DOMANDE PRIMA PARTE

Descrivere la differenza tra architettura e organizzazione di un calcolatore

L'architettura di un elaboratore è costituita dall' insieme degli attributi visibili al programmatore, come le istruzioni, il numero di bit usati per rappresentare i dati, i meccanismi di I/O e le tecniche di indirizzamento alla memoria. L'organizzazione riguarda gli aspetti hardware trasparenti al programmatore, come i segnali di controllo, le interfacce tra periferiche e programmatore e la tecnologia della memoria.

Descrivere in cosa consiste l'architettura di Von Neumann

L'architettura di Von Neumann è alla base di quasi tutti i calcolatori odierni. Essa consiste in una memoria centrale, alla quale è accessibile per indirizzo e contiene dati e istruzioni, un'unità logico-aritmetica (ALU) in grado di operare su dati binari, un'unità di controllo (UC) che interpreta ed esegue le istruzioni e dei dispositivi di I/O, i quali servono per comunicare con il calcolatore e viceversa. ALU e UC vanno a formare la CPU, nella quale sono presenti dei registri, una sorta di memoria interna.

Spiegare a cosa serve il bus di sistema, com'è strutturato e in che modo viene usato dal calcolatore

Il bus di sistema connette le varie componenti del calcolatore, è composto da più linee che trasmettono in parallelo un dato, dove ogni linea può trasportare un bit alla volta. Ci sono due tipi di linee: dedicate, cioè linee separate per indirizzi e dati (meno contesa del bus ma costi e dimensioni maggiori), e multiplexate, cioè una linea per indirizzi e dati con linea di controllo che segnala il tipo delle informazioni che transitano (economiche ma circuiti complessi e prestazioni limitate). Gli eventi sono determinati da dei cicli di uguale durata, detti clock (sequenze 0-1), nel caso di temporizzazione sincrona, oppure dipendenti da un evento precedente nel caso di temporizzazione asincrona. La gestione del bus può essere centralizzata (cioè c'è un controller che decide a chi assegnare il bus) o distribuita (cioè l'autogestione dei moduli). Ci sono 4 tipi di bus: il bus dati, dove viaggiano i dati, il bus indirizzi, dove viaggiano le destinazioni dei dati, bus di controllo, che controllano l'uso dei bus precedentemente citati comprendendo segnali di temporizzazione e comandi e, infine, il bus di alimentazione elettrica.

Descrivere in dettaglio la gestione I/O da programma

Nell' I/O da programma i dati si scambiano tra processore e modulo I/O. Quando si incontra un'istruzione che richiede un intervento I/O la CPU invia un comando al modulo I/O appropriato. Il modulo lo esegue e setta il bit di stato una volta finito. La CPU periodicamente controlla il bit di stato, in questo modo l'I/O non interrompe la CPU non informandola direttamente. I comandi che la CPU può inviare 4 tipi di comandi al modulo I/O: controllo, dice al modulo cosa fare, test, controlla il bit di stato, di lettura e di scrittura, trasferiscono il dato da/verso il dispositivo.

Nel contesto di gestione dell'I/O si spieghi la differenza fra tecnica I/O memory mapped e I/O isolated. Discuterne vantaggi e svantaggi

Nella gestione di I/O ci sono due modi di indirizzamento. Nell'indirizzamento I/O memory mapped non c'è distinzione tra indirizzi di memoria e dispositivi, quindi vengono indirizzati nella stessa maniera. Nell'indirizzamento I/O isolated invece gli spazi di indirizzamento sono separati, il bus quindi ha una linea specifica per verificare se l'indirizzo è rivolto a una periferica o alla memoria; è un metodo più ordinato ma raddoppia il numero degli indirizzi.

Spiegare il vantaggio di utilizzare il sistema di interruzioni

Utilizzare un sistema di interruzioni evita che la CPU resti ferma ad aspettare il completamento delle istruzioni, come per esempio aspettare la fine di un trasferimento dei dati). Attraverso i segnali di interrupt un dispositivo può attirare l'attenzione della CPU, cosi che, dopo aver eseguito i comandi in corso, possa dedicarsi ad esso. In questo modo il processore non avrà periodi di inattività e il tempo verrà ottimizzato.

