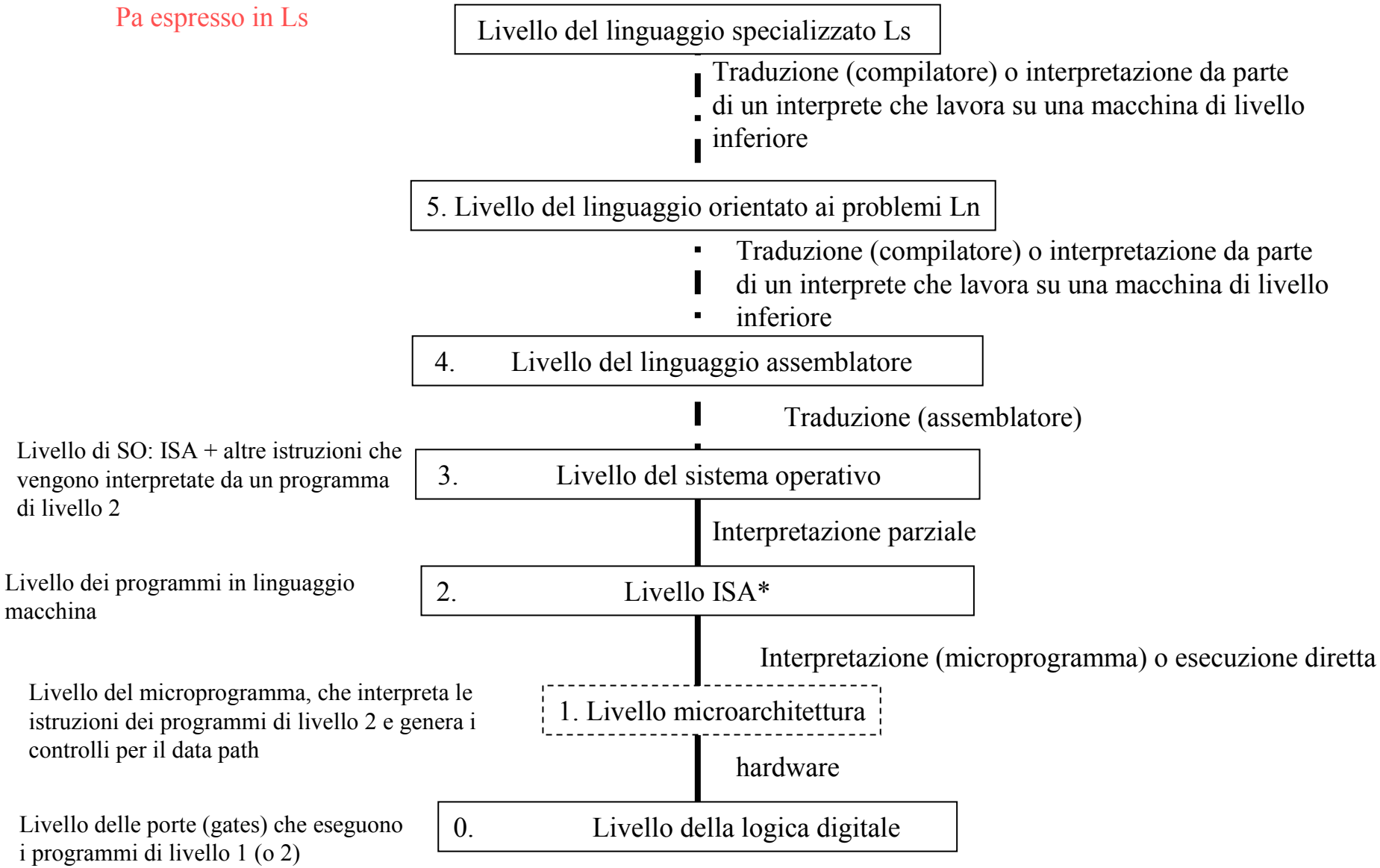
Calcolatori Elettronici B
a.a. 2004/2005

Tecniche di Controllo

Massimiliano Giacomin

Architetture

descrivono il calcolatore a diversi livelli di astrazione



*ISA: Instruction Set Architecture

Il calcolatore è progettato come una sequenza di macchine a diversi livelli, ognuna costruita sulle macchine definite ai livelli precedenti.
Ogni livello rappresenta una distinta astrazione, caratterizzata da differenti oggetti e operazioni.
I tipi di dati, le operazioni e le caratteristiche di ogni livello sono chiamate **architettura.**

- Dal livello 5 i linguaggi in cui esprimere i programmi sono chiamati linguaggi di alto livello: forniscono dati e operazioni per descrivere soluzioni di problemi in termini comprensibili per persone esperte in un certo campo.
- I programmi del livello 4 sono in forma simbolica; vengono generalmente tradotti (o a volte interpretati) da altri programmi.
- I programmi ai livelli 1, 2 e 3 sono sempre interpretati; la forma interpretata dalla macchina è sempre numerica.

