PRIMA PARTE

1 – INTRODUZIONE

Descrivere la differenza tra architettura e organizzazione di un calcolatore.

L'Architettura è un insieme di caratteristiche visibili al programmatore, cioè attributi che hanno un impatto diretto sull'esecuzione logica di un programma (istruzioni, spazio e tecnica di indirizzamento)
L'organizzazione fa riferimento alle unità operative e le loro interconnessioni (interfacce tra calcolatore e periferiche, segnali di controllo, tecnologia delle memorie).
Gli attributi architetturali sono gli elementi hardware che costituiscono la struttura dell'elaboratore, quelli organizzativi sono dati dal modo in cui i primi vengono implementati.
Per esempio, la disponibilità o meno dell'operazione moltiplicazione è un aspetto architetturale, il modo in cui viene implementata (circuito dedicato o somme ripetute) organizzativo.

3 – COMPONENTI E CONNESSIONI

Descrivere in cosa consiste l'architettura Von Neumann

La progettazione di quasi tutti i calcolatori odierni è basata su concetti sviluppati da John Von Neumann. Essa è nota come Architettura di Von Neumann ed è basata su 3 concetti chiave:
1: i dati e le istruzioni risiedono in un'unica memoria di lettura e scrittura
2: i contenuti di questa memoria sono accessibili per indirizzo, indipendentemente dal tipo di informazione rappresentata;
3: l'esecuzione avviene in modo sequenziale, da un'istruzione a quella immediatamente successiva.

Differenza tra programmazione software e programmazione cablata

Per eseguire un programma, possiamo costruire i componenti logici in modo che il risultato sia quello voluto. Questo è il modo di costruire un "programma cablato", cioè in forma hardware, che non può essere modificato. Esso è un sistema non flessibile, che può eseguire solo istruzioni determinate. Tramite circuiti generici accetta segnali di controllo e produce risultati. Per nuovi programmi basta quindi dare i giusti segnali di controllo. La programmazione invece è molto più facile perché invece che ridefinire ogni volta l'hardware basta fornire una nuova sequenza di codici (ossia una nuova istruzione). Questo processo viene detto "programmazione software".

Descrivere in dettaglio il ciclo completo di fetch/execute delle istruzioni.

Il ciclo completo di fetch/execute delle istruzioni delle istruzioni può essere diviso in 7 stati, alcuni dei quali si possono ripetere più volte:

1. **Calcolo indirizzo istruzione:** determina l'indirizzo della prossima istruzione da eseguire
2. **Fetch:** legge l'indirizzo dalla memoria e lo trasferisce nel registro del processore noto come IR (instruction register)
3. **Decodifica Istruzione:** poi analizzata per determinare il tipo di operazione da eseguire l'operando da utilizzare
4. **Calcolo Indirizzo Operando:** determina l'indirizzo dell'operando
5. **Lettura Operando:**
6. **Operazione Sui Dati**: esegue l'operazione indicata dall'istruzione
7. **Memorizzazione Risultato:** il risultato viene scritto nella memoria e viene inviato alla periferica

Spiegare la differenza tra interruzioni multiple e interruzioni annidate, discutendo le differenti modalità di trattamento da loro richieste.

Per trattare le interruzioni si possono usare due approcci, il primo è disabilitare gli interrupt durante l'elaborazione di un interrupt, questi interrupt disabilitati rimarranno pendenti fino alla riabilitazione da parte della CPU e verranno gestiti in sequenza senza quindi tener conto di eventuali priorità da parte di qualcuno di essi. Il secondo invece consiste nel definire delle priorità e consentire quindi ad un interrupt di priorità maggiore di essere trattato subito, interrompendo quindi la gestione di un interrupt a priorità inferiore.

Descrivere in dettaglio il ciclo di esecuzione con trattamento delle interruzioni.

All'inizio di ogni ciclo esecutivo il processore legge un'istruzione dalla memoria. Dopo il prelievo di tale istruzione il processore incrementa il valore del registro PC (Program Counter) in modo che la prossima istruzione eseguita sia quella posizionata all'indirizzo di memoria immediatamente successivo. L'istruzione viene prima caricata in un registro del processore noto come IR (instruction register) e poi analizzata per determinare il tipo di operazione da

eseguire e gli operandi da utilizzare. In seguito per ogni operando si determina il suo indirizzo e subito dopo viene letto. Infine viene eseguita l'operazione indicata nell'istruzione e il risultato viene scritto nella memoria e viene inviato alla periferica. Dopodiché il processore controlla se è avvenuto qualche interrupt, se non vi sono interrupt pendenti, il processore procede al ciclo di prelievo e legge la successiva istruzione del programma altrimenti opera come segue: sospende l'esecuzione del programma in esecuzione salvandone il contesto (memorizzando quindi la l'indirizzo della prossima istruzione ed altri dati rilevanti per l'attività corrente) imposta il PC all'indirizzo di partenza di una routine per la gestione dell'interrupt. Il processore poi procede al ciclo di fetch e legge la prima istruzione del programma di gestione dell'interrupt. Quando questa routine termina, il programma riprende l'esecuzione del programma utente dal punto dell'interruzione.

