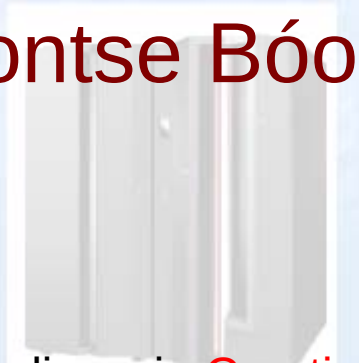


# Arquitectura del MIPS: Introducción

Montse Bóo Cepeda

# Estructura del curso

1. Evolución y caracterización de los computadores.
2. Arquitectura del MIPS: Introducción.
3. Tipo de datos.
4. El repertorio de instrucciones.
5. Aritmética del computador.
6. El camino de datos.
7. Sección de control.
8. El camino de datos multiciclo.
9. Sección de control multiciclo.
10. Entrada/Salida (I/O).

# Esquema de contenidos

1. Repertorio de instrucciones MIPS
2. Programa almacenado
3. Sistemas de almacenamiento de datos

# Introducción: La visión del programador

- El funcionamiento de la CPU está determinado por las instrucciones que ejecuta.
- El conjunto de instrucciones distintas que puede ejecutar se denomina **repertorio de instrucciones**
- En los computadores actuales las instrucciones se presentan como números  y se almacenan en memoria (**programa almacenado**)
- El lenguaje utilizado por el ordenador se denomina **lenguaje máquina**. El lenguaje **ensamblador** es una representación simbólica más cercana al humano.

# Visión software: Jerarquía de traducción

**Lenguaje de alto nivel**

```
temp    = v[k];
v[k]    = v[k+1];
v[k+1]  = temp;
```

```
TEMP  = V(K)
V(K)  = V(K+1)
V(K+1) = TEMP
```

Compilador C

Compilador Fortran

**Lenguaje ensamblador** (MIPS)

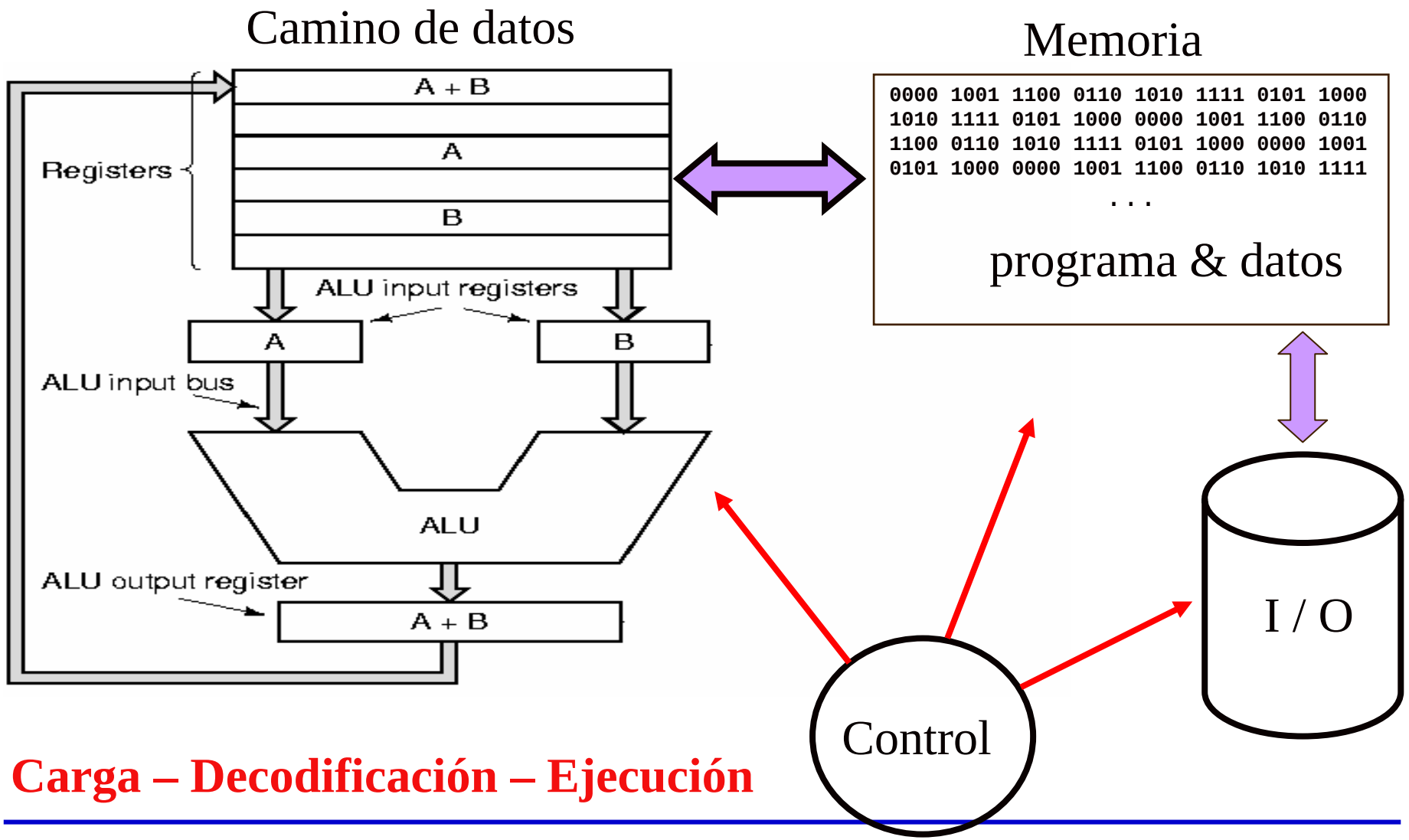
```
lw  $t0,  0($2)
lw  $t1,  4($2)
sw $t1,  0($2)
sw $t0,  4($2)
```