I confini tra hardware e software sono sfumati:

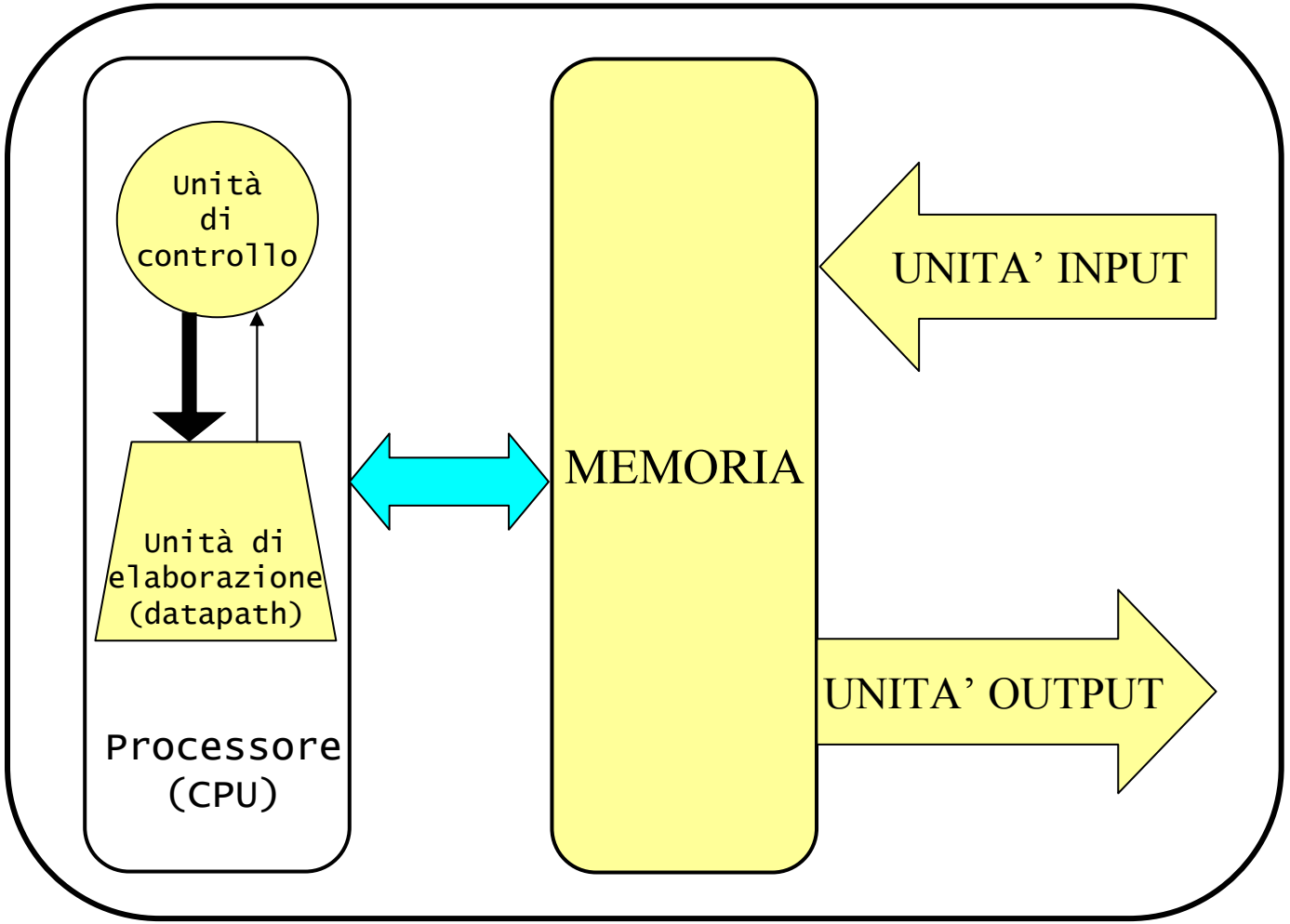
Hardware and and software are logically equivalent (Tanenbaum)
Hardware is just petrified software (K: Panetta Lenz)

ma “dovendo scegliere”:

> E' vero che il software non potrebbe esercitare i poteri della sua leggerezza se non mediante la pesantezza dell’hardware; ma è il software che comanda, che agisce sul mondo esterno e sulle macchine, le quali esistono solo in funzione del software, si evolvono in modo d'elaborare programmi sempre più complessi. (Italo Calvino)

