Unidad 2:
LA ARQUITECTURA DE UN COMPUTADOR

BLOQUE I - Los fundamentos de la Informática y de la Seguridad

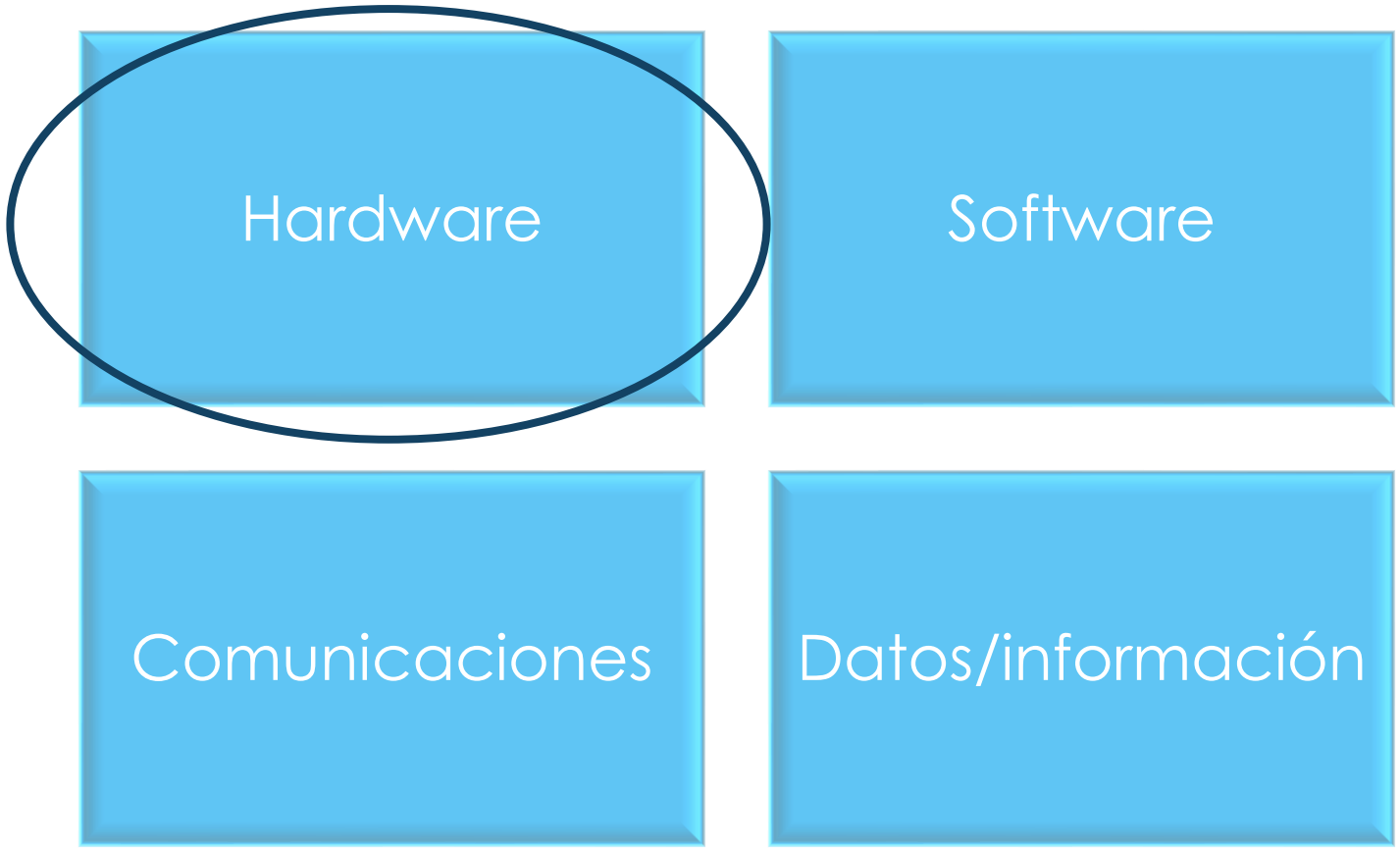
CONTENIDOS

1. Tipos de computador.
2. Modelo Von Neumann.
3. Lenguaje ensamblador.
4. ¿Cómo se ejecuta una instrucción?

1. Tipos de computador

- ¿Recordáis lo que hablamos en la unidad anterior?