Spiegare a cosa serve il bus di sistema, com'è strutturato e in che modo viene utilizzato in un calcolatore.

Il Bus di sistema è un mezzo di comunicazione condiviso che connette le componenti di un calcolatore e che permette lo scambio di dati fra di esse. Esso è composto da più linee (che va da 50 a 100) che trasmettono in parallelo un dato (sotto forma di segnali che indicano i bit 0 e 1) e tali linee vengono classificate in tre gruppi funzionali: dati, indirizzi e controllo. Le linee dati forniscono il percorso su cui viaggiano i dati tra i moduli del sistema e la loro ampiezza (ossia il numero di linee) varia da 32 a qualche centinaio ed è indispensabile per le prestazioni del sistema. Le linee indirizzi invece indicano la sorgente o la destinazione dei dati e per queste linee, l'ampiezza determina la quantità totale di memoria di un sistema. Infine le linee di controllo vengono utilizzate per controllare l'accesso e l'uso delle linee dati e indirizzi. Il bus in un calcolatore opera come segue: se un modulo desidera inviare dati a un altro deve prima di tutto ottenere l'uso del bus e poi trasferire i dati mentre se desidera richiedere i dati deve, dopo aver ottenuto l'uso del bus, prima trasferire una richiesta all'altro modulo sulle appropriate linee di indirizzo e controllo e poi attendere l'invio dei dati.

4 – MEMORIA CACHE

Nel contesto di una gerarchia di memoria, discutere la modalità e la granularità del trasferimento delle informazioni fra i vari livelli della gerarchia.

E' possibile organizzare gerarchicamente le memorie. Scendendo lungo la gerarchia, troviamo un decrescente costo per bit, una capacità crescente, un tempo di accesso crescente e una decrescente frequenza di accesso da parte del processore. In questo modo, memorie più piccole, più costose e più veloci sono integrate da memorie più grandi, economiche e più lente. La CPU utilizza direttamente il livello più alto della gerarchia. Ogni livello inferiore, deve contenere tutti i dati presenti ai livelli superiori (più altri). La dimensione di un blocco è la quantità minima indivisibile di dati che occorre prelevare (e copiare) dal livello inferiore. L'indirizzo di un dato diviene l'indirizzo del blocco che lo contiene, sommato alla posizione del dato all'interno del blocco.

Nel contesto di una gerarchia di memoria spiegare perché la memoria viene suddivisa in blocchi e, relativamente alle prestazioni della cache, discutere pregi e difetti dell'adozione di una dimensione di blocco elevata.

Per realizzare un'organizzazione gerarchica della memoria, che soddisfi i parametri di velocità, ampiezza e costo, conviene suddividere la memoria in blocchi. La dimensione di un blocco è la quantità minima indivisibile di dati che occorre prelevare (copiare) dal livello inferiore. L'indirizzo di un dato diviene l'indirizzo del blocco che lo contiene sommato alla posizione del dato all'interno del blocco.
Se si adotta un blocco con dimensione elevata si guadagna in termini di costi e capacità ma si perde in termini di velocità.

Nel contesto di una gerarchia di memoria, spiegare i possibili modi di realizzazione del mapping dei blocchi, discutendo vantaggi e svantaggi.

I possibili metodi di mapping dei blocchi in una gerarchia di memoria sono tre: diretto (direct mapping), associativo (n-way set) e set associativo (n-way set associative).
L'indirizzamento diretto è la tecnica più semplice in quanto assegna ad ogni blocco una sola possibile linea di cache. Per accedere alla cache poi, ogni indirizzo in memoria centrale viene diviso in tre campi: tag, linea, parola. Questo metodo di traduzione si distingue per la semplicità con cui quest'ultima avviene da indirizzo ILI (memoria) ad ILS (cache) e per la veloce determinazione di hit o miss. D'altro canto necessita di contraddistinguere il blocco presente in cache tramite un'etichetta (tag) e porta inoltre ad un numero elevato di swap per accedere ai dati di blocchi adiacenti. Il secondo metodo invece supera lo svantaggio dell'indirizzamento diretto poiché ogni blocco della memoria centrale può essere caricato su qualsiasi linea della cache. Così facendo l'indirizzo di memoria presenta solamente due campi: tag e parola e garantisce nel complesso una massima efficienza di allocazione. L'unico svantaggio è dato dalla complessità circuitale richiesta per esaminare in parallelo i tag di tutte le linee di cache.

L'indirizzamento set-associativo invece è un compromesso che unisce i punti di forza dei precedenti riducendo i loro svantaggi. Qui la cache è suddivisa in insiemi (set) di k linee e il blocco può essere assegnato a qualunque linea dell'insieme, l'indirizzo in memoria è suddiviso nei campi tag, set e parola. Questo tipo di indirizzamento vanta una buona efficienza di allocazione a fronte di una discreta complessità di ricerca.