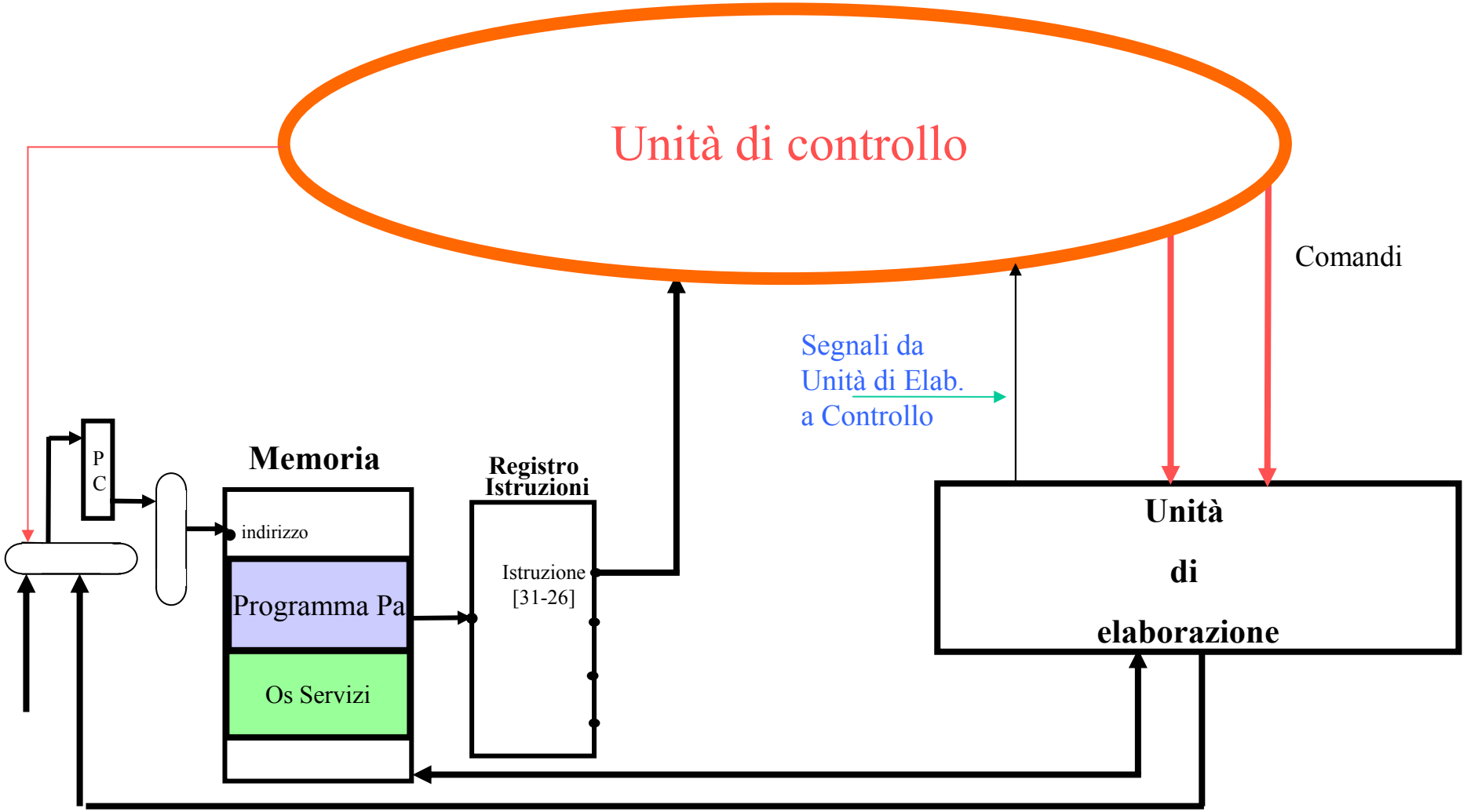
Schema del calcolatore

- Livelli 0-1 -



Schema del processore (e memoria)

Durante l'esecuzione di un programma applicativo Pa, i circuiti interpretano le istruzioni del programma in linguaggio macchina costituito dal < Pa (tradotto) ∘ i servizi OS>