1. Tipos de computador



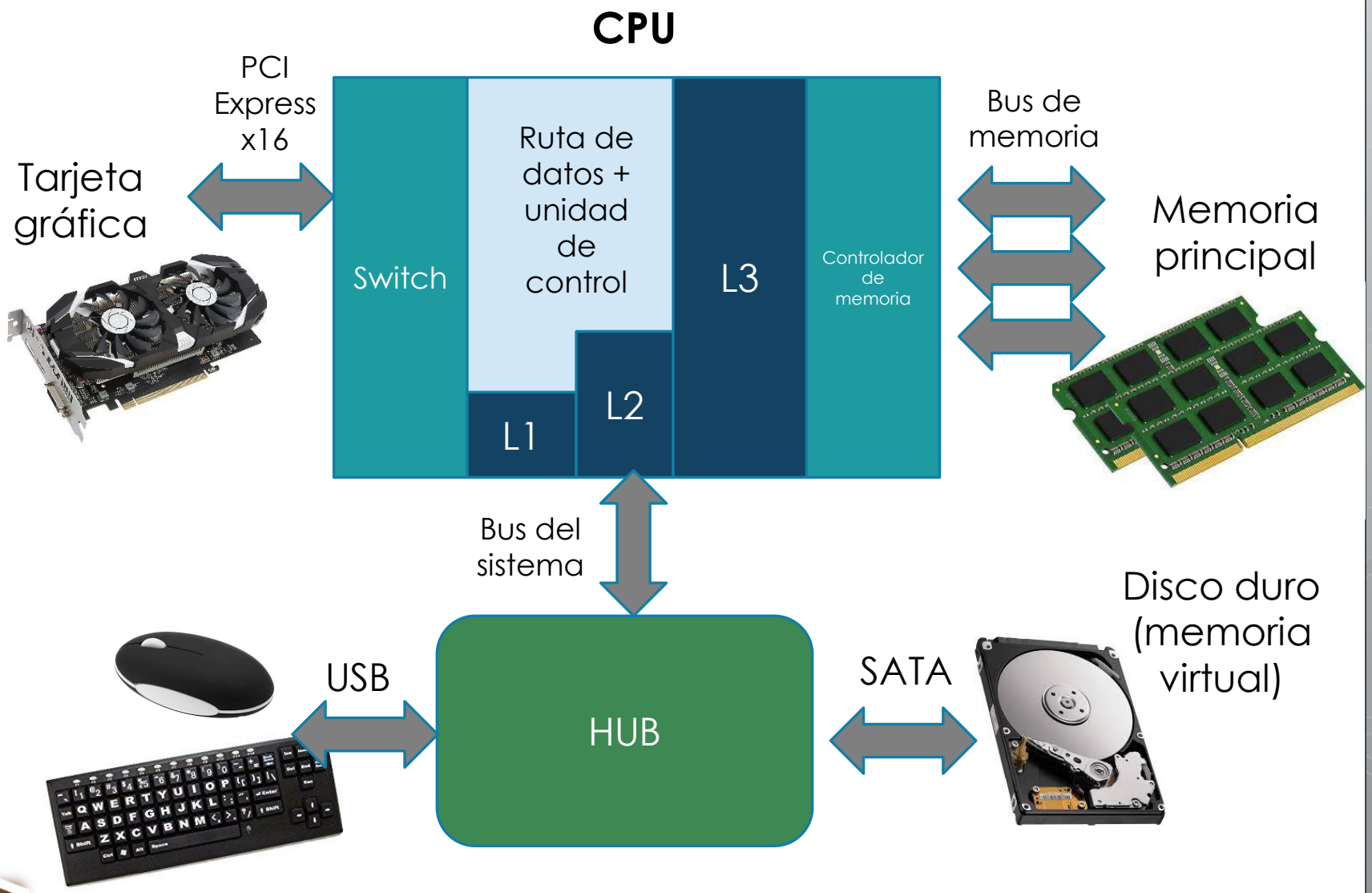
1. Tipos de computador

- Vamos a centrarnos, de momento, en los computadores personales, servidores, ordenadores portátiles, etc.
- Pregunta: ¿Has abierto/desmontado alguna vez un ordenador?

