SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Interconnessione tra registri

Prof. Daniele Gorla

Interconnessione fra registri

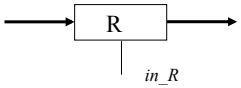
SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Per venir elaborate o memorizzate, le informazioni devono essere trasferite da un registro all'altro all'interno di un elaboratore.

Il trasferimento di informazioni tra registri viene realizzato tramite **reti di interconnessione.**

N.B.: Se i registri sono a scaricamento parallelo (PIPO/SIPO), l'informazione in realtà viene *copiata* dal registro sorgente a quello destinazione.

Non ci interessa la modalità di caricamento/scaricamento; assumiamo che un registro è



dove le linee spesse denotano *n* bit (se il registro è formato da *n* FF) e la linea *in_R* abilita il registro alla scrittura (come la linea *load* per il PIPO)

4 tipi di interconnessione

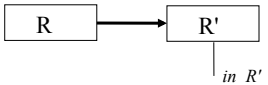
SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

	destinazione prefissata	**destinazione variabile**
sorgente prefissata	(punto-punto) tramite porte logiche o buffer tri-state	con decodificatore
sorgente variabile	con multiplexer	tramite mesh o bus

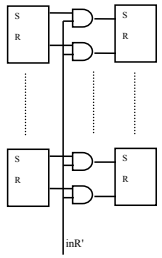
Sorgente e destinazione prefissata: porte logiche

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Per trasferire il contenuto di R in R', basta collegare l'output di R all'input di R' e settare *in_R'* ogni volta che si vuole effettuare il trasferimento:



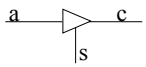
Ciò in realtà rappresenta in maniera schematica la seguente interconnessione, assumendo che i registri siano sequenze di FF di tipo SR:



Sorgente e destinazione prefissata: Buffer tri-states

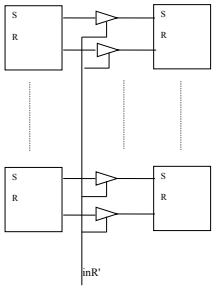
SAPIENZA Università di Roma Dipartimento di Informatica

Il buffer tri-states è un interruttore elettronico, schematicamente rappresentato come



che può assumere tre stati (da cui il nome):
- circuito aperto: s = 0;
- circuito chiuso e uscita 0: se s=1 e a=0;
- circuito chiuso e uscita 1: se s=1 e a=1.

Invece che usare porte AND, nella realizzazione precedente possiamo usare buffer tri-states con segnale di controllo il bit *in_R'*:



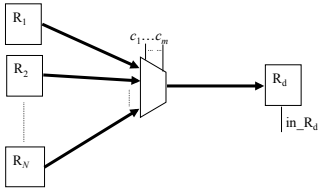
5

Sorgente variabile e destinazione prefissata: interconnessione con multiplexer

SAPIENZA Università di Roma Dipartimento di Informatica

Il registro sorgente può essere un qualsiasi registro R_i di un insieme di N registri, mentre il registro destinazione R_d è fissato.

Questo può essere fatto usando un MUX le cui entrate sono gli output degli N registri sorgente e la cui uscita viene data in ingresso al registro destinazione:



I segnali di selezione del multiplexer $c_1,\ldots c_m$ sono $m = \lceil \log_2 N \rceil$ e forniscono la codifica binaria dell'indice i del registro R_i il cui contenuto deve essere copiato in R_d.

6

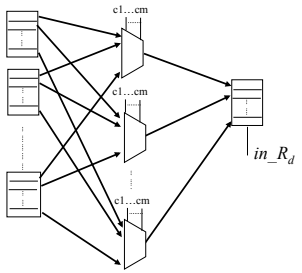
Interconnessione tramite MUX: dettaglio (a linee singole)

SAPIENZA Università di Roma Dipartimento di Informatica

Nella rappresentazione precedente, le frecce marcate rappresentano n linee; quindi, anche i MUX sono in realtà n!!

