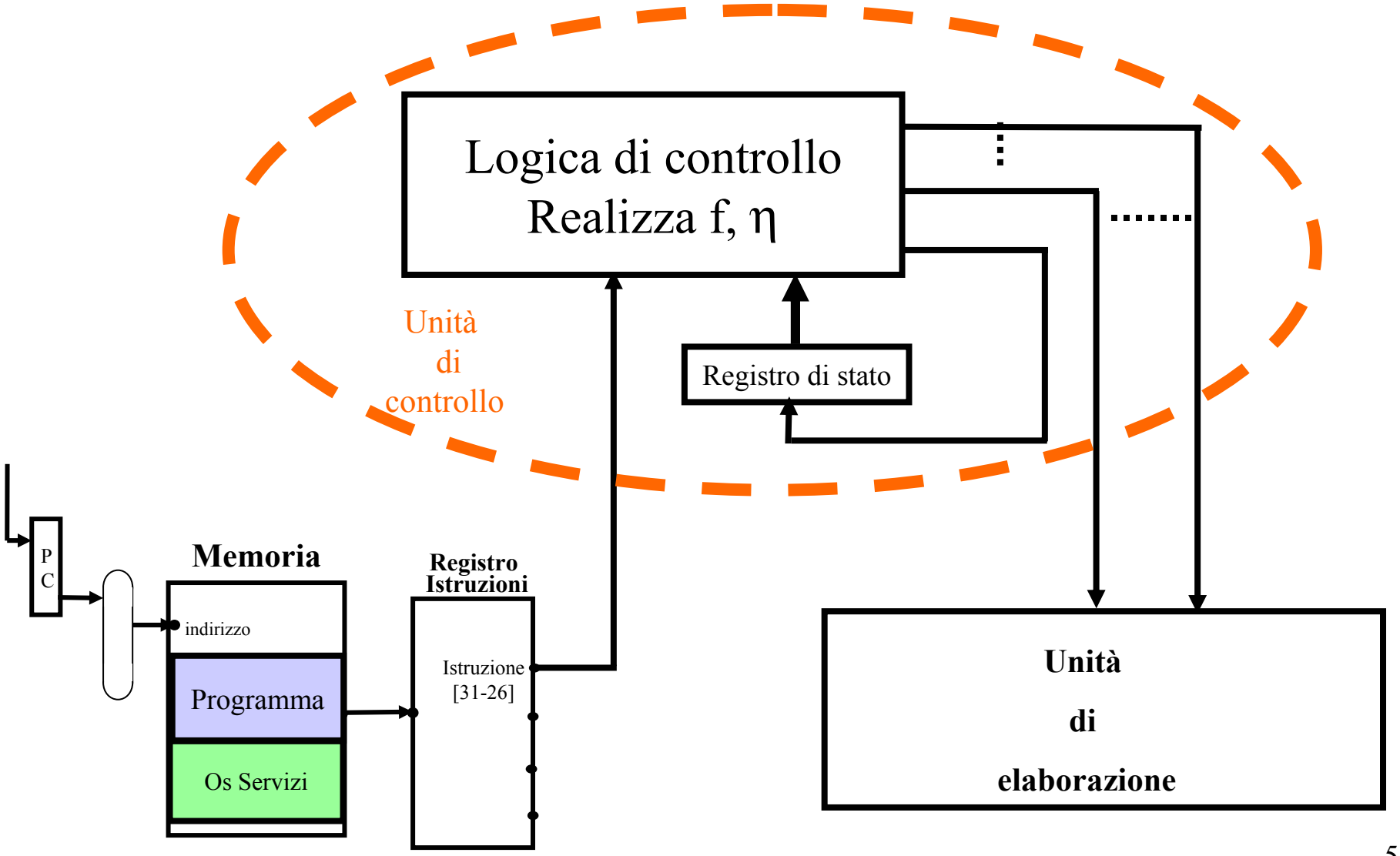
