NUMERO: **2530A** ANNO: 2022

A P P U N T I

STUDENTE: Carosella Angelica

MATERIA: Calcolatori Elettronici - Prof. Sonza Reorda

CALCOLATORI ELETTRONICI

1. INTRODUZIONE

Sistema di elaborazione: (calcolatore) sistema che riceve input, elabora dati (operazioni logiche ed aritmetiche) ed invia informazioni in output. Spesso sono sistemi a
processore. Un calcolatore deve rappresentare informazioni, risolvere problemi
mediante algoritmi (operazioni automatizzate) e memorizzare dati.

I. **ERA MECCANICA** (~ 1945)

- Macchina di Pascal (‘600: +, -, ruota dentata)
- Macchina di Leibniz (evoluzione, più ruote dentate: +, -, x, /)
- Telaio di Jacquard (primo esempio di programmabile, input su schede perforate)
- Macchine di Babbage (navigazione, un solo algoritmo, output: incisione su piatto di rame)
- Z1, Z2, Z3 e Z4 di Zuse (primo programmabile funzionante, relè elettromagnetici, rappresentazione binaria di informazioni)
- Mark (I e II) di Aiken (nastri di carta perforata)

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 3 di 114

- **3ª generazione** (‘65)

  Circuiti a transistor sostituiti dai circuiti integrati, memorie a semiconduttore, microprogrammazione, architetture a pipeline.

III. **ERA VLSI** (1975 ~)

Very Large Scale of Integration: possibilità di integrare migliaia di transistor in un solo chip, inserito in un package su un die con dei pin.

Pin: terminazioni metalliche che stabiliscono contatti elettrici.
Die: sottile piastrina di silicio su cui viene realizzato il chip.
Package: contenitore in cui viene inserito il circuito integrato.

Circuito integrato: chip assemblato su un die e connesso mediante dei pin, inserito in un package.

Legge di Moore: il numero di transistor che possono essere integrati in un circuito raddoppia ogni 18/24 mesi (migliorano tecnologia e architettura): siamo arrivati ad alcuni miliardi.

ARCHITETTURE DEI PROCESSORI

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 5 di 114

- **SUPERSCALARI** (anni ‘90)
  Set di istruzioni di tipo RISC, più istruzioni per colpo di clock, architettura a pipeline e unità funzionali multiple.

- **MULTICORE** (anni 2000)
  Più processori (core) integrati su un unico dispositivo.

II. **Special purpose (embedded)**

Microcontrollore: dispositivo integrato su un unico circuito integrato, composto da processore, memorie (RAM, ROM, flash), porte e interfacce per I.O. e contatori.

ASIC (Application Specific IC): circuito integrato realizzato per eseguire un’applicazione specifica. Se contiene uno o più processori è un system on chip.

System On Chip: dispositivo specifico integrato su un unico chip, composto da uno o più processori / microcontrollori, memorie, interfacce e moduli specifici.
Corrisponde a un intero sistema di elaborazione e svolge una specifica applicazione.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 7 di 114

linee per trasferire dati tra di loro (prodotto dal progettista).

- Comportamento : funzione di trasformazione dai dati input ai dati output (ricevuto come specifica dal progettista).

Progetto: individuare una connessione tra i componenti in modo da ottenere il comportamento desiderato e rispettare le specifiche (velocità, durata…).

Ciclo di vita: specifica (documento che descrive comportamento e vincoli del sistema), progetto (solitamente top-down, produzione di un modello utilizzabile e validazione mediante simulazioni), produzione e operatività.

Un progetto può avvenire a diversi **livelli** (ad ogni livello viene iterato lo stesso flusso: specifiche, sintesi, verifica e realizzazione) di astrazione:

I. Elettrico (più basso e dettagliato)
II. Transistor
III. Porte logiche (circuiti combinatori o sequenziali)
IV. Registri
V. Sistema

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 9 di 114

solo 2 valori di tensione, quindi 2 valori logici (0 per V basse e 1 per V alte).

- AND : 1 se entrambi 1
- NAND : 0 se entrambi 1
- OR : 1 se almeno uno 1
- NOR : 1 se nessuno 1
- EXOR : 1 se alterni
- EXNOR : 1 se uguali
- NOT : 1 se 0

Insieme di porte completo: insieme le cui porte possono realizzare qualsiasi funzione combinatoria.