Descrivere le politiche di scrittura write through e write back evidenziando differenze, vantaggi e svantaggi di entrambe.

Write through:
Ogni dato modificato nella cache viene contemporaneamente modificato nella memoria centrale. In questo modo i dati sono sempre coerenti tra i vari livelli di memoria. Lo svantaggio principale però è che per frequenti scritture sul medesimo blocco si verifica un aumento di traffico nel bus con conseguente collo di bottiglia.
Write back:
La scrittura in memoria centrale avviene solo quando il corrispettivo blocco in cache viene rimpiazzato. Ciò consente un ottimizzazione del traffico tra livelli ma causa periodi di incoerenza tra di essi, inoltre occorre sempre ricordare se sono avvenute operazioni di scrittura nel blocco tramite "dirty bit". Il problema di questa tecnica è che parti della memoria centrale non sono aggiornate, e dunque gli accessi tramite moduli I/O possono essere consentiti solo attraverso la cache.

Nel contesto di una gerarchia di memoria, spiegare come funziona la politica di scrittura write back. Discutere criticamente i problemi che possono insorgere nell'adottarla.

La politica di scrittura write back è una tecnica alternativa alla write through che evita il collo di bottiglia creato da quest'ultima minimizzando le scritture in memoria e applicando gli aggiornamenti solo nella cache. Il problema di questa tecnica è che parti della memoria centrale non sono aggiornate, e dunque gli accessi tramite moduli I/O possono essere consentiti solo attraverso la cache. Ciò richiede circuiti complicati e costituisce un potenziale collo di bottiglia. Oltre a ciò in un organizzazione a bus in cui più di un dispositivo è connesso e dove la memoria centrale è condivisa un'alterazione dei dati in una cache può invalidare tutti i dati nella memoria centrale e anche quelli nelle altre cache eventualmente connesse al bus. Per ovviare al problema esistono 3 approcci: un monitoraggio del bus con write through dove i controllori della cache osservano le linee di indirizzi ed invalidano eventuali scritture in una locazione di memoria condivisa da parte di qualche gestore del bus qualora tale locazione di memoria sia già residente nella cache. Una trasparenza hardware che permetta mediante un hardware aggiuntivo un aggiornamento costante delle memoria centrale e di tutte le cache presenti e una memoria *noncachable,* ovvero una porzione condivisa nella quale gli accessi sono dei cache miss dato che essa non viene mai copiata nella cache. Essa può essere identificata via hardware o tramite indirizzi riservati.

Nel contesto di una gerarchia di memoria, spiegare come i "miss" possono essere categorizzati in diversi tipi e dire quali strategie, per ogni tipo, si possono adottare per diminuirne il numero. Discutere criticamente tali strategie.

I miss in una gerarchia di memoria possono essere caratterizzati in 3 tipi diversi: miss di primo accesso, inevitabile e non riducibile, Miss per capacità insufficiente, quando la cache non può contenere tutti i blocchi necessari all'esecuzione del programma e miss per conflitto, quando più blocchi possono essere mappati (con associazione diretta o a gruppi) su uno stesso gruppo. Le tecniche risoluzione classiche per i miss per capacità insufficiente possono essere una maggiore dimensione del blocco la quale è una buona tecnica per fruire di località spaziale che però causa un aumento di miss per conflitto (a causa del numero ridotto di blocchi disponibili) mentre per i miss per conflitto una soluzione efficiente può essere la maggiore associatività che causa però un incremento del tempo di localizzazione in gruppo ed è soggetta alla regola del 2:1 (cache di N blocchi stessa probabilità di miss di cache a N/2 con associazione a 2 vie).
Altre tecniche di risoluzione possono essere l'adozione di una cache multilivello, la separazione di cache dati e cache istruzioni e l'ottimizzazione dei dati mediante compilatori che permettono le seguenti operazioni ossia il posizionamento accurato delle procedure ripetitive, la fusione di vettori in strutture (località spaziale) e la trasformazioni di iterazioni annidate (località spaziale).

Nel contesto di una gerarchia di memoria, illustrare le possibili tecniche (incluse quelle che coinvolgono il compilatore) che si possono adottare per tentare di minimizzare il numero di miss.