Ensamblador MIPS

**Lenguaje máquina**

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

# Ejecución de un programa




**Carga – Decodificación – Ejecución**

# Instrucciones

- Cada instrucción máquina debe especificar:
  - Código de operación
  - Referencia a operandos fuente
  - Referencia a operandos resultado
  - Referencia a la siguiente instrucción

- Los operandos fuente y resultado pueden estar en alguna de las siguientes localizaciones:
  - Memoria
  - Registro de la CPU
  - Dispositivo de E/S

# Representación de las instrucciones

- Dentro del computador cada instrucción se representa por una secuencia de bits
- La instrucción se divide en campos correspondientes a los elementos constitutivos de la misma (código de operación, operandos, etc.)
- La descripción en campos y bits se denomina **formato de instrucción**

# Operaciones del repertorio de instrucciones

- Aritméticas
- Lógicas
- Transferencia de datos
- De control de flujo
  - Instrucciones de bifurcación o salto
  - Instrucciones de salto condicional
  - Instrucciones de llamada a subrutina
- De control del sistema
- De E/S

# Repertorio de instrucciones escogido: MIPS

- Arquitectura MIPS R2000/R3000
- Pionera de las arquitecturas RISC
- Muy sencilla y uniforme
- Creada en Stanford (John Hennessy)
- Utilizada por NEC, Nintendo, Silicon Graphics y Sony.

# Lenguaje MIPS

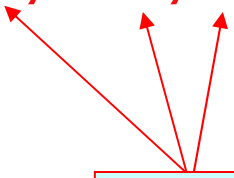
Instrucciones:

- Aritméticas
  - add, sub, mult, div
- Lógicas
  - and, or, ssl (shift left), srl (shift right)
- Transferencia de datos
  - lw (load), sw (store), lui
- Saltos
  - Condicionales: beq, bne, slt
  - No condicionales: j, jr, jal

# MIPS: Ejemplo de sencillez

**SUMA de dos variables en registros $s1 y $s2, resultado en registro $t0.**

```
add $t0, $s1, $s2    # $t0 = $s1 + $s2
```



En MIPS son nombres de registros

- Longitud de instrucción fija: 32 bits
- La arquitectura MIPS realiza la mayor parte de los cálculos con datos almacenados en registros
  - MIPS es sencilla y eficiente
- El acceso a memoria se hace a través de operaciones de carga/almacenamiento (transferencia de datos)

# Registros en el MIPS

- 32 registros de 32 bits

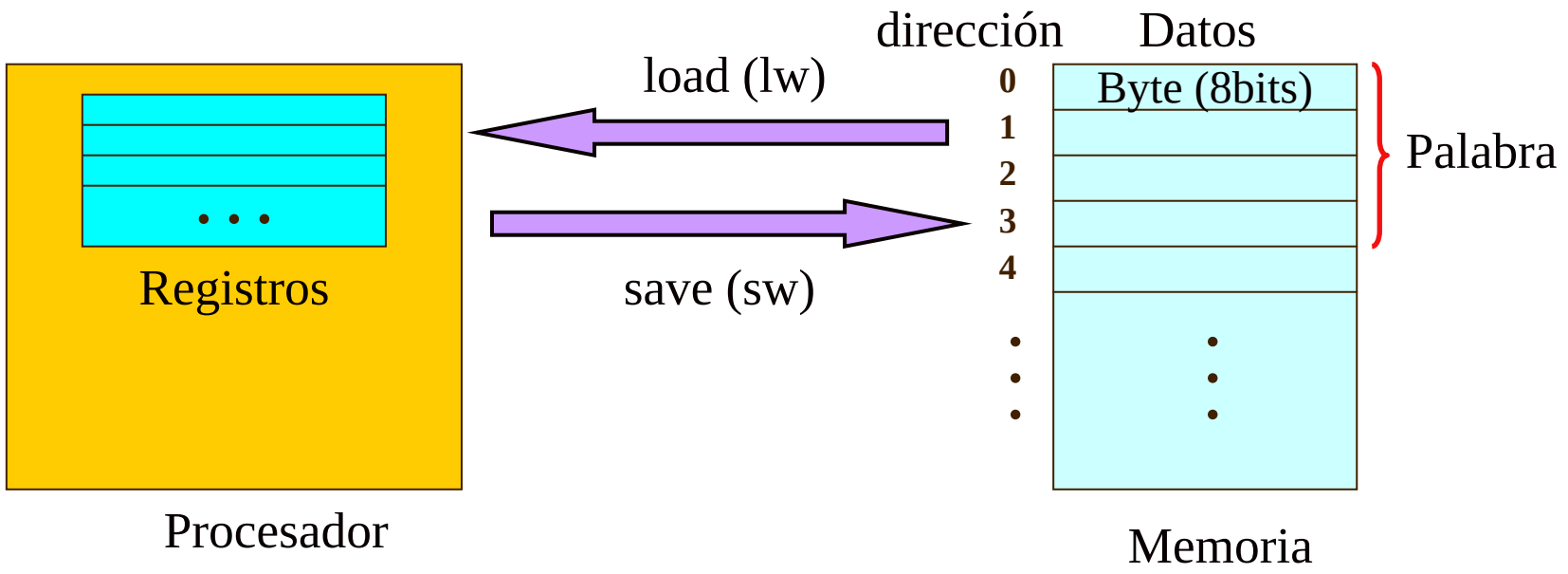
| Número | Nombre | Uso convencional |
|---|---|---|
| 0 | $zero | Valor constante 0 |
| 1 | $at | Reservado por el ensamblador |
| 2-3 | $v0-$v1 | Para resultados y evaluación de expresiones |
| 4-7 | $a0-$a3 | Argumentos |
| 8-15 | $t0-$t7 | Temporales |
| 16-23 | $s0-$s7 | Salvados |
| 24-25 | $t8-$t9 | Temporales |
| 26-27 | $k0-$k1 | Reservado para núcleo de SO |
| 28 | $gp | Puntero global |
| 29 | $sp | Puntero de pila |
| 30 | $fp | Puntero de bloque de activación |
| 31 | $ra | Dirección de retorno |

# Concepto de programa almacenado

- Las instrucciones son representadas como números.
- Los programas están almacenados en memoria
  - Pueden ser leídos/escritos como números.

