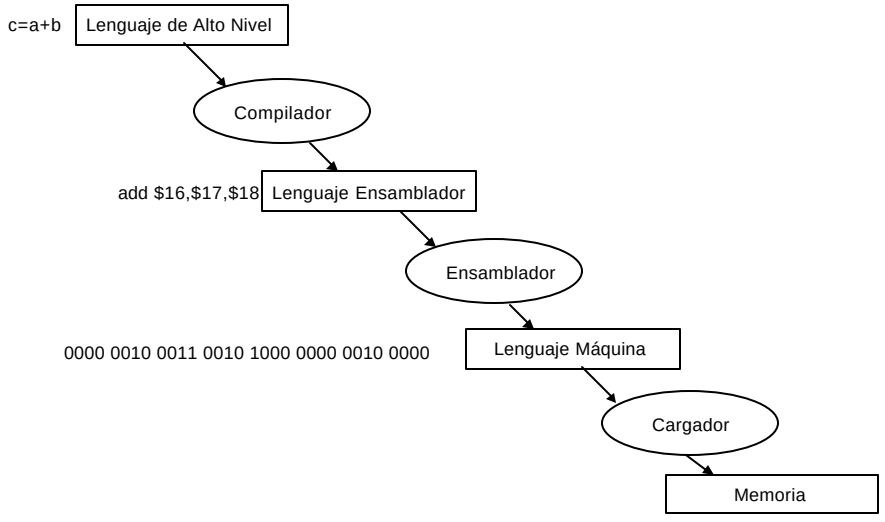
TEMA 6- Arquitectura del R2000
Contenido

6.2.1 Características generales

- **Introducción.**
- **Bancos de registros y organización de la memoria.**
- **Formato y juego de instrucciones.**
- **Modos de direccionamiento.**

Introducción



Arquitectura de un Computador (recordatorio)

- Visión de la máquina que tiene el programador en lenguaje máquina.
- La arquitectura del computador está determinada por:
  - Tipos y formatos de los operandos.
  - Mapas de memoria.
  - Juego de instrucciones.

Tipos de Datos

- Tipos de datos soportados por el MIPS R2000:

Tipo de datos	Tamaño de palabra	Datos representados
Ascii	**8 bits**	**Caracteres**
Byte	**8 bits**	**Números enteros**
Half	**16 bits**	
Word	**32 bits**	
Float	**32 bits**	**Números reales de simple precisión**
Double	**64 bits**	**Números reales de doble precisión**

  - Half y Word. El procesador dispone de instrucciones para el manejo de tipos de datos Half y Word sin signo y con signo (complemento a dos).
  - Float. Representa números reales de simple precisión según el estándar IEEE-754.

Bancos de Registros

- Bancos de registros del MIPS R2000:
  - Banco de registros de **Enteros**, constituido por:
    - 32 registros de 32 bits de propósito general para operaciones con enteros. Se identifican por el carácter especial **$** seguido de un número, del 0 al 31.
      - Ejemplo: add $16, $17, $18
    - 2 registros especiales de 32 bits: HI y LO.
      - Almacenan los resultados de multiplicaciones y divisiones.
      - Se utilizan para operaciones de transferencia de datos.
  - Banco de registros de **Reales**, que puede utilizarse como:
    - 32 registros de 32 bits de propósito general para operaciones en coma flotante de simple precisión según el formato IEEE 754. Se identifican por $f0 a $f31.
    - Los 32 registros anteriores se pueden combinar de dos en dos para tener 16 registros de 64 bits para operaciones en coma flotante de doble precisión.
- El registro $0 tiene permanentemente el valor 0.

Mapa de Memoria (I)

- Memoria. Puede verse como un vector (array) unidimensional.
- Dirección de Memoria. Índice dentro del vector.
- Es posible acceder a la memoria por:
  - Bytes: cualquier dirección 0 - 1 - 2 - 3 - .... etc.
  - Words:
    - Cada palabra ocupa 4 posiciones o direcciones de memoria.
    - Sólo direcciones múltiplo de cuatro: 0 - 4 - 8 - 12 - .... etc.

