

Dept. de Teoría de la Señal y Comunicaciones y Sistemas
Telemáticos y Computación
Área de Telemática (GSyC)

MIPS

Microprocessor without Interlocked Pipeline Stages

Katia Leal Algara

katia.leal@urjc.es

<http://gsyc.escet.urjc.es/~katia/>



Repertorio RISC

- ❑ **RISC**, los microprocesadores de hoy en día!
- ❑ *Reduced Instruction Set Computer*, filosofía de diseño basada en
 - ❑ repertorios de pocas instrucciones muy básicas
 - ❑ rutas de datos con un hardware muy sencillo
 - ❑ multitud de posibilidades para su optimización
- ❑ Los diseños RISC están en el núcleo de todos los procesadores de PC y portátiles
- ❑ ... y se han incorporado con éxito en otro tipo de plataformas y arquitecturas

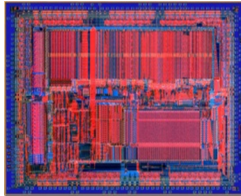
Microprocesadores RISC

- ☐ La serie **MIPS** Technologies Inc., que se encuentra en la mayoría de los computadores de SGI, en la Nintendo 64 y en la Sony PlayStation
- ☐ La línea **IBM POWER**, utilizada en Servidores de IBM
- ☐ La versión **PowerPC** de Motorola e IBM (una versión de la serie IBM POWER) utilizada en los ordenadores Apple Macintosh como el iMac, eMac, Power Mac y posteriores
- ☐ El procesador **SPARC** de SUN Microsystems y el **UltraSPARC**, que se encuentra en todos sus últimos modelos de equipos
- ☐ El **PA-RISC** y el **HP/PA** de Hewlett-Packard
- ☐ El **DEC Alpha**, que se puede encontrar en servidores HP AlphaServer
- ☐ El **ARM**, que se encuentra en dispositivos PALM, y en múltiples PDAs, teléfonos móviles y consolas de videojuegos de Nintendo

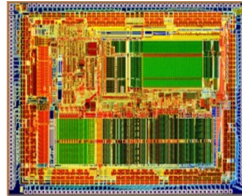
Introducción

Microprocesadores MIPS

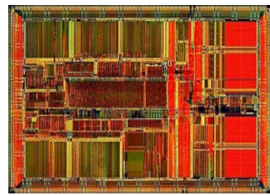
MIPS R2000



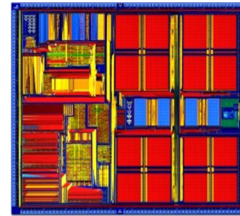
MIPS R3000



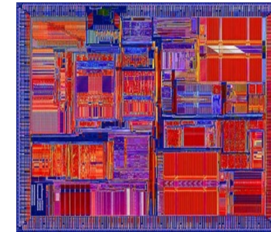
MIPS R4000



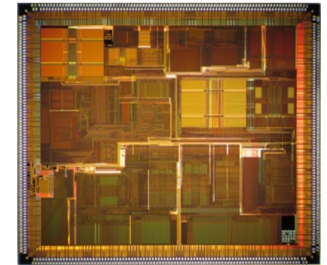
MIPS R5000



MIPS R10000



MIPS R12000



Year	1985	1988	1992	1996	1995	1998
MIPS ISA	MIPS I (32-bit)	MIPS I (32-bit)	MIPS III (64-bit)	MIPS IV (64-bit)	MIPS IV (64-bit)	MIPS IV (64-bit)
Transistor count	110k	110k	2.3 – 4.6m	3.7m	6.8m	7.15m
Process node	2 μm	1.2 μm	0.35 μm	0.32 μm	0.35 μm	0.25 μm
Die size	80 mm ²	40 mm ²	84 – 100 mm ²	84 mm ²	350 mm ²	229 mm ²
Speed	12 – 33 MHz	20 – 40 MHz	50 - 250 MHz	150 – 266 MHz	180 – 360 MHz	270 – 400 MHz

Flagship devices

DECstation 2100 and 3100 workstations

Sony PlayStation game console

SGI IRIS and Indigo workstations

NASA New Horizons space probe

Nintendo N64 game console

Carrera Computers and DeskStation Technology PCs (Windows NT)

SGI Onyx, Indigo, Indigo2, and Indy workstations

SGI O2 and Indy workstations

Cobalt Qube servers

HP LJ4000 laser printers

SGI Indigo2 and Octane workstations

SGI Onyx and Onyx2 supercomputers

NEC Cenju-4 supercomputers

Siemens Nixdorf servers

SGI Octane 2, Onyx 2, and Origin workstations

¿Por qué MIPS?

