (dirección)
4		IR (dirección) → MAR
5		Memoria → MDR
6		MDR → RX

Ciclo de instrucción y microoperaciones

Esquema completo del CICLO DE INSTRUCCIÓN desglosado en microoperaciones:

3. Ciclo de ejecución de la instrucción: es diferente de una instrucción a otra

Ej.: ADD R3, X (suma el contenido de la posición X de memoria al registro R3)

Etapa	Microoperación	Ciclo
1	IR (dirección) → MAR	Búsqueda de operando
2	Memoria → MDR	Búsqueda de operando
3	R3 + MDR → R3	Ejecución de instrucción

Ciclo de instrucción y microoperaciones

Esquema completo del CICLO DE INSTRUCCIÓN desglosado en microoperaciones:

4. Ciclo de almacenamiento del resultado: en este ejemplo se guarda el resultado en la misma dirección de memoria donde se encontraba el operando X.

Etapa	Microoperación
1	IR (dirección) → MAR
2	R3 → MDR
3	MDR → Memoria

Camino de datos

- La unidad de control debe **ejecutar** las microoperaciones en el orden adecuado para que el resultado sea correcto. La temporización de las microoperaciones es la **SECUENCIACIÓN**. Hace falta un reloj para marcar los tiempos de las distintas microoperaciones.

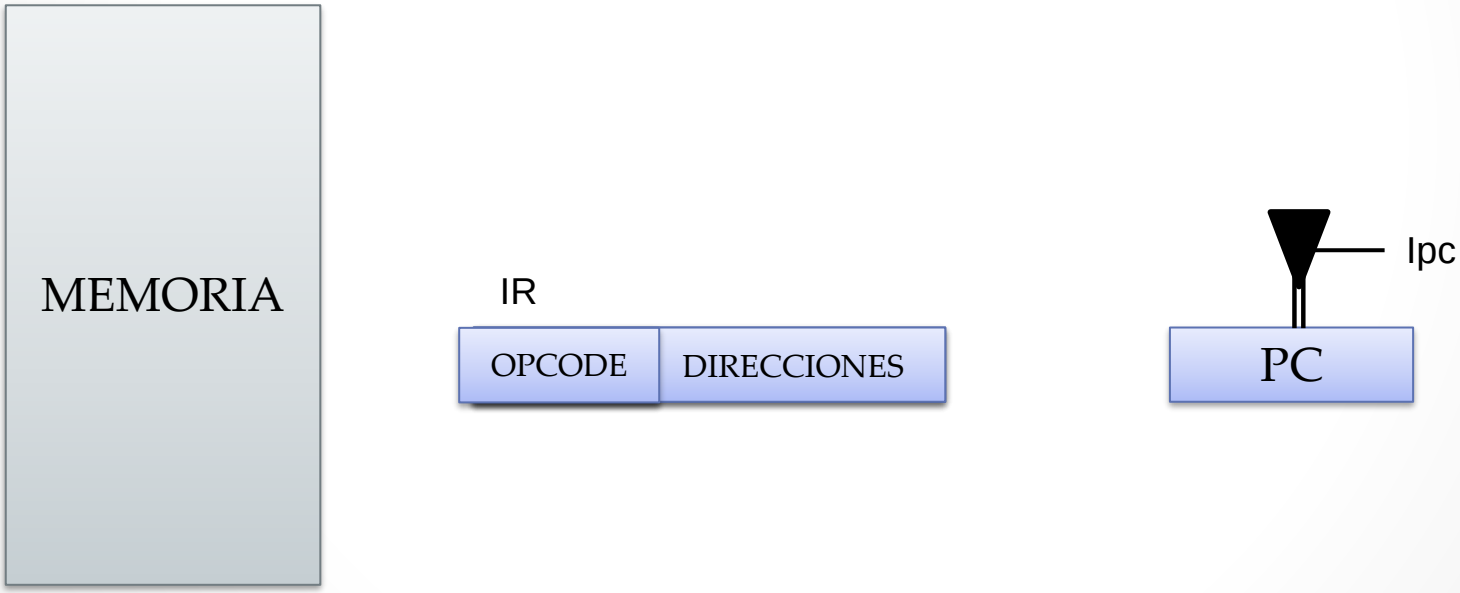
- **CAMINO DE DATOS**: conjunto de elementos necesarios para que se puedan llevar a cabo las instrucciones.

- Es necesario conocer el camino de datos para diseñar correctamente todos los circuitos necesarios para controlarlo.

Camino de datos ejemplo

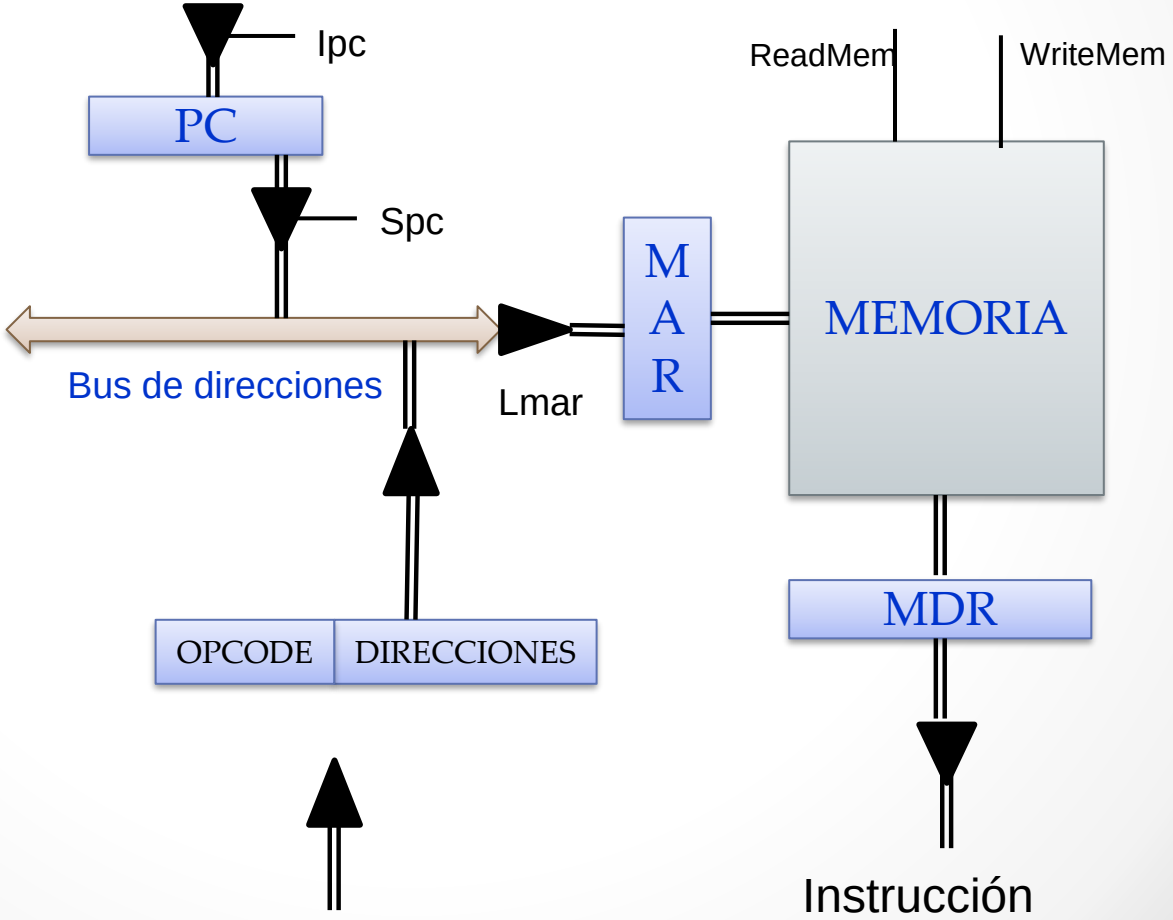
Construcción de un camino de datos

1) Necesitamos un lugar para almacenar las instrucciones, un registro que nos indique la instrucción que se está ejecutando y algún medio que nos permita avanzar a la siguiente instrucción.



