LABORATORIO DI ARCHITETTURA DEI CALCOLATORI

lezione n° 18

Prof. Rosario Cerbone

rosario.cerbone@libero.it

http://digilander.libero.it/rosario.cerbone

a.a. 2006-2007

Il BUS del calcolatore

- Il calcolatore elettronico è un insieme di unità funzionali:
  - unità centrale di elaborazione (CPU), o processore
  - unità funzionali di memoria, o banchi di memoria
  - unità funzionali di interfacciamento alle periferiche
- Le unità sono interconnesse tramite un organo di collegamento: il BUS
- Lo scopo del BUS è quello di effettuare tutti i trasferimenti di informazioni tra le unità funzionali del calcolatore
- Le informazioni da trasferire sono costituite da istruzioni e dati
- Ogni trasferimento costituisce un'operazione sul bus

BUS interni ed esterni

- I BUS del calcolatore si distinguono in:
- **BUS interni**, confinati all'interno di una singola unità funzionale, e che collegano i blocchi funzionali contenuti nell'unità
- **BUS esterni**, che si estendono all'esterno dell'unità funzionale e che la collegano alle altre unità funzionali. Sono solitamente standardizzati

Controllo del BUS

- In ogni istante, una sola unità funzionale possiede il controllo del BUS, cioè decide quali operazioni di trasferimento eseguire
- Generalmente la CPU possiede il controllo del BUS, ma può anche cedere temporaneamente questo ruolo ad altre unità funzionali
- L'unità funzionale che detiene il controllo del BUS si chiama MASTER:
  - decide quale operazione eseguire, lettura oppure scrittura, e in quale istante di tempo
  - decide qual è l'unità funzionale da cui leggere oppure su cui scrivere
- Le rimanenti unità funzionali, che non detengono il controllo del BUS, si chiamano SLAVE

Ruoli e modalità di connessione al bus

Master	Slave	Esempio
CPU	Memoria	Prelievo istruzioni e lettura/scrittura dati
CPU	Unità di I/O	Ricezione/invio dati da/a un'unità di I/O
CPU	Coprocessore	La CPU dà istruzioni al coprocessore
I/O	Memoria	Accesso diretto alla memoria (DMA)*
Coprocessore	CPU	Il coprocessore legge operandi dalla CPU

- Per potersi collegare al BUS, le unità MASTER e SLAVE sono dotate di un componente (o blocco funzionale) di interfaccia, chiamato BUS DRIVER
- Il BUS driver è in grado di:
  - collegarsi e scollegarsi elettricamente al o dal BUS (p. es. tramite la tecnica di alta impedenza)
  - amplificare opportunamente i segnali da trasmettere/ricevere sul/dal BUS

Struttura del BUS

- Un generico BUS esterno di calcolatore ha una struttura simile alla piedinatura della CPU, cui va collegato.
- È diviso in tre componenti:
  - **BUS degli indirizzi** (address BUS)
  - **BUS dei dati** (data BUS)
  - **BUS di controllo** (control BUS)
- La corrispondenza con i piedini della CPU non è però sempre uno-a-uno
- Versioni del bus:
  - molti tipi di BUS sono stati progressivamente potenziati con:
    - aumento della larghezza del BUS indirizzi
    - aumento della larghezza del BUS dati
    - qualche aggiunta di segnali al BUS di controllo

Funzionamento del BUS

- Le attività del calcolatore si sviluppano per cicli di BUS: in ogni ciclo avviene un'operazione.
- Le operazioni sono governate dal MASTER che detiene il controllo del BUS. Gli SLAVE non possono dare inizio a un'operazione in modo autonomo.
- Le operazioni sono trasferimenti di informazioni tra MASTER e SLAVE e può essere necessario specificare se il trasferimento è relativo a memoria o a interfaccia di I/O.
  - Operazione di lettura: un dato viene trasferito da uno SLAVE al MASTER. Lo SLAVE è la sorgente dell'informazione.
  - Operazione di scrittura: un dato viene trasferito dal MASTER a uno SLAVE. Il MASTER è la sorgente dell'informazione.
- La direzione del trasferimento (lettura o scrittura) è relativa al MASTER che detiene (in quel momento) il controllo del BUS