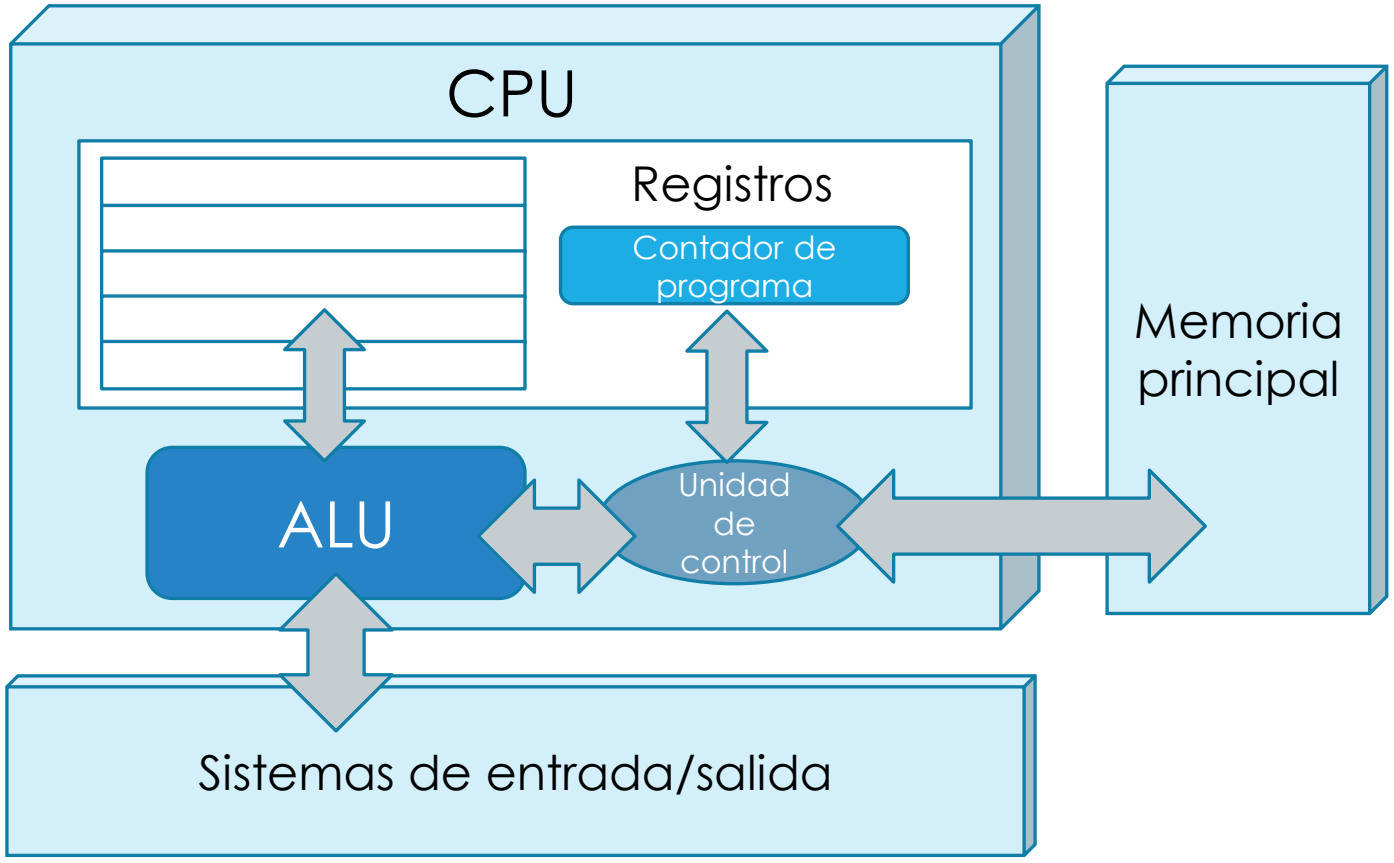
¿Qué diferencia al PC de otros tipos de computador? Se han definido estándares que todos los fabricantes siguen para abaratar costes

1. Tipos de computador

- Pregunta: ¿Puedes dibujar un esquema de conexión de los componentes principales de esta placa base?