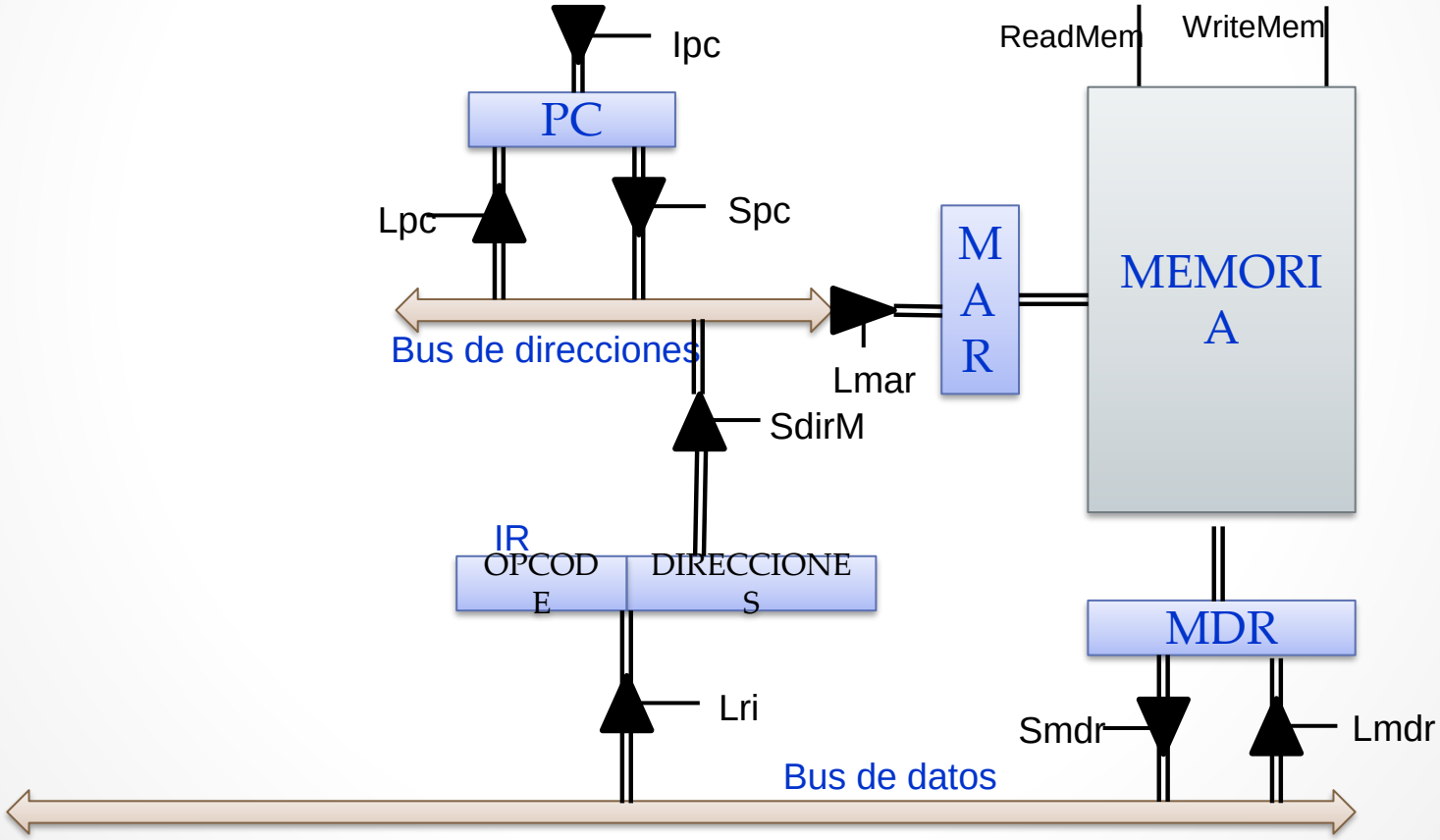
Construcción de un camino de datos

Hardware (camino de datos) que ejecuta la búsqueda y captura de la instrucción (ciclo de fetch)



Construcción de un camino de datos

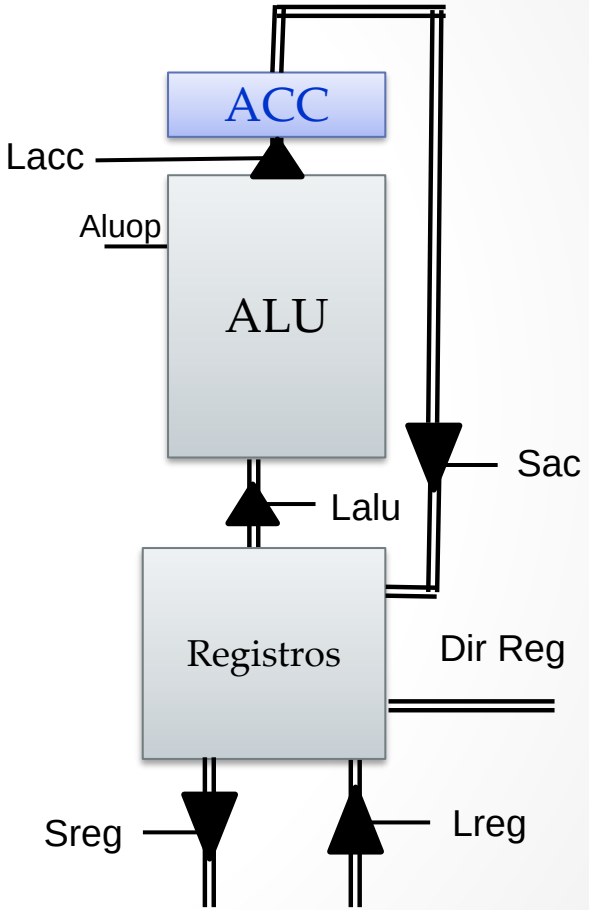
Hardware (camino de datos) que ejecuta la búsqueda y captura de la instrucción (ciclo de fetch)



Construcción de un camino de datos

2) Instrucciones tipo R o aritmético-lógicas: leer dos registros, realizar alguna operación con la ALU sobre el contenido de los registros y escribir el resultado. Ej: *add, sub, and, or.*

En la figura vemos el hardware (camino de datos) que ejecuta instrucciones tipo R