Un **sistema** può essere:

- Combinatorio (porte logiche, ingressi attuali)
- Sequenziale (flip-flop, ingressi attuali + istanti precedente)

1. CIRCUITI COMBINATORI

Sistemi che non coinvolgono il tempo, descritti con tavole di verità (**i** ingressi, righe= $\mathbf{2^i}$) e funzioni booleane, e costituiti da porte logiche. I valori delle uscite dipendono solo dagli ingressi in quell'istante.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 11 di 114

- $a + a = a$
- $a \cdot a = a$
- $\overline{a} + \overline{b} = \overline{a \cdot b}$
- $a + b = a \cdot b$

Cicuiti a 2 livelli (soluzione per progettare circuiti minimi): espressione booleana sottoforma di somma di prodotti (uscite di diverse porte AND pilotate da una sola porta OR) oppure prodotto di somme (uscite di diverse porte OR pilotate da una sola porta AND).

Terminologia:

- Letterale (x affermata o negata)
- Minterm (AND di tutte le variabili, affermata o negata)
- Maxterm (OR di tutte le variabili, affermata o negata)
- Cubo (non necessariamente tutte le variabili)

Un circuito è **minimo** se il numero di porte è minimo **(**la profondità è minore di un certo valore, il $FANIN_{max}$ è minore di un certo valore e il $FANIN_{min}$ è maggiore di un certo valore**)** e

- Righe e colonne sono sempre così
- Uno 0 è una variabile negata, un 1 è una variabile affermata
- I valori don’t care possono valere 0 o 1 a seconda della necessità

Ritardo porta: tempo massimo dopo cui si può garantire che le uscite abbiano assunto il valore esatto ($K = t_0 + ð$), dipende dalla porta e dalla tecnologia. K è proporzionale alla profondità del circuito.

Cammino critico: cammino lungo il quale il ritardo con cui si propaga una variazione in ingresso è massimo oppure che passa per il maggior numero di porte logiche.

La lunghezza (somma dei ritardi delle porte lungo il cammino) determina:

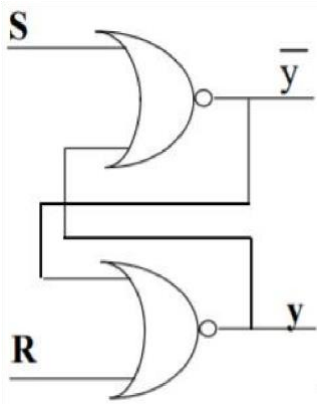
- tempo minimo da aspettere affinchè vengano prodotte le uscite
- frequenza massima con cui si possono fornire gli ingressi al circuito

Livello di una porta:

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 15 di 114

i. Flip-flop SR asincrono
ii. Flip-flop SR sincrono
iii. Flip-flop D
iv. Flip-flop Master Slave

i. Flip-flop SR asincrono

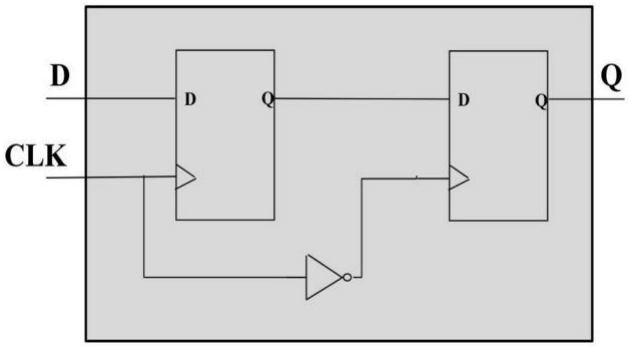


Costituito da 2 porte NOR (una delle possibili implementazioni), 2 ingressi SET e RESET, 2 uscite y e ~~y~~.

Sono possibili 4 configurazioni:

- S=R=0 → y mantiene il valore
- S=0, R=1 → y è forzato a 0
- S=1, R=0 → y è forzato a 1
- S=R=1 → configurazione vietata: dipende dal ritardo delle porte (1 se $\Delta_1 > \Delta_2$, 0 se $\Delta_1 < \Delta_2$).