5	1 Byte de datos
4	1 Byte de datos
3	1 Byte de datos
2	1 Byte de datos
1	1 Byte de datos
0	1 Byte de datos

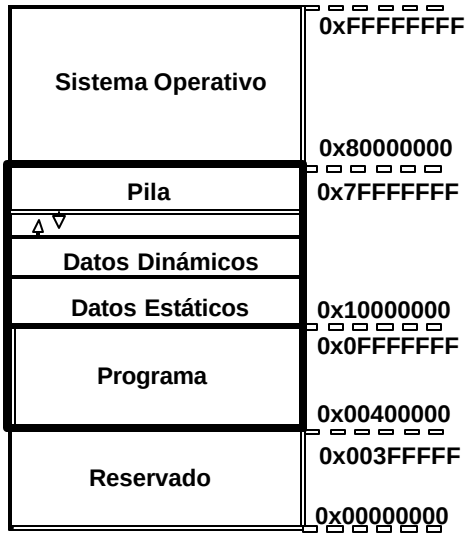
Acceso a byte

20	4 Bytes de datos
16	4 Bytesde datos
12	4 Bytes de datos
8	4 Bytes de datos
4	4 Bytes de datos
0	4 Bytes de datos

Acceso a WORD

Mapa de Memoria (II)

- Espacio direccionable en el MIPS R2000 → 4 Gbytes
  - 2^{32} bytes con direcciones que van desde 0 a $2^{32} - 1$
  - 2^{30} palabras (4 bytes) con direcciones: 0, 4, 8, ..., $2^{32} - 4$
- Memoria accesible por el usuario se encuentra en el rango [0x00400000, 0x7FFFFFFF]



7

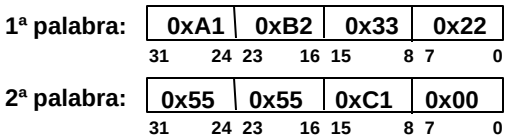
Mapa de Memoria (III)

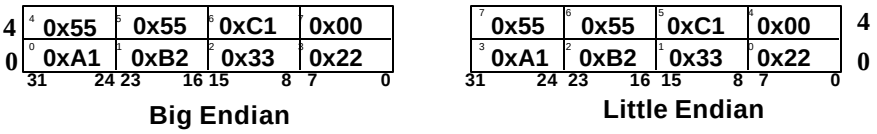
- Existen dos formas de almacenar información en memoria:
  - Formato *big endian*: El byte de **mayor peso** del dato o de la instrucción a ser almacenado se escribe en la dirección más baja.
  - Formato *little endian*: El byte de **menor peso** del dato o de la instrucción a ser almacenado se escribe en la dirección más baja.
- El procesador MIPS R2000 utiliza la organización *big endian*.
- El simulador utiliza la misma organización que el computador sobre el que se ejecuta. El procesador Intel 80x86 utiliza la organización *little endian*.

8

Mapa de Memoria (IV)

- Almacenamiento en memoria de las dos palabras siguientes utilizando ambas organizaciones:



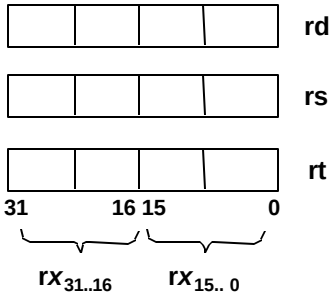


Juego de Instrucciones (I)

- Define el conjunto de operaciones que puede realizar el procesador -> Lenguaje del compilador.
- Cada instrucción en lenguaje ensamblador tiene un formato según el cual se codificará finalmente la instrucción en lenguaje máquina (0's y 1's).
- Tipos de instrucciones:
  - Instrucciones aritméticas.
  - Instrucciones lógicas.
  - Instrucciones de carga y almacenamiento.
  - Instrucciones de movimiento.
  - Instrucciones de comparación.
  - Instrucciones de salto condicional.
  - Instrucciones de salto incondicional.

