Aide mémoire ALMO
Jeu d'instructions MIPS

Instructions Arithmétiques/Logiques entre registres

Assembleur		Opération		Format
Add	Rd, Rs, Rt	Add overflow detection	Rd <- Rs + Rt	R
Sub	Rd, Rs, Rt	Subtract overflow detection	Rd <- Rs - Rt	R
Addu	Rd, Rs, Rt	Add no overflow	Rd <- Rs + Rt	R
Subu	Rd, Rs, Rt	Subtract no overflow	Rd <- Rs - Rt	R
Addi	Rt, Rs,I	Add Immediate overflow detection	Rt <- Rs + I	I
Addiu	Rt, Rs,I	Add Immediate no overflow	Rt <- Rs + I	I
Or	Rd, Rs, Rt	Logical Or	Rd <- Rs or Rt	R
And	Rd, Rs, Rt	Logical And	Rd <- Rs and Rt	R
Xor	Rd, Rs Rt	Logical Exclusive-Or	Rd <- Rs xor Rt	R
Nor	Rd, Rs, Rt	Logical Not Or	Rd <- Rs nor Rt	R
Ori	Rt, Rs, I	Or Immediate unsigned immediate	Rt <- Rs or I	I
Andi	Rt, Rs, I	And Immediate unsigned immediate	Rt <- Rs and I	I
Xori	Rt, Rs, I	Exclusive-Or Immediate unsigned immediate	Rt <- Rs xor I	I
Sllv	Rd, Rt, Rs	Shift Left Logical Variable 5 lsb of Rs is significant	Rd <- Rt << Rs	R
Srlv	Rd, Rt, Rs	Shift Right Logical Variable 5 lsb of Rs is significant	Rd <- Rt >> Rs	R
Srav	Rd, Rt, Rs	Shift Right Arithmetical Variable 5 lsb of Rs is significant	Rd <- Rt >>* Rs	R
Sll	Rd, Rt, sh	Shift Left Logical	Rd <- Rt << sh	R
Srl	Rd, Rt, sh	Shift Right Logical	Rd <- Rt >> sh	R
Sra	Rd, Rt, sh	Shift Right Arithmetical	Rd <- Rt >>* sh * : with sign extension	R
Lui	Rt, I	Load Upper Immediate 16 lower bits of Rt are set to zero	Rt <- I ‖ "0000"	I

Instructions Arithmétiques/Logiques (suite)

Assembleur		Opération		Format
Slt	Rd, Rs, Rt	Set if Less Than	Rd <- 1 if Rs<Rt else 0	R
Sltu	Rd, Rs, Rt	Set if Less Than Unsigned	Rd <- 1 if Rs<Rt else 0	R
Slti	Rt, Rs, I	Set if Less Than Immediate sign extended Immediate	Rt <- 1 if Rs < I else 0	I
Sltiu	Rt, Rs, I	Set if Less Than Immediate unsigned immediate	Rt <- 1 if Rs < I else 0	I
Mult	Rs, Rt	Multiply	Rs * Rt LO <- 32 low significant bits HI <- 32 high significant bits	R
Multu	Rs, Rt	Multiply Unsigned	Rs * Rt LO <- 32 low significant bits HI <- 32 high significant bits	R
Div	Rs, Rt	Divide	Rs / Rt LO <- Quotient HI <- Remainder	R
Divu	Rs, Rt	Divide Unsigned	Rs / Rt LO <- Quotient HI <- Remainder	R
Mfhi	Rd	Move From HI	Rd <- HI	R
Mflo	Rd	Move From LO	Rd <- LO	R
Mthi	Rs	Move To HI	HI <- Rs	R
Mtlo	Rs	Move To LO	LO <- Rs	R

