CPU
MEMORIE
    MEMORIA PRINCIPALE
    MEMORIA CACHE
    MEMORIA SECONDARIA E OFF-LINE
    MEMORIA VIRTUALE
Input/Output - I/O
    INTERRUPT
    DMA
BUS
    ARBITRAGGIO CENTRALIZZATO

CPU

Composta da:

- ***unità di elaborazione*** → *contiene registri, ALU, ecc.*
- ***unità di controllo*** → *genera segnali di controllo per l'unità di elaborazione*

L'**unità di elaborazione** è composta da:

- IR → *instruction register*
- R_i → *registri*
- TEMP → *registro per uso privato CPU*
- PC → *program counter*
- MAR → *memory address register*
- MDR → *memory data register*
- ALU → *unità aritmetico-logica*
- $BUS_{interno}$

Le **unità di controllo** possono essere di 2 tipi:

- ***cablate*** → *normale circuito sequenziale*
    - Vantaggi:
        - Minimizza area di silicio utilizzata
        - Elevata efficienza in termini di velocità delle istruzione
        - Minimizza il costo del circuito
    - Svantaggi:
        - La dimensione della MSF può essere troppo grande
        - Una volta che il circuito è stato sintetizzato fisicamente, la modifica è complessa.
- ***microprogrammate*** → *esegue microistruzioni e assegna valori prefissati ai segnali di controllo*
    - Vantaggi:
        - Modificare un'istruzione equivale a modificare il contenuto della memoria di microprogramma
    - Svantaggi:
        - L'esecuzione di un'istruzione richiede una serie di accessi alla memoria di microprogramma
        - La presenza della memoria di microprogramma e della relativa logica fa crescere il costo HW

MEMORIE

La memoria di un calcolatore è organizzata in livelli caratterizzati da diversi tipi di memoria in termini di velocità, dimensione e quindi costo.

Gerarchia di memoria *(tempo di accesso e dimensioni crescenti, costo decrescente)*:

- ***Memoria interna*** → *Registri (SRAM), cache*
- ***Memoria principale*** → *RAM (DRAM)*
- ***Memoria secondaria*** → *Cache disco, HDD, SSD*
- ***Memoria off-line*** → *Nastri magnetici, dischi ottici*

Alcuni parametri importanti per le memorie sono:

- ***costo***
- ***velocità***
- ***modi di accesso***
- ***alterabilità***
- ***durevolezza del contenuto***
- ***affidabilità***
- ***caratteristiche fisiche***
- ***consumo***

Per accedere alle memoria possono esserci diversi modi di accesso.
Sono principalmente:

- ***sequenziale*** → *informazioni lette/scritte solo in un ordine prefissato (es. nastri)*
- ***diretto*** → *ogni blocco ha un indirizzo e la ricerca nel blocco è sequenziale (es. HDD)*
- ***casuale*** → *ogni dato ha un indirizzo, informazioni lette/scritte in qualsiasi ordine (es. RAM, ROM)*
- ***associativo*** → *confronto tra contenuto cella e quello specificato in una maschera (es. alcuni tipi di cache)*

La velocità di una memoria è espressa attraverso 3 parametri:

- ***Tempo di accesso*** → *tempo tra la ricezione della richiesta (lettura/scrittura) e quello in cui la richiesta è eseguita*
- ***Tempo di ciclo*** → *tempo tra l'inizio di un ciclo di accesso alla memoria e l'inizio del ciclo successivo*
- ***Tasso di trasferimento*** → *velocità con la quale i dati possono essere trasferiti da/verso la memoria (bit/s, o bps)*

MEMORIA PRINCIPALE

Caratteristiche:

- *ogni cella può essere indirizzata indipendentemente*
- *tempi di accessi uguali e costanti per ogni cella*

Sono presenti diversi segnali di controllo, che permettono alla memoria di sapere il tipo di operazione richiesta, la disponibilità degli indirizzi sull'Abus, la disponibilità dei dati sul Dbus, ecc. Esempi di segnali di controllo sono:

- ***Chip Select (CS)*** *→ per poter leggere o scrivere*
- ***Output Enable (OE)*** *→ per poter abilitare l'uscita del chip su un bus condiviso (nel caso di più chip ad un singolo bus)*
- ***Write Enable (WE)*** *→ per poter effettuare una operazione di scrittura*