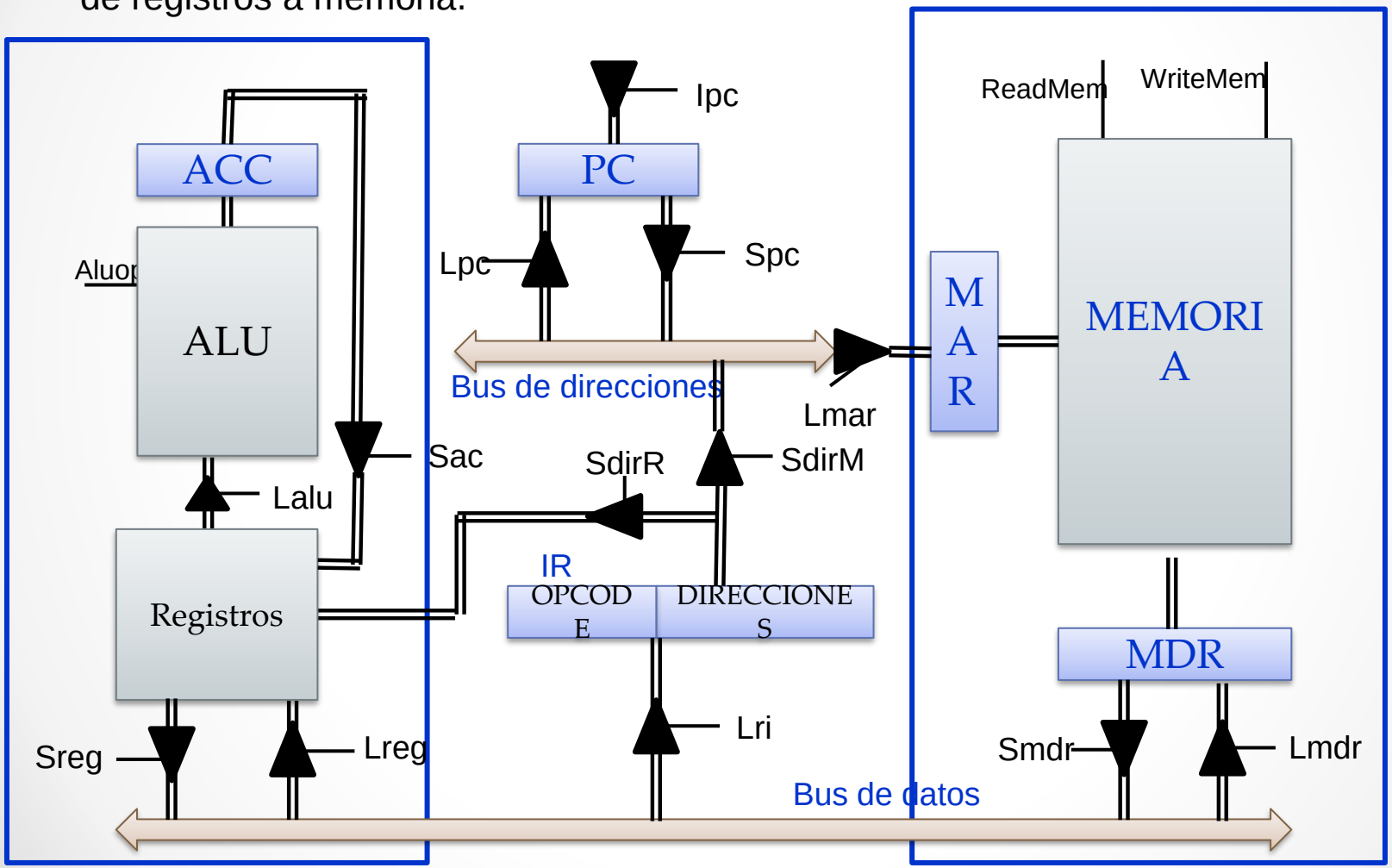
Construcción de un camino de datos

- Las instrucciones aritméticas tienen 3 operandos: por cada instrucción es necesario leer dos palabras como operandos y devolver el resultado.

-En esta máquina simple, los operandos están en memoria y hay que llevarlos a cada una de las entradas de la ALU. El resultado se guarda en el acumulador para posteriormente ser enviado a memoria o realimentado a la ALU.

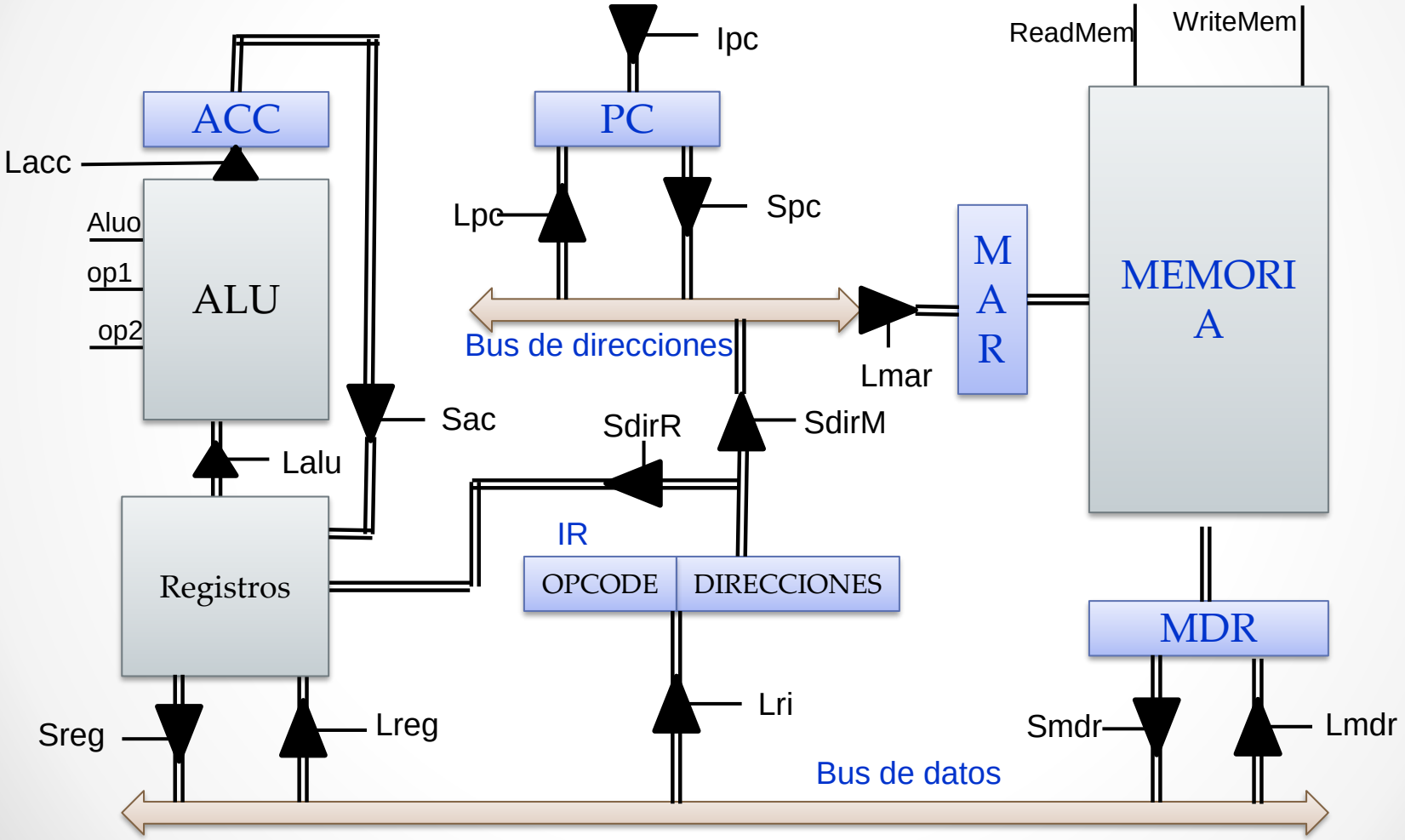
Construcción de un camino de datos

3) Instrucciones de carga y almacenamiento: transferencias de memoria a registro o de registros a memoria.