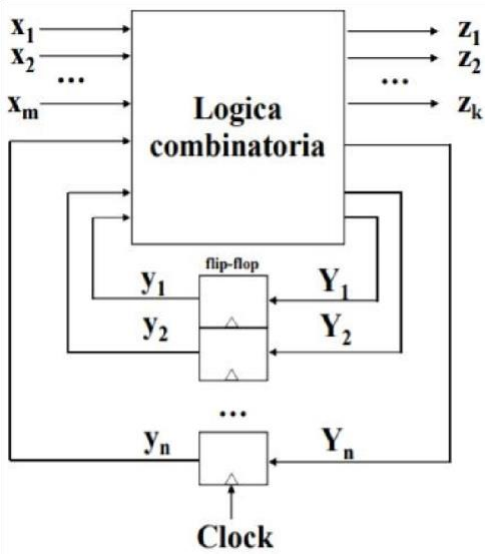
iv. **Flip-flop Master Slave**



Costituito da 2 flip-flop D in cascata + segnale di clock + porta NOT. Il primo flip-flop (detto master) riceve il clock affermato, il secondo slave riceve il clock negato.

- Clock = 0 → master bloccato, slave conserva lo stato di memoria del master
- Clock = 1 → slave bloccato, master memorizza D

MODELLO DI HUFFMAN



Il modello più generale di circuito sequenziale è il Modello di Huffman (sincrono), che è una FSM di Mealy.
La logica combinatoria riceve gli ingressi x e i valori

IV. **LIVELLO DI REGISTRI**

Al livello dei registri, l'unità di dato manipolata è la parola (gruppo di bit di una certa dimensione): i componenti utilizzati possono essere combinatori (porte logiche) o sequenziali (porte logiche + flip-flop).

Componenti combinatori:

- Porte logiche operanti su parole
- Multiplexer
- Decodificatori
- Codificatori
- Moduli aritmetici
  - sommatori
    - i. combinatorio
      - i. combinatorio modulare (ripple carry adder / carry-lookahead) ii. seriale
  - ALU
  - comparatori

Componenti sequenziali:

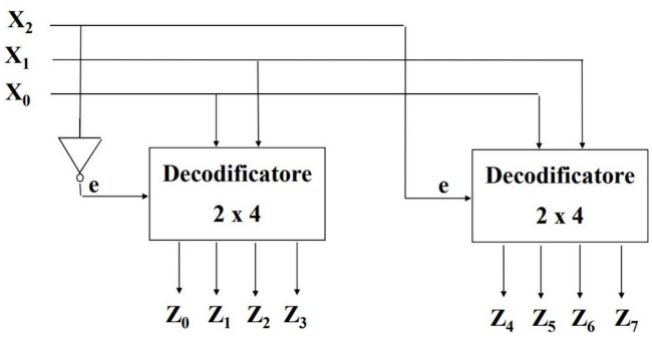
- Registri
- Contatori

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 21 di 114

- **S** → negato sulla porta di sinistra, affermato sulla porta di destra
- **S = O** → passa l'ingresso di sinistra (nel MUX)
- **S = 1** → passa l'ingresso di destra (nel MUX)

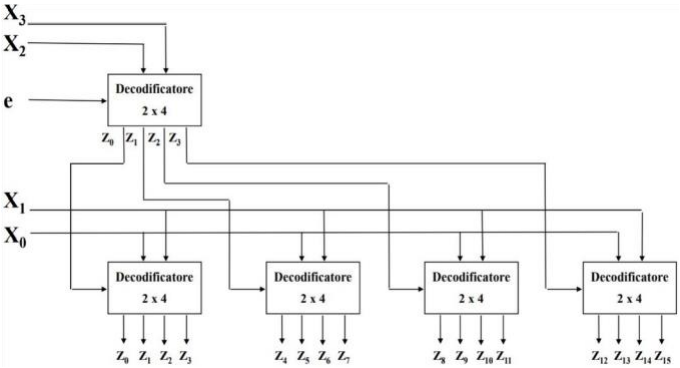
La tavola di verità di un MUX 2 x 1 ha **8 righe** (2^3, 3 ingressi: S, X_0, X_1).