Juego de Instrucciones (II)

- En las diferentes instrucciones se utiliza la siguiente nomenclatura:
  - rs, rd y rt representan el registro fuente, destino y transformable respectivamente.
  - r*x* $_{31..16}$: Parte alta del registro r*x*.
  - r*x* $_{15..0}$: Parte baja del registro r*x*.



Juego de Instrucciones (III)

- En las diferentes instrucciones se utiliza la siguiente nomenclatura:
  - desp+rs, da una dirección de memoria, resultado de sumar al contenido del registro *rs* una cantidad *desplazamiento*.
  - M[desp+rs] es el contenido de la posición de memoria *desp+rs*.

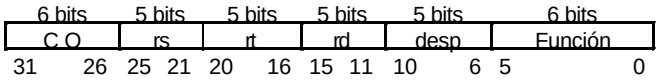
0x00000020	0x0012
0x00000016	0x0032
0x00000012	0x7654
0x00000008	0x0543
0x00000004	0x0000
0x00000000	0x0100

rs = 0x00000004
desp = 4 → **Desp+rs = 0x00000008** → **M[desp+rs] = 0x0543**

Formato de Instrucciones

- Todas las instrucciones del MIPS R2000 tienen un tamaño de 32 bits.
- El formato de las instrucciones indica cómo se codifican dichas instrucciones en código máquina.
- En el MIPS R2000 se distinguen tres tipos de formatos en función de los componentes del procesador que utilizan las instrucciones:
  - Formato **R** o de tipo registro.
  - Formato **I** o de tipo inmediato.
  - Formato **J** o de salto incondicional.
- Cada uno de los componentes que utiliza la instrucción se especifica mediante una serie contigua de bits denominada “campo”.

Formato de Instrucciones
Formato R



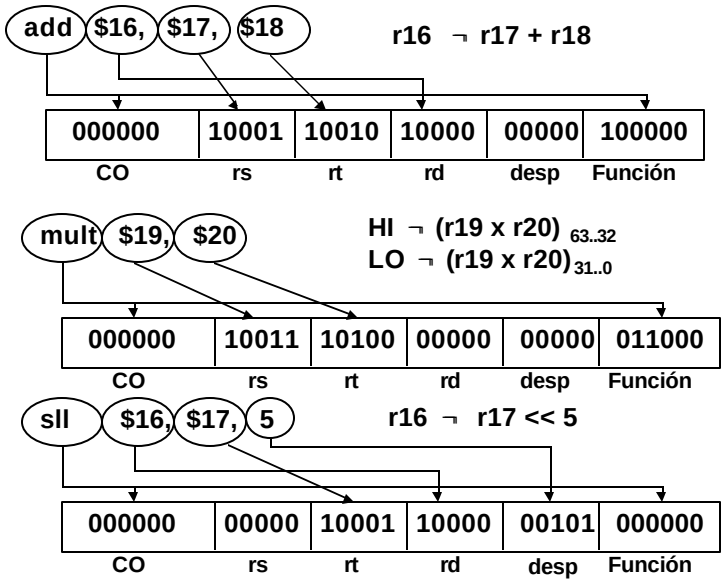
Campos	Nº de bits	Descripción
CO	6	Código de operación de la instrucción, impuesto por el diseñador, indica el tipo de instrucción
rs	5	Codificación binaria del primer registro fuente
rt	5	Codificación binaria del segundo registro fuente
rd	5	Codificación binaria del registro destino que se utilice
desp	5	Codificación del desplazamiento
Función	6	Indica el tipo de operación: aritmética o lógica