Sono inoltre presenti altri tipi di segnali come i segnali di stato:

- ***Errore*** *→ La memoria ha individuato una parola con un dato errato*
- ***MFC (Memory Function Complete)*** *→ La memoria ha completato l'operazione di lettura/scrittura*

Il **costo di una RAM** dipende soprattutto dalla complessità dei dispositivi di accesso, che può essere ridotta tramite un'opportuna organizzazione delle celle di memoria.
Ci sono 2 tipologie principali:

- ***Organizzazione a vettore***
- ***Organizzazione a matrice bidimensionale***
    - Vantaggi:
        - Minor costo HW
        - Layout più compatto

Nel caso di RAM con organizzazione a matrice bidimensionale, esistono 2 segnali particolari:

- ***RAS (Row Address Strobe)*** *→ indica che l'indirizzo inviato corrisponde al numero di riga*
- ***CAS (Column Address Strobe)*** *→ indica che l'indirizzo inviato corrisponde al numero di colonna*

Alcuni altri tipi di memoria principale, oltre alla RAM, sono:

- ***ROM (Read Only Memory)*** *→ utilizzata principalmente per programmi di sistema, contenuto definito prima della realizzazione fisica*
- ***PROM (Programmable Read Only Memory)*** *→ scrittura eseguita dopo la produzione per mezzo di attrezzature definite "programmatori"*
- ***EPROM (Electrically Programmable Read Only Memory)*** *→ possono essere riscritte un numero arbitrario di volte previa cancellazione del contenuto mediante esposizione prolungata a luce ultravioletta*
- ***EEPROM (Electrically Erasable Programmable Read Only Memory)*** *→ possono essere riprogrammate byte per byte*
- ***Flash*** *→ usano un solo transistor per bit.*
    - **Tipo NOR** *→ stesso comportamento di una RAM ma più lente, resistono per un numero limitato di cancellazioni/riscritture*
    - **Tipo NAND** *→ possono essere lette/scritte solo a blocchi, sono più veloci e più resistenti*

Quadro di sintesi:

Tipo	Categoria	Cancellazione	Scrittura	Volatilità
RAM	r/w	elettricamente	elettricamente	si
ROM	r-only	impossibile	in fase di prod.	no
PROM	r-only	impossibile	elettricamente	no
EPROM	r-mostly	luce UV	elettricamente	no
EEPROM	r-mostly	elettricamente	elettricamente	no
Flash	r-mostly	elettricamente	elettricamente	no

Esistono 2 tipi principali di RAM:

Tipo RAM	Descrizione
SRAM (Static)	La singola cella corrisponde ad un flip flop
DRAM (Dynamic)	La singola cella corrisponde a un condensatore e a un transistor. L'informazione è memorizzata sotto forma di carica del condensatore. Richiedono un rinfresco periodico dell'informazione (lettura fittizia). La lettura è di tipo distruttivo (Destructive Read-Out).

Per aumentare l'affidabilità delle memorie dinamiche sono stati adottati dei metodi noti come codici di protezione, i più comuni sono:

- ***Codice di parità*** → *viene calcolato e aggiunto un bit che indica lo "stato iniziale" della parola senza errori, il cosiddetto bit di parità*
    - Può essere di 2 tipi:
        - **bit di parità dispari** → *se nella parola il numero di bit posti ad "1" è pari*
        - **bit di parità pari** → *se nella parola il numero di bit posti ad "1" è dispari*
- ***Codici di Hamming*** → *permettono anche di correggere eventuali errori verificatisi in memoria. In paricolare:*
    - Permettono di rilevare e correggere tutti gli errori singoli
    - Permettono di rilevare (ma non correggere) tutti gli errori doppi

---

MEMORIA CACHE

Normalmente collocata tra CPU e bus, la memoria cache permette così un alleggerimento del bus, e permette una migliore compatibilità con le architetture multiprocessore.