Specifica e realizzazione del controllo

<u>Circuiti di controllo</u>: normalmente, sono presenti due parti

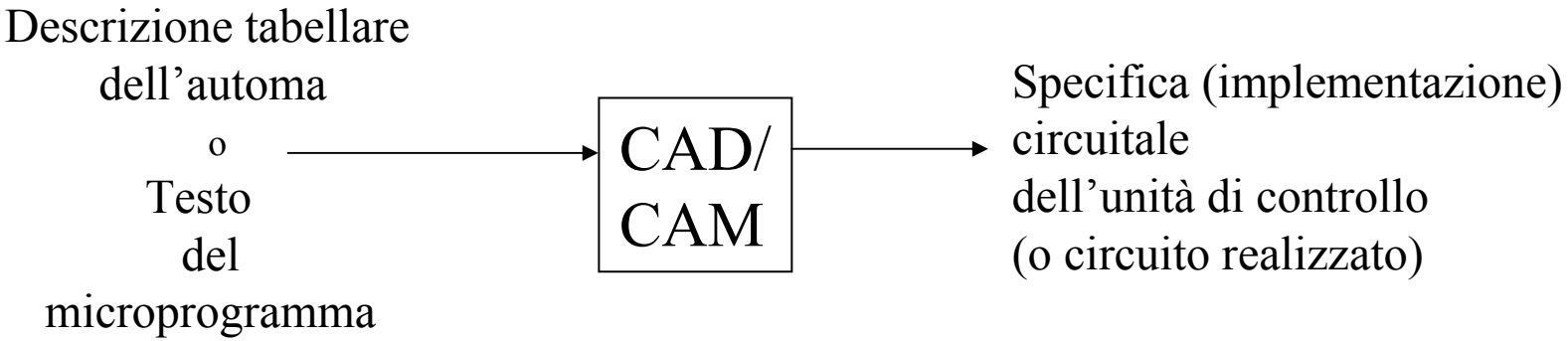
- **Combinatori** (reti combinatorie): per parte del processo di decodifica e di controllo (es. controllo della ALU), ma vedi anche controllo PIPELINE!
  - Specifica: Funzioni logiche | Tabelle di verità
  - Realizzazione: PLA | ROM
- **Sequenziali**: controllo "ad alto livello", p.es. sequenzializzazione operazioni nel controllo multiciclo.
  - Specifica: diagramma a stati finiti (FSM) | microprogramma
  - Realizzazione:

    Macchina a stati finiti (stato esplicito) | Sequenzializzatore (controllo microprogrammato)

(si può passare da una delle due tecniche di specifica ad una delle due tecniche di realizzazione)

Importanza della specifica

Data una specifica, esistono strumenti di progettazione assistita che ne permettono la traduzione in circuiti.
P.es. per il controllo multiciclo



NB: E' assolutamente necessario

- Aver bene in mente le differenze tra specifica ed implementazione
- Saper distinguere nei vari casi la natura sequenziale o combinatoria di una specifica unità di controllo (faremo vari esempi)

Faremo riferimento ad un sottoinsieme delle istruzioni MIPS:

- Istruzioni aritmetiche: add, sub, and, or, slt

  add rd, rs, rt // rd ← rs + rt

  slt rd, rs, rt // rd = 1 se rs < rt, 0 altrimenti

- Istruzioni di accesso a memoria:

  lw rt, offset(rs) // rt ← M[rs+offset]

