Architettura degli Elaboratori I

Corso di Laurea Triennale in Informatica

Università degli Studi di Milano

Dipartimento di Informatica “Giovanni Degli Antoni”

Edizione 2 (Cognomi H-Z), A.A. 2022-2023, Nicola.Basilico@unimi.it

Riepilogo segnali di controllo

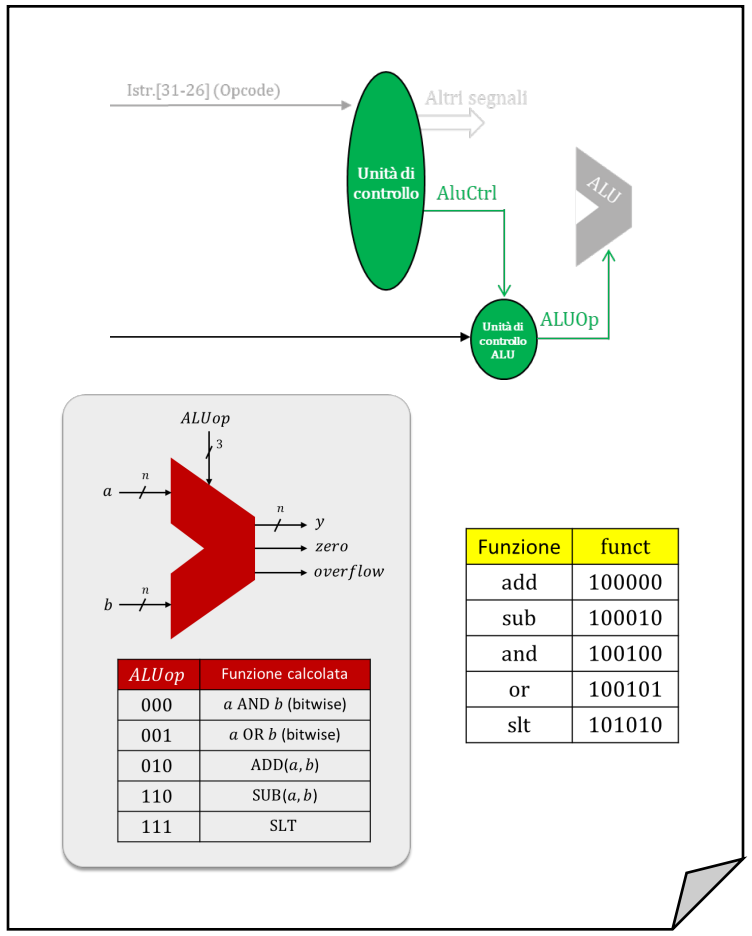
Segnali di lettura/scrittura: hanno un effetto solo quando posti a 1

RegWrite	Attiva la scrittura nel Register File: il dato D viene scritto nel registro indirizzato da WriteAddr
MemRead	Attiva la **lettura in memoria**: sull'uscita DataRead viene posto il valore della parola di memoria che ha indirizzo Addr
MemWrite	Attiva la **scrittura in memoria**: il dato D vene scritto nella parola di memoria che ha indirizzo Addr

Segnali di selezione

	Se posto a 0	Se posto a 1
ALUSrc	Il secondo operando della ALU viene dal Register File: uscita DataRead_2	Il secondo operando della ALU è l'estensione su 32 bit del campo IMMEDIATO dell'istruzione (bit 15-0)
MemToReg	Il dato D da scrivere nel Register File proviene dall'uscita della ALU	Il dato D da scrivere nel Register File dall'uscita di lettura DataRead della memoria
RegDst	L'indirizzo del registro destinazione è estratto dal campo r_t dell'istruzione (bit 20-16)	L'indirizzo del registro destinazione è estratto dal campo r_d dell'istruzione (bit 15-11)
Branch AND Zero	L'indirizzo da scrivere in PC è $PC + 4$ (a meno che Jump non sia posto a 1)	L'indirizzo da scrivere in PC è $PC + 4$ a cui si somma il valore del campo IMMEDIATO dell'istruzione (bit 15-0) esteso su 32 bit e moltiplicato per 4
Jump	L'indirizzo da scrivere in PC è $PC + 4$ (a meno che Branch AND Zero non sia posto a 1)	L'indirizzo da scrivere in PC ottenuto aggiungendo al valore del campo PSEUDO-INDIRIZZO dell'istruzione (bit 0-15) due zeri a destra e i 4 bit più significativi di PC a sinistra