Il principio di funzionamento delle memorie di tipo cache è definito dalla località dei riferimenti, che può essere:

- ***località temporale*** → *è molto probabile che un programma faccia nuovamente accesso ad una cella a cui aveva già fatto accesso poco prima*
- ***località spaziale*** → *è molto probabile che un programma faccia accesso ad una cella vicina ad un'altra a cui aveva fatto recentemente accesso*

In particolare, se ad un certo istante viene portato in cache un intero blocco di memoria a cui un programma sta accedendo, è molto probabile che per un certo lasso di tempo, tutte le parole di memoria a cui dovrà fare successivamente accesso, apparterranno a quel blocco.

Considerando le memorie cache, in termini di prestazioni, vengono definite 3 principali grandezze:

- ***h*** → *hit ratio della cache*
- ***C*** → *tempo di accesso alla cache*
- ***M*** → *tempo di accesso in memoria quando il dato non è in cache*

Il tempo medio di accesso in memoria per la CPU, sarà dato dalla seguente formula:

$$t_{medio} = hC + (1 - h)M$$

Una cache è comporta da un certo numero di *linee*, ad ognuna delle quali viene associato un campo identificativo *tag*.
La struttura è simile ad una tabella.

Supponiamo di avere una RAM di 2kB, una cache a 4 linee ognuna da 32B (ogni blocco nella RAM è quindi 32B).
Per rappresentare tutti gli indirizzi della RAM sono necessari 11bit.
Ogni parola in un blocco può essere rappresentata da 5bit ($32 = 2^5$).
I rimanenti 6bit rappresentano il numero del blocco.
Supponiamo di voler prendere la parola all'indirizzo di memoria ***00011100010***; per calcolare in che linea di cache andrà a finire il dato, basta dividere il numero di blocco per il numero di linee, il resto di questa operazione sarà il numero di linea cercato.
Il numero di blocco, nel nostro caso è ***000111*** (= 7), dividendolo per le linee di cache (4) otteniamo un resto di 3.

NUM. LINEA	TAG (6 bit)	LINEA (32B)
0		
1		
2		
3	***000111***	Qui andrà a finire tutto il blocco 000111. In particolare, verrà caricata la parola che cercavamo, cioè quella all'indirizzo (interno al blocco) ***00010***

In generale, per definire in quale linea di cache vada potenzialmente posizionato un certo blocco, esistono diverse funzioni di traduzione (mapping):

Tipo	Caratteristiche
direct mapping	Ogni blocco **i** della memoria principale è posizionato in corrispondenza fissa della linea **k** della cache, secondo la formula: $k = i \% N$.
associative mapping	Ogni blocco della memoria principale può essere memorizzato in un qualsiasi blocco della cache.
set associative mapping	Le linee della cache sono suddivide in **S** insiemi. Un blocco **i** è associato all'insieme **k** se: $k = i \% N$. Il blocco **i** può essere posizionato in una qualunque delle linee dell'insieme **k**.

Esistono anche diversi algoritmi per capire quale linea della cache dovrà essere rimpiazzata con un nuovo dato in entrata. Vengono chiamati algoritmi di rimpiazzamento e sono:

- ***LRU (Least Recently Used)*** → *il più utilizzato, rimpiazza la linea di cache meno utilizzata recentemente*
- ***FIFO (First-In First-Out)*** → *il più economico, rimpiazza la linea di cache che è stata inserita per prima*
- ***LFU (Least FrequentlyUsed)*** → *il più efficace, rimpiazza la linea di cache meno utilizzata*
- ***random*** → *semplice ed efficiente, rimpiazza una linea scelta casualmente*

Normalmente quando un dato in cache viene modificato dalla CPU, possono verificarsi delle incongruenze tra cache e memoria principale. Per risolvere queste incongruenze, esistono 2 principali meccanisci di aggiornamento della memoria principale:

- ***write-back*** → *funziona grazie all'aggiunta del cosiddetto **dirty bit**, cioè un flag che ricorda se il blocco è stato modificato da quando è stato caricato in cache. Quando il blocco viene rimosso dalla cache, se il **dirty bit** è settato a 1, allora si effettua l'aggiornamento della memoria con il contenuto attuale del blocco in cache*
  - Svantaggi:
    - *sostituzione lenta*
    - *se l'architettura è multiprocessore, potrebbero verificarsi delle inconsistenze tra le cache dei diversi processori*
    - *se si verifica un system failure prima che la copia viene rimossa dalla cache (e quindi copiata stabilmente in memoria), si possono perdere i dati modificati fino a quel momento*
- ***write-through*** → *ogni volta che la CPU effettua un'operazione di scrittura, quest'ultima la si effettua sia in cache che in memoria principale*