- La memoria: Contiene instrucciones y datos.
  - Las instrucciones son cargadas automáticamente (control).
  - Los datos son transferidos explícitamente entre memoria y procesador y viceversa →INSTRUCCIONES DE TRANSFERENCIA DE DATOS

# Concepto de programa almacenado

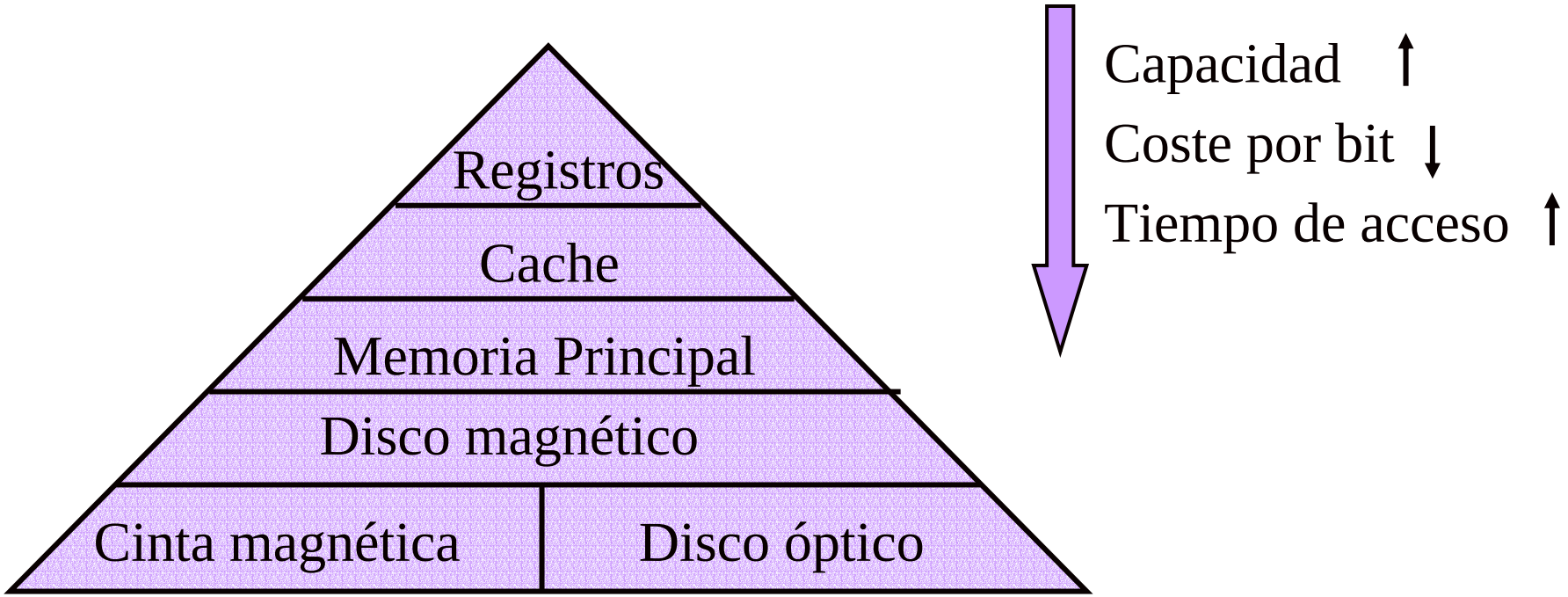


# ¿Dónde están los operandos?:

Los operandos pueden estar:

- En la instrucción:
  - Rápido y simple.
  - Sólo para constantes
- En la memoria principal:
  - Direccionamiento implica muchos bits.
  - Acceso lento
- En los registros de la CPU:
  - Hay pocos registros: el direccionamiento implica pocos bits
  - Acceso rápido