Unità di controllo ALU



ALUop	Funzione calcolata
000	a AND b (bitwise)
001	a OR b (bitwise)
010	ADD(a, b)
110	SUB(a, b)
111	SLT

Funzione	funct
add	100000
sub	100010
and	100100
or	100101
slt	101010

- AluCtrl: comunica all'unità di controllo della ALU quale tipo di istruzione la CPU sta eseguendo, ci interessano solo le istruzioni aritmetico-logiche, gli accessi a memoria e la branch; con jump la ALU non è necessaria
- ALUOp: configura la ALU perché esegua una delle operazioni supportate

AluCtrl	ALUOp
Istruzione Aritmetico-Logica (tipo R)	La ALU fa l'operazione indicata da funct
Accesso a memoria (lw o sw)	La ALU fa una somma
Branch on equal	La ALU fa una differenza

- Iniziamo scegliendo una codifica per AluCtrl il cui valore dipende da Opcode

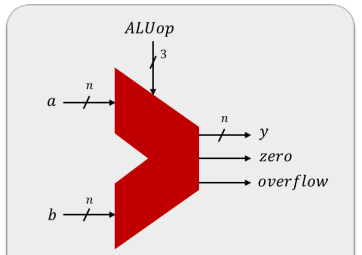
Opcode	AluCtrl
000000 (tutte le tipo R)	10
100011 (lw)	00
101011 (sw)	00
000100 (beq)	01

AluCtrl = 10 → Aritmetico-logica

AluCtrl = 00 → Accesso a memoria

AluCtrl = 01 → Branch

Unità di controllo ALU



ALUop	Funzione calcolata
000	a AND b (bitwise)
001	a OR b (bitwise)
010	ADD(a, b)
110	SUB(a, b)
111	SLT

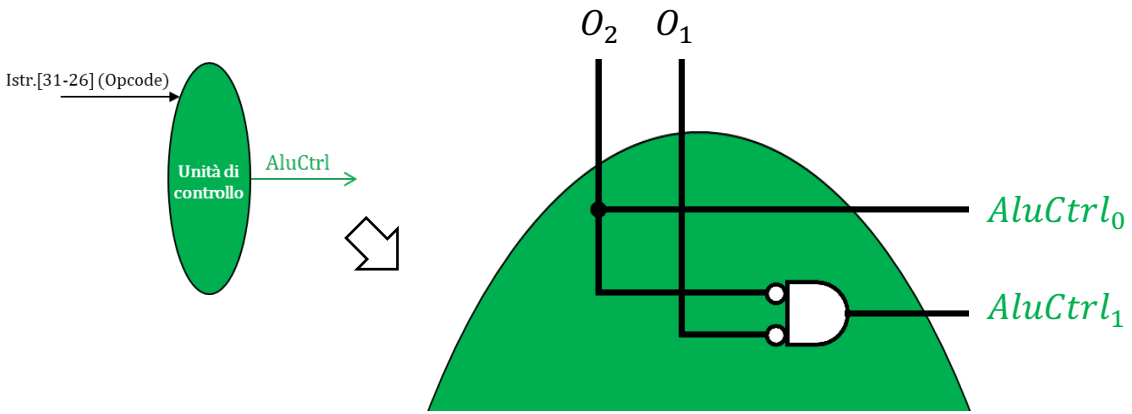
Funzione	funct
add	100000
sub	100010
and	100100
or	100101
slt	101010