In aggiunta a questi meccanismi, nei sistemi multiprocessore, si aggiungono altri controlli per garantire la coerenza della cache:

- ***Bus Watching con Write-through*** → *il controllore di ciasciuna cache rileva le modifiche dal bus e invalida (attraverso un bit di validità) le linee corrispondenti nella propria cache*
- ***Non-cacheable Memory*** → *la memoria condivisa non può essere trasferita in cache*

Per garantire maggiore efficienza, può essere conveniente utilizzare più livelli di cache:

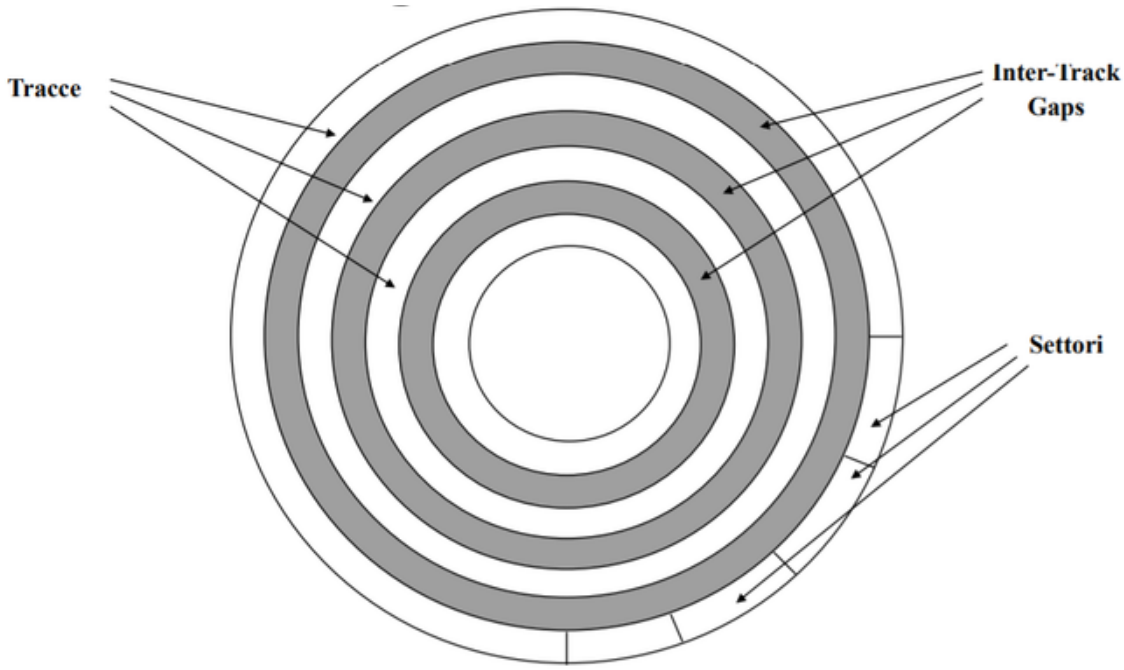
- ***cache di primo livello (C1)*** → *piccola e veloce*
- ***cache di secondolivello (C2)*** → *più grande ma più lenta*
- ***cache di terzo livello (C3)*** → *ancora più grande ma ancora più lenta*

MEMORIA SECONDARIA E OFF-LINE

Sono memorie caratterizzate da ***non volatilità***, ***basso costo*** ed ***elevato tempo di accesso***.

Esistono diversi tipi di memoria secondaria:

- ***Memorie a disco magnetico (HDD)*** → *si tratta di dischi ricoperti di materiale magnetico su cui esistono una serie di tracce concentriche separate da zone di gap, ogni traccia è suddivisa in settori, i quali vengono considerati come un unico blocco nelle fasi di lettura e scrittura. In queste due fasi, la testina si posiziona alla giusta distanza dal centro (sulla traccia desiderata) e il disco inizia a ruotare fino a quando il settore cercato non giunge sotto la testina, che leggerà/scriverà il contenuto*