CPU
PCI Express x16
Tarjeta gráfica
Switch
Ruta de datos + unidad de control
L1
L2
L3
Controlador de memoria
Bus de memoria
Memoria principal
Bus del sistema
HUB
USB
SATA
Disco duro (memoria virtual)

2. Modelo Von Neumann



2. Modelo Von Neumann

- Según este modelo en un computador:
  - Tanto los programas como los datos se almacenan en una memoria común. Las instrucciones de los programas y los datos tienen diferente modo de uso, pero su estructura no se representa en memoria de manera codificada.
  - Cada posición de memoria se identifica con un número único, llamado dirección. Hacen falta instrucciones para leer y escribir en estas direcciones de memoria.
  - Cada programa se ejecuta de forma secuencial comenzando por la primera instrucción. Para cambiar esta secuencia se utilizan instrucciones específicas de control de flujo.

2. Modelo Von Neumann

- Pregunta: ¿Los ordenadores que utilizamos en nuestra vida cotidiana siguen este modelo todavía?

3. Lenguaje ensamblador

- Según el modelo Von Neumman para escribir programas informáticos es necesario que el computador pueda ejecutar, como mínimo, tres tipos de instrucciones diferentes. Esto se sigue cumpliendo en la actualidad:

Accesos a memoria (lecturas y escrituras)	Operaciones aritmético/lógicas	Control de flujo

3. Lenguaje ensamblador

- En los primeros computadores, los programadores empleaban directamente este lenguaje máquina o ensamblador, a muy bajo nivel, para escribir sus programas.
- Con el paso del tiempo, se comenzó a trabajar con lenguajes de alto nivel y con herramientas de tipo compilador que traducen estos programas escritos en alto nivel a lenguaje máquina o ensamblador.
  - Lo veremos más adelante en la asignatura.

3. Lenguaje ensamblador

- El funcionamiento y diseño de un computador está completamente determinado por el repertorio de instrucciones máquina o ensamblador que debe ejecutar (el conjunto de instrucciones permitido o ISA, Instruction Set Architecture).
  - El procesador debe ser capaz de ejecutar correctamente todas y cada una de las instrucciones incluidas en este repertorio.
- Por eso se suele comenzar el diseño de un nuevo procesador diseñando su repertorio de instrucciones.

3. Lenguaje ensamblador

- Hasta los años 80 la mayor parte de los repertorios de instrucciones eran de tipo CISC (Complex Instruction Set Computer).
- Las arquitecturas de tipo CISC manejan repertorios con un gran número de instrucciones complejas.
  - Con gran variedad de tipos de datos, de operaciones, etc,
- Esto permite implementar instrucciones de alto nivel directamente o con un número pequeño de instrucciones ensamblador.

3. Lenguaje ensamblador

- Pero a partir de ese momento la tendencia comenzó a cambiar, imponiéndose los repertorios de tipo RISC (Reduced Instruction Set Computer).
  - Los procesadores actuales o son completamente RISC o, aunque aparezcan como CISC hacia el exterior, en realidad mantienen en su núcleo una arquitectura que incorpora las técnicas típicas de diseño RISC (como es el caso de las arquitecturas x86).
- En el caso de las arquitecturas RISC, el repertorio está compuesto por pocas instrucciones y muy básicas.

3. Lenguaje ensamblador

Tipo de operación	Ejemplos
Aritmético-lógicas	add,substract,and,or
Acceso a memoria	load,store
Control de flujo	branch,jump
Sistema	Llamadas al SO
Coma flotante	add,divide,compare
Cadenas de caracteres	compare,cat

Estudiando las frecuencias de aparición de las instrucciones en arquitecturas RISC se suele ver que las más utilizadas con las más sencillas, por orden: load, branch, compare, store, add, and.

3. Lenguaje ensamblador

- Ejemplo:
  - Vamos a sumar cuatro números enteros de manera que resultado = A+B+C+D.
  - Tenemos a continuación los códigos ensamblador para dos arquitecturas diferentes:
    - Arquitectura x86 (ensamblador IA32) como ejemplo de arquitectura CISC.
    - Arquitectura nanoMIPS (ensamblador subconjunto del MIPS64) como ejemplo de arquitectura RISC.
  - Incluso en una operación tan sencilla se observan diferencias ya que se toman decisiones de diseño diferentes en procesadores que ejecutan los dos tipos de repertorio.