- ☐ Muchas alternativas RISC: ARM o PowerPC, ¿por qué MIPS?
- ☐ **Una cuestión de docencia**
 - ☐ Arquitectura de referencia utilizada en la mayor parte de los textos docentes disponibles para la enseñanza de Arquitectura de Computadores
 - ☐ Microprocesador estándar estudiado en muchas universidades del mundo
- ☐ Principales motivos
 - ☐ **Diseño sencillo y público**
 - ☐ En la década de los 90 uno de cada tres microprocesadores que salían al mercado estaban basados en un núcleo MIPS
- ☐ Los repertorios de microprocesadores RISC son muy parecidos entre sí, conociendo uno podemos entender fácilmente otros identificando sus particularidades

Ejemplos de aplicación

- ❑ No hay que considerar sólo los procesadores diseñados para PC's y portátiles. Existen otras muchas aplicaciones de las arquitecturas estudiadas

P: ¿Qué tienen en común las videoconsolas y la arquitectura de computadores?

R: Las videoconsolas, al igual que teléfonos móviles o las tabletas poseen, al menos, un microprocesador que gobierna su funcionamiento

- ❑ Se trata de diseños RISC, en la actualidad casi siempre basados en la arquitectura PowerPC o ARM
- ❑ Por su parte, los **productos que hoy se basan en la arquitectura MIPS** se pueden clasificar en
 - ❑ Digital Home: Multimedia Players, Top Box, HD Media Device
 - ❑ Mobile: Netbooks, Multimedia Players, Navigation Devices, Readers
 - ❑ Networking: Access Points, Modems, Routers, Wireless Routers

Investigadores pioneros

❑ John L. Hennessy

- ❑ En 1981 el doctor Hennessy reunió a varios investigadores para centrarse en la arquitectura de computadores RISC
- ❑ Hennessy ayudó a transferir esta tecnología a la industria y a darla a conocer con sus múltiples publicaciones
- ❑ En 1984, **cofundó la compañía MIPS** Computer Systems, hoy día MIPS Technologies
- ❑ En los últimos años, su investigación se ha centrado en la Arquitectura de Computadores de altas prestaciones

❑ David A. Patterson

- ❑ Dirigió el diseño e **implementación del RISC 1**, probablemente el primer procesador RISC producido en VLSI (*Very Large Integration Scale*)
- ❑ Su enseñanza ha sido galardonada en múltiples ocasiones por el ACM, el IEEE y la Universidad de California

Nociones básicas

☐ **CISC**

- ☐ Juego de instrucciones más complejo => ruta de datos más compleja
- ☐ Compilador más sencillo, código ensamblador más corto
- ☐ Hardware más complicado