- ***Unità a stato solido (SSD)*** → *si tratta di unità realizzate con tecnologie elettroniche, che non hanno parti in movimento e permettono una lettura random*
  - Vantaggi (rispetto agli HDD):
    - *più resistenti agli shock fisici*
    - *minore tempo di accesso*
    - *minore latenza*
    - *minore consumo*
  - Svantaggi:
    - *più costose*

Per quanto riguarda invece le memorie off-line:

- ***Memorie a nastro magnetico*** → *supporto di plastica flessibile su cui vengono memorizzate un certo numero di tracce, leggibili/scrivibili contemporaneamente. I dati sono organizzati in record, separati fra loro da apposite zone di gap*
- ***Cartucce*** → *supporti rimovibili collegati con il sistema mediante appositi connettori (vedi cartucce gioco Nintendo DS)*

- ***Memorie ottiche*** → *diffuse sotto forma di dischi ottici, hanno elevati tempi di accesso ma al contempo un'elevata affidabilità*
    - Tipologie di dischi ottici:

Tipo	Descrizione
CD	Non cancellabili, memorizzano informazioni audio.
CD-ROM	Non cancellabili, memorizzano informazioni in forma elettronica. I dati sono organizzati su una sola traccia a spirale, sotto forma di fosse (pits) e piazzole (lands) che riflettono in maniera diversa il raggio laser utilizzato per percorrere le tracce.
CD-RW	Riscrivibili un numero arbitrario di volte. Funzionano grazie ad una combinazione di argento, indio, antimonio e tellurio.
DVD	Utilizzati principalmente per la memorizzazione di contenuti video. Esistono diversi tipi di DVD, come i DVD-ROM, DVD-Video, DVD-Audio, DVD-R, ecc.
Blue-Ray Disk (BD)	Sono il formato utilizzato per memorizzare dati e video ad alta definizione. Possono memorizzare grandi quantità di dati. Sono così chiamati a causa del *raggio blu* che serve per leggere il disco.

MEMORIA VIRTUALE

Si tratta di un meccanismo attraverso il quale il processore:

- *può fare accesso ad uno spazio di indirizzamento molto maggiore delle dimensioni della memoria principale realmente disponibile*
- *utilizza indirizzi logici che vengono poi trasformati in indirizzi fisici*

La traduzione da indirizzi logici a fisici è realizzata attraverso un componente HW chiamato ***Memory Management Unit (MMU)***.

Il compito principale della ***MMU*** è quello di verificare se la parola richiesta si trovi fisicamente nella memoria principale:

- ***se si*** → *produce l'indirizzo fisico corrispondente con cui si accede alla memoria principale*
- ***se no (page fault)*** → *richiede l'intervento del SO che provvede a spostare in memoria principale il relativo blocco di memoria*

La memoria è suddivisa in blocchi chiamati **pagine**.

Per effettuare la traduzione da indirizzi logici a fisici la ***MMU*** sfrutta una tabella detta ***Memory Address Table (MAT)***.

La ***MAT*** contiene una entry per ogni pagine virtuale.
Per ogni entry si memorizza:

- *l'indirizzo fisico al quale la pagina si trova nella memoria principale*
- *la posizione della pagina in memoria secondaria*

Per velocizzare l'accesso alla ***MAT*** le entry più accedute recentemente, vengono memorizzate in un'apposita memoria definita ***TLB*** posta direttamente a porta della ***MAT***.
Si può dire che la ***TLB*** svolga il ruolo di cache della ***MAT***.

Input/Output - I/O

Si tratta di un sotto-sistema che permette al sistema di elaborazione di interagire e di scambiare informazioni con il modno esterno.
Il sotto-sistema di I/O è composto da:

- *Dispositivi di Input/Output (**periferiche**)*
- *unità di controllo di questi dispositivi*ù
- *software per la loro gestione*

L'accesso alle periferiche viene realizzato mediante registri (***porte***).

Esempio:
Una stampante è vista dalla CPU come 3 registri:

- *un registro per i dati (output)*
- *un registro per le informazioni di controllo (output)*
- *un registro per le informazioni di stato (input)*

Per far eseguire delle operazioni alla stampante la CPU eseguirà letture/scritture sui 3 registri.