3. Lenguaje ensamblador

Arquitectura x86

```
mov A, %eax
add B, %eax
add C, %eax
add D, %eax
mov %eax, resultado
```

Arquitectura nanoMIPS

```
lw R1,A(R0)
lw R2,B(R0)
lw R3,C(R0)
lw R4,D(R0)
add R1,R1,R2
add R3,R3,R4
add R1,R1,R3
sw R1,resultado(R0)
```

Arquitecturas RISC	Fabricante	Última versión
MIPS	MIPS Technologies	MIPS VI (año 2014)
PowerPC	IBM	Power 10 (año 2021)
Alpha	HP	EV7 (año 2003)
SPARC	Sun Microsystems	SPARC M8 (año 2017)
ARM	ARM Holdings	ARM v9 (año 2021)

Arquitecturas híbridas CISC/RISC (x86)	Fabricante	Última versión
Core	Intel	Core i9 (año 2017)
Zen+	AMD	Ryzen (año 2019)

4. ¿Cómo se ejecuta una instrucción?

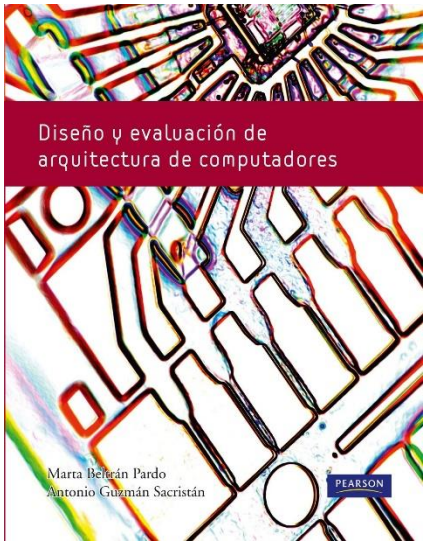
- Las evoluciones principales que se observan en la actualidad respecto del modelo Von Neumman son:
  - Los procesadores son capaces de ejecutar varias instrucciones al mismo tiempo, de desordenar su ejecución (parece que es secuencial, pero no lo es en realidad), de planificar sus recursos de manera eficiente, de predecir lo que va a pasar.
  - No se utiliza una única memoria, sino una jerarquía: memoria caché, memoria principal, memoria virtual.
  - Las instrucciones y los datos no se almacenan juntos, por lo menos en los primeros niveles de esta jerarquía.

4. ¿Cómo se ejecuta una instrucción?

- La mayor parte de computadores actuales se basan en arquitecturas de 32 ó 64 bits. Esto significa que:
  - Los registros internos del procesador son de 32 ó 64 bits.
  - La palabra de memoria es de 32 ó 64 bits.
  - La dirección de memoria es de 32 ó 64 bits.
  - La longitud de los operandos habituales es de 32 ó 64 bits.
  - La longitud de las instrucciones ensamblador es de 32 ó 64 bits.

4. ¿Cómo se ejecuta una instrucción?

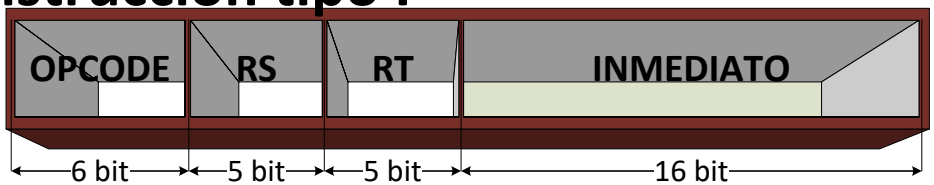
- Supongamos que trabajamos con una arquitectura de 32 bits denominada nanoMIPS (ejemplo de la diapositiva 19 ¿te acuerdas?).
  - Que funciona sin ningún tipo de optimización para el procesador, ejecuta las instrucciones secuencialmente (en orden) y de una en una.
- Se trata de un repertorio de instrucciones muy sencillo en el que sólo están presentes las siguientes instrucciones para números enteros:
  - Acceso a memoria: LW y SW.
  - Operaciones aritmético-lógicas: ADD, SUB, AND, OR y SLT.
  - Control de flujo: BEQ.