Unità funzionali e collegamenti al BUS

- Poiché è il MASTER a dare inizio a un'operazione di lettura o scrittura, solo il MASTER può immettere un indirizzo sul BUS degli indirizzi
- Gli SLAVE non possono generare indirizzi e corrispondono ai diversi indirizzi, secondo la mappa di indirizzamento del calcolatore
- I collegamenti delle unità funzionali al BUS dati variano, a seconda che l'unità funzionale (MASTER o SLAVE) sia in lettura e scrittura, sola lettura o sola scrittura
- I collegamenti delle unità funzionali al BUS di controllo sono quelli meno regolari e classificabili e dipendono dalle funzioni dell'unità considerata

Temporizzazione del BUS

- Per realizzare la scansione dei cicli di BUS esistono due metodi fondamentali:
- **BUS sincrono**: funzionamento governato da un segnale di clock
- **BUS asincrono**: funzionamento governato solo dall'interazione tra master e slave

BUS sincrono

- Il BUS di controllo contiene una linea per il **segnale di clock**, a frequenza prestabilita.
- Il clock viene distribuito a tutte le unità funzionali collegate al BUS.
- Il segnale di clock scandisce le varie transizioni di segnale e il passaggio da un ciclo di BUS al ciclo successivo
- Tutte le unità sanno sempre quando si passa al ciclo successivo
- La durata del ciclo di bus dipende dai dispositivi e dai tempi di latenza/propagazione

BUS asincrono

- Non esiste alcun segnale di clock comune alle unità funzionali collegate al BUS del calcolatore
- Le transizioni di segnale e il passaggio da un ciclo di BUS al ciclo successivo non sono sincronizzati
- Le unità funzionali osservano il BUS di controllo: quando avviene una transizione di segnale significa che si verifica un avanzamento dell'operazione
- Il bus è dotato di opportuni segnali di controllo (di cui vengono osservate le transizioni) che realizzano un protocollo di sincronizzazione a segnale (ad es. **full-handshake**)

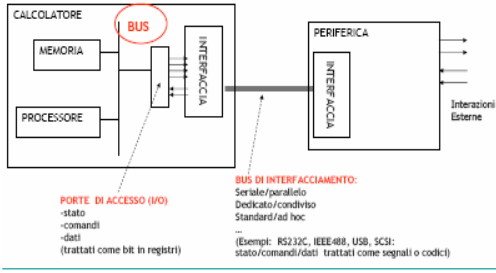
Confronti

- **Il BUS sincrono**
- vantaggi:
  - semplicità di controllo e maggiori frequenze di trasferimento
- svantaggio:
  - "sprechi" di tempo (ogni operazione si deve svolgere in un numero intero di cicli di clock):
    - quando un'operazione si potrebbe completare in meno di un ciclo di clock
    - quando un'operazione richiede un numero frazionario di cicli di clock
    - Necessità di progettare il circuito di distribuzione del clock in modo accurato
- **Il BUS asincrono**
- vantaggi:
  - maggior efficienza nell'uso dei cicli (l'operazione si completa esattamente nel tempo necessario)
  - Semplicità nella realizzazione delle interfacce
- svantaggi:
  - frequenza di trasferimento limitata dalla necessità per ogni dato trasferito di uno scambio di due segnali di controllo tra master e slave

Arbitraggio del bus: cambio di MASTER

- Normalmente la CPU ha il ruolo di MASTER tra le unità funzionali.
- Tuttavia in determinate circostanze anche altre unità funzionali possono assumere temporaneamente il ruolo di MASTER, per scopi particolari:
- Unità di I/O: possono diventare MASTER per trasferire dati direttamente con la memoria, senza bisogno della CPU (DMA)
- Coprocessore: può diventare MASTER per prelevare operandi dalla memoria
- Il BUS del calcolatore può avere, in ogni istante, un unico MASTER.
- In caso di cessione del ruolo di MASTER da un'unità funzionale a un'altra, occorre dunque un meccanismo di ARBITRAGGIO DEL BUS, che ne regoli l'utilizzo da parte delle unità funzionali

Collegamento di una periferica al calcolatore



Collegamento di una periferica al calcolatore

- I dispositivi connessi a un bus devono avere:
  - Un'interfaccia di accesso al bus
  - Un protocollo per usarlo correttamente
- L'indirizzamento dei diversi dispositivi può essere:
- integrato (memory mapped I/O). Parte dello spazio di indirizzi di memoria è riservato ai porti di I/O. Non ci sono istruzioni dedicate
- separato per ogni dispositivo (I/O isolato). Lo spazio degli indirizzi per i porti di I/O è separato da quello per la memoria. Uso di istruzioni dedicate (IN e OUT)