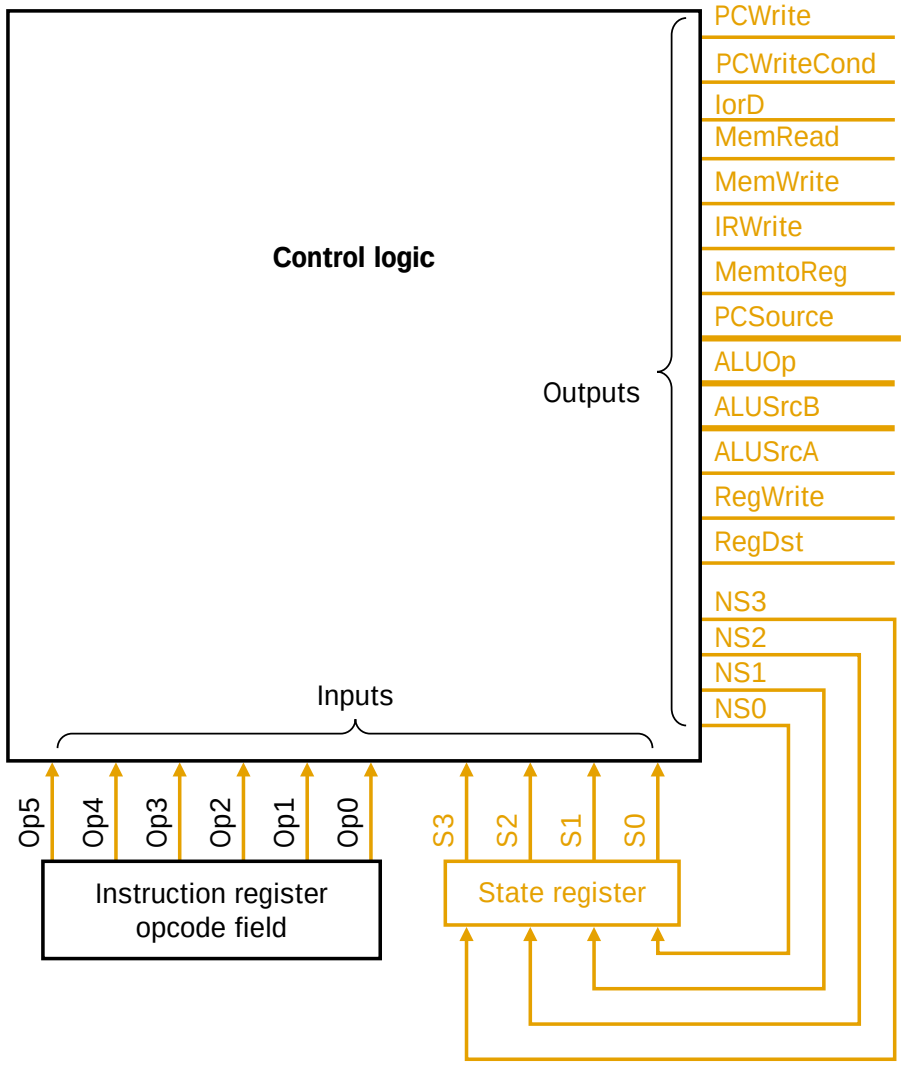
Controllo con macchina a stati finiti