- MUX con m bit in uscita → **m** MUX 2 x 1 in serie.
- MUX con k ingressi → **k–1** MUX 2 x 1 in cascata.
- Implementare **funzioni booleane** con un MUX è veloce ma poco efficiente: con **n** ingressi serve un MUX **2^n x 1** (quante sono le righe della tavola di verità del circuito: tutti i possibili valori restituiti dalla funzione) e **n** segnali di controllo (ingressi valutati dalla funzione).



E’ costituito da 2 decoder e una porta NOT: il segnale del bit più significativo X_2 arriva negato al 1° decoder e affermato al 2° decoder (funziona da **enable**). Per realizzare un decoder 3 x 8 con un proprio enable, lo si collega ad entrambi I codificatori attraverso 2 porte **AND**, a cui arriva anche il segnale X_2.

III. **Decoder 4 x 16** (5 decoder)



E’ costituito da 5 decoder e un enable: il 1° decoder fa da **enable** per i successivi 4 (ha in ingresso e, X_3, X_2), gli altri hanno in ingresso il segnale di “enable” e i bit meno significativi X_1 e X_0.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 25 di 114

quella attiva e il caso in cui nessuna linea è attiva), alcuni codificatori sono dotati di segnali di **enable** (se è inattivo tutte le uscite sono a 0) e di **input active** (va a 1 se sono attivi enable e almeno una linea di ingresso).

5. MODULI ARITMETICI

In base alla loro velocità e al tipo di dati e operazioni che supportano avranno una certa complessità.

I. SOMMATORI

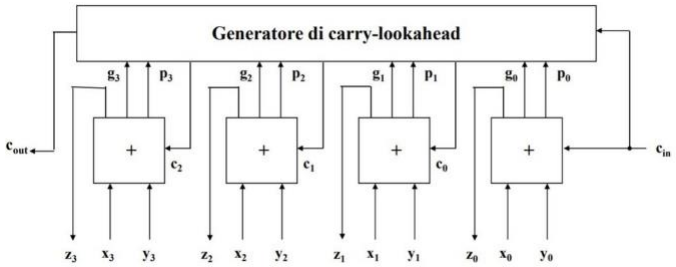
Moduli che eseguono la somma di 2 numeri su n bit.

i. Sommatori combinatori

Circuito a 2 livelli che permette di sommare **2** numeri (parole) di **n** bit e produrre **n+1** bit di uscita. Può essere sintetizzato da una tavola di verità o da una funzione booleana. E' la soluzione migliore in termini di velocità e costo hardware, ma il progetto va eseguito per ogni n.

adder è il carry in del successivo (C_0=0).
E' facile da progettare, ma ha un alto costo in termini di hardware e ritardi (**n · d**, d: ritardo di ogni modulo).

b) **Carry-lookahead**



A differenza del Ripple carry adder, permette di ridurre il ritardo **(3·d):** il carry di ogni modulo è generato in base a dei segnali (g e p) che arrivano in **parallelo** dai full-adder **(modificati).**

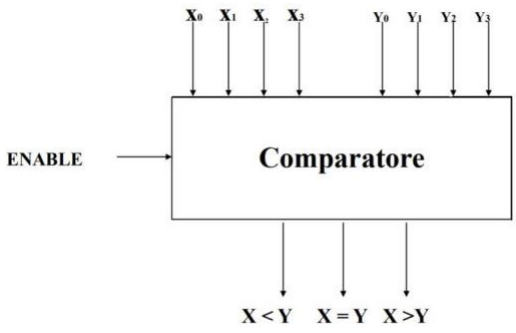
g_i = $x_i y_i$ (generazione) / **p_i** = $x_i + y_i$ (propagazione)
Ci = gi + pi ·gi-1 + pi ·pi-1 ·ci-2.