Construcción de un camino de datos:

Esquema completo

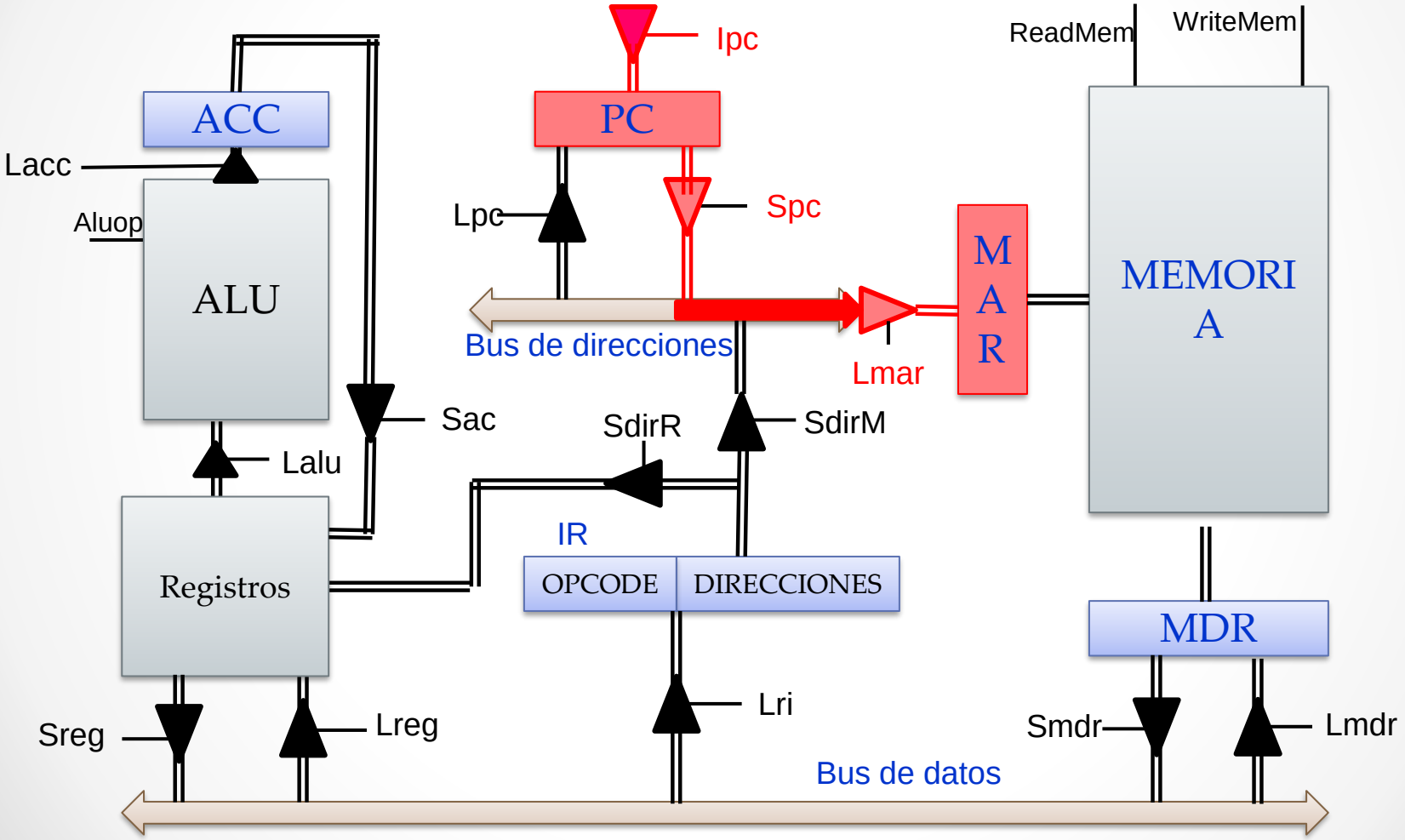


Señales de la Unidad de Control

Señales	Acción
ReadMem	Lectura de Memoria. La instrucción o dato cuya dirección sea la indicada por MAR, pasará al MDR
WriteMem	Escritura de Memoria. El dato que esté en el MDR se escribirá en la posición de memoria cuya dirección indique MAR
Smdr	Save MDR. Transfiere el dato del MDR al bus de datos
Lmdr	Load MDR. Carga el dato del bus de datos al registro MDR
Lri	Load IR. Carga el IR con el dato que está en el bus de datos
SdirM	Store dir a Memoria. Transfiere la dirección, que se encuentra en el campo de direcciones del IR, al bus de direcciones.
SdirR	Store dir a Registro. Selecciona el registro cuya dirección se encuentra en el campo de direcciones del IR.
Lmar	Load MAR. Carga el registro MAR con la dirección que se encuentre en el bus de direcciones.
Ipc	Ipc. Incrementa el registro PC.
Lpc	Load PC. Carga el registro PC, con la dirección que se encuentre en el bus de direcciones
Spc	Store PC. Transfiere la dirección que se encuentra en el PC al bus de direcciones.
Sac	Store ACC. Transfiere el dato que está en el acumulador al banco de registros.
Lalu	Load ALU. Carga el dato que se encuentra en el banco de registros a una de las entradas de la ALU.
Sreg	Store Reg. Guarda el dato que está en un registro en el bus de datos
Lreg	Load Reg. Carga el dato que está en el bus de datos en un registro
Aluop	Operación de la ALU. Indica qué operación aritmética o lógica se va a llevar a cabo en la ALU.