# Sistemas de almacenamiento de datos.

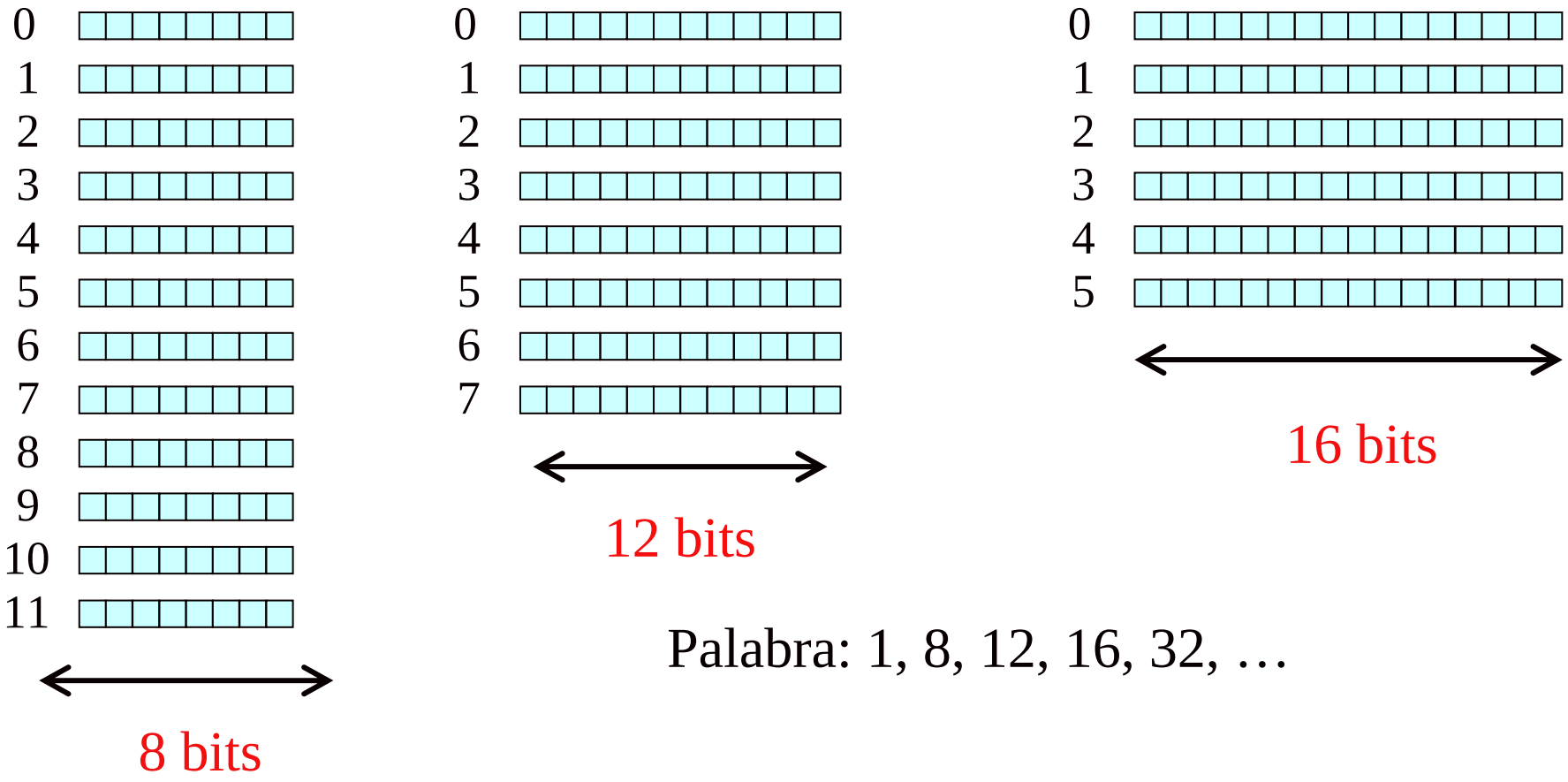


# Memorias de acceso aleatorio

<table>
<tr><th>Tipo de memoria</th><th>Propósito</th><th>Borrado</th><th>Modo de escritura</th><th>Volatilidad</th></tr>
<tr><td>Random Access Memory (RAM)</td><td>Lec/Esc</td><td>Eléctrico</td><td>Eléctrico</td><td>Volátil</td></tr>
<tr><td>Read Only Memory (ROM)</td><td rowspan="2">Lectura</td><td rowspan="2">No posible</td><td>Proceso fabricación</td><td rowspan="6">No volátil</td></tr>
<tr><td>Programmable ROM (PROM)</td><td>Eléctrico (fusibles)</td></tr>
<tr><td>Erasable PROM (EPROM)</td><td rowspan="3">Lectura (reprogramable)</td><td>Luz UV (chip)</td><td rowspan="3">Eléctrico</td></tr>
<tr><td>Electrically Erasable PROM (EEPROM)</td><td>Eléctrico (nivel byte)</td></tr>
<tr><td>Flash</td><td>Eléctrico (nivel bloque)</td></tr>
</table>

# Organización de la memoria

- Organizando una memoria de 96 bits:



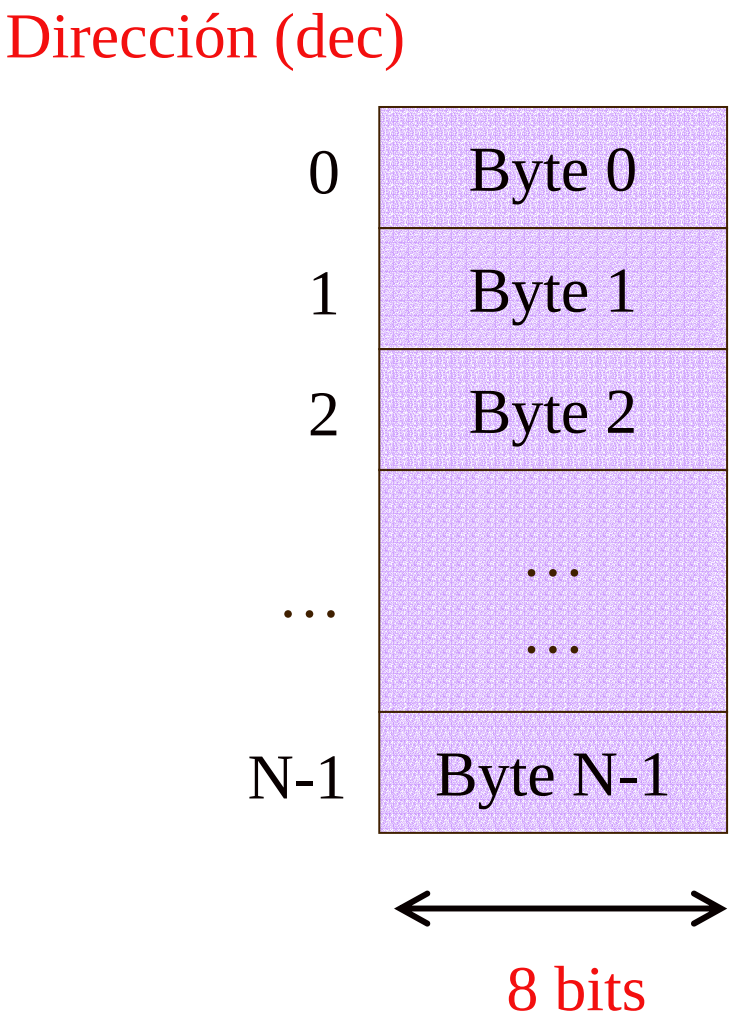

Palabra: 1, 8, 12, 16, 32, …

# Organización de la memoria

- Ejemplos

| | Bus datos | Longitud registros | Unidad direccionable |
|---|---|---|---|
| 8086 | 16 | 16 | Byte |
| 8088 | 8 | 16 | Byte |
| 486 | 32 | 32 | Byte |
| Pentium | 64 | 32 | Byte |
| 68000 | 16 | 32 | Byte |
| 68020 | 32 | 32 | Byte |
| PowerPC 750 | 64 | 64 | Byte |