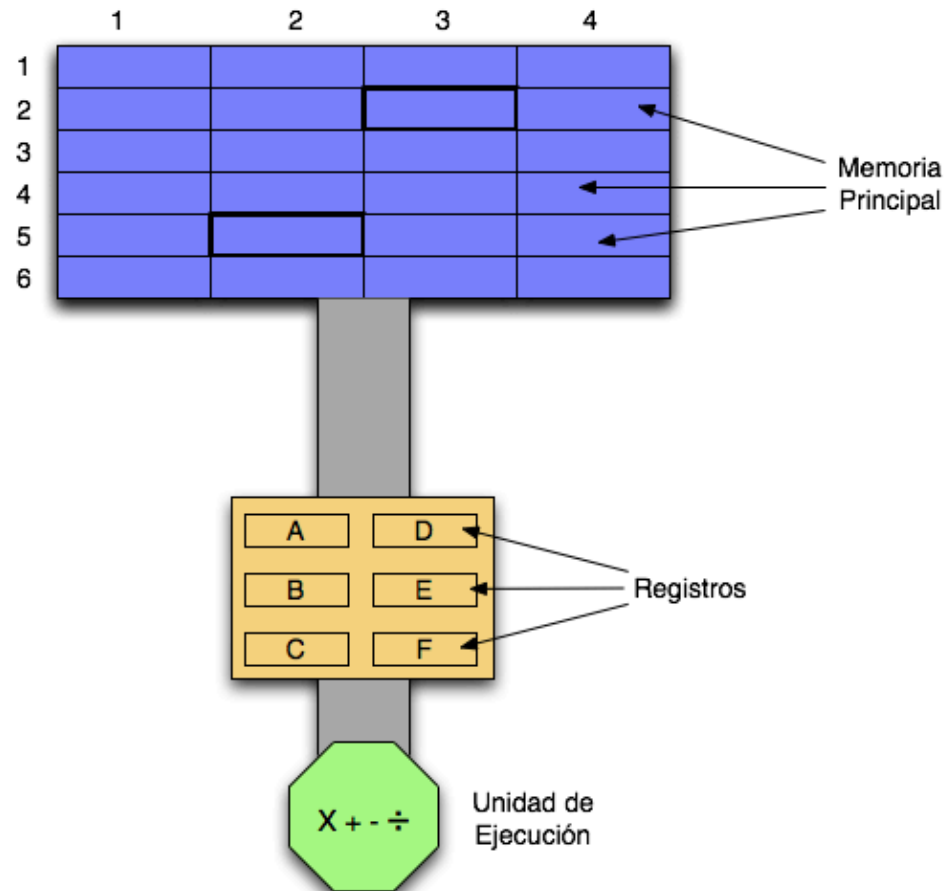
☐ **RISC**

- ☐ Reducen la complejidad del procesador
- ☐ Aumentan la velocidad de procesamiento: lógica cableada, pocos accesos a memoria
- ☐ Repertorio simple; pocos tipos de datos y modos de direccionamiento
- ☐ Programas más largos pero más fácilmente optimizables

RISC Vs CISC

Ejemplo: multiplicación

□ $(2:3) \times (5:2) \Rightarrow (2:3)$

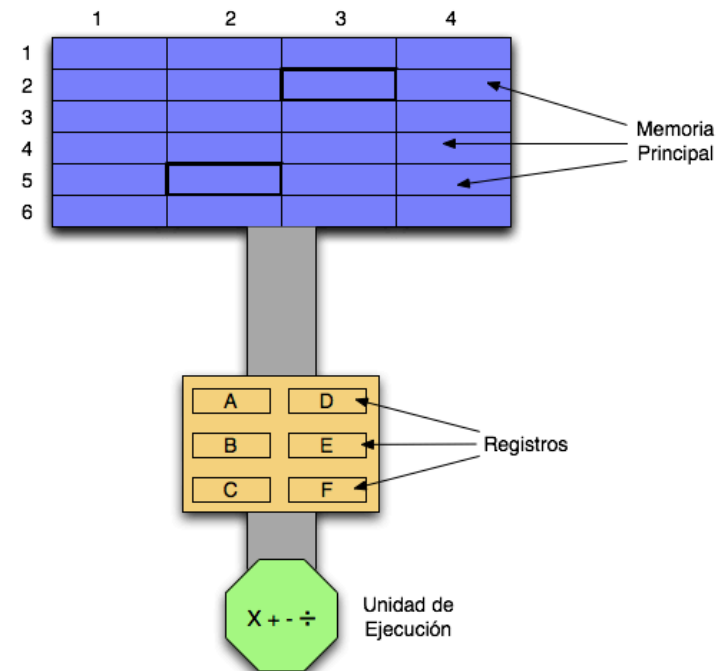


Solución CISC

- ❑ **Objetivo:** completar una tarea en el menor número de líneas de código ensamblador posibles
- ❑ Construcción de un microprocesador capaz de comprender y ejecutar una serie de operaciones complejas
- ❑ La tarea completa puede ser llevada a cabo con una única instrucción específica, *MULT*
- ❑ $(2:3) \times (5:2) \Rightarrow (2:3)$
- ❑ Lenguaje de alto nivel:
 $a = a * b$

MULT (2 : 3) , (5 : 2)

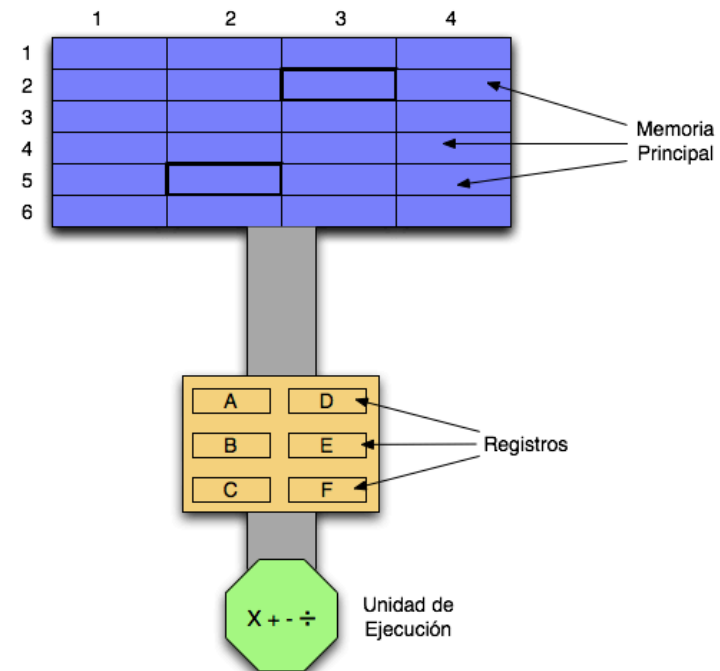
- ❑ ¿Ventajas, inconvenientes?