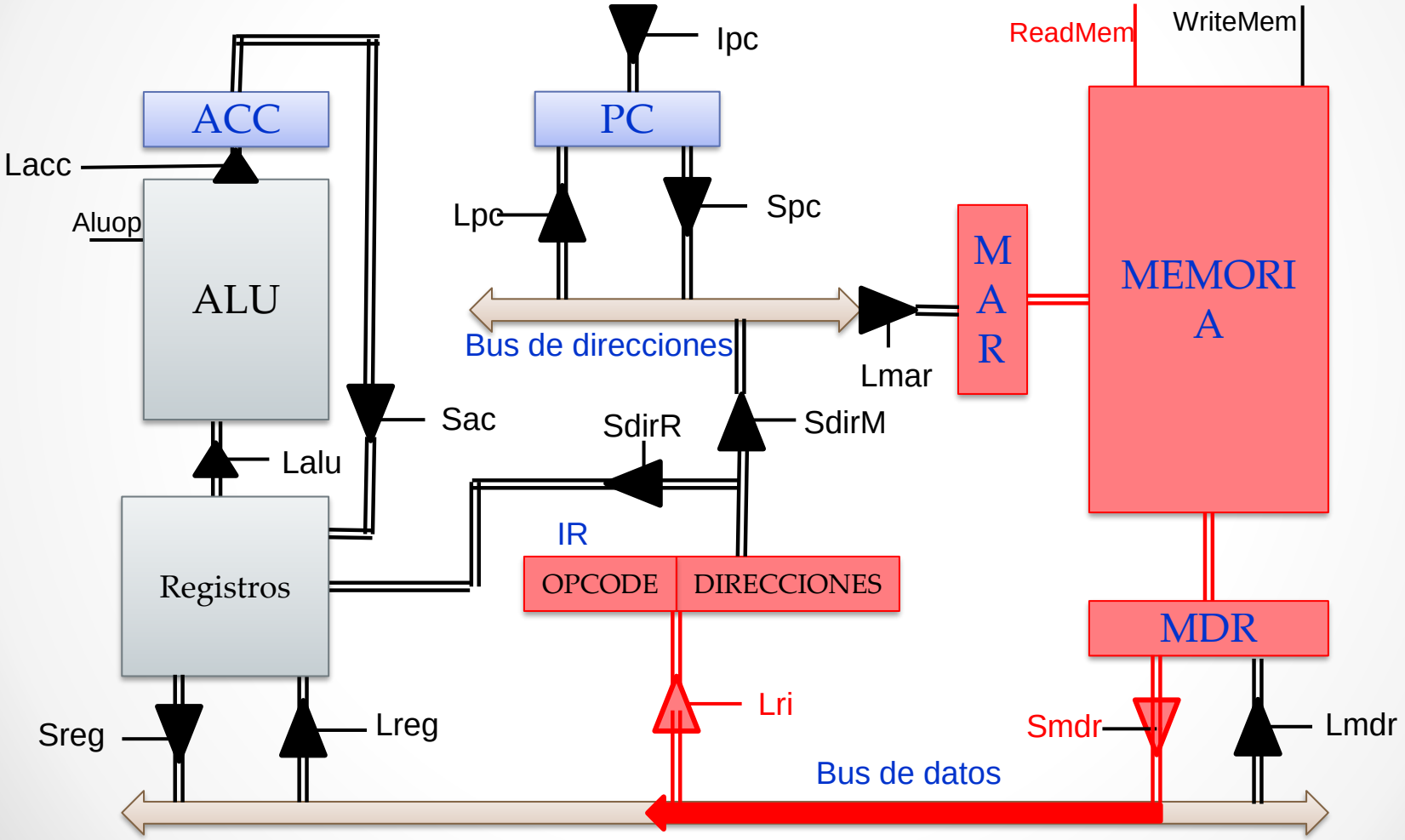
Operaciones

El Contenido del registro contador de programa pasa al MAR. Se incrementa el PC



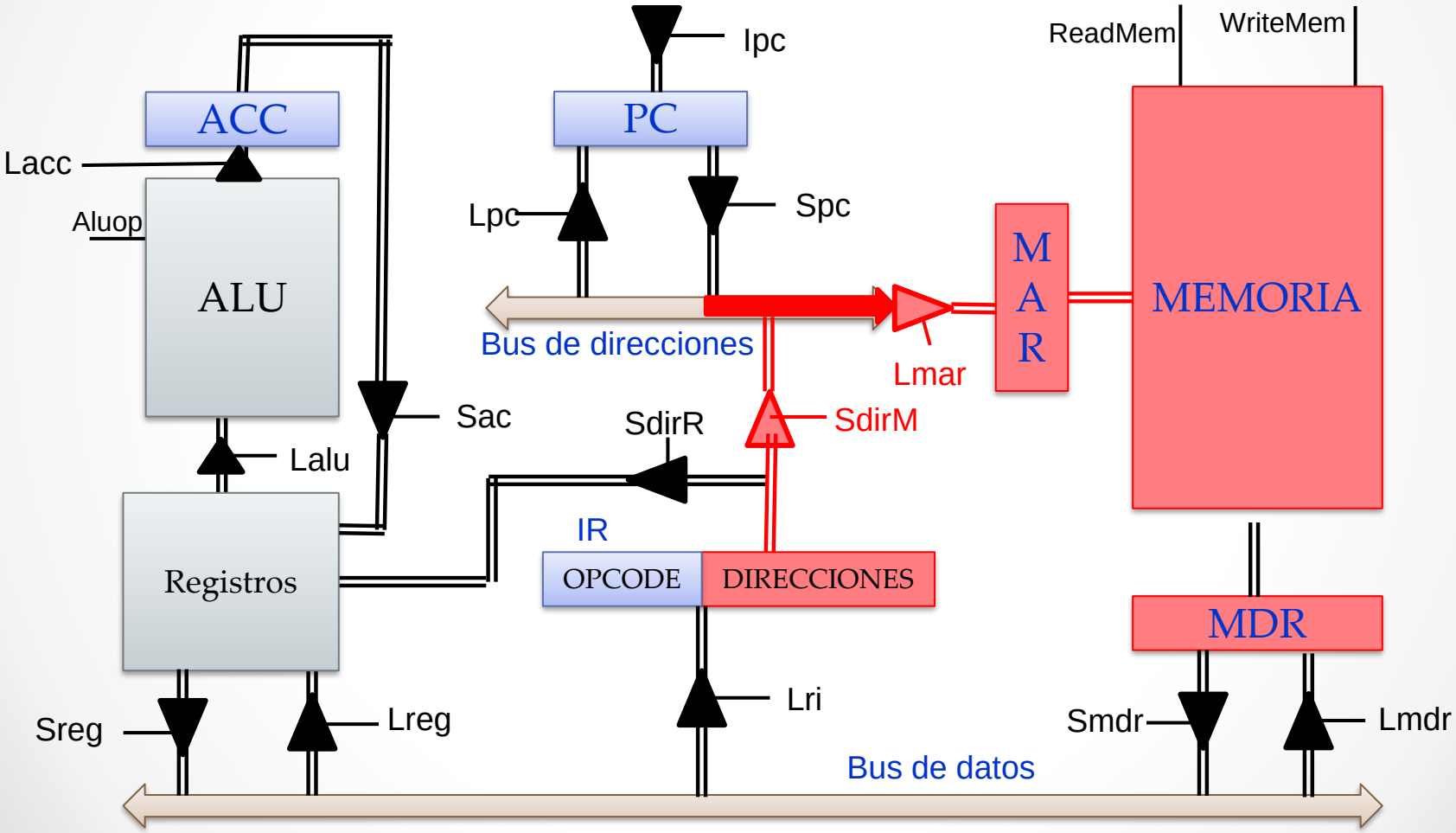
Operaciones

Se lee (ReadMem) la dirección de memoria indicada en el MAR y el contenido se pasa al MDR. Activando SMDR pasa al bus de datos. Activando LRI se guarda en el IR