Spiegare la differenza fra interruzioni multiple e interruzioni annidate discutendo criticamente le differenti modalità di trattamento da loro richieste

Le interruzioni possono essere multiple o annidate. Quelle multiple avvengono quando sono disabilitati gli interrupt durante l'elaborazione di un interrupt. Gli interrupt, in pratica, restano pendenti fino alla terminazione di quello in esecuzione. Questo metodo non tiene conto do alcuna priorità o tempistica. Nelle interruzioni annidate invece vengono dati dei gradi di priorità agli interrupt permettendo a un interrupt con priorità maggiore di interrompere l'esecuzione di un interrupt con priorità minore.

Si descriva in dettaglio il ciclo completo di fetch execute delle istruzioni

Calcolo Indirizzo Istruzione: determina l'indirizzo dell'istruzione successiva
Fetch: legge l'istruzione dalla memoria e la trasferisce nel processore
Decodifica Istruzione: analizza l'istruzione e determina i tipi di operandi
Calcolo Indirizzo Operando: determina l'indirizzo dell'operando in base al formato dell'istruzione
Lettura Operando: legge l'indirizzo dell'operando da memoria o da periferica
Operazione su Dati: esegue l'istruzione
Memorizzazione Risultato: scrive il risultato in memoria o in una periferica

Descrivere in dettaglio il ciclo di esecuzione con trattamento delle interruzioni

All'inizio di ogni ciclo il processore esegue il fetch dell'istruzione il cui indirizzo, scritto su PC, viene subito dopo incrementato. L'istruzione viene caricata sul registro IR e viene analizzata per determinare il tipo dell'istruzione e gli operandi, che verranno prelevato da memoria o da periferica. Poi si esegue l'istruzione e il risultato viene salvato in memoria. La CPU a questo punto controlla se è avvenuto un interrupt. Se si salva l'indirizzo dell'istruzione attuale e imposta PC

all'indirizzo del programma che gestisce l'interruzione, lo esegue e una volta finito rimette su PC l'istruzione precedentemente salvata. Se non ci sono interrupt il programma prosegue normalmente.

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

Nell' I/O interrupt driven la CPU rilascia un comando al modulo I/O, può essere sia di lettura che di scrittura, poi prosegue con altre istruzioni mentre il modulo I/O ottiene/riceve dati dalla periferica. Una volta finito il modulo interrompe la CPU inviandole un segnale di interrupt così che, una volta completata l'istruzione corrente, possa poi occuparsi dell'interrupt. In questo modo il modulo I/O invia i dati alla CPU che, una volta completato il trasferimento, torna ad occuparsi di altre istruzioni. In pratica il modulo I/O, una volta che la CPU fa richiesta di un comando di lettura/scrittura, fa il lavoro al posto della CPU e una volta finito lancia un segnale di interrupt, così da trattare quell'operazione come un interrupt.

Si descrivano le caratteristiche e le operazioni principali della memoria a semiconduttore considerando sia le memorie RAM che ROM

Le memorie a semiconduttore si basano su una cella di memoria con tre proprietà: può assumere due stati stabili (0 e 1), vi si può scrivere per cambiare lo stato e si può leggere lo stato. Tutte le memorie a semiconduttore sono ad accesso casuale, si accede cioè alla cella tramite un circuito dedicato. La più comune è la RAM (Random Access Memory), nella quale si possono leggere e scrivere dati tramite segnali elettrici. Essa è definita volatile in quanto una volta staccata la corrente elettrica si perdono tutti i dati. Esistono due tipi di RAM, la SRAM E LA DRAM. La SRAM, o RAM statica, è una memoria che utilizza dei transistor per memorizzare dati binari, viene definita statica perché non necessita di alcun refresh delle cariche. La DRAM, o RAM dinamica, è una memoria che utilizza dei condensatori per salvare dati binari, viene detta dinamica perché necessita di un refresh periodico delle cariche dato che i condensatori tendono a scaricarsi.
Oltre alla RAM anche la ROM (Read Only Memory) è una memoria a semiconduttore. Essa è una memoria non volatile ma vi possibile scrivervi una volta sola, infatti in esse si trovano elementi del BIOS o programmi necessari al compilatore salvati in fase di progettazione. Le ROM possono essere di più tipi: ci sono le PROM, che sono rom programmabili una sola volta, le EPROM, cancellabili per intero grazie ai raggi UV, le EEPROM, che possono essere cancellate e scritte elettronicamente, e le memorie flash, le quali possono essere cancellate più velocemente delle EPROM.