Solución RISC

- ❑ **Objetivo:** usan instrucciones sencillas que se puedan ejecutar rápidamente
- ❑ Arquitecturas basadas en **registros de propósito general** que operan siempre sobre operandos almacenados en el procesador, cerca de la unidad de ejecución
- ❑ MULT se convierte en:
LOAD, PROD, STORE

```
LOAD  A, (2:3)
LOAD  B, (5:2)
PROD  A, B
STORE (2:3), A
```



Solución RISC

- ❑ ¿Menos eficiente? Más instrucciones => más RAM
- ❑ Más trabajo para el compilador ...
- ❑ ... sin embargo, el **tiempo de ejecución es similar** al MULT de la arquitectura CISC, con la ventaja de necesitar un hardware más sencillo que deja espacio para registros y que es más fácilmente optimizable

```
LOAD  A, (2:3)
LOAD  B, (5:2)
PROD  A, B
STORE (2:3), A
```

Vs

```
MULT (2:3), (5:2)
```

Generalidades

- ☐ MIPS32, arquitectura RISC basada en **registros de propósito general** de tipo carga/almacenamiento
- ☐ Los **operandos** de una instrucción siempre deben estar almacenados en registros dentro del procesador (no puedan estar en memoria)
- ☐ Los resultados siempre se devuelven a registros dentro del procesador
- ☐ La arquitectura del MIPS32 se basa en un juego de instrucciones de longitud fija
- ☐ **32 registros de propósito general** de 32 bits
- ☐ **32 registros para coma flotante** (F0-F31)

Generalidades: gestión de memoria

- ☐ Un conjunto de registros refleja la configuración de las cachés, **MMU** (*Memory Management Unit*), **TLB** y otras funcionalidades en modo supervisor
- ☐ La información proporcionada por los registros de configuración permite la implementación de sistemas operativos en tiempo real
- ☐ Incluye opciones de control de caché bien definidas. El tamaño de las cachés de datos e instrucciones puede variar desde los 256 bytes a los 4 MBytes
- ☐ La memoria caché de datos puede emplear las políticas ***write-back*** o ***write-through***
- ☐ El mecanismo de gestión de memoria puede emplear TLB (*Translation Lookaside Buffer*) ó BAT (*Block Address Translation*)
- ☐ Con TLB, el MIPS32 cumple con los requisitos de gestión de memoria de Windows CE, Linux y Android

Generalidades

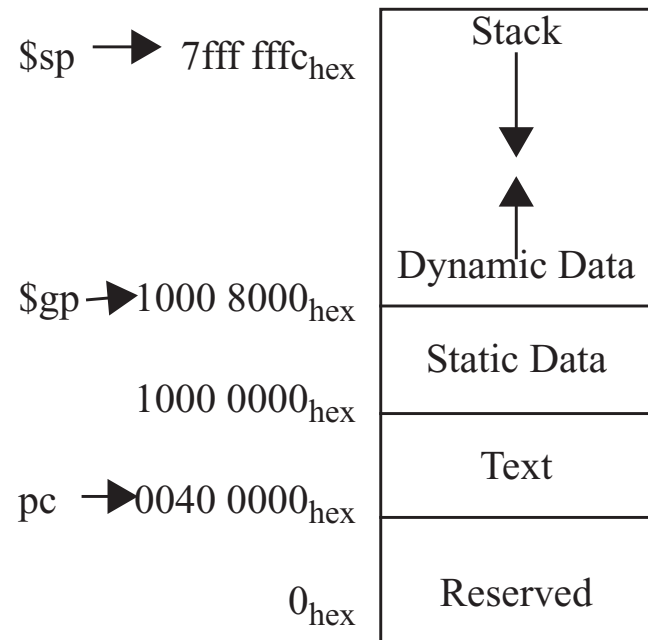
- ☐ Las instrucciones de **longitud fija** de 32-bits permiten una decodificación más fácil
- ☐ 32 x 32-bit General Purpose Register (**GPR**)
- ☐ Instrucciones RISC robusta de load/store RISC con 3-operandos para la mayoría de los formatos (3 registros, 2 registros + inmediato), opciones de branch/jump e instrucciones de salto retardado
- ☐ 32 bits de espacio de direcciones virtuales; hasta 36 bits de espacio de direcciones físicas
- ☐ **Modos de direccionamiento** sencillos
- ☐ Soporte para variables de 8-bit, 16-bit y 32-bit
- ☐ Soporte para multiplicación y división entera
- ☐ Soporte opcional para número en coma flotante de precisión simple (32 bits) y doble (64 bits)
- ☐ Soporte para los sistemas ***Big-Endian*** y ***Little-Endian***