Periferiche collegate alla CPU attraverso il bus mediante diverse modalità di collegamento:

- ***memory-mapped I/O** → connessi come normali celle di memoria a cui si può tranquillamente accedere mediante indirizzi*
- ***isolated I/O** → gli spazi di indirizzamento per la memoria e per le porte di I/O sono separati. Per accedere alle porte di I/O si devono utilizzare apposite istruzioni*

Le velocità molto elevate della CPU si contrappongono a quelle spesso molto più lente dei dispositivi periferici, perciò si deve gestire la sincronizzazione tra le componenti.
Possibili soluzioni sono:

- ***I/O programmato** → la gestione delle periferiche è affidata completamente alla CPU che effettua ripetuti test (polling) sullo stato della periferica*
  - Vantaggi:
    - *poco costoso in teermini di HW*
  - Svantaggi:
    - *poco efficiente*
- ***Interrupt*** → si tratta di un segnale asincrono che la periferica invia alla CPU quando ha bisogno di un servizio
  - Vantaggi:
    - *si riduce il tempo per ottentere l'attenzione della CPU*
    - *la CPU non perde tempo ad effettuare ripetuti test (polling) per verificare lo stato della periferica*
- ***Direct Memory Access (DMA)** → i trasferimenti vengono affidati ad un modulo apposito (DMA Controller). Il modulo deve essere in grado di diventare bus master per poter prendere il controllo del bus e gestire i trasferimenti*
  - Vantaggi:
    - *nel caso di trasferimenti di grandi quantità di dati, la CPU è libera di svolgere altri lavori, scambiandosi di continuo il ruolo di bus master con il DMA Controller*

INTERRUPT

Nel caso degli ***Interrupt***, ci sono diverse cose da tenere in considerazione.
Ad esempio, la CPU, dopo aver ricevuto il segnale di ***Interrupt*** deve:

- *sospendere il lavoro che stava svolgendo*
- *salvarne lo stato per poter riprendere successivamente*
- *saltare all'esecuzione di una procedura di servizio dell'interruzione (**Interrupt Service Runtime - ISR**)*

Nota:

> in alcuni casi è necessario evitare che una richiesta di Interrupt interrompa il processore (ad esempio durante lo svolgimento di operazioni essenziali). Per risolvere questo potenziale problema si va ad aggiungere un bit di abilitazione degli interrupt nel registro di stato della CPU che, se settato ad 1 impedisce agli Interrupt di sospendere il lavoro del processore.

Per identificare quale periferiche abbia inviato una richiesta di ***Interrupt*** esistono varie soluzioni:

- ***Linee di interrupt multiple*** → *ogni periferica è collegata alla CPU mediante un diverso segnale di Interrupt*
    - Svantaggi:
        - *il numero di periferiche è quasi sempre superiore di quello dei segnali disponibili*
- ***Polling*** → *un unico segnale di Interrupt, quando la CPU riceve il segnale inizia a scandire linearmente tutte le parole di stato delle periferiche per trovare quella che ha fatto richiesta*
    - Svantaggi:
        - *latenza spesso elevata*
- ***Interrupt vettorizzato*** → *è il meccanismo più utilizzato. Si compone di diverse fasi:*
    - *quando la CPU è pronta a ricevere una richiesta di Interrupt, invia un segnale di Interrupt Acknowledge*
    - *il periferico che ha fatto richiesta pone sul bus il suo codice di identificazione*
    - *il codice viene utilizzato dalla CPU per determinare l'indirizzo della procedura di servizio usandolo come indice per accedere ad un vettore contenente gli indirizzi delle procedure di servizio (**Interrupt Vector Table** o **IVT**, il quale è posto in memoria principale, spesso nei primi indirizzi)*

Nel caso di richieste di Interrupt simultanee è necessario che venga servito per primo il periferico con priorità maggiore. In base alla differente soluzione attuata per le linee di Interrupt si hanno diversi casi di priorità:

- ***Linee di interrupt multiple*** → *la priorità è stabilita direttamente dalla CPU*
- ***Polling*** → *la priorità è data dallo schema di posizionamento e collegamento delle periferiche fra loro*
    - Svantaggi:
        - *per modificare la priorità bisogna effettuare modifiche direttamente sull'HW*
- ***Interrupt vettorizzato*** → *il più delle volte esisterà un Interrupt Controller che deciderà quale sia il primo periferico da servire*
    - Vantaggi:
        - *per definire le priorità basterà programmare l'Interrupt Controller*