Opcode	AluCtrl
000000 (tutte le tipo R)	10
100100 (lw)	00
101100 (sw)	00
000100 (beq)	01

	Opcode						*AluCtrl*	
	O_5	O_4	O_3	O_2	O_1	O_0	$AluCtrl_1$	$AluCtrl_0$
A-R →	0	0	0	**0**	**0**	0	1	0
lw →	**1**	0	**0**	0	1	1	0	0
sw →	1	0	**1**	0	1	1	0	0
beq →	0	0	0	**1**	0	0	0	1
j →	**0**	0	0	0	**1**	0	x	x

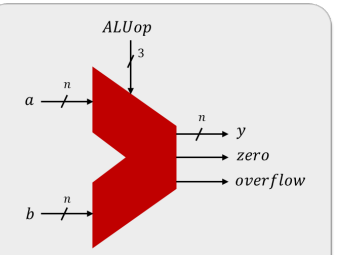
$\text{AluCtrl}_0 = o_2$

$\text{AluCtrl}_1 = \overline{o_2}\,\overline{o_1}$



Unità di controllo ALU

- Ora possiamo sintetizzare l'unità di controllo della ALU che, dato il segnale di controllo AluCtrl e funct, determina ALUOp



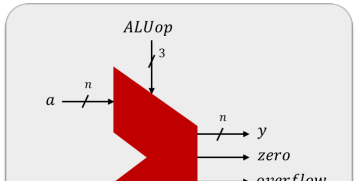
ALUop	Funzione calcolata
000	a AND b (bitwise)
001	a OR b (bitwise)
010	ADD(a, b)
110	SUB(a, b)
111	SLT

Funzione	funct
add	100000
sub	100010
and	100100
or	100101
slt	101010

Opcode	AluCtrl
000000 (tutte le tipo R)	10
100100 (lw)	00
101100 (sw)	00
000100 (beq)	01

	AluCtrl		funct						*ALUOp*		
	$AluCtrl_1$	$AluCtrl_0$	f_5	f_4	f_3	f_2	f_1	f_0	$ALUOp_2$	$ALUOp_1$	$ALUOp_0$
lw	0	0	x	x	x	x	x	x	0	1	0
sw	0	0	x	x	x	x	x	x	0	1	0
beq	0	1	x	x	x	x	x	x	1	1	0
add	1	0	1	0	0	0	0	0	0	1	0
sub	1	0	1	0	0	0	1	0	1	1	0
and	1	0	1	0	0	1	0	0	0	0	0
or	1	0	1	0	0	1	0	1	0	0	1
slt	1	0	1	0	1	0	1	0	1	1	1

Unità di controllo ALU



ALUop	Funzione calcolata
000	a AND b (bitwise)
001	a OR b (bitwise)
010	ADD(a, b)
110	SUB(a, b)
111	SLT

Funzione	funct
add	100000
sub	100010
and	100100
or	100101
slt	101010

Opcode	AluCtrl
000000 (tutte le tipo R)	10
100100 (lw)	00
101100 (sw)	00
000100 (beq)	01

	AluCtrl		funct						*ALUOp*		
	$AluCtrl_1$	$AluCtrl_0$	f_5	f_4	f_3	f_2	f_1	f_0	$ALUOp_2$	$ALUOp_1$	$ALUOp_0$
lw	0	0	x	x	x	x	x	x	0	1	0
sw	0	0	x	x	x	x	x	x	0	1	0
beq	0	1	x	x	x	x	x	x	1	1	0
add	1	0	1	0	0	0	0	0	0	1	0
sub	1	0	1	0	0	0	1	0	1	1	0
and	1	0	1	0	0	1	0	0	0	0	0
or	1	0	1	0	0	1	0	1	0	0	1
slt	1	0	1	0	1	0	1	0	1	1	1

