COMPUTER SYSTEMS OVERVIEW

Diapositiva 2
Processor: controlla le operazioni del computer e svolge le sue funzioni di elaborazione dei dati
Main Memory: memorizza dati e programmi
I/O Modules: trasferiscono dati tra il computer (nel senso di CPU e memoria) e l'ambiente esterno. anche la memoria secondaria, cioè l'hard disk è un elemento esterno!

Diapositiva 3
il processore svolge funzioni di comando e controllo e scambia dati con la memoria principale sfruttando dei particolari registri:
MAR - Memory Address Register: indirizzo di memoria per la prossima operazione di lettura o scrittura.
MBR - Memory Buffer Register: contiene i dati da scrivere in memoria o quelli appena letti dalla memoria.
I/O AR (Input/Output Address Register) e I/O BR (Input/Output Buffer Register) svolgono le stesse funzioni dei due precendenti ma in relazione alle periferiche di I/O.
un modulo di memoria è un insieme di locazioni numerate sequenzialmente che contengono dati in notazione binaria interpretabili come dati o come istruzioni.
un modulo di I/O trasferisce i dati dai dispositivi esterni al processore e alla memoria e viceversa e contiene un buffer al suo interno per conservare temporaneamente i dati se non possono essere inviati nel momento in cui il modulo li riceve, così da inviarli appena possibile.

Diapositiva 6
in alcuni casi i registri dati sono di tipo generico e possono essere usati per scopi diversi e con dati di qualsiasi tipo, altri invece sono specializzati, ad es. esistono dei registri dedicati alle operazioni in virgola mobile.
Anche per i registri indirizzo l'uso può essere generico o specifico a seconda del tipo di indirizzamento.

Diapositiva 7
INDEX REGISTER: nell'indirizzamento indicizzato si somma un certo indice ad un valore base per ottenere l'esatto indirizzo.
SEGMENT POINTER REGISTER: con l'indirizzamento segmentato la memoria cosiderata suddivisa in blocchi di lunghezza variabile e l'indirizzo si ottiene sommando ad un valore base, che indica l'inizio del segmento, un valore che indica l'offset necessario per raggiungere la cella interessata.
STACK POINTER REGISTER: nell'indirizzamento tramite stack c'è bisogno di uno stack pointer che punti alla cima della pila.
sono parzialmente visibili all'utente anche i condition code e i flag.
(vedi control and status register)

Diapositva 8
PC, detto anche Instruction Pointer (IP)
il PSW è un registro o un insieme di registri, che contiene informazioni d stato, tipicamente flags.

Diapositiva 9
questi valori vengono in genere utilizzato nei salti condizionati.

Diapositiva 10
il processore preleva l'istruzione e la esegue. l'esecuzione di un'istruzione può consistere di diverse operazioni, dipende dalla natura stessa dell'operazione.
il processore preleva un'istruzione dalla memoria. il PC viene incrementato in modo da puntare alla successiva istruzione. l'istruzione appena prelevata viene inserita nell'Instruction Register. L'istruzione contiene dei bit che indicano che tipo di operazione il processore deve eseguire, viene quindi interpretata ed eseguita. a seconda di questi bit esistono diversi tipi di istruzioni.

Diapositiva 13
1I940 è la prima istruzione e viene messa nell'IR. l'opcode 1 indica che si deve caricare nell'accumulatore il valore presente all'indirizzo 940. nell'accumulatore va il valore 0003 e il PC si sposta di una posizione. la prossima istruzione 5I941 viene caricata nell'IR. L'opcode 5 indica che si aggiunga al contenuto dell'accumulatore il contenuto dell'indirizzo specificato nell'istruzione che è in questo caso 941. quindi nell'accumulatore avviene la somma 0003+0002. Il PC si sposta e la prossima istruzione è 2I941. 2 indica il salvataggio in memoria del contenuto dell'accumulatore quindi nella cella di indirizzo 941 si scrive il valore 0005.

potremmo avere lo stesso tipo di procedura in cui però invece che con la memoria il processore scambia dati con un dispositivo esterno. in generale quando dei dati devono essere trasferito da un dispositivo esterno alla memoria o viceversa questi dati passano attraverso il processore che di occupa di copiarli da una parte all'altra. talvolta però è preferibile che i trasferimenti avvengano direttamente in memoria senza dover coinvolgere il processore che può così occuparsi si svolgere altre operazioni. si dice quindi che in questi casi i dispositivi esterni hanno un accesso diretto alla memoria, cioè è attivo il DMA (Direct Memory Access)

Diapositiva 14
Quando all'interno di un programma in esecuzione avviene la chiamata ad un'altra procedura allora il programma corrente viene temporaneamente sospeso e si attiva la procedura chiamata. alla fine dell'esecuzione della procedura si torna allo stato immediatamente precedente la sua attivazione e si continua l'esecuzione del programma chiamante. il procedimento può essere annidato senza problemi.