Porte di Porte di accesso (I/O) e bus

- Porte di accesso: registri indirizzabili e leggibili e/o scrivibili dal processore (accessibili anche da interfaccia lato periferica) e circuiti ausiliari:
  - Registro(i) dati: dati forniti dalla periferica alla CPU (ingresso) oppure forniti dalla CPU alla periferica (uscita)
  - Registro comandi: informazioni che determinano il modo di funzionamento della periferica (generate dalla CPU e utilizzate dalla periferica)
  - Registro di stato: informazioni relative allo stato di funzionamento della periferica (generate dalla periferica e utilizzate dalla CPU)
- Bus
  - Bus indirizzi: indirizzi dei registri, forniti dal master del bus (in generale la CPU)
  - Bus di controllo: segnali per specificare l'accesso a porte di ingresso e uscita

Buffer tri-state

- E' il circuito elementare modellabile come un contatto a tre posizioni:
  - in stato di bassa impedenza consente di avere in uscita o il **livello alto** (1) o il **livello basso** (0)
  - in stato di alta impedenza (Z) **isola elettricamente** l'uscita
- L'uscita tri-state viene gestita da un apposito ingresso di controllo (**O**uput **E**nable) che, se non attivo, forza lo stato di alta impedenza

Acquisizione dato

- La acquisizione del dato avviene:
  - a controllo di programma;
  - a interruzione;
  - ad accesso diretto alla memoria (DMA: direct memory access).

A controllo di programma

- La sincronizzazione e il trasferimento vengono eseguiti dal programma di gestione della periferica
- Il programma di gestione della periferica legge il registro di stato dell'interfaccia della periferica (IN registro di stato): se lo stato è "periferica non pronta" il programma torna a leggere il registro di stato; altrimenti esegue il trasferimento del dato tramite:
  - IN registro dati se la periferica è di ingresso (ad es. tastiera)
  - OUT registro dati se la periferica è di uscita (ad es. stampante)

A interruzione

- La parte di programma che legge la porta **NON** è nel programma ma è silente in memoria in una locazione convenuta.
- Quando l'interfaccia della periferica porta il dato alla porta di ingresso, con un segnale allerta il processore.
- Il processore interrompe l'esecuzione del programma in corso e salta automaticamente a eseguire la parte di programma che legge la porta. La lettura avviene come nel caso precedente.
- Al termine il processore riprende il programma interrotto.
- In pratica, la periferica ha deciso **quando** l'istruzione di lettura della porta deve essere eseguita.

DMA

- Quando l'interfaccia della periferica porta il dato alla porta di ingresso, manda un segnale al processore, imponendogli di lasciare libero il bus.
- Appositi circuiti generano un ciclo di bus che forza l'attivazione della porta, genera l'indirizzo in memoria dove deve finire il dato, comanda la memoria alla scrittura.
- Intanto, il processore non utilizza il bus.
- Terminato il ciclo, l'interfaccia della periferica manda un altro segnale al processore, lasciandolo libero di proseguire.
- In pratica, alcuni circuiti di I/O hanno scritto il dato in memoria, pochi nanosecondi dopo il suo arrivo.

A controllo da programma (ingresso)

- Il programma di gestione deve:
- leggere le porte a cui si presentano i bit di stato ed i dati dell'interfaccia della periferica;
- scrivere le porte che forniscono bit di comando all'interfaccia di periferica.
- Il programma che acquisisce il dato, lo memorizza e lo usa è unico.

La tastiera: controllo da programma

- Le operazioni da compiere per acquisire un dato da tastiera sono:
- input dello stato della tastiera in registro R0
- se il bit di codice arrivato è 0, ripeti lettura finché non lo trovi a 1
- codice di tasto è arrivato: viene letto dalla porta e scritto in R1
- copia codice di tasto in cella di memoria
- ... Istruzioni successive (hanno a disposizione il codice di tasto arrivato)

A controllo di programma (ingresso)

- Ideale se:
  - il ciclo di istruzioni per la lettura è inserito nel programma utente;
  - quando il programma utente è in attesa che venga premuto un tasto, non ha null'altro da fare;
  - non c'è nessun altro programma utente da eseguire e, quindi, la soluzione di ripetere all'infinito il ciclo di istruzioni per la lettura non ha controindicazioni.