- Vengono applicati X e Y
- I full-adder producono g e p, inviati in parallelo al generatore
- Il generatore di carry-lookahead produce i carry

II. **ALU**

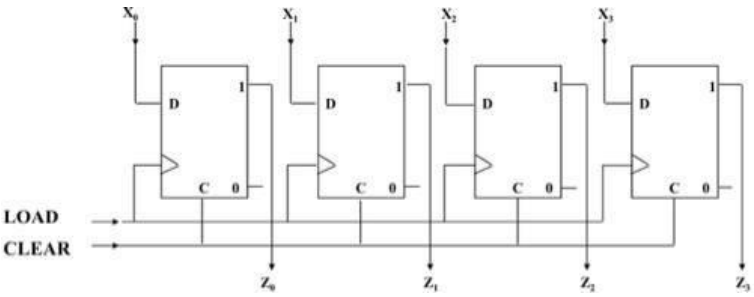
Le **Unità Aritmetico Logiche** (più usate nei processori rispetto ai sommatori) sono moduli che integrano in un unico blocco le principali funzioni aritmetiche (somma, sottrazione…) e logiche (AND, OR…). Sono costituiti da dei segnali di controllo (**m** = $\log_2 k$ bit, specificano il tipo di operazione da effettuare), C_{in} e C_{out} (che permettono di collegare più ALU in cascata), gli ingressi X e Y su **n** bit (parallelismo del processore) e l'uscita Z. Con **k** operazioni supportate, sono necessari $\log_2 k$ segnali di controllo. Il ritardo dipende solo dalle porte logiche interne.

III. **COMPARATORI**



Modulo che permette di confrontare due numeri su 4 bit (si confrontano bit a bit): è costituito da un segnale di **enable** (se è inattivo non può essere attivata nessuna

I. Registri a m bit



Modulo che permette di memorizzare parole di **m** bit: è costituito da m entrate ed m uscite da 1 bit, m flip-flop D e dei segnali load e clear. Quando **load** viene attivato, ciascun flip-flop memorizza 1 degli m bit in ingresso, che vengono mandati in uscita e restano costanti fino al successivo load. Il segnale di **clear** permette di azzerare tutti i flip-flop.

II. Registri a scalamento

Lo scalamento (a destra o a sinistra) può servire per eseguire moltiplicazioni / divisioni per potenze di due, per comunicazioni seriali o per memorizzare dati seriali FIFO (First In First Out, n shift register in parallelo).

7. CONTATORI

Sono moduli su **n** bit (modulo $\mathbf{2^n}$), con n uscite e dei segnali di count, enable e terminal count, il cui valore memorizzato all'interno viene incrementato (o decrementato) di 1 ogni volta che **count** è attivo. Il **terminal count** va a 1 quando il valore interno assume valore massimo (2^n-1).

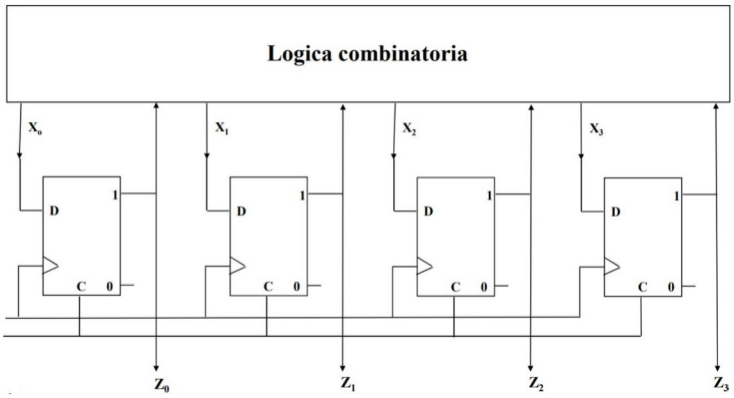
Per realizzare contatori di dimensioni maggiori, si collegano più contatori da 4 bit mediante terminal count → count (quando il primo finisce, si attiva il successivo). Sono usati per contare eventi, come program counter nei processori e come divisori di frequenza (divide per 2^n, genera clock più lenti da un clock).