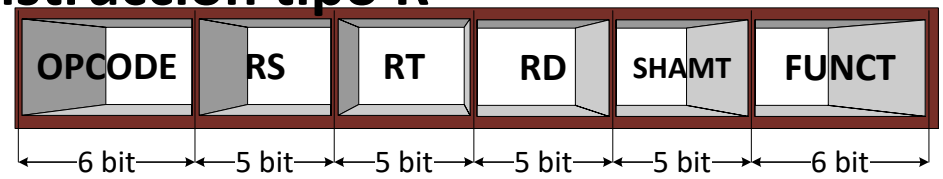
4. ¿Cómo se ejecuta una instrucción?

- Tenemos tres tipos de instrucciones en ensamblador:

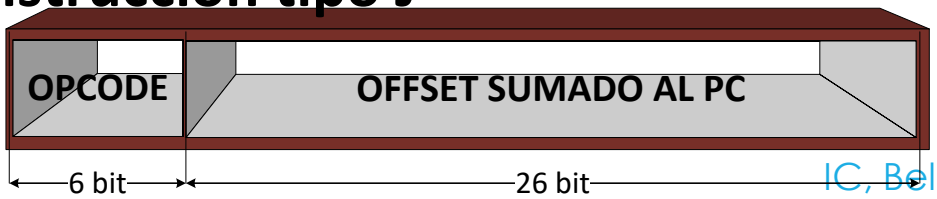
Instrucción tipo I



Instrucción tipo R



Instrucción tipo J



4. ¿Cómo se ejecuta una instrucción?

- **LW y SW (tipo I)**
  - RS (registro fuente): Registro base para el acceso a memoria.
  - RT (registro destino): Registro para los datos.
  - Inmediato: Desplazamiento para el cálculo de la dirección de memoria a la que hay que acceder.

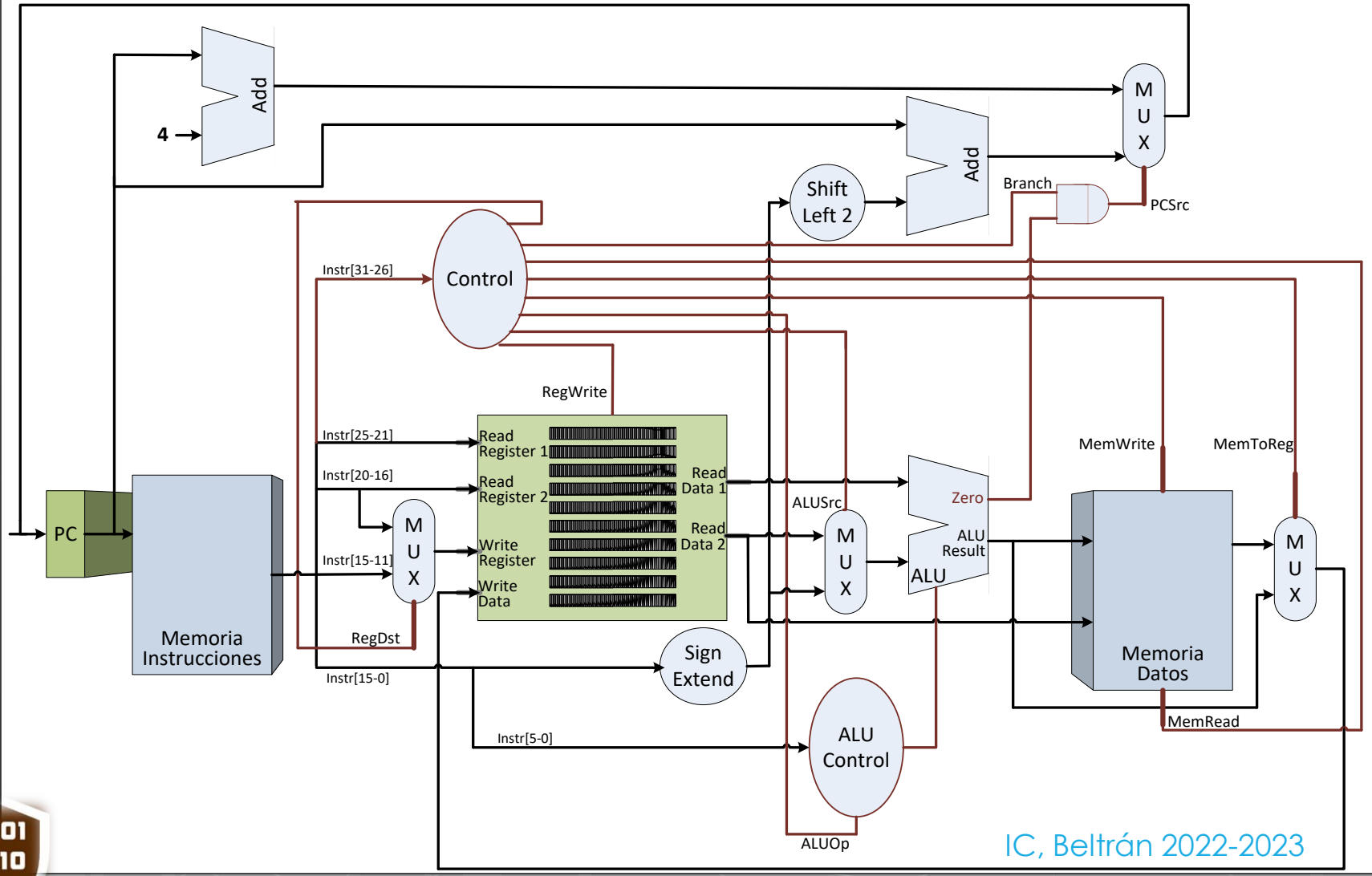
4. ¿Cómo se ejecuta una instrucción?