Formato de Instrucciones
Formato R

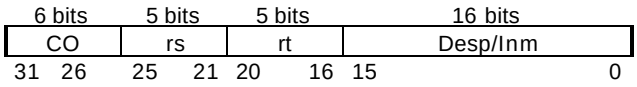
Instrucciones aritméticas	add rd, rs, rt
	addu rd, rs, rt
	sub rd, rs, rt
	subu rd, rs, rt
	mult rs, rt
	multu rs, rt
	div rs, rt
	divu rs, rt

Instrucciones lógicas	and rd, rs, rt or rd, rs, rt xor rd, rs, rt nor rd, rs, rt
Instrucciones de desplazamiento	sll rd, rt, desp srl rd, rt, desp sra rd, rt, desp
Instrucciones de salto	jr rs
Instrucciones de transferencia	mfhi rd mflo rd mthi rs mtlo rs

Formato de Instrucciones
Formato R



Formato de Instrucciones
Formato I



Campos	Nº de bits	Descripción
CO	6	Código de operación de la instrucción, impuesto por el diseñador, indica la operación que realiza la instrucción
rs	5	Codificación binaria del primer registro fuente que se utilice
rt	5	Codificación binaria del segundo registro
Desp/Inm	16	Dato de 16 bits codificado en complemento a 2

- El campo *Desp/Inm* indica:
  - Desplazamiento:
    - En número de palabras para instrucciones BEQ y BNE.
    - En número de bytes para instrucciones de Carga y Almacenamiento.
  - Dato Inmediato.

17

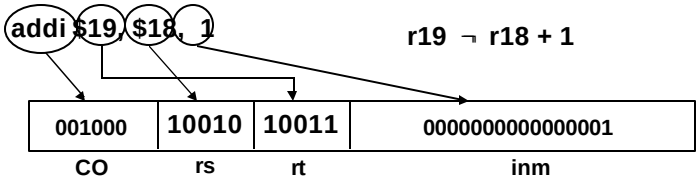
Formato de Instrucciones
Formato I

Instrucciones aritméticas	addi rt, rs, inm addiu rt, rs, inm
Instrucciones lógicas	andi rt, rs, inm ori rt, rs, inm xori rt, rs, inm
Comparación	slti rt, rs, inm
Salto condicional	beq rs, rt, etiqueta bne rs, rt, etiqueta

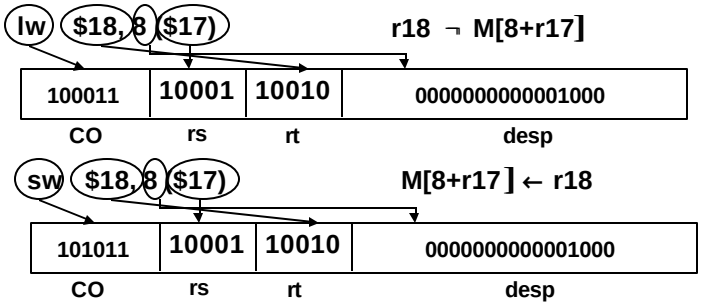
Carga y almacenamiento	lw rt,desp(rs)
	lh rt,desp(rs)
	lb rt,desp(rs)
	sw rt,desp(rs)
	sh rt,desp(rs)
	sb rt,desp(rs)
	lui rt,inm

18

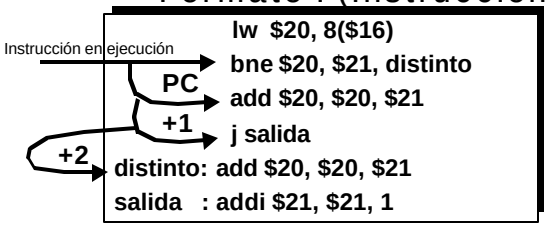
Formato de Instrucciones
Formato I



Instrucciones de Carga y almacenamiento

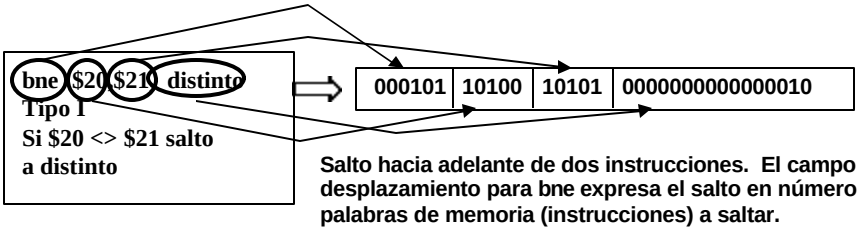


Formato de Instrucciones
Formato I (Instrucciones de salto)