Durante l'esecuzione di un programma applicativo i circuiti interpretano le istruzioni:
del programma costituito dal< programma applicativo ∘ i servizi OS>



- Automa con 10 stati ⇒ 4 bit per codificarli (S3-S0)
  memorizzati con 4 flip-flop di tipo D
- Codifica: assegnamento di un valore a ciascuno stato (useremo numero progressivo)



Dobbiamo realizzare due funzioni logiche combinatorie:

- funzione di uscita $\eta(\underline{s})$
  restituisce 16 segnali di uscita
- funzione di stato futuro $f(\underline{s}, \underline{I})$
  restituisce NS3…NS0

dove $\underline{s} = (S3, S2, S1, S0)$

$\underline{I} = (OP5, \ldots, OP0)$

Tabelle di verità per le **uscite di controllo** (dipendono solo dallo stato):
per semplificarle vengono indicati solo gli stati in cui singole uscite = 1.

PCSource (PCSource1 – PCSource0)

Dal diagramma di stato, si vede che:
- PCSource1 vale 1 nello stato 9
- PCSource0 vale 1 nello stato 8

S3	S2	S1	S0	PCSource1
1	0	0	1	1

S3	S2	S1	S0	PCSource0
1	0	0	0	1

⇒ Equazioni logiche: $PCSource1 = S3\ \overline{S2}\ \overline{S1}\ S0$

$PCSource0 = S3\ \overline{S2}\ \overline{S1}\ \overline{S0}$

IorD

Dal diagramma di stato, si vede che
vale 1 negli stati 3 e 5

⇒ $IorD = \overline{S3}\ \overline{S2}\ S1\ S0 + \overline{S3}\ S2\ \overline{S1}\ S0$

S3	S2	S1	S0	IorD
0	0	1	1	1
0	1	0	1	1

Altre uscite ricavate con considerazioni analoghe…

NB: campi Op don't care (1 riga sta per $2^6 = 64$ righe!)

s3	s2	s1	s0
0	0	0	0
1	0	0	1

a. Tabella di verità per PCWrite

s3	s2	s1	s0
1	0	0	0

b. Tabella di verità per PCWriteCond

s3	s2	s1	s0
0	0	1	1
0	1	0	1

c. Tabella di verità per IorD

s3	s2	s1	s0
0	0	0	0
0	0	1	1

d. Tabella di verità per MemRead

s3	s2	s1	s0
0	1	0	1

e. Tabella di verità per MemWrite

s3	s2	s1	s0
0	0	0	0

f. Tabella di verità per IRWrite

s3	s2	s1	s0
0	1	0	0

g. Tabella di verità per MemtoReg

s3	s2	s1	s0
1	0	0	1

h. Tabella di verità per PCSource1

s3	s2	s1	s0
1	0	0	0

i. Tabella di verità per PCSource0

s3	s2	s1	s0
0	1	1	0

j. Tabella di verità per ALUOp1

s3	s2	s1	s0
1	0	0	0

k. Tabella di verità per ALUOp0

s3	s2	s1	s0
0	0	0	1
0	0	1	0

l. Tabella di verità per ALUSrcB1

s3	s2	s1	s0
0	0	0	0
0	0	0	1

m. Tabella di verità per ALUSrcB0

s3	s2	s1	s0
0	0	1	0
0	1	1	0
1	0	0	0

n. Tabella di verità per ALUSrcA

s3	s2	s1	s0
0	1	0	0
0	1	1	1

o. Tabella di verità per RegWrite

s3	s2	s1	s0
0	0	0	0

p. Tabella di verità per RegDst

Tabelle di verità per le **uscite di stato futuro** (dipendono da stato e campi Op)

NS0: pari a 1 quando stato futuro = 1, 3, 5, 7, 9

Dall'analisi del diagramma di stato, si vede che ad essi si può arrivare:

stato 1 : da stato 0 a prescindere da Op
stato 3 : da stato 2 con Op = "lw" [100011]
stato 5: da stato 2 con Op = "sw" [101011]
stato 7: da stato 6 a prescindere da Op
stato 9: da stato 1 con Op = "j" [000010]

Ciò corrisponde alla tabella di verità:

OP5	OP4	OP3	OP2	OP1	OP0	S3	S2	S1	S0	NS0
X	X	X	X	X	X	0	0	0	0	1
1	0	0	0	1	1	0	0	1	0	1
1	0	1	0	1	1	0	0	1	0	1
X	X	X	X	X	X	0	1	1	0	1
0	0	0	0	1	0	0	0	0	1	1

$$NS0 = \overline{S3}\ \overline{S2}\ \overline{S1}\ \overline{S0} + OP5\ \overline{OP4}\ \overline{OP2}\ OP1\ OP0\ \overline{S3}\ \overline{S2}\ S1\ \overline{S0} + \overline{S3}\ S2\ S1\ \overline{S0} + \overline{OP5}\ \overline{OP4}\ \overline{OP3}\ \overline{OP2}\ OP1\ \overline{OP0}\ \overline{S3}\ \overline{S2}\ \overline{S1}\ S0$$