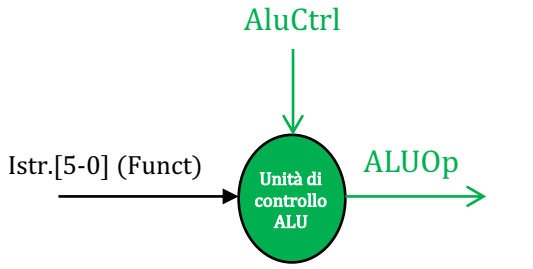
$$AluCtrl_i \rightarrow C_i$$

$$ALUOp_0 = C_1\,\overline{C_0} f_5 \overline{f_4}(\overline{f_3} f_2 \overline{f_1} f_0 + f_3 \overline{f_2} f_1 \overline{f_0})$$

$$ALUOp_1 = (\overline{C_1} + C_0) + (\overline{f_5} + f_4 + f_3 + \overline{f_2} + f_1)$$

$$ALUOp_2 = \overline{C_1} C_0 + C_1 \overline{C_0}(f_5 \overline{f_4} \overline{f_2} f_1 \overline{f_0})$$

Unità di controllo ALU

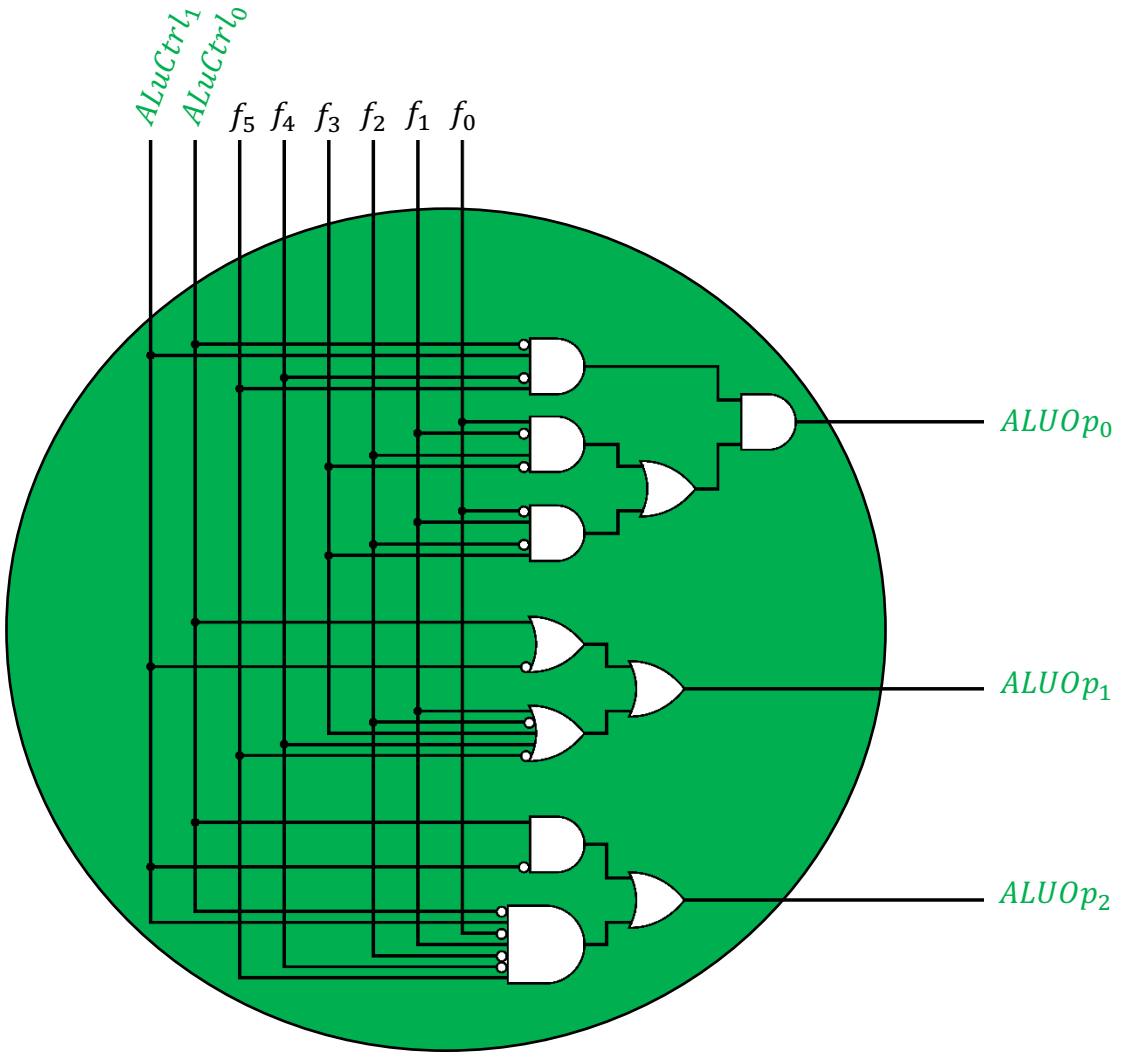


	AluCtrl		funct						*ALUOp*		
	$AluCtrl_1$	$AluCtrl_0$	f_5	f_4	f_3	f_2	f_1	f_0	$ALUOp_2$	$ALUOp_1$	$ALUOp_0$
lw	0	0	x	x	x	x	x	x	0	1	0
sw	0	0	x	x	x	x	x	x	0	1	0
beq	0	1	x	x	x	x	x	x	1	1	0
add	1	0	1	0	0	0	0	0	0	1	0
sub	1	0	1	0	0	0	1	0	1	1	0
and	1	0	1	0	0	1	0	0	0	0	0
or	1	0	1	0	0	1	0	1	0	0	1
slt	1	0	1	0	1	0	1	0	1	1	1

$$ALUOp_0 = C_1\,\overline{C_0} f_5 \overline{f_4}(\overline{f_3} f_2 \overline{f_1} f_0 + f_3 \overline{f_2} f_1 \overline{f_0})$$