Le possibili tecniche per ridurre il numero di miss sono: **maggiore dimensione del blocco** la quale è una buona tecnica per fruire di località spaziale che però causa un aumento di miss per conflitto (a causa del numero ridotto di blocchi disponibili), **maggiore associatività** che causa però un incremento del tempo di localizzazione in gruppo ed è soggetta alla regola del 2:1 (cache di N blocchi stessa probabilità di miss di cache a N/2 con associazione a 2 vie).
Altre tecniche di risoluzione possono essere:

- utilizzare cache multilivello, cioè utilizzare una gerarchia di cache. Vi sarà un cache piccola, molto veloce e sullo stesso cip della CPU, detta cache on-chip, che permetterà alla CPU di accedere ai dati a "tempo 0"(cache L1). Vi è poi un'altra cache(L2), più grande e leggermente più lenta, associare alla cache L1.Ve ne può anche essere una terza(L3), ancor più grande e più lenta, associata alla L2. Le cache sono connesse tra di loro in modo indipendente dal bus di sistema, così da non pesare su di esso. Nonostante l'aumento dei dispositivi tra CPU e memoria centrale, vi è un aumento prestazionale rispetto alla connessione diretta della L1alle memoria centrale, in quanto, grazie alla localizzazione spaziale, vi sarà un alto numero di hit dentro i dati nella L3, la quale, in caso di miss potrà inviare i dati sino alla L1 in un tempo estremamente ridotto rispetto a quello che farebbe la memoria centrale.
- separare la cache in cache istruzioni e cache dati, così da rendere indipendente la scrittura/lettura dei dati dalla singola lettura delle istruzioni
- l'ottimizzazione dei dati mediante compilatori che permettono il posizionamento accurato delle procedure ripetitive, la fusione di vettori in strutture (località spaziale) e la trasformazioni di iterazioni annidate (località spaziale).

5 – MEMORIA INTERNA

Descrivere caratteristiche e le operazioni principali delle memorie a semiconduttore, considerando sia memorie RAM che ROM.

L'elemento base delle memorie a semiconduttore è la cella di memoria che presenta specifiche proprietà: due stati stabili, che rappresentano il bit 0 e 1, la possibilità di scrivere nella cella per impostare lo stato, la possibilità di leggere lo stato, inoltre tutti i tipi di memoria a semiconduttore sono ad accesso casuale (si accede alla parola tramite circuito dedicato).
Il più comune è la memoria RAM la cui caratteristica principale è la possibilità di leggere e scrivere dati in e da memoria in modo semplice e rapido tramite segnali elettrici. Tale memoria è inoltre definita volatile per il fatto che necessità di alimentazione costante se si vuole preservare i dati ed è per questo quindi che la RAM viene utilizzata solo per una memorizzazione temporanea. Essa si divide in 2 tipologie SRAM e DRAM. La DRAM acronimo di Dynamic RAM è composta di celle che memorizzano i dati sotto forma di cariche nei condensatori. L'assenza o presenza di cariche determina lo stato 0 o 1. Poiché i condensatori tendono a scaricarsi naturalmente dopo un certo periodo di tempo le RAM necessitano di un refresh periodico e da ciò deriva l'aggettivo "dinamiche" per il fatto quindi che la carica scompare dinamicamente anche in presenza di alimentazione. Le SRAM utilizzano invece gli stessi elementi base di un processore, infatti i valori binari vengono memorizzati mediante porte logiche e diversamente dalle DRAM non necessitano di un refresh delle cariche.
Un altro tipo di memoria ad accesso casuale è la ROM (read only memory) la quale presenta uno schema di dati predefinito e non modificabile. Tali memorie non sono volatili e mantengono quindi i dati anche in assenza di alimentazione, però questo tipo di dispositivi possono essere solo letti. Il vantaggio di questo memorie è che i dati e programmi di piccole dimensioni possono essere scritti in esse e non devono essere caricati da dispositivi esterni. In esse inoltre, i dati vengono inseriti durante il costoso processo di fabbricazione che richiede poi grande precisione poiché l'errata scrittura di un bit causa la perdita di tutto il lotto di ROM.