Si descriva in dettaglio la memoria DRAM

La DRAM è una variante della RAM, un tipo di memoria a semiconduttore. In essa i dati sono segnali elettrici memorizzati in condensatori. Se c'è corrente essa viene interpretata come 1 da un valore di soglia, se non c'è viene interpretato come 0. Il condensatore tende a scaricarsi, quindi c'è bisogno di circuiti che effettuino periodicamente il refresh, da qui l'attributo Dinamica. Viene chiamata memoria volatile perché mantiene i dati fino a che è alimentata. Quando si

Scrive si applica tensione alla linea di bit e successivamente si applica tensione alla linea degli indirizzi che trasferisce la carica al condensatore. Quando si legge si seleziona prima la linea di indirizzo, poi il valore di carica del condensatore viene comparato a un valore di riferimento per capire se contiene 0 o 1. Si applica un refresh per ripristinare le cariche alla fine della lettura.

Si descriva in dettaglio la memoria SRAM

La SRAM è una variante della RAM, un tipo di memoria a semiconduttore. Essa utilizza gli stessi elementi base del processore, i transistor, per memorizzare i valori binari. Non necessita di refresh, da qui l'attributo Statica. Viene chiamata memoria volatile perché mantiene i dati fino a che è alimentata. La linea d'indirizzo viene usata per aprire o chiudere un interruttore, il quale controlla due transistor. Questi ultimi, quando viene applicato un segnale, vengono accesi permettendo la lettura/scrittura dello stato della cella.

Si faccia un confronto tra SRAM e DRAM

Sia la DRAM che la SRAM sono memorie volatili, le quali mantengono i dati fino a che sono alimentate. Le DRAM in particolare sono più economiche, perché costituite da condensatori, e più dense, ma necessitano di un hardware per i circuiti di refresh, per questo vengono preferite per grandi quantità di dati, come la memoria centrale. Le SRAM invece sono più costose, essendo composte da transistor, ma più veloci delle DRAM, quindi più piccole, per questo vengono utilizzate per la memoria cache.

Spiegare in dettaglio le differenze tra un modulo di memoria DRAM e uno di memoria SRAM, discutendone vantaggi e svantaggi

Vedi risposte sopra.

Spiegare in dettaglio come funziona il codice di correzione di Hamming, dare un esempio concreto di codifica nel caso di memorizzazione di un insieme di 8 bit

Quando i dati stanno per essere memorizzati viene creato un codice che viene memorizzato insieme ad essi. Questo servirà in futuro per vedere se il dato letto è giusto grazie al confronto con il codice salvato. Il più semplice tra i codici per correggere gli errori è il codice di Hamming. Quando i circuiti ricevono i due codici di k bit essi vengono confrontati bit a bit con lo XOR. Ciò che si genera viene chiamato sindrome, ampia k bit, con valore da 0 a 2^k-1 e indica il bit errato. Il codice di Hamming deve rispettare la formula **$2k-1 \geq M + K$** con M che è il numero di bit della parola data e K il numero di bit di controllo che si possono aggiungere. Se la sindrome ha tutti 0 non ci sono errori, se ha un unico bit 1 c'è un errore nel bit di controllo, se contiene più di 1 allora il valore numerico indica la posizione del bit errato. (esempio concreto con tabella)

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

In modo da realizzare un'organizzazione gerarchica della memoria conviene che sia suddivisa in blocchi. La dimensione di un blocco è la quantità minima indivisibile di dati da prelevare dalla memoria di livello inferiore. L'indirizzo di un dato quindi diventa l'indirizzo del blocco che lo contiene sommato alla posizione del dato all'interno del blocco.
Adottando una dimensione di blocco elevata si guadagna in termini di capacità ma si perde in termini di velocità. Inizialmente infatti una dimensione di blocco maggiore porta all'aumento di dati utili in cache, quindi la percentuale di successo aumenta. Blocchi più larghi però portano ad avere meno blocchi in cache, il che può portare a sovrascritture di blocchi in fasi successive al loro prelievo, con una conseguente diminuzione della percentuale di successo.

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

Il mapping è un tipo di algoritmo che esegue l'indirizzamento dei blocchi di memoria centrale nella cache e può essere di tre tipi. Il mapping diretto, più semplice e meno costoso, associa a ogni blocco una sola possibile linea della cache. Per accedervi ogni indirizzo viene suddiviso in tre campi: tag, set e parola. Semplice ma se un programma accede ripetutamente a due blocchi a cui è assegnata la stessa linea di cache c'è un elevato numero di miss. Il mapping associativo invece associa a ogni blocco una qualsiasi linea della cache, cosi che l'indirizzo presenti solo i campi tag e parola. Efficiente ma c'è difficolta nell'esaminare in parallelo i tag di tutte le linee di cache. Il mapping set-associativo infine è un ibrido dei due metodi sopra citati. In esso infatti la cache è suddivisa in set di k linee, ogni blocco può essere assegnato a qualunque linea del set e l'indirizzo è suddiviso nei campi tag, set e parola.