Registros

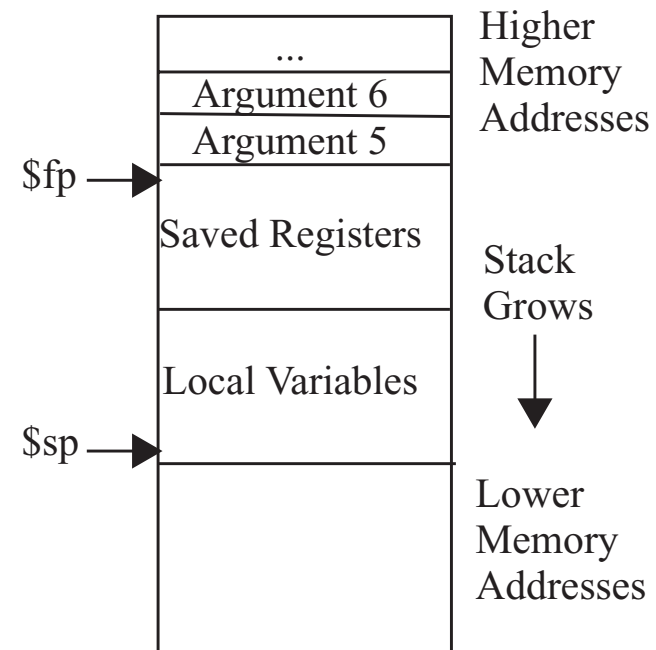
NAME	NUMBER	USE	PRESERVED ACROSS A CALL?
\$zero	0	The Constant Value 0	N.A.
\$at	1	Assembler Temporary	No
\$v0-\$v1	2-3	Values for Function Results and Expression Evaluation	No
\$a0-\$a3	4-7	Arguments	No
\$t0-\$t7	8-15	Temporaries	No
\$s0-\$s7	16-23	Saved Temporaries	Yes
\$t8-\$t9	24-25	Temporaries	No
\$k0-\$k1	26-27	Reserved for OS Kernel	No
\$gp	28	Global Pointer	Yes
\$sp	29	Stack Pointer	Yes
\$fp	30	Frame Pointer	Yes
\$ra	31	Return Address	Yes

Memoria

MEMORY ALLOCATION



STACK FRAME



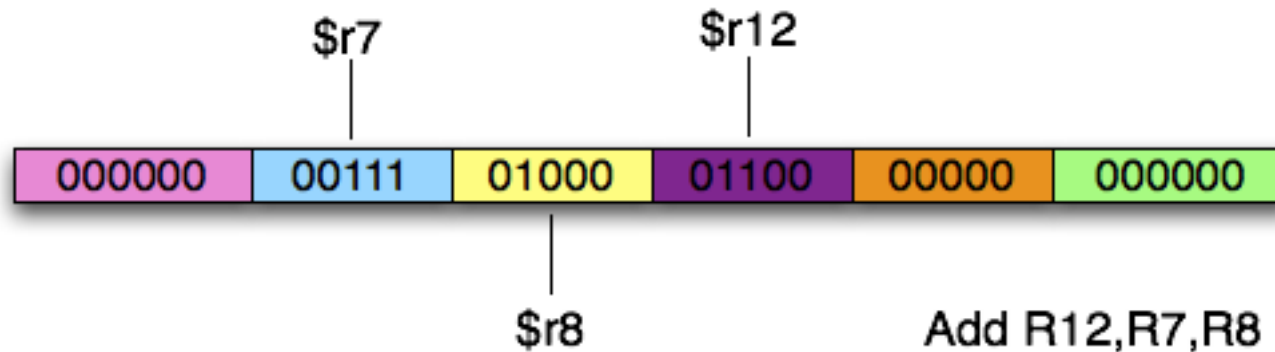
Alineación

DATA ALIGNMENT

Double Word							
Word				Word			
Halfword		Halfword		Halfword		Halfword	
Byte	Byte	Byte	Byte	Byte	Byte	Byte	Byte
0	1	2	3	4	5	6	7