Instructions de lecture/écriture mémoire

Assembleur		Opération		Format
Lw	Rt, I (Rs)	Load Word sign extended Immediate	Rt <- M (Rs + I)	I
Sw	Rt, I (Rs)	Store Word sign extended Immediate	M (Rs + I) <- Rt	I
Lh	Rt, I (Rs)	Load Half Word sign extended Immediate. Two bytes from storage is loaded into the 2 less significant bytes of Rt. The sign of these 2 bytes is extended on the 2 most signficant bytes.	Rt <- M (Rs + I)	I
Lhu	Rt, I (Rs)	Load Half Word Unsigned sign extended Immediate. Two bytes from storage is loaded into the the 2 less significant bytes of Rt, other bytes are set to zero	Rt <- M (Rs + I)	I
Sh	Rt, I (Rs)	Store Half Word sign extended Immediate. The Two less significant bytes of Rt are stored into storage	M (Rs + I) <- Rt	I
Lb	Rt, I (Rs)	Load Byte sign extended Immediate. One byte from storage is loaded into the less significant byte of Rt. The sign of this byte is extended on the 3 most signficant bytes.	Rt <- M (Rs + I)	I
Lbu	Rt, I (Rs)	Load Byte Unsigned sign extended Immediate. One byte from storage is loaded into the less significant byte of Rt, other bytes are set to zero	Rt <- M (Rs + I)	I
Sb	Rt, I (Rs)	Store Byte sign extended Immediate. The less significant byte of Rt is stored into storage	M (Rs + I) <- Rt	I

Instructions de Branchement

Assembleur		Opération		Format
Beq	Rs, Rt, Label	Branch if Equal	PC <- PC+4+(I*4) if Rs = Rt PC <- PC+4 if Rs ≠ Rt	I
Bne	Rs, Rt, Label	Branch if Not Equal	PC <- PC+4+(I*4) if Rs ≠ Rt PC <- PC+4 if Rs = Rt	I
Bgez	Rs, Label	Branch if Greater or Equal Zero	PC <- PC+4+(I*4) if Rs ≥ 0 PC <- PC+4 if Rs < 0	I
Bgtz	Rs, Label	Branch if Greater Than Zero	PC <- PC+4+(I*4) if Rs > 0 PC <- PC+4 if Rs ≤ 0	I
Blez	Rs, Label	Branch if Less or Equal Zero	PC <- PC+4+(I*4) if Rs ≤ 0 PC <- PC + 4 if Rs > 0	I
Bltz	Rs, Label	Branch if Less Than Zero	PC <- PC+4+(I*4) if Rs < 0 PC <- PC+4 if Rs ≥ 0	I
Bgezal	Rs, Label	Branch if Greater or Equal Zero and link	PC <- PC+4+(I*4) if Rs ≥ 0 PC <- PC+4 if Rs < 0 R31 <-PC+4 in both cases	I
Bltzal	Rs, Label	Branch if Less Than Zero and link	PC <- PC+4+(I*4) if Rs < 0 PC <- PC+4 if Rs ≥ 0 R31 <- PC+4 in both cases	I
J	Label	Jump	PC <- PC 31:28 ‖ I*4	J
Jal	Label	Jump and Link	R31 <- PC+4 PC <- PC 31:28 ‖ I*4	J
Jr	Rs	Jump Register	PC <- Rs	R
Jalr	Rs	Jump and Link Register	R31 <- PC+4 PC <- Rs	R
Jalr	Rd, Rs	Jump and Link Register	Rd <- PC+4 PC <- Rs	R

Instructions Systèmes

Assembleur	Opération	Format
Rfe	Restore From Exception SR <- SR 31:4 ‖ SR 5:2 Privileged instruction. Restore the prevous IT mask and mode	R
Break n	Breakpoint Trap SR <- SR 31:6 ‖ SR 3:0 ‖ "00" Branch to exception handler. PC <- "8000 0080" n defines the breakpoint number CR <- cause	R
Syscall	System Call Trap SR <- SR 31:6 ‖ SR 3:0 ‖ "00" Branch to exception handler PC <- "8000 0080" CR <- cause	R
Mfc0 Rt, Rd	Move From Control Coprocessor Rt <- Rd Privileged Instruction . The register Rd of the Control Coprocessor is moved into the integer register Rt	R
Mtc0 Rt, Rd	Move To Control Coprocessor Rd <- Rt Privileged Instruction . The integer register Rt is moved into the register Rd of the Control Coprocessor	R