Con considerazioni analoghe si ricavano le altre uscite di stato futuro

⇒ Abbiamo la specifica di entrambe le funzioni combinatorie

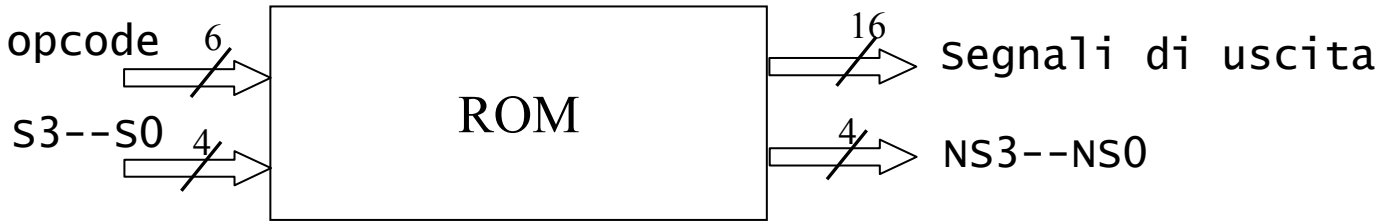
Realizzazione in 2 (*2) modi alternativi:

1) Tramite ROM
   - unica ROM per entrambe le funzioni
   - due ROM separate

2) Tramite PLA
   - unica PLA
   - due PLA separate

Confrontiamo le varie soluzioni

Unica ROM



Dimensione: $\mathbf{2^{10} * 20 = 20\ Kbit}$

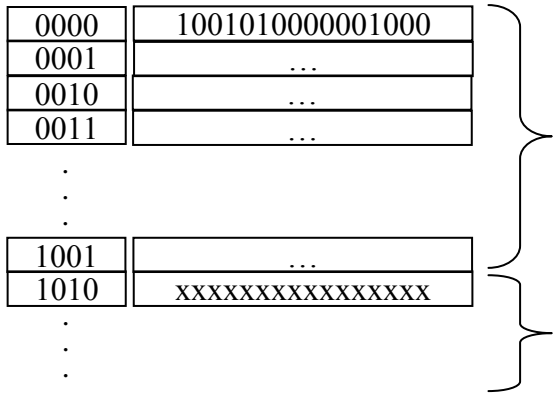
Parola di uscita

Combinazioni possibili dei 10 valori di ingresso

• Definendo un ordine dei bit di ingresso e di uscita, è possibile specificare il contenuto della ROM:

- 10 bit di ingresso (stato + opcode) → 4 bit di stato futuro

- 4 bit indirizzo (stato) → 16 bit di controllo

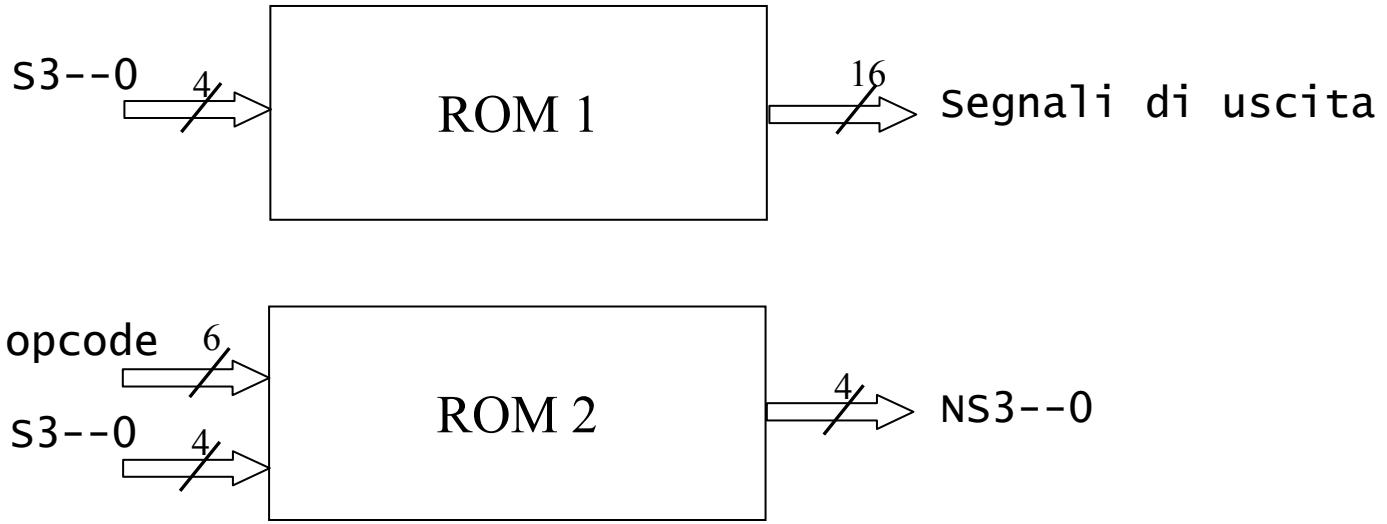


10 parole ripetute 2^6 volte
(6 opcode don't care)

6 parole (non corrispondenti a stato) don't care anch'esse ripetute 2^6 volte

Possiamo evitare lo spreco dovuto alle ripetizioni per 2^6 usando due ROM separate (stato futuro e uscite controllo):
la ROM per le uscite di controllo ha solo 4 ingressi (bit di stato) e quindi evita di ripetere la stessa parola di memoria per tutte le 2^6 combinazioni degli ingressi!

Due ROM separate



Dimensione: **2^4 * 16 + 2^{10} * 4 = 4,3 Kbit**

Nelle ROM ci sono comunque molti elementi duplicati:

- molte combinazioni dell'ingresso non si verificano (ROM le codifica tutte)
  p.es. ho 4 bit per 10 stati: 16-10=6 combinazioni non si verificano
- a volte le uscite non dipendono da [tutti] i valori dell'ingresso.
  p.es. da stato 0 si passa sempre a stato 1 per qualunque valore di opcode, mentre ROM 2 duplica la parola NS1 (0001) per 2^6 volte