Possono essere di diversi tipi:

- **up-down counter:** possono contare avanti e indietro
- **programmable counter:** il valore può essere modificato (si aggiunge un sommatore che riceve il valore corrente e il passo)

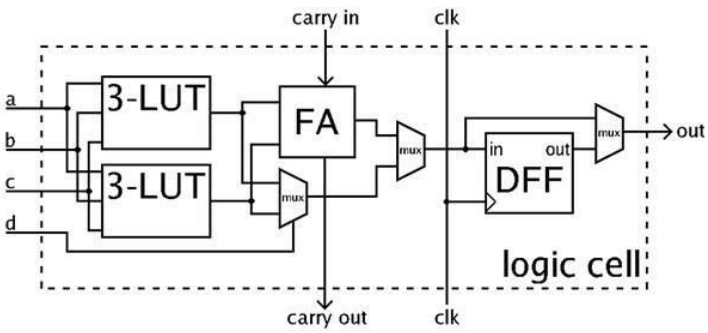
© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 35 di 114

II. **Contatore sincrono**



E' costituito da n FF, un segnale di count e una **logica combinatoria** (dato il valore corrente, elabora il prossimo valore dei FF una
voltà che verrà attivato count sommando o sottraendo 1). Il ritardo è minore (non dipende da n) ma ha un costo più elevato.

8. **FPGA**



Sono circuiti **programmabili** (costituiti da 1 FFD, 1 full adder e 2 LUT che posso decidere come collegare), alternativi agli ASIC, che permettono di implementare funzionalità abbastanza complesse, combinatorie o

9. BUS

Sono strutture circuitali di interconnessione tra più moduli (solo uno scrive, gli altri assumo un valore di alza impedenza, cambiano nel tempo): possono trovarsi in un **circuito**, su una scheda (tra più circuiti) o in un sistema (tra più schede). Tra il singolo modulo e il bus sono presenti delle interfacce.

BUFFER TRI-STATE

L'interfaccia tra il modulo presente sul circuito e il bus (su cui deve scrivere) è costituita da un buffer tri-state: un circuito costituito da un segnale di enable, un ingresso X su n bit e un'uscita Y su n bit.

- **e = 1** → Y = X (si comporta da circuito chiuso)
- **e = 0** → Y = Z (alta impedenza, si comporta come un circuito aperto)

Per assicurare che ci sia un solo segnale di enable attivo in ogni istante, si può utilizzare un **decoder** anteposto alle interfacce.

II. ROM

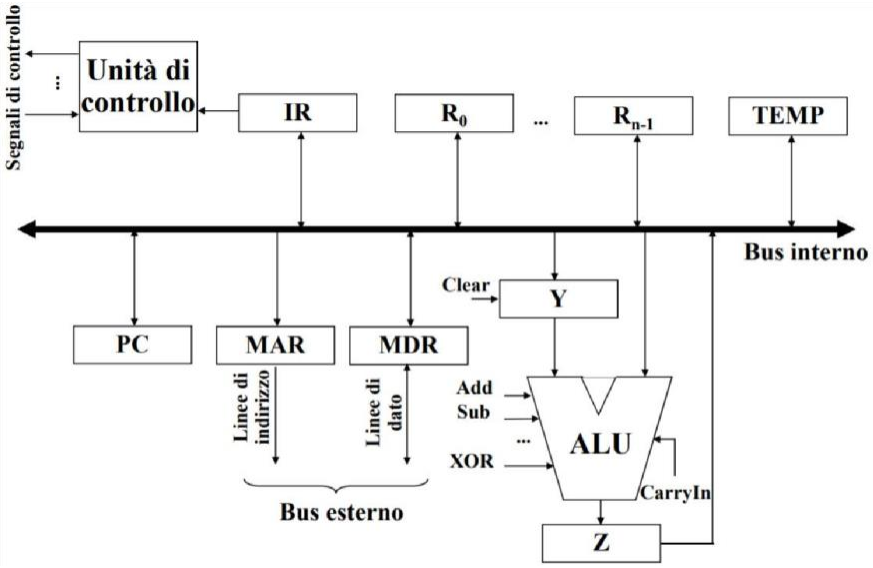
A differenza delle RAM, sono memorie in sola lettura: il loro contenuto viene determinato al momento della produzione.
Mancano read, write e $Data_{in}$.

III. Banchi di memorie

Insieme di moduli di memoria interconnessi, la cui dimensione è pari alla **somma** delle dimensioni dei singoli moduli.