A controllo di programma (uscita)

- Stampa di un carattere:
- L'interfaccia della stampante tramite un bit comunica che è pronta ad accettare un nuovo codice di carattere da stampare.
- Occorrono circuiti analoghi a quelli visti per la tastiera:
- una porta di ingresso, per leggere il bit di pronta ad accettare il dato
- una porta di uscita, su cui il programma scrive il codice del carattere
- il circuito che genera il bit di pronto ad accettare (analogo a quello che generava il bit di codice arrivato).

A controllo di programma

- Per stampare un dato:
- input dello stato della stampante in registro R0
- se il bit di pronta ad accettare è 0, ripeti lettura finché non lo trovi a 1
- copia codice di carattere da cella di memoria in R1
- scrivi codice di carattere sulla porta di uscita

Esempio 1

- Driver per la lettura di un carattere da una periferica di input.
- Registri:
  1. Registro di controllo RC
  2. Registro dato RD
  3. Registro di stato RS
- Protocollo:
  - La CPU porta alto il bit START del registro di controllo (es. bit 0)
  - La periferica risponde ponendo alto il bit READY (es. 5) del registro di stato
  - La CPU legge il dato dal registro dato e lo salva in memoria.

```
start   move.b   #1,RC
wait    btst     #5,RS
        beq      wait
        move.b   RD,MEM
```

Esempio 2

- Driver per la scrittura di un carattere su una periferica di output.
- Registri:
  1. Registro di controllo RC
  2. Registro dato RD
  3. Registro di stato RS
- Protocollo:
  - La CPU controlla il bit READY del registro di stato (es. bit 7). Quando il bit è zero, la periferica è pronta per ricevere il dato.
  - La CPU scrive il dato nel registro dato.
  - La CPU pone alto il bit START (es. bit 1) nel registro controllo. Il dato viene prelevato dalla periferica di output.

```
start   btst     #7,RS
        bne      start
        move.b   D0,RD
        move.b   #2,RC
```

Esempio 3
Il dispositivo videotastiera TERMINAL

- CHIP Name: TERMINAL
- Address 1: 00002000. Address 2: 00002001.

- Address1: Indirizzo registro dati (tastiera in scrittura, video in lettura)
- Address2: Indirizzo registro stato-controllo

Esempio 3
Il dispositivo videotastiera TERMINAL

- Significato dei bit nel registro di Controllo e Stato del dispositivo TERMINAL
- bit 0) abilita interruzione su "Buffer full";
- bit 1) abilita interruzione sull'"ENTER";
- bit 2) cancella video;
- bit 3) pulisci buffer di tastiera;
- bit 4) abilita tastiera;
- bit 5) abilita eco;
- bit 6) stato di "Buffer full";
- bit 7) stato di ENTER inviato.
- Il contenuto del registro CNTRL può essere modificato o da programma o utilizzando il comando Modifica Valore del menù Device.
- Il comando Mostra Registri apre una finestra di dialogo in cui appare il registro CNTRL, il numero di caratteri presenti nel buffer di tastiera e la posizione nel buffer dell'ultimo carattere letto dal processore

Esempio 3
Il dispositivo videotastiera TERMINAL

*Il programma Terminal utilizza l'omonima configurazione e consente di gestire una interazione
*tastiera-video

```
        ORG $8200

TERD    EQU $2000
TERC    EQU $2001
MEM     EQU $8000

QUEST   DC.B    'What is your name-Come ti chiami ? : ',0
ANSW    DC.B    'Hallo-Ciao ',0

BEGIN   MOVEA.L #TERD,A0
        MOVEA.L #TERC,A2
        MOVE.B  #$30,(A2)
        LEA.L   QUEST,A1
OUTPUT  MOVE.B  (A1)+,D0
        CMP.B   #0,D0
        BEQ     CONT
        MOVE.B  D0,(A0)
```

bit 0) abilita interruzione su "Buffer full";

bit 1) abilita interruzione sull'"ENTER";

bit 2) cancella video;

bit 3) pulisci buffer di tastiera;

bit 4) abilita tastiera;

bit 5) abilita eco;

bit 6) stato di "Buffer full";