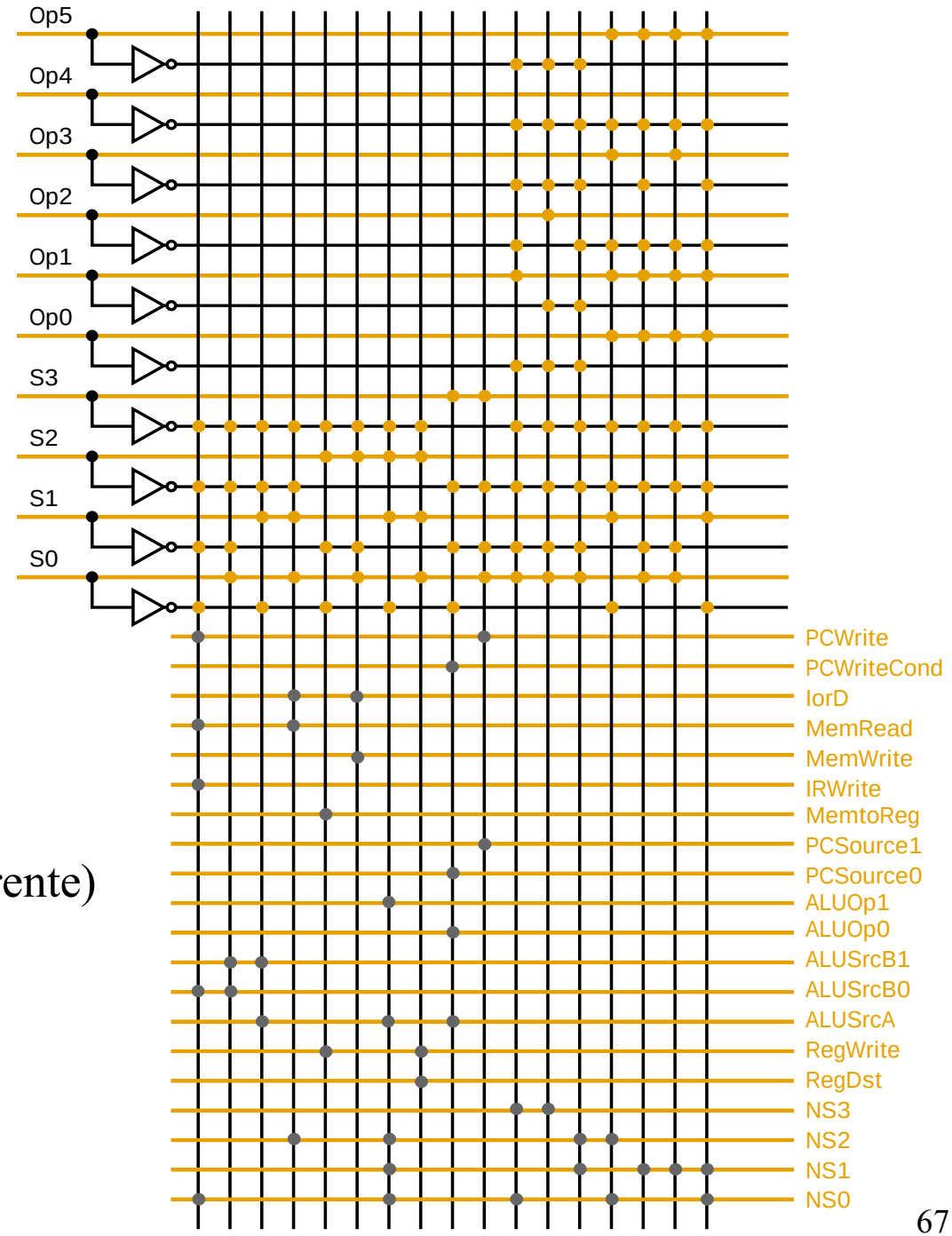
Unico PLA

Ciascuna uscita è OR di termini prodotto (mintermini)

- Rappresentate solo le righe della tabella di verità con uscita 1
- Ciascun mintermine può rappresentare più righe (input don't care)

In questo caso: 17 mintermini
(10 dipendono solo da stato corrente)

Dimensione: proporzionale a
10*17 + 17*20 = 510



Due PLA

1) PLA a 4 ingressi (stato corrente) e 16 uscite di controllo

10 mintermini (quelli che dipendevano solo da stato corrente, usati dalle uscite di controllo)

Dimensione: 4*10 + 10*16 = 200

2) PLA a 10 ingressi (stato e opcode) e 4 uscite di stato futuro

10 mintermini (cfr. figura del PLA: sono quelli usati da uscite di stato futuro, dei quali 7 dipendono da stato e opcode)

Dimensione: 10*10 + 10*4 = 140

⇒ Dimensione totale: proporzionale a

200 + 140 = 340

Riepilogo

	1	2	Riduzione a
ROM	20 Kbit	4.3 Kbit	1/5
PLA	510	340	3/5
Riduzione a	1/40	1/13	

FINORA ABBIAMO VISTO:

- Processore a singolo ciclo [Unità di controllo combinatoria]
  - specifica e implementazione: cfr. reti combinatorie
- Processore multi-ciclo [Unità di controllo sequenziale]
  - **specifica come FSM** e **implementazione come FSM**

    ↓

    Parti combinatorie:
    - specifica con tab. di verità | equazioni logiche
    - realizzazione con ROM |PLA (singole o doppie)

  - ORA VEDIAMO LA **SPECIFICA COME microprogramma**