Diapositiva 15
Per fare in modo che si possa tornare al momento subito precedente la chiamata della procedura è necessario salvare lo stato in cui ci si trova, in particolare l'indirizzo di memoria dell'istruzione successiva. nello stack viene inserita, al momento della chiamata ma prima di attivare la sottoprocedura, l'istruzione seguente alla chiamata che sarà quella da eseguire finita l'esecuzione della procedura. La politica LIFO dello stack fa sì che l'annidamento delle procedure funzioni senza problemi.

Diapositiva 17
tipicamente le periferiche sono più lente della cpu, quindi il processore controlla regolarmente se la periferica ha completato l'esecuzione delle sue operazioni; per questo si parla di busy waiting, perchè il processore resta impegnato e non può fare altro mentre aspetta. questo tipo di tecnica quindi è tanto più inefficiente quanto più la periferica è lenta e viene infatti utilizzata solo con periferiche molto veloci come ad esempio le schede grafiche.

Diapositiva 18
gli interrupt nascono per ovviare al problema del busy waiting. il processore si mette a fare altro e quando la periferica è pronta lo interrompe.
un interrupt è una chiamata di procedura asincrona generata dalla circuiteria di un dispositivo di I/O. è detta asincrona perchè non è noto a priori quanto tale chiamata arriverà.

Diapositiva 19
Program: eventi interni al processore stesso non esterni ma che il processore non sa come gestire. in genere in questi casi il processo viene terminato/ucciso.

Diapositiva 20
nei moderni SO esista un controllore che gestisce gli interrupt che è esterno ala cpu, può essere integrato ma è cmq logicamente diverso.
La chiamata di procedura asincrona è effettuata dall'handler che sa chi ha scatenato l'interrupt. deve essere efficace ma breve, veloce.

Diapositiva 22
l'istruzione macchina in esecuzione quando arriva l'interrupt viene cmq portata a termine perchè non si potrebbe ripristinare lo stato intermedio tra la fase di fetch e quella di esecuzione in seguito al soddisfacimento dell'interrupt.

Diapositiva 23
il programma interrotto deve essere salvato per poi essere ripristinato correttamente alla fine dell'esecuzione dell'interrupt. questo programma non deve rendersi conto della presenza dell'interrupt non deve subire variazioni. es il program counter viene salvato a livello di hardware ma il resto via software in uno stack.

Diapositiva 25
posso servirlo subito dopo aver finito il corrente.
oppure posso disabilitare la possibilità di avere interrupt mentre se ne sta servendo uno.

Diapositiva 26
oppure servo il nuovo interrupt subito interrompendo quello che stavo eseguendo se questo ha una priorità maggiore. si dice che gli interrupt sono annidati

COMPUTER SYSTEMS OVERVIEW

Diapositiva 28
senza interrupt la CPU finisce prima quello che sta facendo poi esegue la richiesta dell'I/O. con gli interrupt invece esegue subito la richiesta e poi continua quello che stava facendo.

Diapositiva 29
il meccanismo degli interrupt elimina i tempi morti dovuti al busy waiting

Diapositiva 31
per facilitare le cose si crea una circuiteria apposita per il trasferimento
 di dati diretto tra le I/O e la memoria.
così nel frattempo la cpu può fare qualcos'altro.
in teoria però le fetch le deve fare cmq la cpu... per risolvere questo problema si usa la cache

Diapositiva 32
in generale si usa la RAM, la memoria mentre il processore usa la cache così non c'è conflitto.
la memoria più è veloce più costa, se costa meno ne metto di più... equilibrio tra velocità e quantità.
Reg. pochi byte
Cache qualche mega
Memory intorno al giga o poco più
HDD centinaia di giga
località temporale: se hai utilizzato un dato molto probabilmente lo riutilizzerai tra poco. è questo che giustifica l'uso della cache!

Diapositiva 34
il disco è molto più lento della ram di diversi ordini di grandezza. anche le innovazioni tecnologiche non sono state grandi come quelle per le ram. quindi il sistema operativo deve poter leggere il meno possibile. sempre per il principio della località mantengo in memoria le ultime cose utilizzate. questa è la disk cache. è una decisione del SO

Diapositiva 35
parti della ram memorizzate nella cache. è una decisione dell'hardware, non è gestita direttamente dal SO. a parte casi particolari con cache particolari...

Diapositiva 36
la multiprogrammazione diminuisce il principio di località e quindi rende meno efficiente il meccanismo della cache.