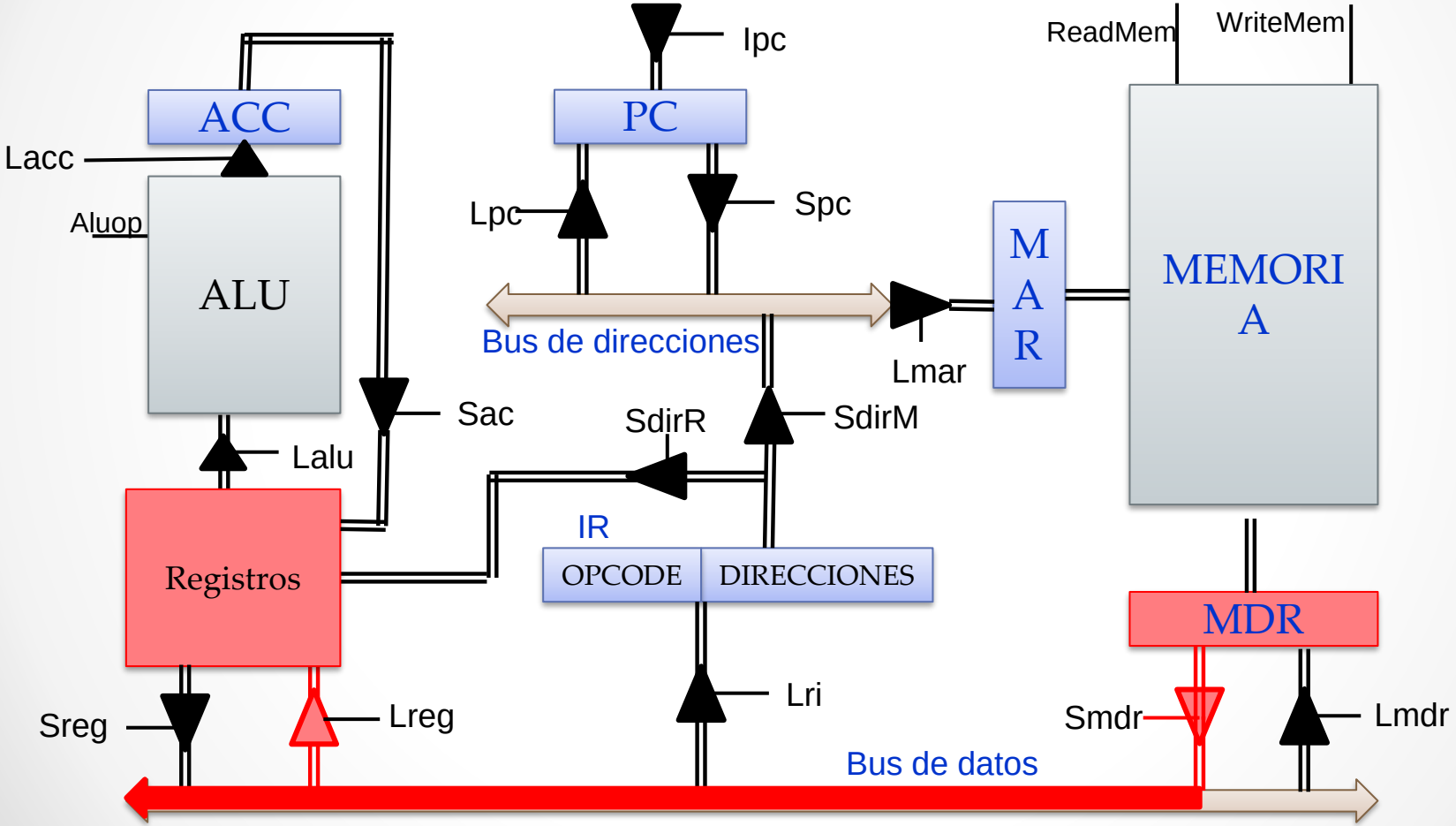
Operaciones

Sdir; hace que el campo de dirección pase al bus de direcciones. Lmar guarda el contenido del bus en el MAR. Readmem guarda el contenido de la memoria en MDR.



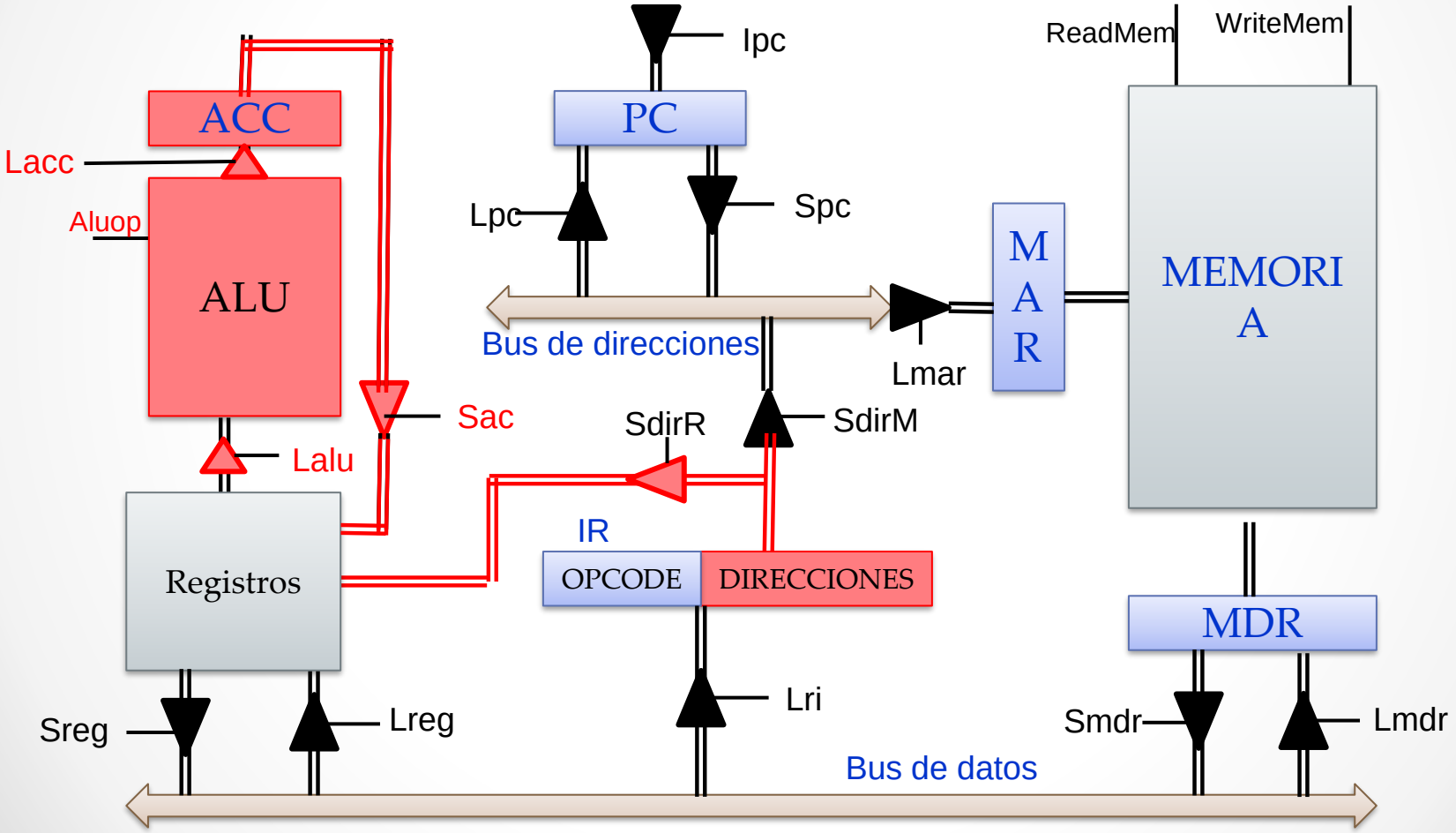
Operaciones

Smdr hace que el MDR pase al bus de datos. Lreg hace que el contenido del bus de datos pase a un registro.