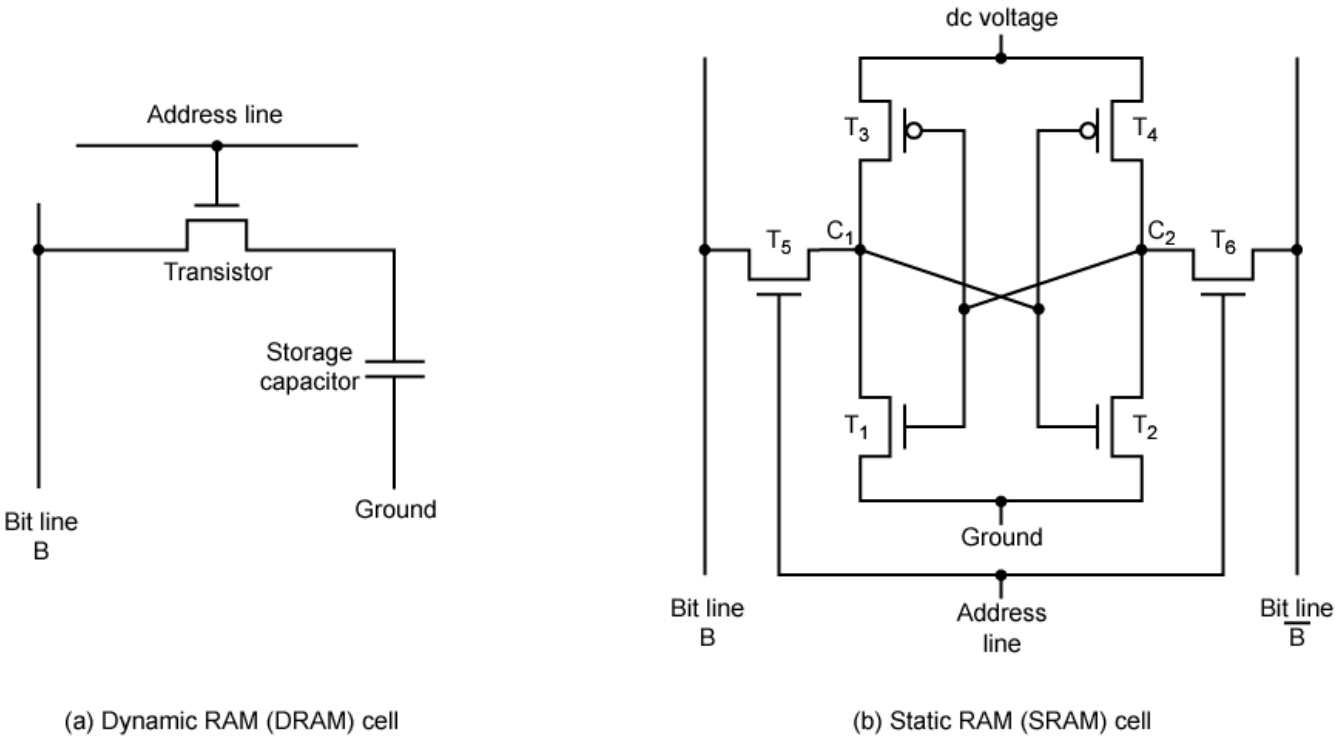
Descrivere in dettaglio le memorie DRAM.

La DRAM acronimo di Dynamic RAM è composta da celle che memorizzano i dati sotto forma di cariche nei condensatori. L'assenza o presenza di cariche determina lo stato 0 o 1. Poiché i condensatori tendono a scaricarsi naturalmente dopo un certo periodo di tempo le ram necessitano di un refresh periodico e da ciò deriva l'aggettivo "dinamiche" per il fatto quindi che la carica scompare dinamicamente anche in presenza di alimentazione.
Per le operazioni di scrittura si applica tensione alla linea di bit (che a seconda dell'intensità determina se bit è 1 o 0) successivamente si applica segnale alla linea degli indirizzi trasferendo la carica al condensatore. Per quelle di lettura invece prima di tutto si seleziona la linea indirizzo poi la carica viene convogliata in una linea di bit collegata ad un amplificatore che confronta la tensione con un valore di riferimento e determina se la cella contiene il bit 1 o 0 e infine si applica un refresh per ripristinare la carica nel condensatore.

Descrivere la struttura e il funzionamento di un modulo di memoria SRAM.

La SRAM è un dispositivo digitale in cui i valori sono memorizzati mediante porte logiche.
Non necessita di refresh.
Nella struttura tipica di una cella SRAM quattro transistor sono connessi in modo da costituire uno stato logico stabile. Nello stato logico 1 il punto C1 è alto e il punto C2 è basso; in questo stato, T1 e T4 sono spenti, mentre T2 e T3 sono accesi. Nello stato logico 0, il punto C1 è basso e il punto C2 è alto; in questo stato T1 e T4 sono accesi mentre T2 e T3 sono spenti. Entrambi gli stati sono stabili finché la cella è alimentata.

La linea di indirizzo controlla due transistor (T5 e T6). Quando a questa linea viene applicato un segnale, i due transistor vengono accesi, consentendo la lettura o la scrittura. Per un'operazione di scrittura, si applica il valore desiderato del bit alla linea B, e il valore negato alla linea $\overline{B}$. Questo forza i quattro transistor (T1, …. T4) allo stato corretto. In lettura, il valore viene letto dalla linea B.



(a) Dynamic RAM (DRAM) cell

(b) Static RAM (SRAM) cell

Figure 5.2 Typical Memory Cell Structures

Spiegare in dettaglio le differenze tra un modulo di memoria DRAM ed un modulo di memoria SRAM, discuterne vantaggi e svantaggi.

Una RAM dinamica memorizza i dati in forma di cariche su condensatori, quindi la sua cella di memoria è più semplice e più piccola di quella statica. Quindi le DRAM sono più dense e meno costose (a parità di capacità). Poiché essi tendono naturalmente a scaricarsi, le DRAM necessitano di circuiti aggiuntivi per il refresh periodico, ma per memorie grandi il costo di questi circuiti è più che compensato dal costo inferiore della singola cella.
Quindi le DRAM tendono a essere favorite per memorie di grandi capacità, infatti vengono usate per la memoria centrale.
Invece le SRAM sono più veloci ma hanno un costo maggiore per cui vengono usate per memorie piccole e veloci: la memoria cache.