bit 7) stato di ENTER inviato

Esempio 3
Il dispositivo videotastiera TERMINAL

```
        BRA     OUTPUT
CONT    MOVE.B  (A2),D1
        AND.B   #$80,D1
        BEQ     CONT
        MOVEA.L #MEM,A1
IN1     MOVE.B  (A0),D0
        MOVE.B  D0,(A1)+
        CMP.B   #13,D0
        BNE     IN1
        LEA.L   ANSW,A1
OUT1    MOVE.B  (A1)+,D0
        MP.B    #0,D0
        BEQ     CONT1
        MOVE.B  D0,(A0)
        BRA     OUT1
CONT1   MOVEA.L #MEM,A1
OUT2    MOVE.B  (A1)+,D0
        MOVE.B  D0,(A0)
        CMP.B   #13,D0
        BNE     OUT2
        STOP    #$2700
        END     BEGIN
```

bit 0) abilita interruzione su "Buffer full";

bit 1) abilita interruzione sull'"ENTER";

bit 2) cancella video;

bit 3) pulisci buffer di tastiera;

bit 4) abilita tastiera;

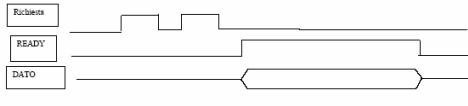
bit 5) abilita eco;

bit 6) stato di "Buffer full";

bit 7) stato di ENTER inviato

Esercizio 18.1

- Scrivere un sottoprogramma che acquisisca da una periferica di input una linea di caratteri con al più N dati e li memorizzi in memoria a partire da un indirizzo assegnato.
- Il sottoprogramma accetta come parametri di ingresso il numero max. di caratteri (N), indirizzo iniziale dell'area di memoria e un carattere terminatore. Il driver termina la lettura o se viene letto il carattere terminale (passato come parametro) o se sono stati acquisiti N caratteri. Il driver restituisce al programma chiamante il numero di caratteri effettivamente letti. Il protocollo utilizzato per la lettura del singolo carattere è quello specificato dal diagramma riportato in figura. Il registro di controllo e stato è mappato in memoria all'indirizzo $8500 e il registro dato all'indirizzo $8501. Il bit di richiesta e il bit ready sono rispettivamente il bit 2 e il bit 5 del registro di controllo e stato.



Esercizio 18.1

```
                org $8000

RCS     equ     $8500
RDATI   equ     $8501
fine    equ     '*'
N       equ     10
mem     ds.b    10
letti   ds.b    1

start   movem.l d0/d1/d2/a0, -(SP)
        move.w  #N,d0
        move.b  #fine,d1
        move.l  #mem,a0
        jsr     leggi
        move.b  d2,letti
        movem   (SP)+, d0/d1/d2/a0
        stop    #$2700
```

Esercizio 18.1

```
leggi   move.l   d4,-(SP)
        clr.b    d2
        add.w    #-1,d0
loop    btst.b   #5,rcs    attesa su ready
        bne      loop
        bset.b   #2,RCS          avvia il protocollo
        bclr.b   #2,RCS
        bset.b   #2,RCS
        bclr.b   #2,RCS
wait    btst.b   #5,RCS          attesa attiva su ready
        beq      wait
        move.b   RDATI,d4                lettura del carattere
        move.b   d4,(a0)+
        add.b    #1,d2
        cmp.b    d4,d1
        beq      return
        dbeq     d0,loop
return  movem.l  (SP)+,d4
        rts
end     start
```

Esercizio 18.2

- Scrivere un sottoprogramma che trasferisca ad una periferica di output una sequenza di N cifre (intero compreso tra 0 e 9) memorizzati a partire da un indirizzo di memoria assegnato.
- Il sottoprogramma accetta come parametri di ingresso il numero di cifre (N), indirizzo iniziale dell'area di memoria. Il protocollo utilizzato per la visualizzazione del singolo carattere è quello specificato dal diagramma riportato in figura. Il registro di controllo è mappato in memoria all'indirizzo $8700, il registro di stato all'indirizzo $8702 e il registro dato all'indirizzo $8701. Il bit Go è il bit 1 del registro di controllo mentre il bit ready è il bit 6 del registro di stato.
- N.B. Il bit ready si abbassa nel momento in cui il processore trasferisce un nuovo carattere da visualizzare nel registro dati.