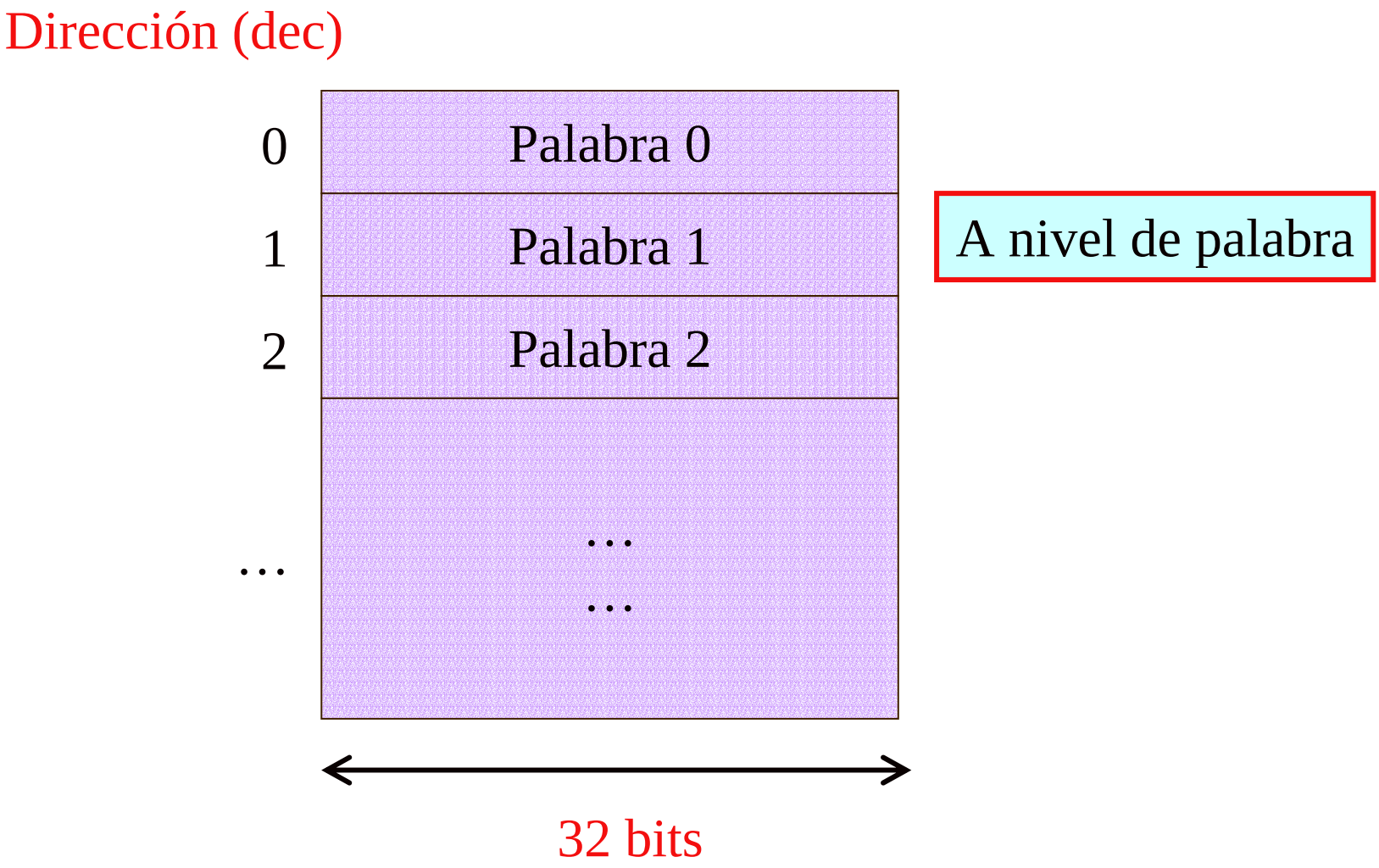
# Direcciones de memoria



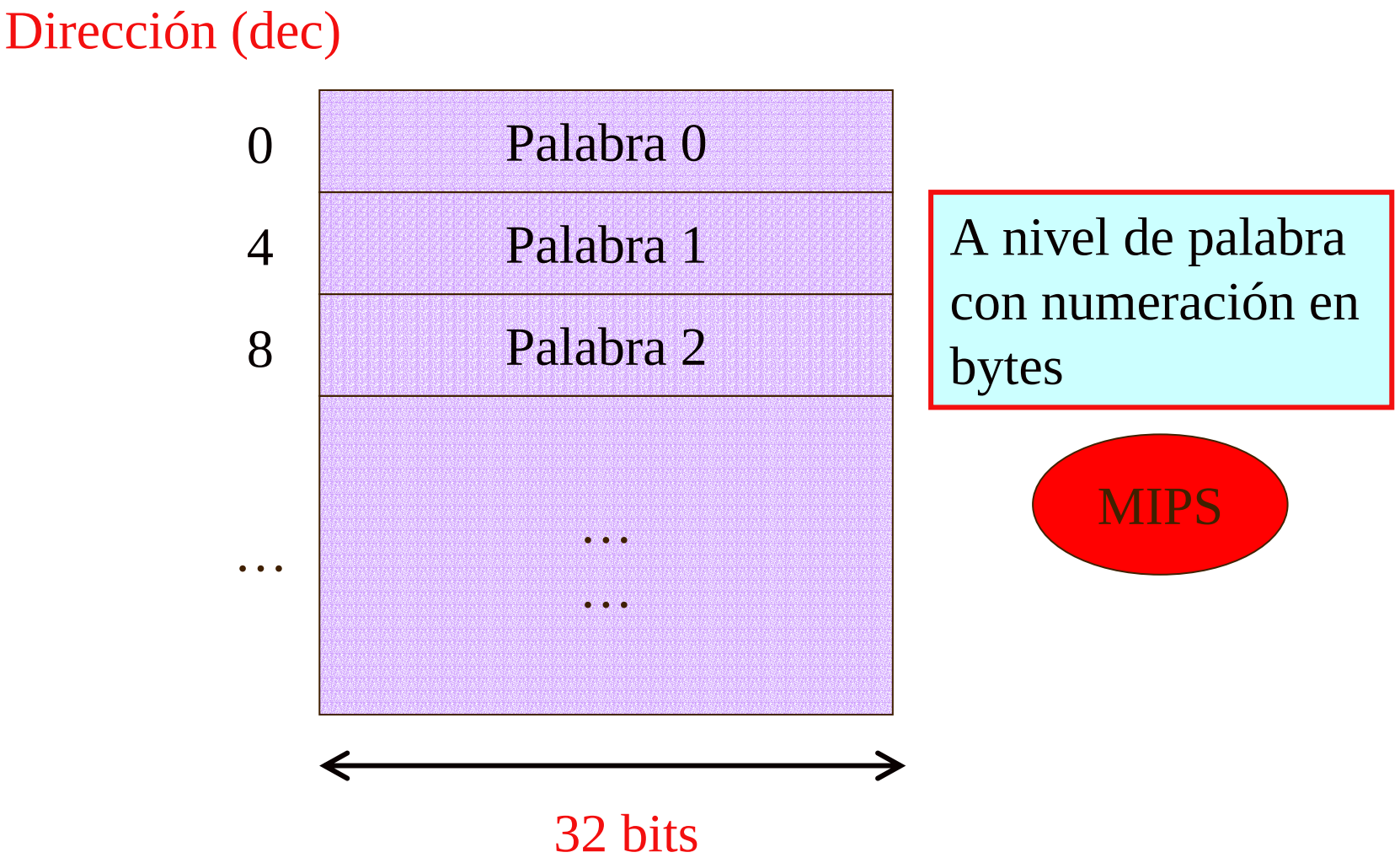

A nivel de byte

Tamaño memoria: N bytes

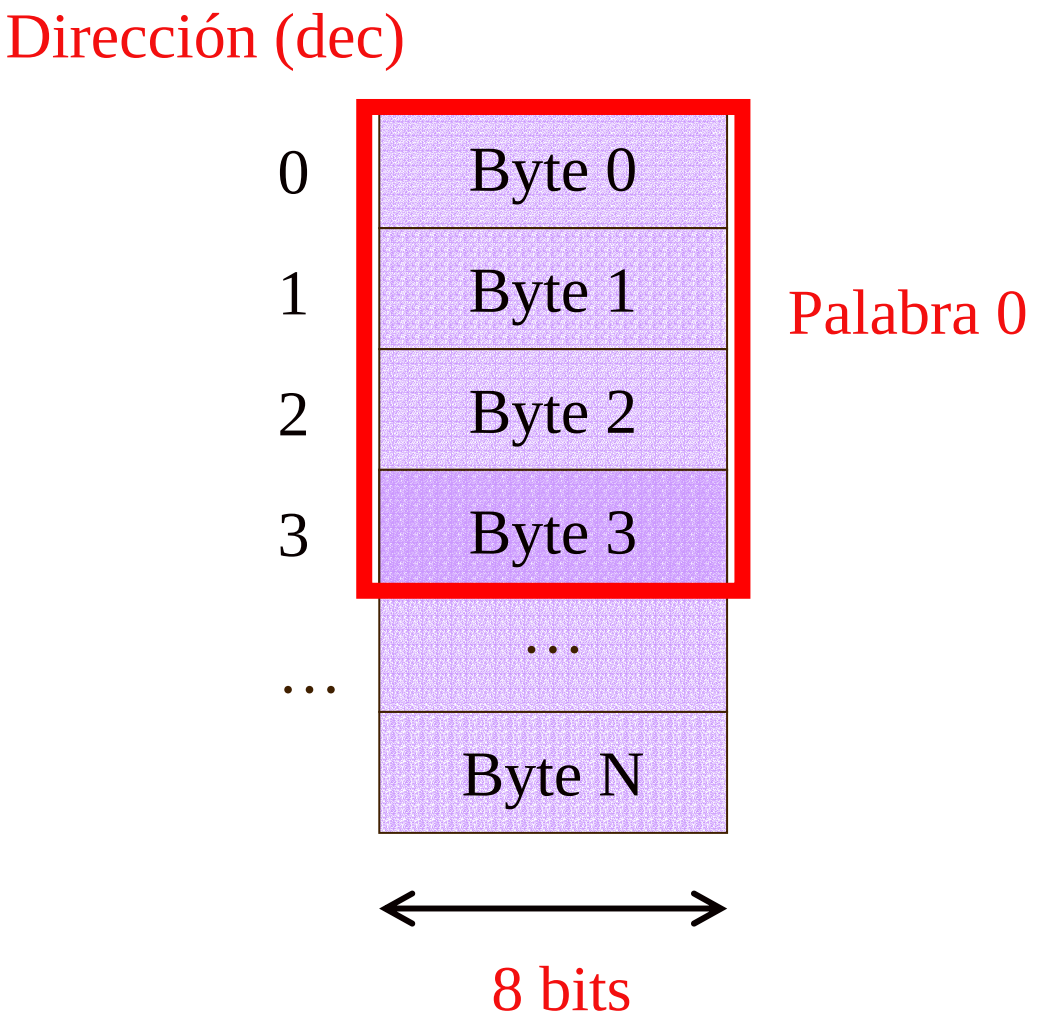
# Direcciones de memoria



# Direcciones de memoria