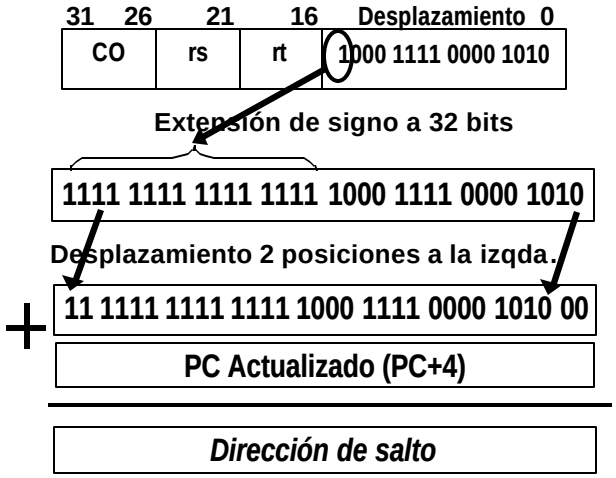
– El registro PC contiene la dirección de la siguiente instrucción.

– Cada instrucción ocupa una palabra (4 bytes) en memoria.
– Por tanto, las direcciones de memoria donde se encuentran las instrucciones siempre serán múltiplos de cuatro: 0, 4, 8, 12, 16, etc.

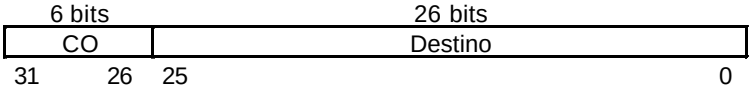


Salto hacia adelante de dos instrucciones. El campo desplazamiento para bne expresa el salto en número de palabras de memoria (instrucciones) a saltar.

Formato de Instrucciones
Formato I (Instrucciones de Salto)



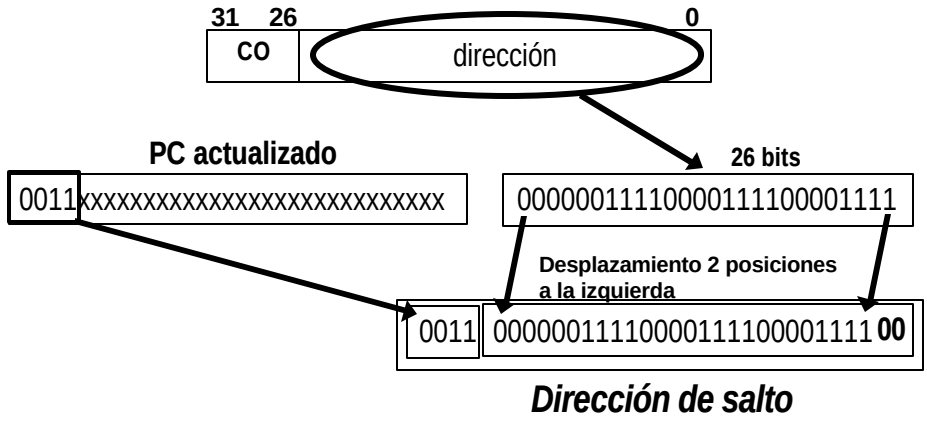
Formato de Instrucciones
Formato J



Campos	Nº de bits	Descripción
CO	6	Código de operación de la instrucción, impuesto por el diseñador, indica la operación que realiza la instrucción
Destino	26	Destino del salto a ejecutar