  sw rt, offset(rs) // M[rs+offset] ← rt

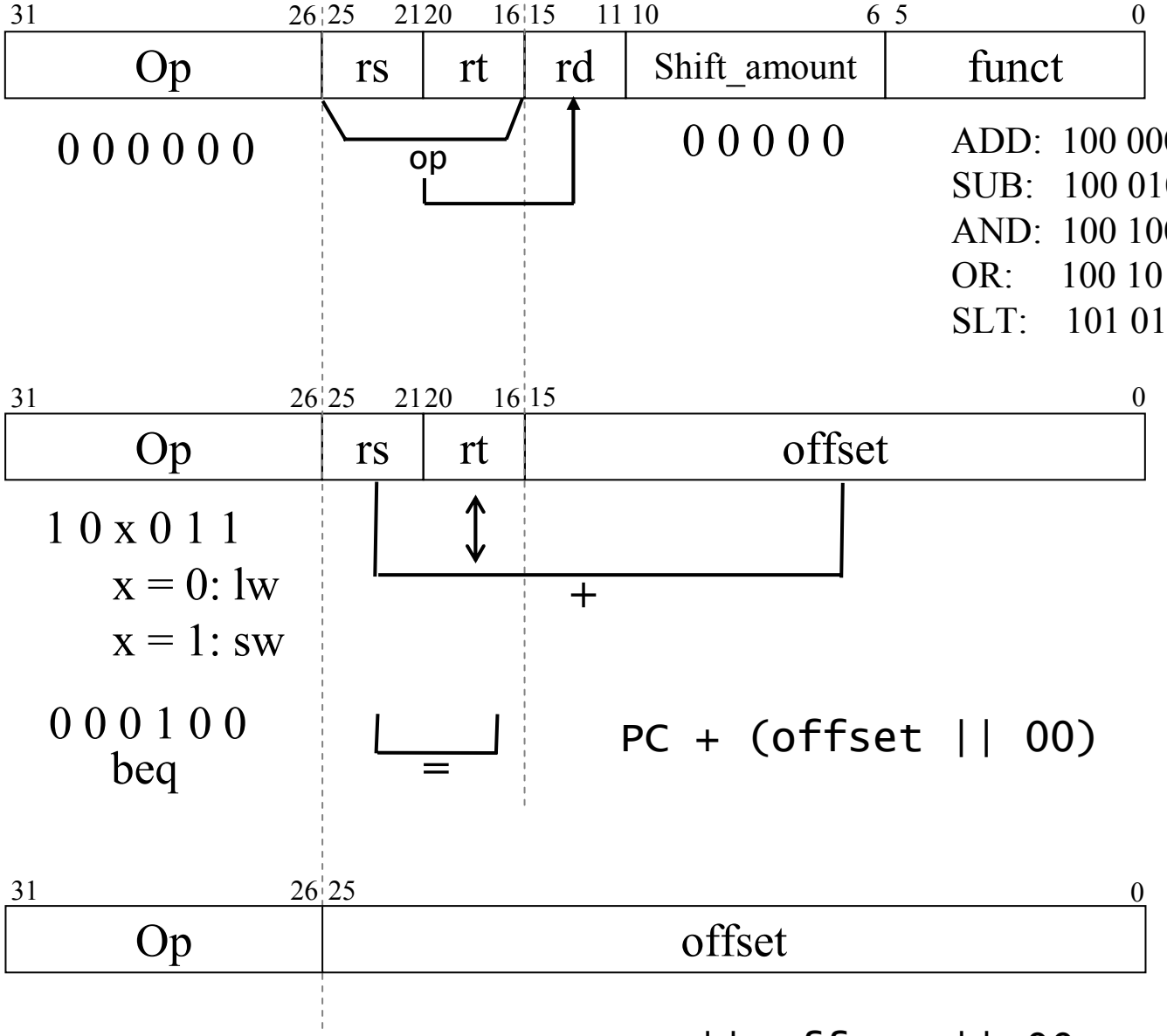
- Istruzioni di salto condizionato:

  beq rs, rt, offset // se rs=rt salta a offset *istruzioni* rispetto a PC
  (aggiornato a istruzione corrente + 4 bytes!)
  in bytes: PC + (offset || 00)

- Salto incondizionato:

  j offset // salta all'indirizzo *in istruzioni* ottenuto da:
  4 bit di PC || offset [30 bit]
  indirizzo in byte è la concatenazione di
  4 bit di PC || offset || 00 [32 bit]

Codifica delle istruzioni viste:

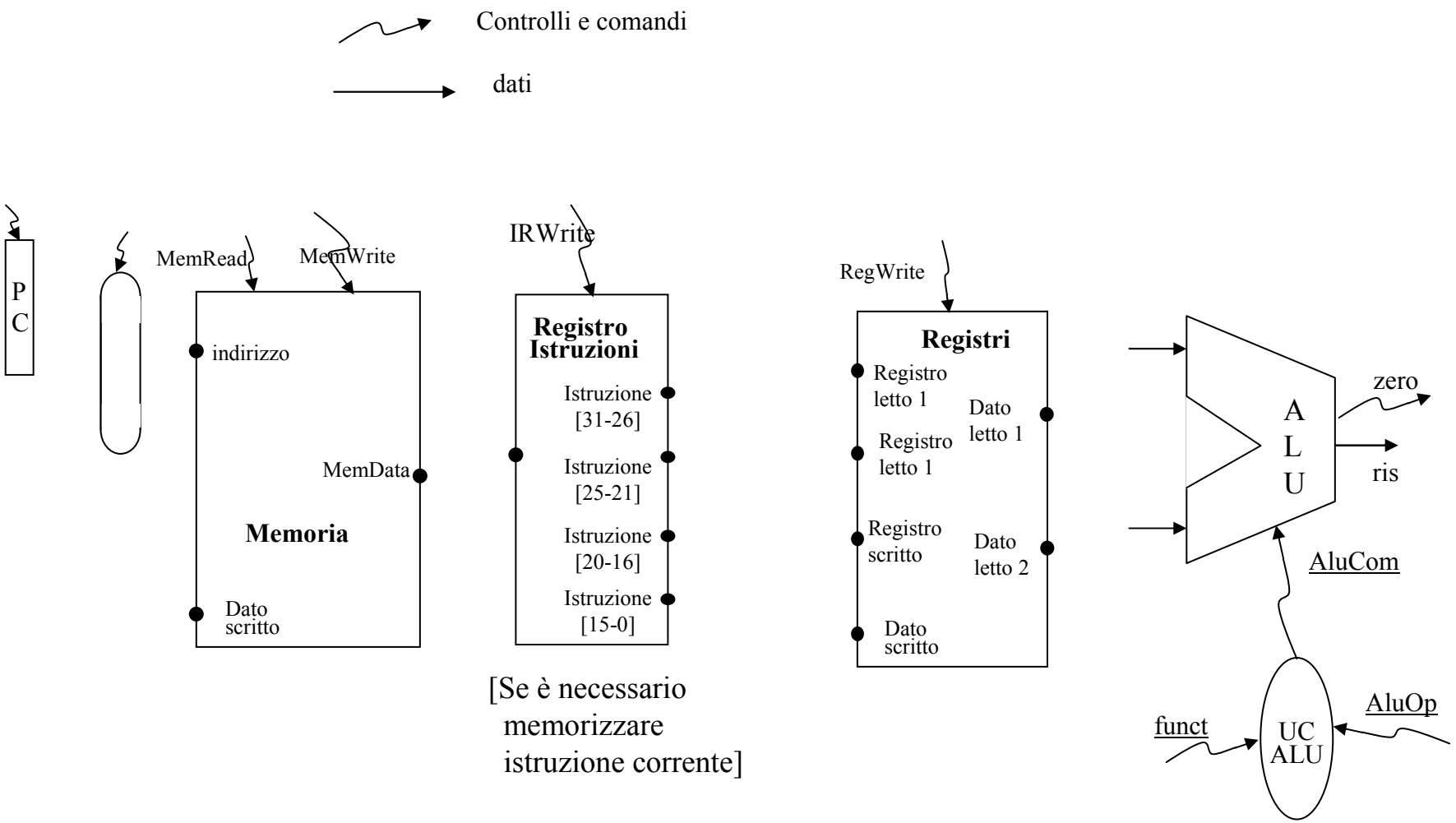


Aritmetiche: Tipo-R

lw, sw, beq: Tipo-I

J: Tipo-J

Glossario Visuale: indica le risorse individuate in base ad una prima analisi

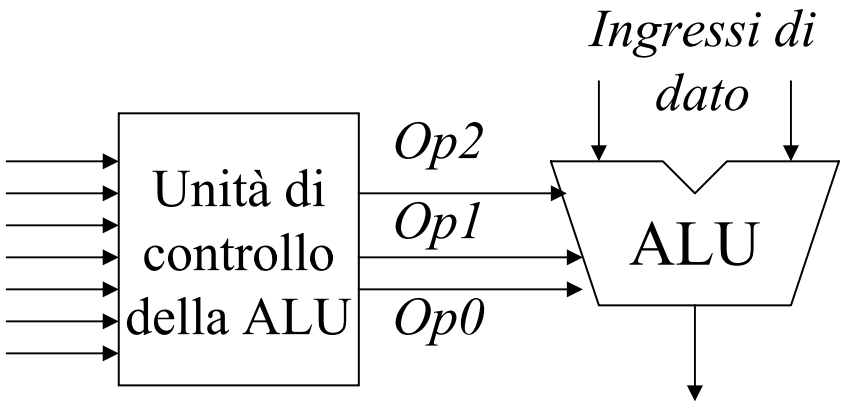


[Se è necessario memorizzare istruzione corrente]

Un esempio di controllo combinatorio: Il controllo della ALU

- Supponiamo che la ALU possa svolgere le 5 operazioni seguenti :
  - AND
  - OR
  - Somma
  - Sottrazione
  - Comparazione di minoranza (set on less than)
- Occorrono 3 ingressi di controllo (della ALU) per codificare le 5 operazioni possibili
- Cioè il blocco logico che implementa l'unità di controllo della ALU avrà 3 uscite, corrispondenti ai 3 bit di controllo che codificano una delle 5 operazioni possibili

Chiamiamo ***Op2***, ***Op1***, ***Op0*** le 3 **uscite** del blocco di controllo della ALU (ovvero l'**ingresso di controllo** della ALU)



Assumiamo la seguente codifica

	Op2	*Op1*	*Op0*
AND	0	0	0
OR	0	0	1
Somma	0	1	0
Sottrazione	1	1	0
Set on less than	1	1	1

- Gli **ingressi** dell'unità di controllo della ALU derivano da:
  - Un campo di controllo di 2 bit (***ALUOp***) che identifica la classe dell'istruzione da eseguire come segue:

Classe dell'istruzione	*ALUOp*	*Operazione*
Load word, store word	00	somma
Branch on equal	01	sottraz.
Tipo-R	10	***funct***