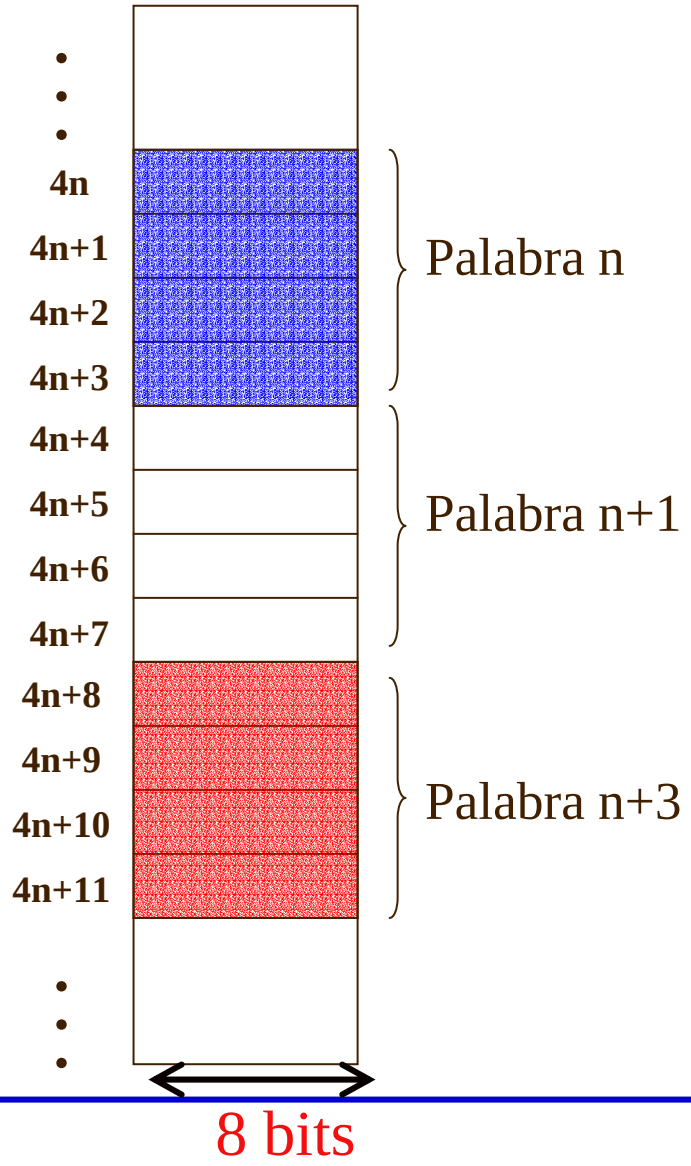
# Direcciones de memoria




A nivel de palabra con numeración en bytes

MIPS

# Direcciones de memoria




A nivel de palabra con numeración en bytes

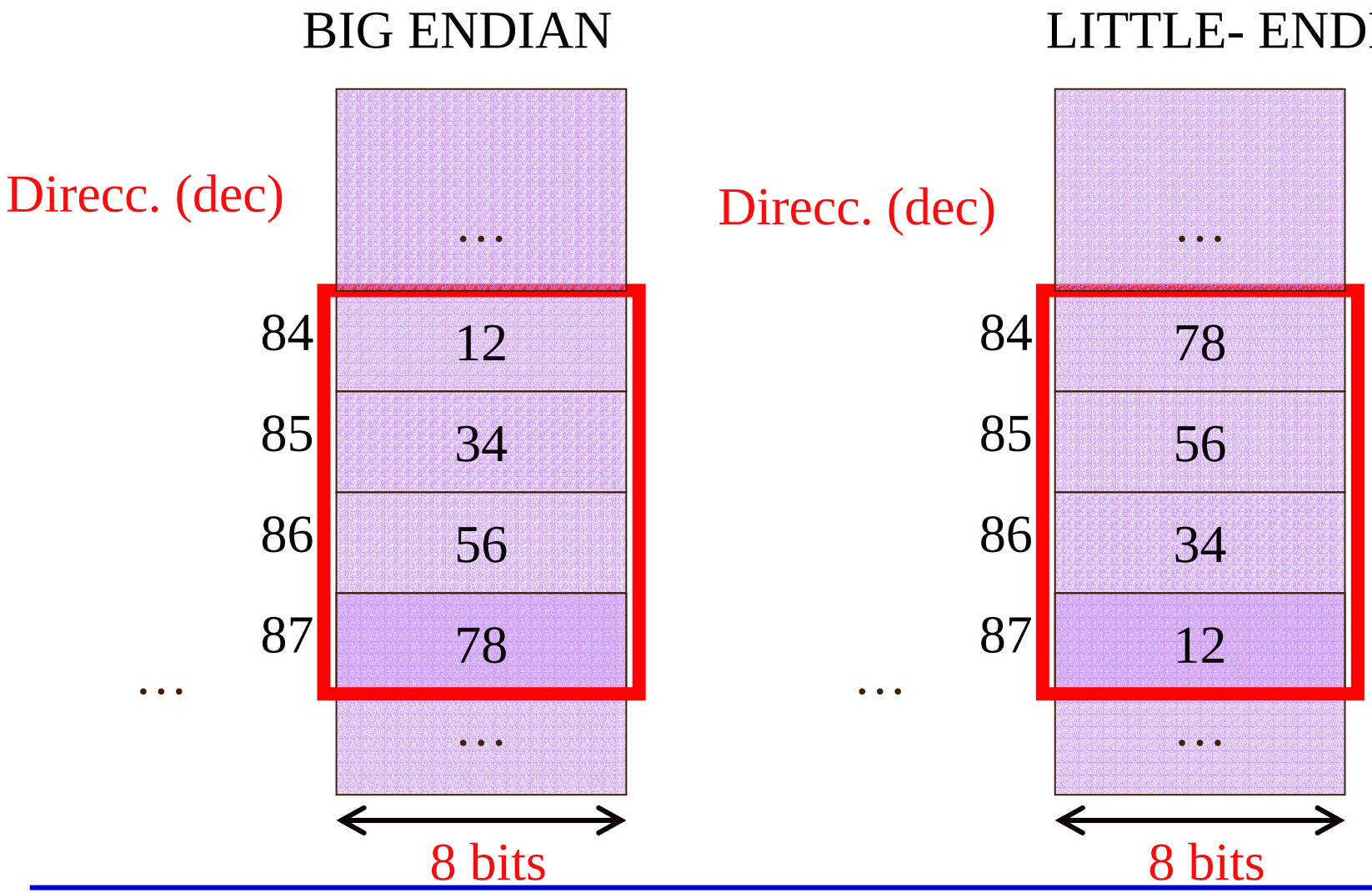
MIPS