Formato de Instrucciones
Formato J

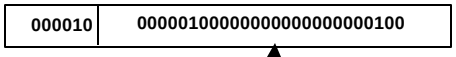
Cálculo de la dirección de salto



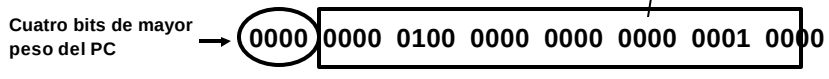
Formato de Instrucciones
Formato J

- Codificar la instrucción de salto j teniendo en cuenta que la instrucción addi del siguiente ejemplo se encuentra en la dirección de memoria 0x400010

```
        lw  $20, 8($16)
        bne $20, $21, distinto
        add $20, $20, $21
        j salida
distinto: add $20, $20, $21
salida  : addi $21, $21, 1
```



- Codificación de la etiqueta:
  - Sólo *26 bits* de los 32 del *Contador de Programa* (*PC*) son modificados por j permaneciendo los otros 6 sin cambios.
  - Salida = 0X400010 = 0000 0000 0100 0000 0000 0000 0001 0000 = dirección de salto.
  - La dirección de salto (32 bits) debe codificarse en el campo destino de 26 bits.
  - Se prescinde de los dos bits de menor peso y de los 4 bits de mayor peso.

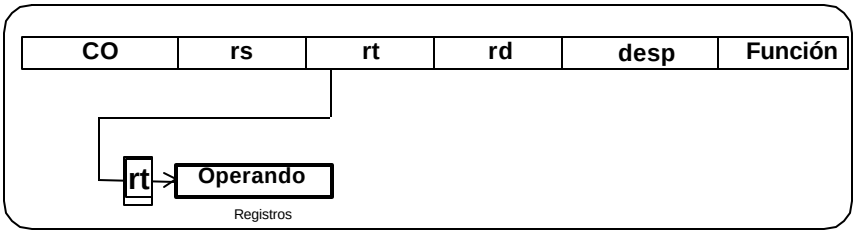


Modos de Direccionamiento (I)

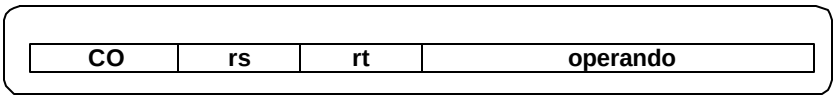
- Los modos de direccionamiento permiten localizar un operando que puede estar ubicado en:
  - En la propia instrucción (Direccionamiento inmediato).
  - En un registro (Direccionamiento a registro).
  - En la memoria (Direccionamiento a memoria).
- El objetivo de los modos de direccionamiento es especificar el lugar donde se encuentra el operando.
- *La dirección efectiva* es la dirección física donde se encuentra un dato.

Modos de Direccionamiento (II)

- Direccionamiento a registro
  - El operando está en un registro cuyo identificador se encuentra codificado en la instrucción.

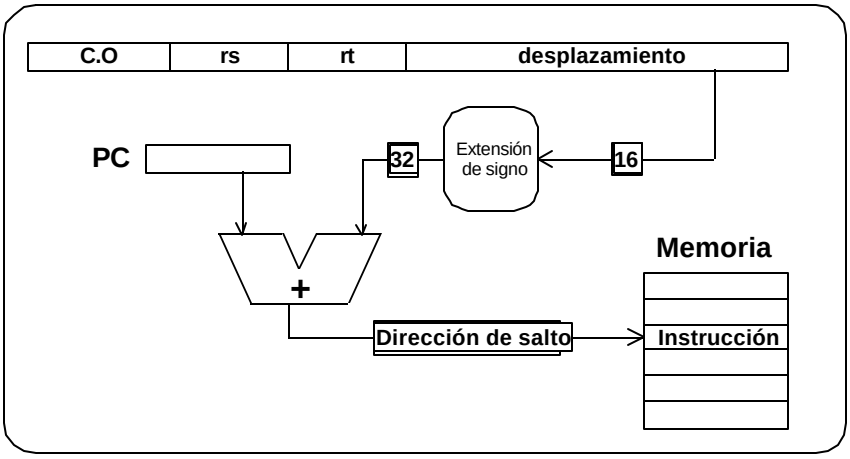


- Direccionamiento inmediato
  - El operando se encuentra codificado en la propia instrucción, representado como dato inmediato.