  - Il campo funzione (***funct***) dell'istruzione (se Tipo-R)
- I bit ALUOp vengono generati dall'unità di controllo principale in base al codice operativo (***op***) della istruzione

Relazione fra ingressi di controllo della ALU e istruzioni

Le operazioni previste da gran parte delle istruzioni del MIPS possono essere eseguite da una ALU in grado di svolgere le 5 operazioni viste

Codice operativo	ALUOp	Operazione	Campo funct	Azione della ALU	Ingresso di controllo della ALU
LW	00	load word	XXXXXX	somma	010
SW	00	store word	XXXXXX	somma	010
Branch equal	01	branch equal	XXXXXX	sottrazione	110
Tipo-R	10	somma	100000	somma	010
Tipo-R	10	sottrazione	100010	sottrazione	110
Tipo-R	10	AND	100100	and	000
Tipo-R	10	OR	100101	or	001
Tipo-R	10	set on less than	101010	set on less than	111

INGRESSI **USCITE**

Tabella di verità dei 3 bit di controllo della ALU

X (ALUOp0 = 1 identifica la riga)

ALUOp		*Funct*						*Operazione*		
ALUOp1	*ALUOp0*	*F5*	*F4*	*F3*	*F2*	*F1*	*F0*	*Op2*	*Op1*	*Op0*
0	0	X	X	X	X	X	X	0	1	0
0	1	X	X	X	X	X	X	1	1	0
1	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	1	0	1	1	0
1	0	1	0	0	1	0	0	0	0	0
1	0	1	0	0	1	0	1	0	0	1
1	0	1	0	1	0	1	0	1	1	1

La presenza di 1 identifica ultime 5 righe

X X X

IN OUT

Tabella di verità dei 3 bit di controllo della ALU

ALUOp		*Funct*						*Operazione*		
ALUOp1	***ALUOp0***	***F5***	***F4***	***F3***	***F2***	***F1***	***F0***	***Op2***	***Op1***	***Op0***
0	0	X	X	X	X	X	X	0	1	0
X	1	X	X	X	X	X	X	1	1	0
1	X	X	X	0	0	0	0	0	1	0
1	X	X	X	0	0	1	0	1	1	0
1	X	X	X	0	1	0	0	0	0	0
1	X	X	X	0	1	0	1	0	0	1
1	X	X	X	1	0	1	0	1	1	1

Bisogna implementare 3 funzioni distinte (per Op2, Op1 e Op0); sfruttando la presenza dei don't care, è possibile semplificare la tabella.
Esaminiamo dapprima la funzione per Op2=1…

ALUOp		*Funct*						*Operazione*		
ALUOp1	*ALUOp0*	*F5*	*F4*	*F3*	*F2*	*F1*	*F0*	*Op2*	*Op1*	*Op0*
0	0	X	X	X	X	X	X	0	1	0
X	1	X	X	X	X	X	X	1	1	0
1	X	X	X	0	0	0	0	0	1	0
1	X	X	X	0	0	1	0	1	1	0
1	X	X	X	0	1	0	0	0	0	0
1	X	X	X	0	1	0	1	0	0	1
1	X	X	X	1	0	1	0	1	1	1

Si vede che, per le ultime due righe evidenziate:

- la presenza di F1=1 (e di ALUOp1 = 1) le identifica tra le altre;
- anche tenendo conto che ALUOp non è mai 11…

➡ Le tre righe per cui Op2 = 1 sono equivalenti alle due righe seguenti:

X 1 X X X X X X (la prima delle tre evidenziate)
1 X X X X X 1 X (la combinazione delle ultime due evidenziate)

In modo equivalente…

ALUOp		*Funct*					
ALUOp1	***ALUOp0***	***F5***	***F4***	***F3***	***F2***	***F1***	***F0***
X	1	X	X	X	X	X	X
1	X	X	X	0	0	1	0
1	X	X	X	1	0	1	0
1	X	X	X	X	0	1	0
1	X	X	X	X	0	1	1
1	X	X	X	X	1	1	0
1	X	X	X	X	1	1	1
1	X	X	X	X	X	1	X

Combinazione

Righe per le quali non deve essere Op2 = 0

Si usano simili considerazioni per le altre due tabelle, ottenendo:

***Op2*=1**

ALUOp		*Funct*					
ALUOp1	*ALUOp0*	*F5*	*F4*	*F3*	*F2*	*F1*	*F0*
X	1	X	X	X	X	X	X
1	X	X	X	X	X	1	X

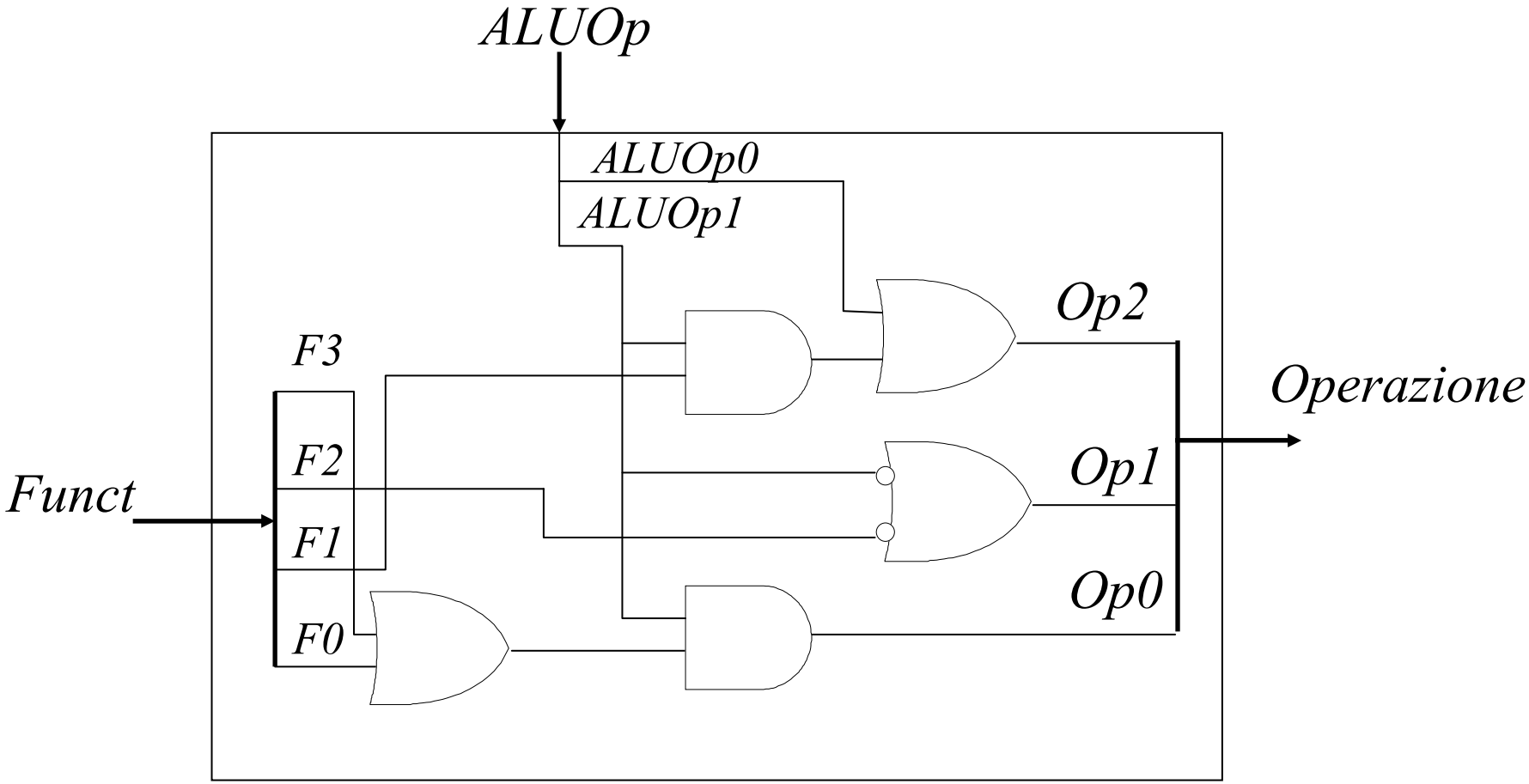
***Op1*=1**

ALUOp		*Funct*					
ALUOp1	*ALUOp0*	*F5*	*F4*	*F3*	*F2*	*F1*	*F0*
0	X	X	X	X	X	X	X
X	X	X	X	X	0	X	X

***Op0*=1**

ALUOp		*Funct*					
ALUOp1	*ALUOp0*	*F5*	*F4*	*F3*	*F2*	*F1*	*F0*
1	X	X	X	X	X	X	1
1	X	X	X	1	X	X	X

Blocco di controllo della ALU: Realizzazione circuitale



NB: nella pratica, per passare da specifica a implementazione si usano strumenti CAD:

- semplificano il processo di realizzazione (riducendo errori)
- utilizzano insiemi strutturati di porte logiche;
  p.es. PLA utilizza approccio a due livelli, generalmente più efficiente

ORA CONSIDERIAMO IL CONTROLLO DEL PROCESSORE.
VEDREMO:

- SINGOLO CICLO
- MULTICICLO
  - FSM
  - MICROPROGRAMMATO
- CON PIPELINE