- Stesso $\mathbf{2^k}$, diverso **n**: più moduli elementari in cascata
- Diverso $\mathbf{2^k}$, stesso **n**: **decoder** con bit aggiuntivi (più significativi) in ingresso, le uscite sono gli enable dei moduli (read, write, $data_{in}$ vanno in parallelo a tutti i moduli)
- Diverso $\mathbf{2^k}$, diverso **n**: decoder, le singole parole stanno in più moduli (serie)

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 41 di 114



- **MAR** (indirizzo): connesso ad Abus (CPU → memoria)
- **MDR** (dato)**:** connesso a Dbus (CPU ←→ memoria), lettura e scrittura

Le **operazioni** svolte dalla CPU (per ogni microistruzione dell'ISA) sono di 4 tipologie elementari:

i. Trasferimento di un dato tra registri ii. Prelievo di un dato (o istruzione) dalla memoria e caricamento in un registro iii. Scrittura di un dato in memoria da un registro iv. Operazione aritmetica o logica (ALU) e scrittura risultato in un registro

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 43 di 114

anche al bus esterno: **MAR** (scrive sull'Abus) e **MDR** (scrive e legge dal dBus). Alla fine di ogni istruzione di lettura o scrittura, una volta completata l'operazione, la memoria attiva il segnale MFC (Memory Function Complete). Il **prelievo** di una parola (load word) in memoria è costituito da **5** microistruzioni (MAR←R_1 - READ - MFC - MDR←dBus - R_2←MDR) precedute dalla fase di FETCH.

iii. Scrittura di un dato in memoria

La **scrittura** di una parola (store word) in memoria è costituita da **4** microistruzioni (MDR←R_1 – MAR←R_2 – WRITE - MFC) precedute dalla fase di FETCH.

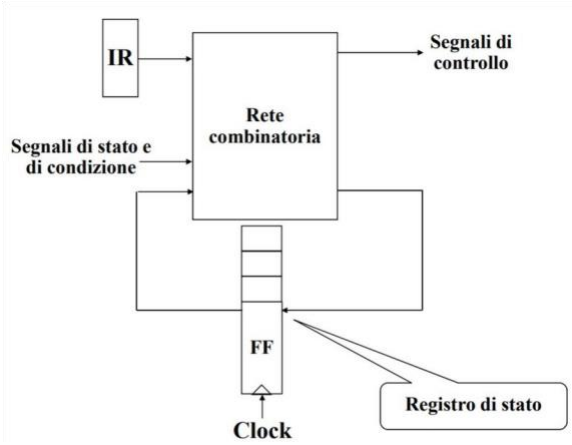
iv. Operazioni della ALU

L'esecuzione di un'**operazione** logica o aritmetica è costituita da **3** microistruzioni (Y←R_1 – Z←op – R2←Z) precedute dalla fase di FETCH.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 45 di 114

II. **Microprogrammate:** operazioni descritte da microistruzioni (contengono i valori dei segnali di controllo), memorizzate in una memoria apposita (CISC)

I. UC cablata



Le UC cablate sono progettate come **FSM**: lo stato del circuito è rappresentato dal valore presente ad ogni colpo di clock in un registro di stato (FF). Per ogni stato corrispondente a una combinazione di ingressi (segnali da esterno / unità di elaborazione e IR) vengono prodotte le uscite (segnali di controllo per l'unità di elaborazione) e lo stato futuro: viene costruita una tabella degli stati e implementata in hardware. E' **vantaggiosa** in termini di hardware, costo e velocità, **svantaggiosa** in termini di progetto e flessibilità (per modificare o aggiungere istruzioni è

- **Microprogrammazione orizzontale**

  E’ il caso più semplice: le microistruzioni contengono 1 bit per ogni segnale di controllo. La **velocità** è massima (i segnali sono pilotati senza dover essere manipolati), ma la **lunghezza** delle microistruzioni può essere eccessiva e alcune combinazioni possono non verificarsi mai.

- **Microprogrammazione verticale**

  I bit dei segnali di controllo vengono memorizzati codificati: sono suddivisi in **k** gruppi (classi di compatibilità) da $\log_2 n$ bit (**n** bit di controllo decodificati da ogni gruppo mediante un **decoder**, non per forza lo stesso n° di bit per gruppo). Una **classe di compatibilità** è un gruppo di segnali di controllo
  compatibili a 2 a 2 (solo 1 dei 2 viene attivato per periodo di clock). La **lunghezza** delle microistruzioni è ridotta, ma la **velocità** è minore.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 49 di 114