Spiegare cosa sono gli errori soft, come ovviare a tali errori e fare eventualmente un esempio.

Le memorie primarie sono soggette ad errori. Questi possono essere classificati in guasti hardware, che sono permanenti, e guasti software, chiamati anche "soft error", che sono casuali e non distruttivi. Essi infatti alterano i contenuti di una o più celle di memoria, senza però danneggiarla fisicamente. Per rilevare ed eventualmente correggere gli errori soft è necessario usare dei codici a correzione d'errore: ciò consiste nel memorizzare, oltre agli M bit di dati, un codice di controllo calcolato tramite una funzione dei dati. Al momento della lettura viene calcolato un nuovo codice di controllo che poi viene confrontato con quello memorizzato e può produrre 3 risultati: nessun errore rilevato, viene rilevato un errore ed è possibile correggerlo, perciò i bit dati e i bit per la correzione vengono inviati ad un correttore che produce un insieme corretto di M bit da emettere, oppure viene rilevato un errore ma non è possibile correggerlo quindi ciò viene segnalato.
Un esempio di codice a correzione d'errore è il codice di Hamming.

Come funziona il codice di correzione Hamming? Dare un esempio concreto di codifica nel caso di memorizzazione di un insieme di 4 bit.

Il codice di correzione di Hamming consiste nel memorizzare, oltre agli M bit di dati, K bit come codice di controllo tale che $2^K-1 \geq M+K$. Quindi si ottiene una parola formata da M+K bit le cui posizioni sono numerate da 1 a M+K e in cui i bit di controllo nelle posizioni delle potenze di due. Ogni bit di controllo è calcolato facendo l'OR esclusivo tra i bit del dato il cui numero di posizione contiene un 1 nella stessa posizione di quel bit di controllo. Quando poi la

parola viene letta viene calcolato un nuovo codice di controllo che poi viene confrontato con quello memorizzato tramite l'OR esclusivo e il risultato viene chiamato sindrome. Se la sindrome contiene tutti 0 allora non è stato rilevato alcun errore, se contiene un unico bit a 1 si è avuto un errore nei bit di controllo e non è necessaria alcuna correzione, mentre su contiene più di un bit a 1 per correggere la parola è necessario complementare il bit nella posizione del valore numerico della sindrome.
Su parole di 4 bit utilizziamo il diagramma di Eulero Venn per dividere il piano in 3 cerchi che determinano 7 regioni, i 4 bit dati verranno assegnati agli scomparti interni. I restanti scomparti vengono riempiti con i cosiddetti bit di parità. Ciascun bit di parità è scelto in modo che il numero totale di 1 nel proprio cerchio sia pari. Ora se avvenisse una modifica dei bit dati l'errore sarebbe facilmente rilevato e potrebbe essere corretto cambiando il determinato bit che lo causa.

6 – MEMORIA ESTERNA

Descrivere l'organizzazione e formattazione dei dati nei dischi rigidi.

Nei dischi magnetici la testina è in grado di leggere e scrivere su una porzione del disco rotante. Ciò origina la disposizione fisica dei dati in anelli concentrici, chiamati tracce (track). Le tracce hanno la stessa larghezza della testina, e ne esistono migliaia per ciascun piatto. Tracce adiacenti sono separate da spazi (gaps). Ciò minimizza gli errori dovuti al disallineamento della testina o all'interferenza tra i campi magnetici. Il trasferimento dati avviene per settori, che generalmente sono un centinaio per traccia, di lunghezza fissa o variabile (odiernamente ammonta a 512 byte), i settori adiacenti sono inoltre separati da spazi. I bit più vicini al centro del disco ruotano attorno al punto fisso, come la testina, più lentamente dei bit esterni, questa velocità viene quindi compensata per permettere alla testina di leggere tutti i bit alla stessa velocità facendo ruotare il disco a velocità angolare costante.
Nei moderni sistemi viene utilizzata la tecnica nota come registrazione a più zone che nella quale la superficie è divisa in un certo numero di aree all'interno delle quali il numero di bit per traccia è costante. Le zone più lontane dal centro contengono un numero maggiore di bit, e di settori, rispetto a quelle più vicine e quindi consente una capacità di memorizzazione più elevata al costo di circuiti leggermente più complessi.
La formattazione invece è l'operazione con la quale si prepara il disco per renderlo idoneo all'archiviazione e consiste ad esempio nell'inserimento di un criterio che determini la posizione di un dato settore, il suo inizio e la sua fine e un punto di partenza sulla traccia. Questi requisiti sono rispettati tramite dati di controllo memorizzati sul disco con i quali esso viene inizializzato.

Discutere le ragioni per cui è stato sviluppato il sistema RAID. Inoltre si descriva in dettaglio tutti i livelli del RAID.