# Big Endian, Litte Endian

- Orden de almacenamiento en memoria de los bytes dentro de una palabra
- Big Endian: El byte más significativo en la dirección más baja
- Little-Endian: El byte más significativo en la dirección más alta

# Big Endian, Litte Endian

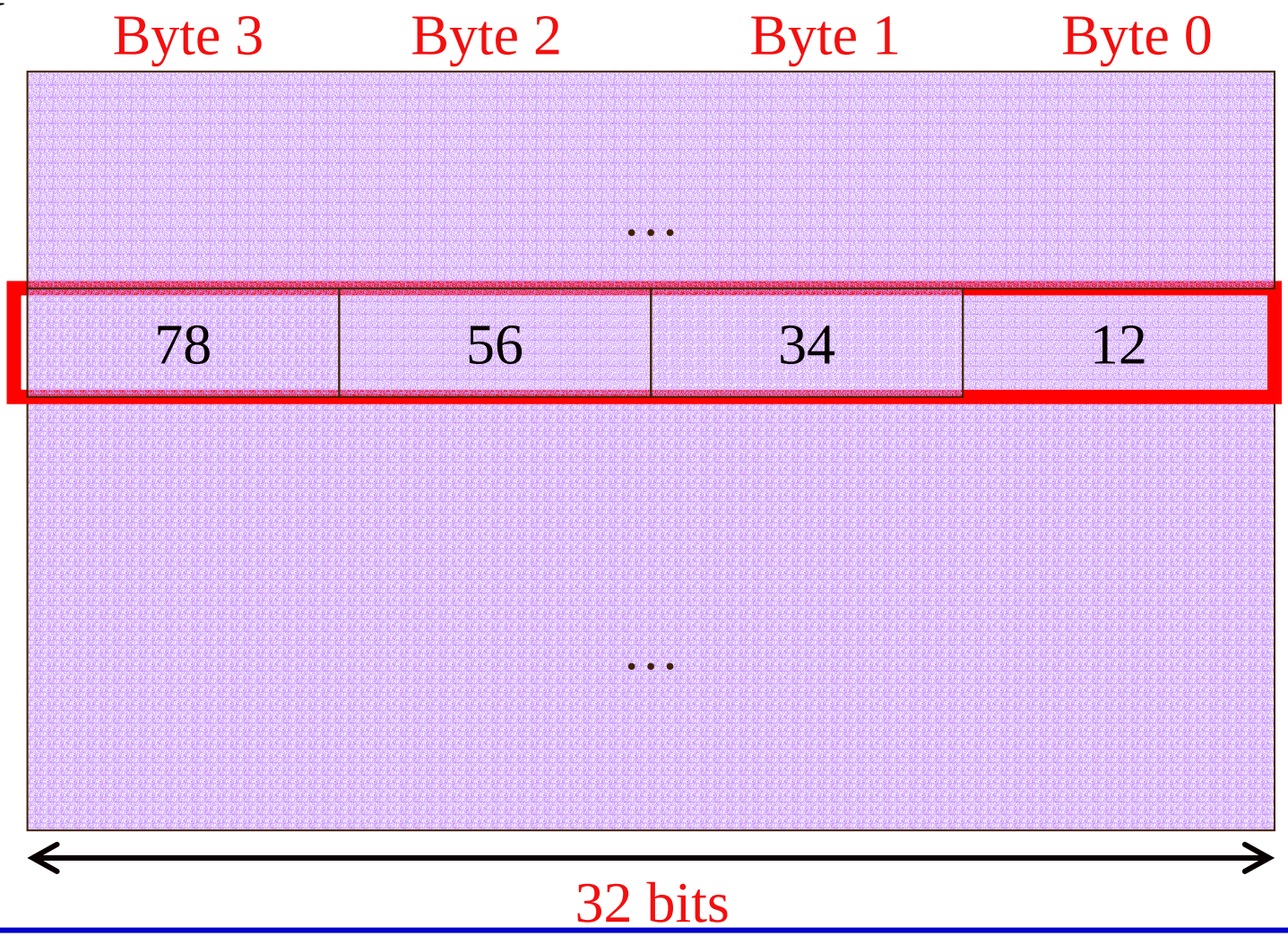
Dato: 12345678 (hex)
Dirección: 84(dec)



# Big Endian, Litte Endian

Dato: 12345678 (hex)
Dirección: 84(dec)

BIG ENDIAN



# Big Endian, Litte Endian

Dato: 12345678 (hex)
Dirección: 84(dec)