DMA

Nel caso del ***DMA*** i traferimenti possono avvenire in modi diversi:

- ***a blocchi (burst transfer)*** → *una volta acquisito il controllo del bus (bus master), il DMA lo mantiene occupato per tutta la durata del trasferimento*
    - Vantaggi:
        - *trasferimenti al massimo della velocità*
    - Svantaggi:
        - *la CPU non può utilizzare il bus per tutta la durata del trasferimento (a volte anche molto lunga)*
- ***con cycle stealing*** → *il trasferimento viene effettuato dividendo i dati in piccoli blocchi occupando il bus per periodi limitati di tempo*
    - Vantaggi:
        - *la CPU non è bloccata per periodi troppo lunghi*
    - Svantaggi:
        - *velocità di trasferimento più lenta*
- ***in trasparent DMA*** → *il DMA Controller è capace di rilevare quando la CPU non utilizza il bus ed in quei momenti ne diventa il master*
    - Vantaggi:
        - *la CPU non è praticamente rallentata dal trasferimento*

Nota:

> nel possibile caso di DMA Controller multipli, bisognerà aggiungere la circuiteria per la gestione dei conflitti.

BUS

Si tratta di una struttura condivisa che interconnette due o più dispositivi.
Le unità connesse al bus utilizzano 2 tipi di dispositivi:

- ***driver*** → *per pilotare le linee del bus*
- ***receiver*** → *per leggere i valori sul bus*

Generalmente un bus è composto da 3 gruppi di segnali:

- ***segnali di dato*** → *normalmente sono in numero pari ad un multiplo di 8, possono essere bidirezionali o unidirezionali (in quest'ultimo caso dovranno essere presenti 2 linee)*
- ***segnali di indirizzo*** → *identificano lo slave con cui il master vuole dialogare*
- ***segnali di controllo*** → *forniscono informazioni di stato, di temporizzazione, di tipo di dati presenti sul bus, ecc*

Nota:

> esistono bus le cui linee trasportano a seconda dei momenti, differenti tipi di dato. Ad esempio in alcuni bus le linee di dato e quelle di indirizzo sono uguali (vengono condivise) e una circuiteria aggiuntiva gestisce il tipo di dato trasportato. Si utilizzano principalmente per ridurre il numero di linee (segnali).

Si possono avere 2 tipi di bus:

- ***bus singolo*** → *configurazione più semplice che comprende un singolo bus su cui vengono trasportate tutte le informazioni*
- ***bus multiplo*** → *configurazione che comprende l'utilizzo di più bus a seconda delle unità collegate. È utile nei casi in cui si desiderino prestazioni elevate o nei casi in cui i dispositivi connessi hanno diverse caratteristiche*

In un sistema a bus le unità connesse sono di 2 tipi:

- ***master*** → *inizia ogni procedura di trasferimento e sceglie lo slave con cui dialogare*
- ***slave*** → *risponde ai comandi dell'unità master (es. memorie, periferici)*

Per gestire le tempistiche con cui si svolgono le operazioni sul bus si sono affermate 2 soluzioni:

- ***bus sincroni***
  - Caratteristiche:
    - *le unità collegate utilizzano lo stesso segnale di clock oppure utilizzano clock separati ma alla stessa frequenza, scambiandosi periodicamenti segnali di sincronizzazione.*
    - *la frequenza del clock è impostata dal dispositivo più lento*
    - *ogni unità di dato è trasferita in un periodo di tempo prefissato*
- ***bus asincroni***
  - Caratteristiche:
    - *ogni comunicazione può avere una sua velocità, determinata da appositi segnali*
    - *si ottiene la massima flessibilità a spese di una maggiore complessità del protocollo*