Nel contesto di una gerarchia di memoria spiegare come i MISS possono essere categorizzati in diversi tipi e dire quali sono le strategie (anche quelle che coinvolgono il compilatore), per ogni tipo, che si possono adottare per tentare di diminuirne il numero. Discutere tali strategie

I miss possono essere di tre tipi: di primo accesso, miss inevitabile, di capacità insufficiente, quando la cache non può contenere altri blocchi, e di conflitto, quando più blocchi possono essere mappati sullo stesso gruppo. Come soluzioni ai problemi di miss sono state ideate diverse strategie. Una per esempio Sfrutta la località dei riferimenti aumentando la dimensione del blocco, ma aumentano i miss di conflitto essendoci meno blocchi disponibili. Per i miss di conflitto una soluzione efficiente può essere l'incremento dell'associatività, causando però un incremento del tempo di localizzazione ed è soggetta alla regola 2:1, dove una cache a N blocchi con associazione diretta ha una probabilità di miss ha una probabilità di miss uguale a una cache N/2 con associazione a due vie. Altre tecniche comprendono l'adozione di una cache multilivello, una cache separata tra dati e istruzioni e ottimizzazione tramite compilatore.

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

Le memorie a semiconduttore sono soggette ad errori. A differenza degli errori hard, che sono guasti permanenti, gli errori soft sono danni alla memoria non permanenti. Essi vengono rilevati e, eventualmente, corretti usando codici di correzione, come il codice di Hamming (vedi sopra).

Nel contesto di una gerarchia di memoria spiegare come funzionano le politiche di scrittura write through e write back. Discuterne criticamente i problemi che possono sorgere adottandole, vantaggi e svantaggi di ciascuna

I dati presenti in memoria possono essere modificati da componenti diversi in momenti diversi. Si cerca, quindi, di adottare una politica di scrittura adeguata. La politica di write through consente di scrivere in memoria ogni dato in contemporanea a quando viene modificato nella cache. Con questo metodo i dati sono sempre coerenti tra i vari livelli di memoria, ma aumenta notevolmente il traffico, con il rischio di causare un collo di bottiglia. La politica di write back invece consente di scrivere in memoria centrale solo quando il medesimo blocco in cache viene rimpiazzato. Questo metodo evita il collo di bottiglia, ma può causare dei lunghi periodi di incoerenza tra i vari livelli, in più bisogna sempre controllare il bit che segna se è avvenuto o meno un rimpiazzamento del blocco in cache. Infine, gli accessi, anche I/O, devono passare per la cache, il che rende i circuiti molto più complessi.

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

È possibile organizzare gerarchicamente le memorie in base a fattori come velocità / prezzo / grandezza. Scendendo lungo la gerarchia, troviamo un minor costo per bit, una capacità maggiore, un tempo di accesso crescente e una minore 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 prelevabile 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.

Discutere il modo in cui le informazioni sono organizzate in un CD-ROM (formato dati)

Il CD è un dispositivo ottico che sfrutta una serie di pozzetti per memorizzare una serie di dati binari su una superficie di policarbonato. Un laser distingue i pit (pozzetti) dai land, distinguendo così il segnale digitale. La traccia forma una spirale che comincia vicino al centro e finisce sul bordo in modo che i settori mantengano una lunghezza costante. Le informazioni quindi vengono lette a velocità lineare costante, il disco quindi ruota più lentamente per gli accessi sul lato esterno rispetto a quelli vicini al centro.
I CD-ROM a differenza dei CD non sono cancellabili e presentano dei dispositivi di correzione degli errori per assicurare la correttezza del trasferimento.

Descrivere l'organizzazione e formattazione dei dati nei dischi rigidi

I dischi magnetici sono dei piatti di vetro rotanti rivestiti di un materiale magnetizzabile. La lettura/scrittura viene effettuata da un braccio meccanico dotato di due testine, una per la lettura e una per la scrittura, che stanno ferme mentre il disco ruota. Nel caso di scrittura la testina viene polarizzata e memorizza 0 e 1 sul disco sotto forma di campi magnetici. Nel caso della lettura invece la testina è fatta di un sensore magneto-resistivo, che permette alla testina di capire la direzione del campo magnetico (0 e 1 hanno campi magnetici opposti). Nei dischi i dati sono disposti in anelli concentrici (tracce) separati da spazi. Ogni traccia è composta da diversi settori, che sono l'unità minima di lettura, anch'essi separati da spazi. Per far si che la testina legga alla stessa velocità tutti i dati, i dischi vengono fatti ruotare con velocità angolare costante.
La formattazione è il processo con il quale il disco viene inizializzato con dei dati invisibili all'utente che permettono di gestire ad esempio il posizionamento della testina e la memorizzazione.