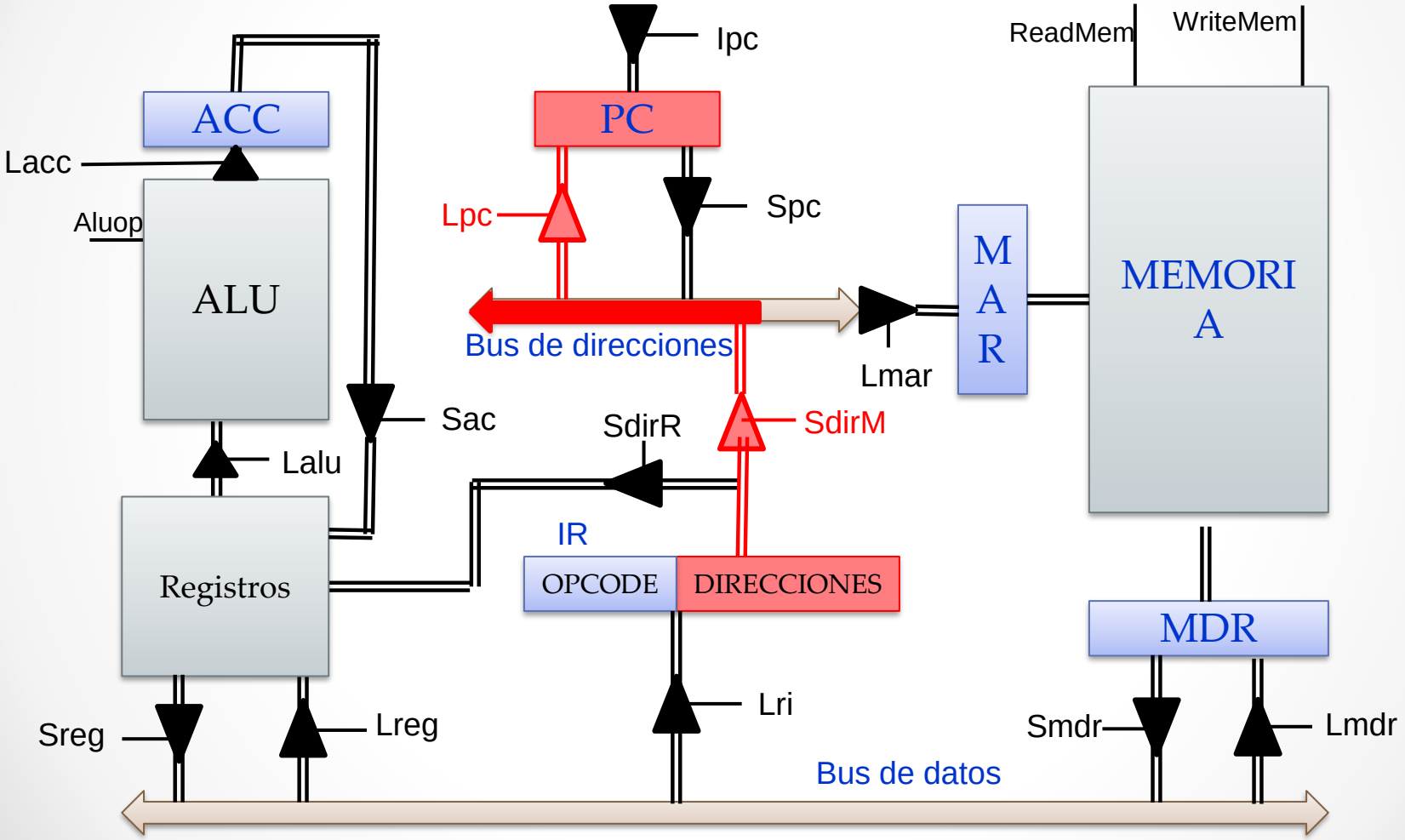
Operaciones

SdirR hace que el campo de direcciones seleccione un registro. Lalu carga el contenido de ese registro en la ALU. Aluop indica la operación a realizar. Lacc carga el resultado en el acumulador. Sac guarda el contenido del acumulador en un registro.



Operaciones

Activando Sdir, el campo de direcciones pasa al bus de direcciones. Activando LPC, el contenido del bus de direcciones pasa al PC. Esto nos permite hacer un salto.



Microoperaciones y Señales de control

El opcode de una instrucción determinada tendrá que activar las señales de control necesarias para que se ejecuten todas las microoperaciones necesarias. Para esta máquina, las microoperaciones más importantes son las siguientes. Hay que destacar que el PC se incrementa después de haber llevado el valor actual al MAR y que Aluop no es una única señal sino varias que permiten determinar qué operación hay que realizar.

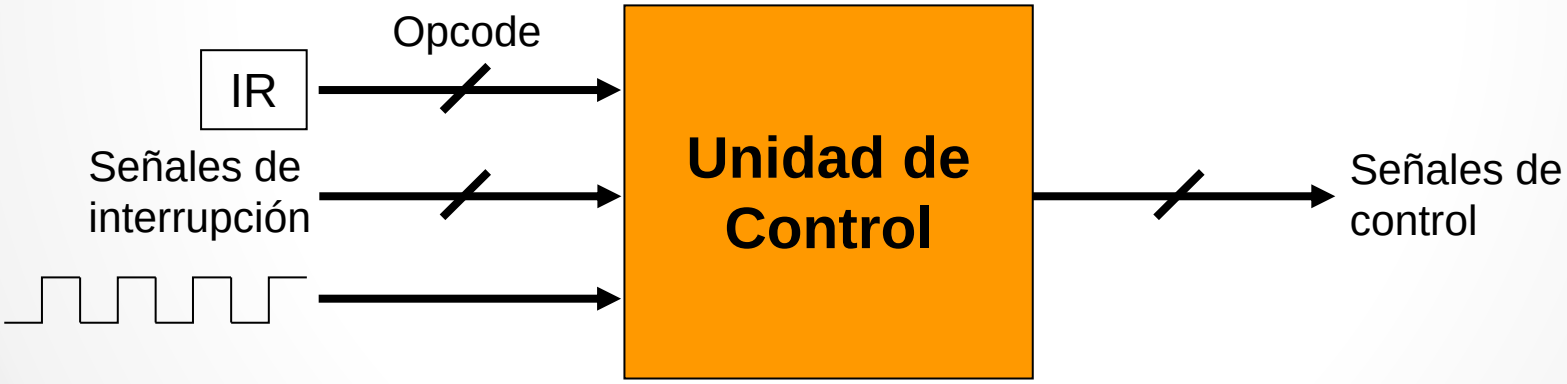
OPERACIÓN	IPC	LPC	SPC	Lmar	ReadMem	Smdr	Lri	SdirM	Lreg	SdirR	Lalu	Lacc	AluOp	Sacc
Búsqueda de instrucción														
MAR← PC	1		1	1										
MDR← MEM(MAR)					1									
IR ← MDR						1	1							
Búsqueda operando en memoria														
MAR ← IR(dirM)				1				1						
Reg ← MDR														
Búsqueda operando en registro														
Reg ← IR(dir)										1				
Instrucción aritmética														
Instrucción aritmética											1	1	1	1
Salto														
Salto		1						1						

Diseño de la unidad de control

Diseño de la Unidad de Control

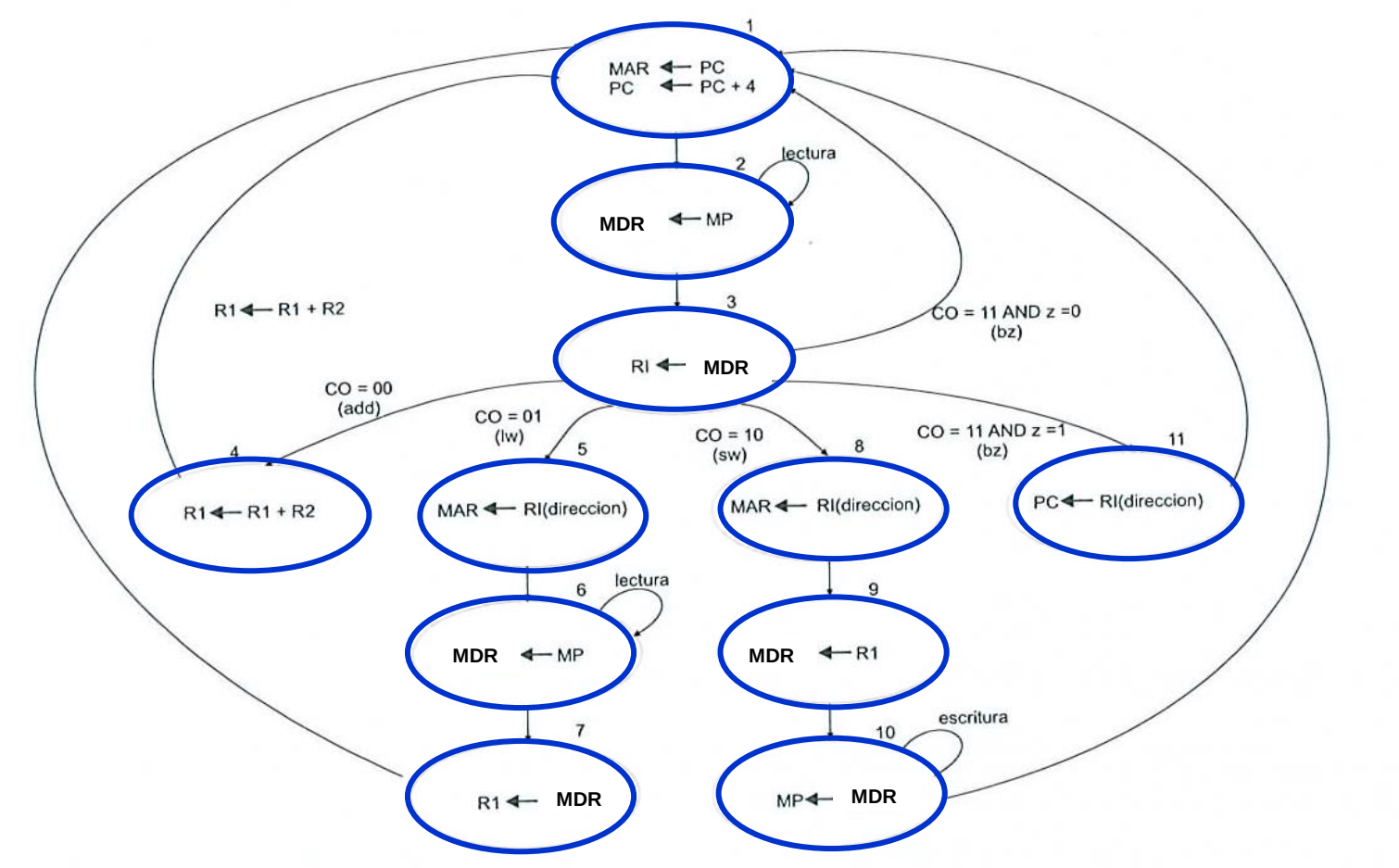
- La unidad de control genera el valor de todas las señales dependiendo del valor de las siguientes entradas:

  - código de operación (opcode), que se encuentra en IR.
  - señales de interrupción
  - reloj (señal de temporización)



Diseño de la Unidad de Control

- Para diseñar la unidad de control es necesario obtener primero la máquina de estados que define su funcionamiento.