- Il **primo** FF di ogni registro sorgente è connesso con il **primo** MUX, la cui uscita va al **primo** FF di R_d;
- il **secondo** FF di ogni registro sorgente è connesso con il **secondo** MUX, la cui uscita va al **secondo** FF di R_d;
- …

Le linee di controllo sono le stesse per tutti i MUX.



7

Sorgente prefissata e destinazione variabile: interconnessione con decodificatore

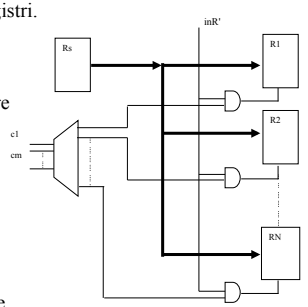
SAPIENZA Università di Roma Dipartimento di Informatica

Il registro sorgente R_s è fissato, mentre il registro destinazione può essere un qualsiasi registro R_i di un insieme di N registri.

Questo può essere fatto mandando in input a ogni registro destinazione l'output del registro sorgente e usando un decodificatore per abilitare la scrittura sul registro destinazione desiderato:

Gli ingressi del decodificatore $c_1,\ldots c_m$ sono $m = \lceil \log_2 N \rceil$ e forniscono la codifica binaria dell'indice i del registro R_i dove copiare l'informazione contenuta in R_s.

Le uscite del decodificatore vengono messe in AND con un segnale "globale" di scrittura (*in_R'*) e forniscono i segnali in_R_i per i registri destinazione.



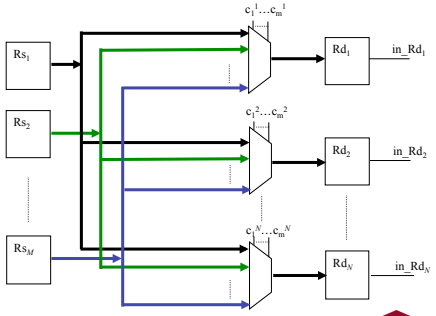
8

Interconnessione molti a molti tramite mesh

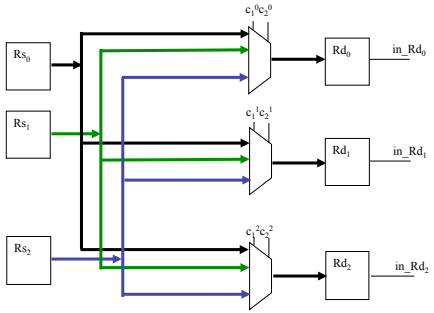
Interconnessione tra M registri sorgente e N registri destinazione

Per realizzare la rete occorrono N (gruppi di n) multiplexer, uno per ogni registro destinazione:

N.B.: Ogni MUX ha le sue proprie linee di controllo (che sono $m = \lceil \log_2 M \rceil$), in modo da poter abilitare il trasferimento da ogni sorgente a ogni destinazione.



Esempio di trasferimento in una mesh



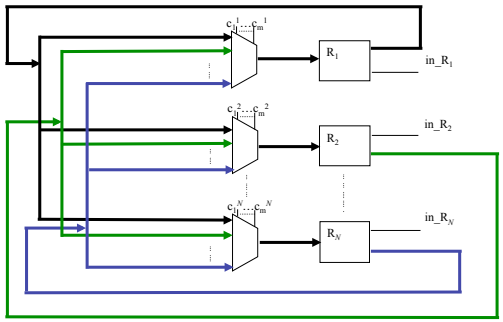
Vogliamo trasferire Rs_i in $Rd_{(i+1) \text{ MOD } 3}$.