- **Aritmético-lógicas (tipo R)**
  - RS: Registro que contiene el operando 1.
  - RT: Registro que contiene el operando 2.
  - RD: Registro destino.
  - Shamt (shift amount): Indica el desplazamiento en las instrucciones shift.
  - Funct: Junto con el opcode indica el tipo de operación que se debe realizar.

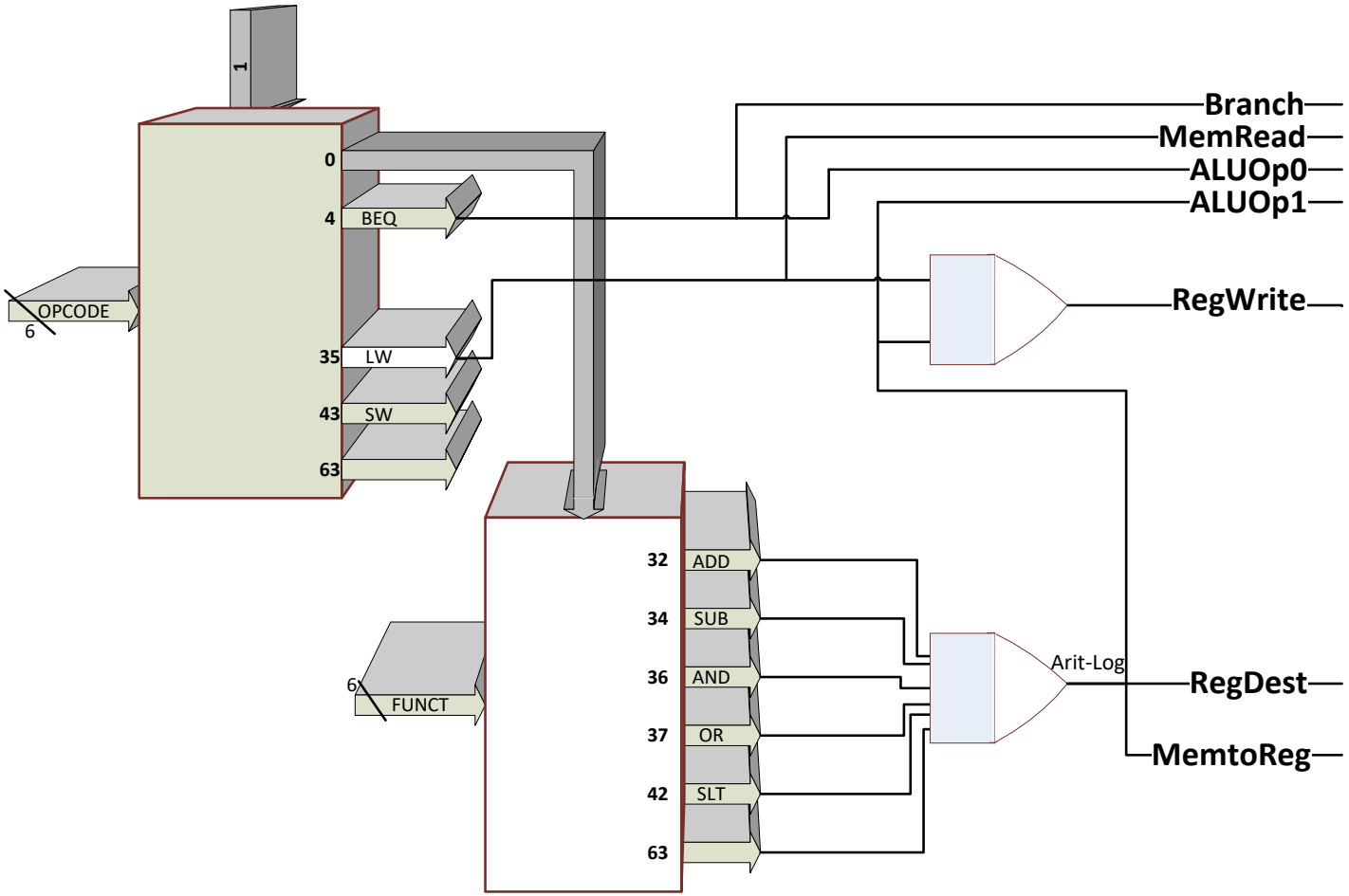
4. ¿Cómo se ejecuta una instrucción?