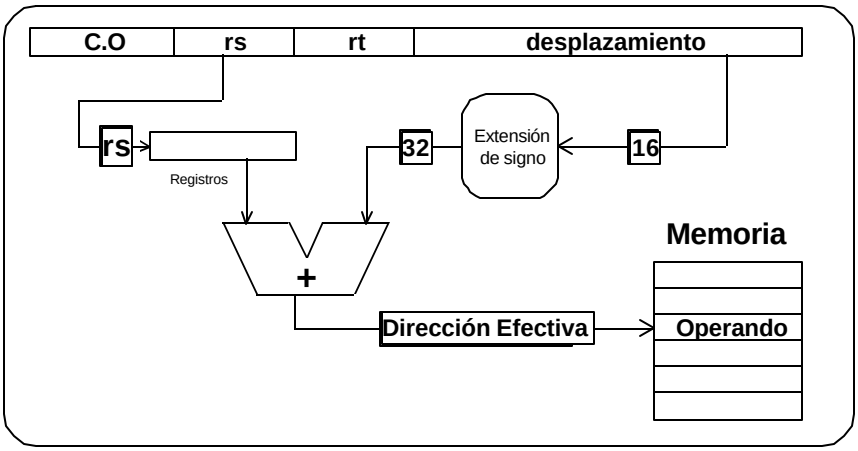
Modos de Direccionamiento (III)

- Direccionamiento relativo al contador de programa
  - Es un caso particular del direccionamiento relativo donde el registro que contiene la dirección base es de forma implícita el PC.



Modos de Direccionamiento (IV)

- Direccionamiento indexado.
  - La dirección efectiva del operando se obtiene sumando un desplazamiento que se encuentra en la instrucción, con la dirección base almacenada en el registro especificado por la instrucción.



Juego de Instrucciones
Instrucciones Aritméticas

Sintaxis	Formato	Descripción
add rd, rs, rt	R	rd ← rs + rt
addi rt, rs, inm	I	rt ← rs + inm
sub rd, rs, rt	R	rd ← rs – rt
mult rs, rt	R	Multiplica rs por rt dejando los 32 bits de menor peso en el registro HI y los 32 bits de mayor peso en LO
div rs, rt	R	Divide rs entre rt dejando el cociente en el registro LO y el resto en el registro HI

Juego de Instrucciones
Instrucciones Lógicas

Sintaxis	Formato	Descripción
and rd, rs, rt	R	rd ← rs and rt, la operación lógica indicada se realiza bit a bit
nor rd, rs, rt	R	rd ← rs nor rt
xor rd, rs, rt	R	rd ← rs xor rt
or rd, rs, rt	R	rd ← rs or rt
andi rt, rs, inm	I	rt← rs and inm, (el dato inmediato es de 16 bits y se extiende con 16 ceros)
ori rt, rs, inm	I	rt← rs or inm
xori rt, rs, inm	I	rt← rs xor inm
sll rd, rt, desp	R	rd← rt << desp, desplazamiento a izquierdas, conforme desplaza se rellena con 0
srl rd, rt, desp	R	rd← rt >> desp, desplazamiento a derechas, conforme desplaza se rellena con 0

Juego de Instrucciones
Instrucciones de Carga y Almacenamiento

Sintaxis	Formato	Descripción
lw rt, desp(rs)	I	rt ← M[desp+rs],
lh rt, desp(rs)	I	rt ← M[desp+rs], carga media palabra (16 bits) y extiende el signo
lb rt, desp(rs)	I	rt ← M[desp+rs], carga 1 byte (8 bits) y extiende el signo
sw rt, desp(rs)	I	M[desp+rs] ← rt
sh rt, desp(rs)	I	M[desp+rs] ← rt Almacena la parte baja de rt en memoria
sb rt, desp(rs)	I	M[desp+rs] ← rt Almacena el byte menos significativo en memoria
lui rt, inm	I	$rt_{31..16}$ ← inm $rt_{15..0}$ ←0