RAID (redundant array of indipendent disk) è un insieme di dischi fisici visti dal sistema operativo come una singola unità. Vista la difficoltà nell'incrementare le prestazioni di un singolo disco venne elaborato il sistema RAID per aumentare le prestazioni di un calcolatore grazie all'utilizzo di più dischi in parallelo. In questo modo si garantiva una più veloce gestione delle diverse operazioni di input/output poiché i dati vengono distribuiti su più dischi, una maggior capacità e, con l'aggiunta della ridondanza, una maggiore affidabilità visto che i dati sono recuperabili in caso di guasto di uno dei dischi.
RAID 0:
I suoi dischi sono divisi in strisce nelle quali i dati sono distribuiti a rotazione, con un algoritmo di round robin, che parte dal primo disco, percorre gli altri fino all'ultimo e poi riprende dal primo.
Non c'è ridondanza, quindi se si perde l'informazione contenuta in una striscia si perdono anche le altre strisce dello stesso disco, e non c'è modo di recuperarla.
I dischi eseguono la ricerca dei settori in parallelo, garantendo una lettura più efficiente rispetto a quella che sarebbe offerta da unico disco molto grande, e una maggiore velocità di accesso all'informazione.
Richieste multiple di dati hanno bassa probabilità di coinvolgere lo stesso disco, diminuendo i conflitti di risorse e aumentando la velocità di ricerca che è più frequentemente in parallelo.
RAID 1:
La ridondanza viene effettuata duplicando tutti i dati. Vengono fatte due copie di dati che vengono memorizzati su dischi separati. La lettura e scrittura avviene su entrambi i dischi.
E' garantita una grande sicurezza dei dati in quanto un recupero in caso di guasto è molto semplice, perché basta sostituire il disco non funzionante e ricopiare l'informazione.
E' però un sistema molto costoso poiché richiede doppie risorse.
RAID 2:
Un sistema RAID 2 è composto da dischi sincronizzati tra loro, cioè la testina di ciascun disco si trova nella stessa posizione su ogni disco, per consentire un accesso parallelo sincronizzato su di essi e memorizzare in maniera

distribuita una quantità di informazione molto piccola (singolo byte o parola) sui vari dischi; per accedere all'informazione completa bisogna accedere simultaneamente su tutti i dispositivi.
Usa un codice di Hamming per la correzione d'errore di errori singoli e il rilevamento di errori doppi.
RAID 2 sarebbe efficace nel caso si verificassero molti errori, ma vista l'alta affidabilità dei dischi non viene usato.
RAID 3:
RAID 3 è composto da dischi sincronizzati tra loro in cui i dati vengono distribuiti in piccole strisce (come raid 2).
Necessita di un solo un disco ridondante in quanto memorizza i bit di parità per ogni insieme corrispondente di bit.
Dati presenti su un disco difettoso possono essere ricostruiti a partire dai dati sui dischi rimanenti e dalle informazioni sulla parità.
Velocità di trasferimento molto alta, ma può eseguire una sola richiesta di I/O alla volta
RAID 4
Ogni disco opera indipendentemente: adatti per alti ritmi di richiesta, non per alti tassi di trasferimento.
Abbiamo bisogno di un parity disk che contiene le informazioni di parità relativa agli altri dischi.
L'unità di memorizzazione non è di singoli byte/bit ma blocchi.
Non commercializzato perché si crea collo di bottiglia nel disco di parità perché ogni volta che si fa una modifica su uno di essi si deve aggiornare la parità del disco di parità.
RAID 5
Simile a RAID 4 ma la parità viene distribuita su tutti i dischi.
L'allocazione dei blocchi di parità avviene con round robin su tutti i vari dischi ed evita il collo di bottiglia.
Soluzione comunemente utilizzata sui server di rete.
RAID 6
Simile al RAID 5 ma con due parità calcolate in modo diverso, quindi offre alta affidabilità dei dati al costo di un ulteriore disco e una penalità in scrittura di circa il 30% rispetto al RAID 5.

Discutere il modo in cui le informazioni sono organizzate in un CD-ROM.

Il CD è un dispositivo ottico non cancellabile di policarbonato dove le informazioni registrate digitalmente sono stampate come una serie di microscopici pozzetti (pit) sulla superficie. Tali informazioni, scritte da un laser ad alta intensità, vengono poi recuperate da un laser a bassa potenza.
Le informazioni sono organizzate su una traccia a spirale che inizia vicino al centro e si svolge verso il bordo del disco. Le informazioni sono quindi impacchettate uniformemente sul disco in segmenti della stessa dimensione e questi vengono letti ruotando il disco a velocità variabile (più veloce vicino al centro e più lenta all'esterno). I pit sono poi letti dal laser a velocità lineare costante.
I dati su un CD-ROM sono organizzati come una sequenza di blocchi. Un tipico formato consiste nei seguenti campi:

- Sync: inizio del blocco: 1 B di 0, 10 B di 1, 1 B di 0. (12 B)
- Header: contiene l'indirizzo del blocco e il byte di modo. (4 B)
    - Modo 0: campo dati vuoto
    - Modo 1: 2048 B di dati e 288 B di codice per la correzione d'errore
    - Modo 2: 2336 B di dati utente senza codice di correzione d'errore
- Data: i dati utente
- Auxiliary: codice per la correzione d'errore o dati utente (modo 2)

7 – INPUT/OUTPUT

Descrivere in dettaglio la gestione da programma I/O.