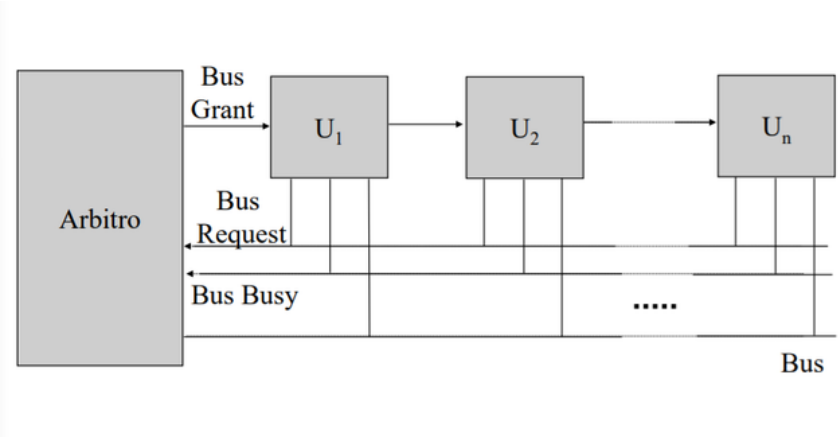
Ad ogni istante un solo dispositivo può funzionare da master del bus.
Per gestire l'arbitraggio ci sono due modalità:

- ***arbitraggio centralizzato*** → *esiste un solo arbitro (solitamente la CPU, ma non per forza)*
- ***arbitraggio distribuito*** → *ogni modulo contiene la logica che permette di definire il nuovo master*

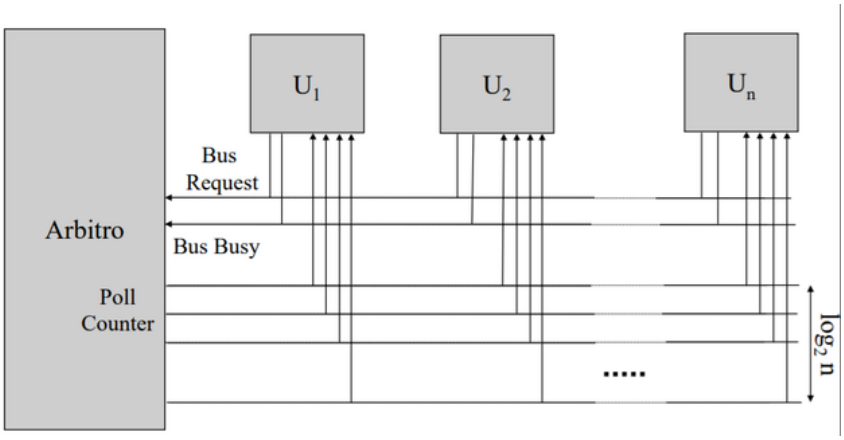
ARBITRAGGIO CENTRALIZZATO

Ne esistono 3 tipi:

- ***Daisy chaining***
  - Funzionamento:
    - *un'unità fa richiesta del bus mediante un apposito segnale (**BUS REQUEST**) attendendo che il bus sia libero (**BUS BUSY**)*
    - *l'arbitro attiva il segnale di **BUS GRANT***
    - *ogni unità, quando riceve il segnale di **BUS GRANT**, controlla se ha richiesto (**BUS REQUEST**) il controllo del bus:*
      - ***se si** → attiva **BUS BUSY** e prende il controllo del bus*
      - ***se no** → attiva **BUS GRANT** verso l'unità subito dopo*



- ***Polling***
  - Funzionamento:
    - *un'unità fa richiesta del bus mediante un apposito segnale (**BUS REQUEST**) attendendo che il bus sia libero (**BUS BUSY**)*
    - *l'arbitro scandisce tutte le unità mettendo sulle linee **POLL COUNTER** l'identificativo di ciascuna, in sequenza*
    - *quando un'unità è indirizzata:*
      - ***se avevo fatto richiesta** → l'arbitro interrompe la scansione e l'unità attiva B**US BUSY** prendendo il controllo del bus*
      - ***se non aveva fatto richiesta** → l'arbitro passa all'unità successiva incrementando il **POLL COUNTER** (o modificandolo, se la sequenza non è "lineare")*



- ***Richieste indipendenti***
  - Funzionamento:
    - *l'unità i-esima fa richiesta del bus mediante un apposito segnale (**BUS REQUEST i**) attendendo che il bus sia libero (**BUS BUSY**)*
    - *l'arbitro gestisce tutte le richieste e concede il bus all'unità con priorità massima (**BUS GRANT j**) tra tutte quelle che hanno fatto richiesta*
    - *l'unità j prende il controllo del bus*