Descrivere la gestione dell'I/O tramite DMA

Il DMA (Direct Memory Access) è un modulo addizionale collegato al bus che sostituisce la CPU per la maggior parte delle attività di I/O. Si occupa di trasferire i dati da o verso la memoria senza passare attraverso il processore, il quale viene coinvolto solo a fine e a inizio trasferimento. Prima di tutto la CPU comunica al DMA il tipo di operazione (lettura/scrittura), l'indirizzo del dispositivo, l'indirizzo iniziale in memoria del blocco dei dati coinvolto e la quantità di dati, poi prosegue con altre attività. Il DMA nel frattempo si occupa del trasferimento dei dati e invia un segnale di interrupt alla CPU quando ha terminato. Ci sono due modi in cui il DMA può accedere al canale dati: una parola alla volta, sotraendo di tanto in tanto il controllo sul canale alla CPU (cycle stealing) oppure per blocchi, prendendo possesso del canale per una serie di trasferimenti (burst mode).

Discutere le ragioni per cui è stato sviluppato il sistema RAID. Inoltre, si descriva in dettaglio il livello 0, 1, 2, 3, 4, 5, 6 del RAID

Il sistema RAID (Redundant Array of Indipendent Disks) Venne elaborato per aumentare le prestazioni di un calcolatore. Esso utilizza più dischi che operano in parallelo, garantendo così una maggiore efficienza nelle gestioni I/O, in più, con la ridondanza, garantisce anche una maggiore affidabilità, poiché i dati sono distribuiti in più dischi e sono recuperabili facilmente. Esso prevede 7 configurazioni:
RAID 0: non include la ridondanza dei dati, se ci sono due richieste di blocchi che si trovano in dischi diversi vengono servite in parallelo. I dati sono distribuiti in strisce distribuite a rotazione sui dischi. In caso di guasto i dati sono irrecuperabili;
RAID 1: la ridondanza è ottenuta duplicando i dati. Si usa lo striping dei dati, dove ogni striscia di dati si trova fisicamente su due dischi, quindi una lettura può essere soddisfatta dal disco da cui si accede più velocemente al dato richiesto. La scrittura viene eseguita in parallelo in entrambi i dischi, quindi le prestazioni sono condizionate dalla scrittura più lenta;
RAID 2: sfrutta l'accesso parallelo, quindi le testine si trovano nella stessa posizione in ogni momento. Si usa lo striping dei dati con strisce molto piccole, vengono generati codici di correzione di Hamming memorizzati in più dischi, perciò se si riscontra un errore il controllore può riconoscere e correggere l'errore istantaneamente in modo che il tempo di accesso per la lettura non venga rallentato. Mai commercializzata;
RAID 3: Solo un disco ridondante che memorizza i bit di parità per ogni striscia dei dischi dei dati. In caso di guasto si accede al disco di parità e i dati vengono ricostruiti con lo XOR con i bit degli altri dischi, oppure vengono generati al momento;

RAID 4: ogni disco opera indipendentemente dagli altri. Usa lo striping dei dati con strisce grandi e utilizza una striscia di parità bit a bit sulle corrispondenti strisce di ciascun disco dati, e i bit di parità sono memorizzati nella corrispondente striscia nel disco di parità. Ottimo per richieste di I/O per dati di grandi dimensioni;
RAID 5: Simile al RAID 4, solo che distribuisce le strisce di parità su tutti i dischi seguendo lo schema round robin. Per un sistema a *n* dischi la striscia di parità si trova su un disco diverso dalle prime *n* strisce, poi si ripete, in questo modo si evita il collo di bottiglia. Usato nei server di rete;
RAID 6: si effettuano due calcoli di parità memorizzati in blocchi separati su dischi differenti, in modo che se si richiedono n dischi effettivamente saranno n+2 dischi. Questo rende possibile rigenerare i dati anche se due dischi contenenti i dati utente si guastano. Altissima affidabilità, ma scrittura lenta.