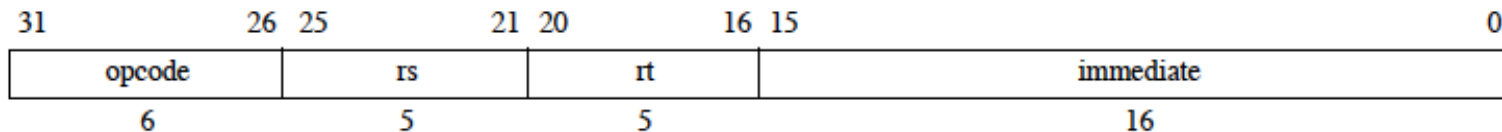
Instruction Set Architecture, ISA

- ❑ Consiste en unas 111 instrucciones, cada una de las cuales **se codifica con 32 bits** (codificación de longitud fija)



- ❑ Suma los valores contenidos en los registros 7 y 8, y guarda el resultado de la suma en el registro 12
- ❑ El procesador identifica el tipo de instrucción mediante los dígitos binarios correspondientes a los campos primero y último
- ❑ Los operandos están representados en los campos azul, amarillo y morado
- ❑ El campo naranja representa un valor que no se utiliza en este caso concreto: el campo *Shift Amount*

Instrucciones tipo I (Immediate)



☐ Load/Store

- ☐ **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

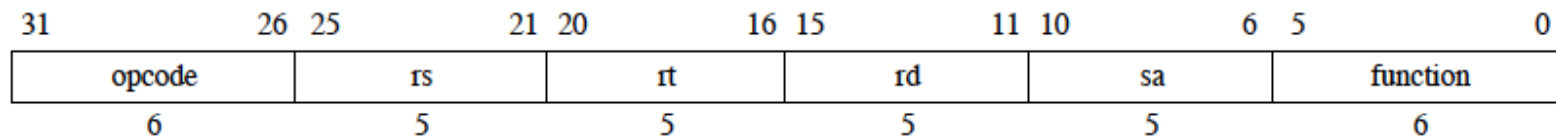
☐ Aritmético-lógicas con direccionamiento inmediato

- ☐ **RS (registro fuente)**: operando 1
- ☐ **RT (registro destino)**: registro destino de la operación
- ☐ **Inmediato**: operando 2, directamente su valor

☐ Saltos condicionales/incodicionales

- ☐ **RS (registro fuente)**: registro de condición (para la comparación)/Registro que contiene la dirección destino del salto
- ☐ **RT (registro destino)**: registro de condición (para la comparación)/No se utiliza
- ☐ **Inmediato**: desplazamiento respecto del PC/0

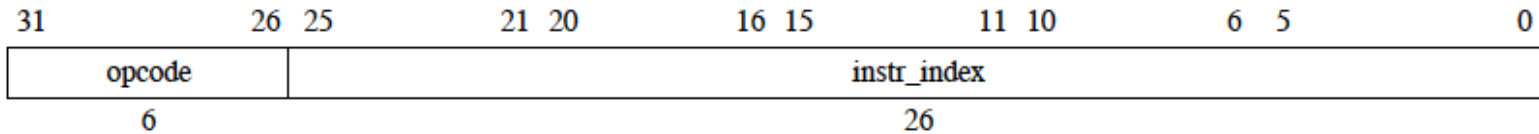
Instrucciones tipo R (Register)



☐ **Aritmético-lógicas** registro-registro

- ☐ **RS (registro fuente)**: operando 1
- ☐ **RT (registro destino)**: operando 2
- ☐ **RD**: registro destino
- ☐ **sa (Shift Amount)**: indica el desplazamiento para las instrucciones de tipo Shift
- ☐ **function**: junto con el OpCode indica el tipo de operación que se debe realizar

Instrucciones tipo J (Jump)



- ☐ **Salto incondicional y retorno de procedimiento** que utilizan direccionamiento con desplazamiento relativo al PC
 - ☐ **opCode**: Código de la operación
 - ☐ **Instr_index**: offset relativo el PC