Repertori d'instruccions del MIPS32

				R	op	rd	rs	rt	shamt	funct
				I	op	rs	rt	immediate / offset		
				J	op	target				
Grp	*Significat de la instrucció*	*Format en assemblador*	*Operació*	*MF*	*31 26*	*25 21*	*20 16*	*15 11*	*10 6*	*5 0*
	Load **W**ord	**lw** rt, *offset*(rs)	rt ← MemWord[$(rs)_u$ + $offset_s$] ***(W*)***	I	100011	rs	rt	*offset*		
	Load **B**yte	**lb** rt, *offset*(rs)	rt ← MemByte[$(rs)_u$ + $offset_s$] ***(S*) (U*)***	I	100000	rs	rt	*offset*		
	Store **W**ord	**sw** rt, *offset*(rs)	MemWord[$(rs)_u$ + $offset_s$] ← (rt) ***(W*)***	I	101011	rs	rt	*offset*		
	Store **B**yte	**sb** rt, *offset*(rs)	MemByte[$(rs)_u$ + $offset_s$] ← $(rt)_{7..0}$	I	101000	rs	rt	*offset*		
	Load **U**pper **I**mmediate	**lui** rt, *immediate*	rt ← *immediate* || 0^{16}	I	001111	00000	rt	*immediate*		
	Move **F**rom **Hi** Register	**mfhi** rd	rd ← (Hi)	R	000000	00000	00000	rd	00000	010000
	Move **F**rom **Lo** Register	**mflo** rd	rd ← (Lo)	R	000000	00000	00000	rd	00000	010010
	Add Word	**add** rd, rs, rt	rd ← $(rs)_s$ + $(rt)_s$ ***(U*)***	R	000000	rs	rt	rd	00000	100000
	Add Immediate Word	**addi** rt, rs, *immediate*	rd ← $(rs)_s$ + $immediate_s$ ***(U*)***	I	001000	rs	rt	*immediate*		
	Substract Word	**sub** rd, rs, rt	rd ← $(rs)_s$ - $(rt)_s$ ***(U*)***	R	000000	rs	rt	rd	00000	100010
	Multiply Word to GPR	**mul** rd, rs, rt	rd ← $((rs)_s$ * $(rt)_s)_{31..0}$	R	011100	rs	rt	rd	00000	000010
	Multiply Word	**mult** rs, rt	Hi ← $((rs)_s$ * $(rt)_s)_{63..32}$; Lo ← $((rs)_s$ * $(rt)_s)_{31..0}$ ***(U*)***	R	000000	rs	rt	00000	00000	011000
	Divide Word	**div** rs, rt	Lo ← $(rs)_s$ div $(rt)_s$; Hi ← $(rs)_s$ mod $(rt)_s$ ***(U*)***	R	000000	rs	rt	00000	00000	011010
	Set on **L**ess Than	**slt** rd, rs, rt	***if*** ($(rs)_s$ < $(rt)_s$) ***then*** rd ← 1 ***else*** rd ← 0 ***(U*)***	R	000000	rs	rt	rd	00000	101010
	Set on **L**ess **T**han **I**mmediate	**slti** rt, rs, *immediate*	***if*** ($(rs)_s$ < $immediate_s$) ***then*** rd ← 1 ***else*** rd ← 0 ***(U*)***	I	001010	rs	rt	*immediate*		
	Branch On **Eq**ual	**beq** rs, rt, *offset*	***if*** ((rs) == (rt)) ***then*** PC ← (PC) + $offset_s$ || 0^2 ***else*** PC ← (PC) + 4	I	000100	rs	rt	*offset*		
	Branch On **N**ot **E**qual	**bne** rs, rt, *offset*	***if*** ((rs) != (rt)) ***then*** PC ← (PC) + $offset_s$ || 0^2 ***else*** PC ← (PC) + 4	I	000101	rs	rt	*offset*		
	Branch On **G**reater **T**han **Z**ero	**bgtz** rs, *offset*	***if*** ($(rs)_s$ > 0) ***then*** PC ← (PC) + $offset_s$ || 0^2 ***else*** PC ← (PC) + 4	I	000111	rs	00000	*offset*		
	Branch On **L**ess **T**han **Z**ero	**bltz** rs, *offset*	***if*** ($(rs)_s$ < 0) ***then*** PC ← (PC) + $offset_s$ || 0^2 ***else*** PC ← (PC) + 4	I	000001	rs	00000	*offset*		
	Jump	**j** *target*	PC ← $(PC)_{31..28}$ || *target* || 0^2	J	000010	*target*				
	Jump **R**egister	**jr** rs	PC ← (rs) ***(L*)***	R	000000	rs	00000	00000	00000	001000
	Jump **A**nd **L**ink	**jal** *target*	$31 ← (PC) + 4; PC ← *target* || 0^2	J	000011	*target*				
	And	**and** rd, rs, rt	rd ← (rs) *and* (rt)	R	000000	rs	rt	rd	00000	100100
	Or	**or** rd, rs, rt	rd ← (rs) *or* (rt)	R	000000	rs	rt	rd	00000	100101
	E**x**clusive **Or**	**xor** rd, rs, rt	rd ← (rs) *xor* (rt)	R	000000	rs	rt	rd	00000	100110
	Shift Word **L**eft **L**ogical	**sll** rd, rt, *n*	rd ← $rt_{(31 - n)..0}$ || 0^n	R	000000	000000	rt	rd	*n*	000000
	Shift Word **R**ight **A**rithmetic	**sra** rd, rt, *n*	rd ← $rt_{31}{}^n$ || $rt_{31..n}$	R	000000	000000	rt	rd	*n*	000011
	Shift Word **R**ight **L**ogical	**srl** rd, rt, *n*	rd ← 0^n || $rt_{31..n}$	R	000000	000000	rt	rd	*n*	000010

(U*) Tenen variant sense signe (afegint una ***u*** al mnemònic). La instrucció ***lbu*** canvia el camp ***op*** pel valor 100100, ***addiu*** i ***sltiu*** incrementen en 1 el camp ***op*** i la resta incrementen en 1 el camp ***funct***.
(S*) S'extén el signe del byte llegit a 32 bits al registre. ***(L*)*** El valor d' ***rs*** ha d'estar alineat a paraula. ***(W*)*** L'adreça ha d'estar alineada a paraula. ***V*** || 0^n Indica la concatenació del valor ***V*** amb ***n*** zeros.