$Rs_0 \rightarrow Rd_1$:
$c_1^1 c_2^1 = 00$, $in\_Rd_1 = 1$

$Rs_1 \rightarrow Rd_2$:
$c_1^2 c_2^2 = 01$, $in\_Rd_2 = 1$

$Rs_2 \rightarrow Rd_0$:
$c_1^0 c_2^0 = 10$, $in\_Rd_0 = 1$

Registri sorgente e destinazione non distinti



Mesh: vantaggi e svantaggi

Vantaggio: trasferimenti in parallelo

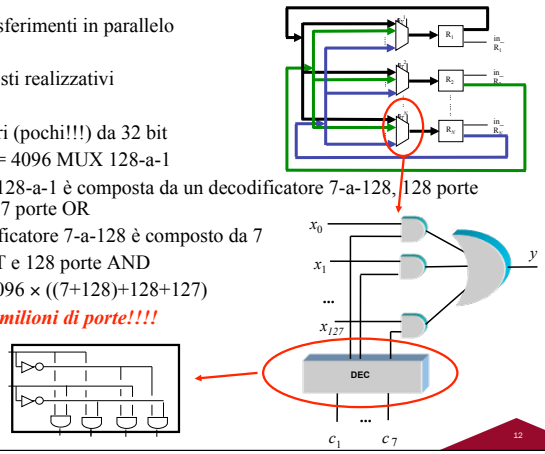
Svantaggio: costi realizzativi

Es.: 128 registri (pochi!!!) da 32 bit

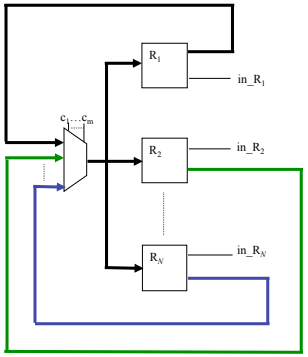
- 128 × 32 = 4096 MUX 128-a-1
- un MUX 128-a-1 è composta da un decodificatore 7-a-128, 128 porte AND e 127 porte OR
- un decodificatore 7-a-128 è composto da 7 porte NOT e 128 porte AND

TOTALE: 4096 × ((7+128)+128+127)

≈ 1,6 milioni di porte!!!!



Una mesh più economica



Svantaggio: parallelismo molto ridotto (solo trasferimenti con stessa sorgente)

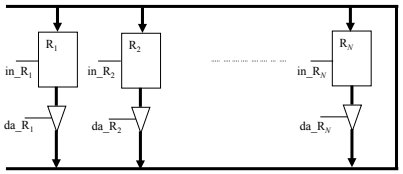
Vantaggio: molto più economica

Es.: 128 registri da 32 bit
- 32 MUX 128-a-1
- un MUX 128-a-1 è composta da un decodificatore 7-a-128 (7 porte NOT e 128 porte AND), 128 porte AND e 127 porte OR

TOTALE: 32 × ((7+128)+128+127)
≈ 12,5 mila porte

Interconnessione molti-a-molti tramite bus

Riducendo il parallelismo, si può realizzare l'interconnessione di prima in maniera ancora più economica tramite un *bus*, cioè un fascio di *n* linee che interconnettono tutti i registri, sia in entrata che in uscita.



N.B.: per evitare conflitti nell'uso del bus, ogni uscita è controllata da un (insieme di *n*) buffer tri-states, ognuno regolato dal segnale di controllo da_R_i che vale 1 se il registro sorgente è R_i. Ovviamente, bisognerà garantire che al più uno di tali segnali valga "1" ad ogni istante.

Es. (128 registri da 32 bit): 128×32 ***≈ 4100 buffer tri-states***

Utilizzo di bus e mesh

2 tipi di memoria:
- *centrale* (nel microprocessore): poche decine di registri veloci
- *di massa*: milioni di registri realizzati con tecnologia più economica

2 tipi di interconnessione:
- mesh tra i registri del microprocessore
- bus per trasferire informazioni dalla memoria di massa a quella centrale

Progettare una rete di interconnessione

Si distinguono 3 parti:
- i **registri** coinvolti ($R_1..R_N$)
- la **rete di interconnessione** che, nel caso più generale, consente di trasferire il contenuto di ogni registro in ogni altro registro
- la **rete combinatoria di controllo** che analizza alcune condizioni interne (contenuto dei registri) o esterne (segnali **S**) e genera gli opportuni comandi per i circuiti che costituiscono la rete di interconnessione