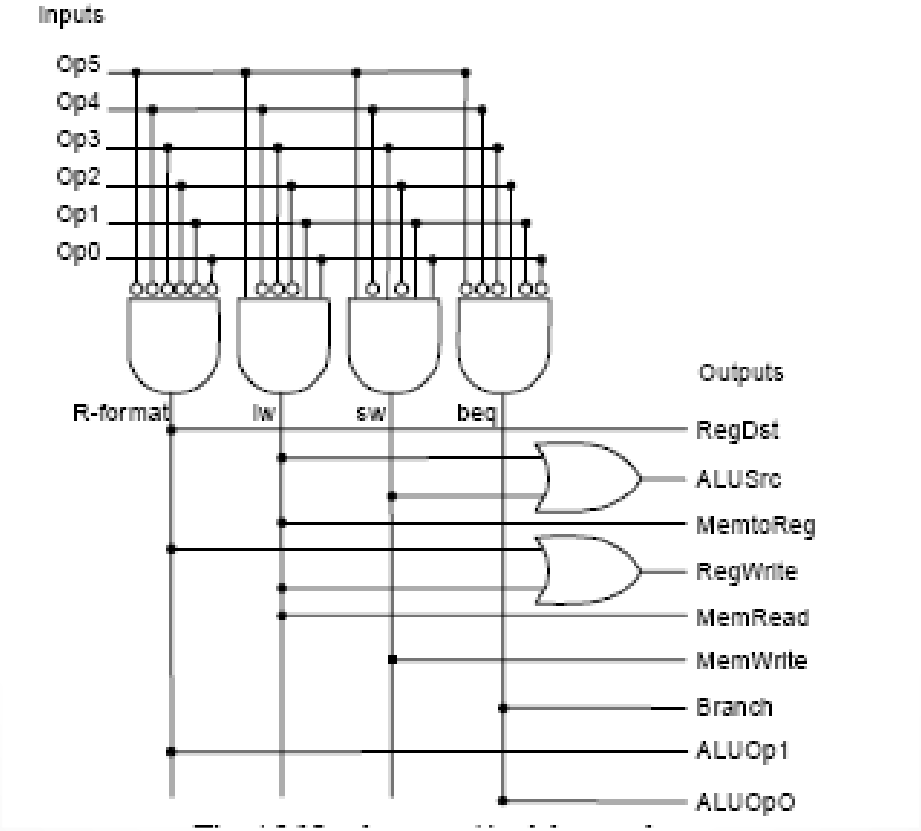
Diseño de la Unidad de Control

- Cada estado de la máquina se corresponde con un ciclo de reloj, durante el cual la unidad de control debe mantener activadas las señales de control necesarias para ejecutar la operación elemental correspondiente.

- A partir de la máquina de estado hay dos formas de abordar el diseño de la Unidad de Control: **UC cableada** y **UC microprogramada**

Diseño de la Unidad de Control

1. UNIDAD DE CONTROL CABLEADA: se construye mediante puertas lógicas con métodos de diseño de circuitos combinacionales y secuenciales



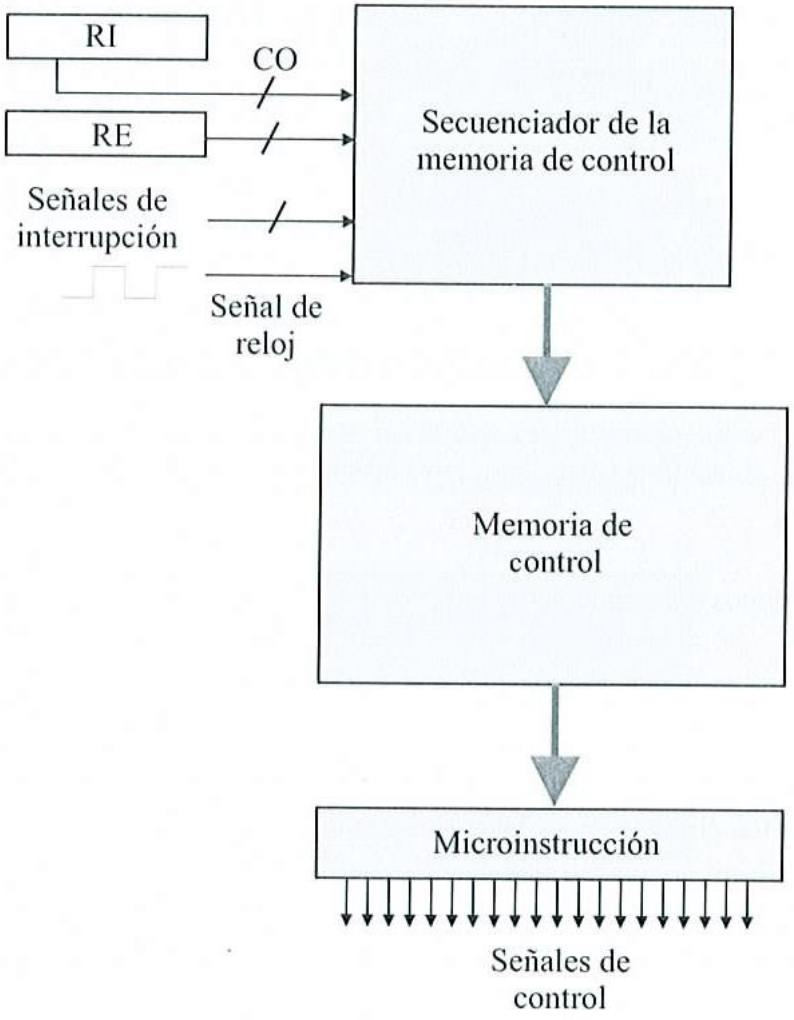
Diseño de la Unidad de Control

2. UNIDAD DE CONTROL MICROPROGRAMADA: utiliza una memoria de control para almacenar el estado de las diferentes señales de control durante cada uno de los ciclos de cada instrucción. En este tipo de UC:

- *Microinstrucción*: conjunto de bits que identifican las señales de control que debe generar la UC en cada ciclo de reloj.
- *Microprograma*: secuencia ordenada de microinstrucciones que debe generar la UC para ejecutar cada instrucción máquina.
- *Firmware o Microcódigo*: conjunto de todos los microprogramas de un computador.

Diseño de la Unidad de Control

Estructura de una unidad de control microprogramada:



- *Memoria de control:* almacena todas las microinstrucciones correspondientes al juego de instrucciones del computador.
- *Secuenciador*: en cada ciclo de reloj el secuenciador genera la dirección de la memoria de control donde se encuentra almacenada la microinstrucción a ejecutar en ese ciclo de reloj.