Nell'I/O da programma i dati vengono scambiati tra processore e modulo I/O. Quando il processore sta eseguendo un programma e incontra un'istruzione correlata con l'I/O, esso esegue l'istruzione inviando un comando al modulo di I/O appropriato che eseguirà l'azione richiesta e alla fine imposterà i bit appropriati nel suo registro di stato, ma senza interrompere la CPU. Quindi è compito del processore controllare periodicamente lo stato del modulo I/O finché non rileva il completamento dell'operazione. Questa tecnica presenta lo svantaggio di tenere occupato inutilmente il processore. Ci sono 4 tipi di comandi che il processore può inviare al modulo I/O e sono: Controllo (avvia una periferica e le comunica cosa fare), test (testa le condizioni di stato dei moduli di I/O), lettura (ottiene i dati dalla periferica attraverso il modulo I/O) e scrittura (impone al modulo di trasmettere tramite bus i dati alla periferica).

Differenza tra I/O memory mapped e separato

Nell'I/O memory mapped il processore tratta i registri dei moduli di I/O come locazioni di memoria e per accedervi utilizza le stesse istruzioni macchina per accedere alla memoria.

Invece con l'I/O separato il bus deve essere dotato di una ulteriore linea di comando che specifica se l'indirizzo si riferisce a una cella di memoria o a una periferica. In tal modo lo spazio può essere interamente disponibile per l'I/O o per la memoria.
La tecnica memory mapped ha i vantaggi che non necessita di istruzioni speciali (il software di controllo può essere scritto interamente in linguaggi di alto livello) e consente una più agevole protezione (è sufficiente nascondere le aree di I/O allo spazio di indirizzamento utente); ma presenta gli svantaggi che non si presta all'uso di cache (occorre disabilitarla) e non è compatibile con architetture a bus multipli (occorre filtrare gli emessi dalla CPU e instradarli sul bus appropriato).

Descrivere in che modo vengono gestite le interruzioni (sia per la componente hardware che per quella software) nel caso di I/O input driven.

In questo caso, per evitare che il processore debba costantemente verificare lo stato del dispositivo di I/O, è il modulo di I/O a interrompere la CPU mentre questa esegue altro lavoro.
Infatti dopo che la CPU invia il comando di lettura al modula di I/O continua svolgere altro lavoro mentre il modulo di I/O procede alla lettura dei dati dalla periferica e appena ha terminato invia un segnale di interrupt alla CPU. Il processore conclude l'operazione corrente e poi, constatata la presenza di interrupt, invia un segnale di riconoscimento al dispositivo che ha inviato tale interrupt che rimuoverà quindi il proprio segnale. Il processore poi salva il contesto del programma ponendo PSW (ovvero lo stato del processore) e la locazione della prossima istruzione da eseguire contenuta nel PC in cima alla pila di sistema. Infine il processore scrive nel PC l'indirizzo della prima istruzione della routine di gestione dell'interrupt considerato. Dopodiché il controllo viene trasferito al programma di gestione dell'interrupt che procederà nel modo seguente: vengono salvate le restanti informazioni del processo dai registri alla pila, successivamente avviene l'elaborazione dell'interrupt e, quando questa è completa, i valori dei registri salvati vengono recuperati dalla pila e riportati nei registri ed infine i valori di PSW e PC vengono ripristinati quindi riprende l'esecuzione del programma precedentemente interrotto

Descrivere gestione dell'I/O tramite DMA.

Il DMA (direct access memory) è un modulo hardware addizionale che consente lo scambio immediato dei dati tra il modulo di I/O e la memoria centrale senza il coinvolgimento del processore.
Quando la CPU desidera leggere o scrivere un blocco di dati, invia un comando al modulo DMA insieme alle seguenti informazioni: lettura/scrittura, indirizzo dispositivo interessato, indirizzo iniziale in memoria del blocco dati coinvolto nell'operazione, quantità di dati da trasferire. Il processore continua poi con altro lavoro mentre il DMA trasferisce l'intero blocco, una parola per volta, direttamente da o verso la memoria senza passare per il processore e solo quando il trasferimento è completato invia un segnale di interrupt al processore.
Il DMA è connesso al bus di sistema e può accedere al canale dati in 2 modi:

- una parola alla volta, sottraendo di tanto in tanto alla CPU il controllo del canale (cycle stealing)
- per blocchi, prendendo in possesso il canale per una serie di trasferimenti (burst mode), più efficace perché l'acquisizione del canale è onerosa

Una configurazione ideale per consumare meno cicli di bus è creare un percorso tra DMA e I/O senza passare per il bus di sistema in modo che il DMA utilizzi il bus solo per scambiare dati con la memoria.