Juego de Instrucciones
Instrucciones de Movimientos de Datos

Sintaxis	Formato	Descripción
mfhi rd	R	rd ← HI
mflo rd	R	rd ← LO
mthi rs	R	HI ← rs
mtlo rs	R	LO ← rs

Juego de Instrucciones
Instrucciones de Comparación

Sintaxis	Formato	**Descripción**
slt rd, rs, rt	R	Si rs < rt entonces rd ← 1 Si no rd ← 0
slti rt, rs, inm	I	Si rs < inm entonces rt ←1 Si no rt ← 0

Juego de Instrucciones
Instrucciones de Salto Condicional

Sintaxis	Formato	**Descripción**
beq rs, rt, etiqueta	I	Si rs = rt entonces salta a la dirección etiqueta PC← etiqueta
bne rs, rt, etiqueta	I	Si rs <> rt entonces salta a la dirección etiqueta PC ← etiqueta

Juego de Instrucciones
Instrucciones de Salto Incondicional

Sintaxis	Formato	Descripción
j etiqueta	J	PC← etiqueta, salta a la dirección etiqueta
jal etiqueta	J	Salta a la dirección etiqueta guardándose previamente la dirección de retorno en $31 $31← PC+4 PC← etiqueta
jr rs	J	PC ← rs, salta a ladirección contenida en el registro rs

Codificación Lenguaje Máquina

- El programa ensamblador es el encargado de codificar en código máquina las instrucciones de un programa escrito en ensamblador.
- La codificación de cada instrucción depende del tipo de formato (R, I o J) al que pertenece.

Codificación Lenguaje Máquina
Instrucciones tipo R

Instrucción	CO	rs	rt	rd	Num desp	Función
add	0	*rs*	*rt*	*rd*	0	0x20
and	0	*rs*	*rt*	*rd*	0	0x24
div	0	*rs*	*rt*	0	0	0x1a
jr	0	*rs*	0	0	0	8
mfhi	0	0	0	*rd*	0	0x10
mflo	0	0	0	*rd*	0	0x12
mthi	0	*rs*	0	0	0	0x11
mtlo	0	*rs*	0	0	0	0x13
mult	0	*rs*	*rt*	0	0	0x18
nor	0	*rs*	*rt*	*rd*	0	0x27
or	0	*rs*	*rt*	*rd*	0	0x25
slt	0	*rs*	*rt*	*rd*	0	0x2a
sub	0	*rs*	*rt*	*rd*	0	0x22
xor	0	*rs*	*rt*	*rd*	0	0x26
sll	0	*rs*	*rt*	*rd*	desp	0x00
srl	0	*rs*	*rt*	*rd*	desp	0x02
	6	*5*	*5*	*5*	5	6

37

Codificación Lenguaje Máquina
Instrucciones tipo I

Instrucción	CO	rs	rt	desp/Inm
addi	8	*rs*	*rt*	Inm
andi	0xc	*rs*	*rt*	Inm
beq	4	*rs*	*rt*	Desplazamiento en palabras
bne	5	*rs*	*rt*	Desplazamiento en palabras
lw	0x23	*rs*	*rt*	Desplazamiento en bytes
lui	0xf	0	*rt*	Inm
ori	0xd	*rs*	*rt*	Inm
sw	0x2b	*rs*	*rt*	Desplazamiento en bytes
xori	0xe	*rs*	*rt*	Inm
	6	*5*	*5*	16

Instrucciones tipo J

Instrucción	C.O.	Dirección
j	2	destino del salto
jal	3	destino del salto
	6	26

38