- **BEQ (tipo I)**
  - RS (registro fuente): Registro de condición (para la comparación.
  - RT (registro destino): Registro de condición (para la comparación).
  - Inmediato: Desplazamiento respecto del PC del destino de salto (por si la condición se cumple y el salto se toma).

RUTA DE DATOS DEL PROCESADOR + L1 DE MEMORIA CACHÉ



UNIDAD DE CONTROL



4. ¿Cómo se ejecuta una instrucción?

Fetch (F):

- Se busca en la memoria caché la instrucción que está almacenada en la dirección que indica el contador de programa (Program Counter o PC).
- Se deja preparado el contador de programa (sumando o restando una cantidad fija que depende de la arquitectura y del repertorio de instrucciones) para buscar la siguiente instrucción.

4. ¿Cómo se ejecuta una instrucción?

Decode (D):

- Se decodifica la instrucción separando sus diferentes campos.
- El código de operación de la instrucción indica qué tipo de instrucción es, y por tanto, qué tipo de operación se debe realizar en la ruta de datos.
- Si es necesario, se leen 1 ó 2 operandos de los registros del procesador.

4. ¿Cómo se ejecuta una instrucción?

Execution (X):

- Se ejecuta la operación que indicaba el opcode.
- Normalmente, utilizando para ello algún tipo de ALU o de unidad funcional aritmético-lógica.

4. ¿Cómo se ejecuta una instrucción?

Memory Access (M):

- Si es necesario acceder a memoria caché de datos para leer o escribir, el acceso se realiza en esta fase.

Writeback (W):

- Si es necesario volcar algún resultado a un registro, se realiza esta escritura.

Para practicar un poco

1. Busca una página web que te permita configurar tu propio PC y escoge una placa base, un procesador, una memoria, etc. Fíjate en el tipo de especificaciones que te proporcionan, en las métricas que se usan para el rendimiento, en las diferencias de precio, intenta dibujar el diagrama de conexión entre los componentes, etc.
2. Juega un rato con este simulador online de un computador sencillo que sigue el modelo Von Neumann: http://vnsimulator.altervista.org/
3. ¿Te suena alguna vulnerabilidad para la seguridad relacionada con el diseño de los procesadores que se haya descubierto en los últimos meses? Busca información sobre alguna de ellas e intenta entender cuál ha sido la causa.

Referencias

- Fotografías
  - https://unsplash.com
- Iconos
  - https://www.flaticon.es/
- Figuras nanoMIPS
  - Libro “Diseño y evaluación de arquitectura de computadores”, Marta Beltrán y Antonio Guzmán. Pearson.