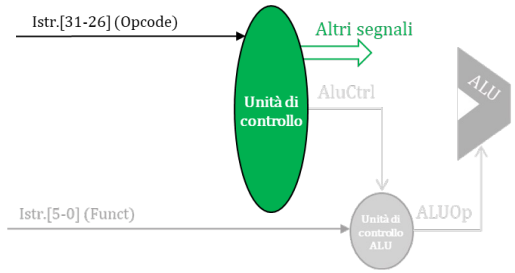
$$ALUOp_1 = (\overline{C_1} + C_0) + (\overline{f_5} + f_4 + f_3 + \overline{f_2} + f_1)$$

$$ALUOp_2 = \overline{C_1} C_0 + C_1 \overline{C_0}(f_5 \overline{f_4} \overline{f_2} f_1 \overline{f_0})$$



Unità di controllo principale

	o_5	o_4	o_3	o_2	o_1	o_0	Reg Write	Reg Dst	ALU Src	Mem Read	Mem Write	Mem ToReg	Branch	Jump
A-R	0	0	0	**0**	**0**	0	1	1	0	0	0	0	0	0
lw	**1**	0	**0**	0	1	1	1	0	1	1	0	1	0	0
sw	1	0	**1**	0	1	1	0	x	1	0	1	x	0	0
beq	**0**	0	0	**1**	0	0	0	x	0	0	0	x	1	0
j	0	0	0	0	**1**	**0**	0	x	x	0	0	x	x	1



$\text{RegWrite} = \overline{o_2}\,\overline{o_1} + o_5\overline{o_3}$

$\text{RegDst} = \overline{o_2}\;\overline{o_1}$

$\text{ALUSrc} = o_5$

$\text{MemRead} = o_5\overline{o_3}$

$\text{MemWrite} = o_3$

$\text{MemToReg} = o_5\overline{o_3}$

$\text{Branch} = \overline{o_5}o_2$

$\text{Jump} = o_1\overline{o_0}$

Unità di controllo finale