

Domande 1

Note Title

11/07/2007

2) le due μ istruzioni relative a
chiarante e decodifico dell'istruzione
sono le stesse della dispendenza

ch ϕ , IC \rightarrow IND, "read" \rightarrow op, set RDYout, ch1
ch1. (RDYin, or (ESIT0) = ϕ -) nep, ch1
(= 1 ϕ) DATAIN \rightarrow IR
REG [DATAIN.R1] \rightarrow A
REG [DATAIN.R2] \rightarrow B
reset RDYin
DATAIN - COP \rightarrow PC
(= 11) ESIT0 \rightarrow N [29..31]
reset RDYin
trattamento - eccezioni

(si potrebbero considerare anche altre
versioni, per due corelle)

di conseguenza, l'esecuzione delle nuove istruzioni può essere implementata con le due μ-istruzioni che seguono:

$\mu\text{mem} = \emptyset$. $A + B \rightarrow MD$, "read" \rightarrow op, set RDYout

$\mu\text{mem} = 1$. $(RDYin, or(ESTIO), (REG[IR.R3] - DATAIN), int$
 $= \emptyset \rightarrow \text{nop}, \mu\text{mem} = 1$

$(= 11 \rightarrow)$ come in ch1: hot ecc

$(= 1\emptyset 1\emptyset)$ $IC + 1 \rightarrow ic$, $ch \emptyset$

$(= 1\emptyset 11)$ $IC + 1 \rightarrow IC$, $hot - int$

$(= 1\emptyset\emptyset\emptyset)$ $IC + IR.TARGET \rightarrow IC$,
 $ch \emptyset$

$(= 1\emptyset\emptyset 1)$ $IC + IR.TARGET \rightarrow ic$,
 $hot - int$

Valutazione del tempo medio di
elaborazione:

fetch e decodifica: $2\tau + t_A$

esecuzione: $2\tau + t_A$

$$4\tau + 2t_A$$

b) assumendo che $N \geq \emptyset$
(e quindi compilando il for
come se fosse un repeat)

loop: IF-MEM \equiv $R_{base}, R_i, R_k, \text{continue}$
ADD R_i, R_k, R_{temp}
STORE R_{base}, R_i, R_{temp}

continue: INC R_i
IF R_i, R_N, loop

c) la compilazione con l'ASM RISC "tradizionale" sarebbe stata invece la seguente:

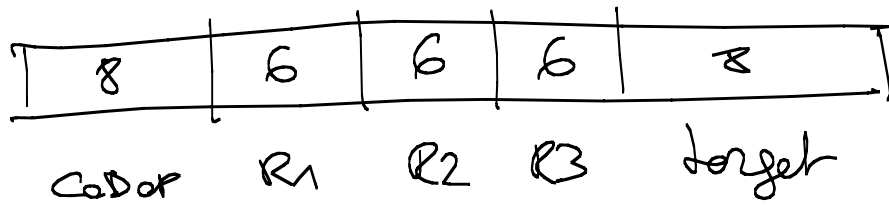
```
loop: LOAD  RbaseX, Ri, Rtemp
      IF=   Rtemp, Rk, continue
      ADD   Ri, Rk, Rtemp
      STORE RbaseX, Ri, Rtemp
continue: INC  Ri
          IF<  Ri, Rk, loop
```

c'è una istruzione in più
in ogni iterazione

Il numero dei fault non cambia:
Si può assumere che 6 istruzioni
(perde) invece che 5 non
cambiano il pattern di accesso
in cache: se i blocchi fossero di 2^3
perde (o più grandi), il loop
potrebbe occupare un solo blocco,
in caso contrario ($< 2^3$, ma
questo non è molto realistico)
occuperebbe 2 blocchi. In ogni
caso si può assumere che non
venge mai scaricato dalla cache.

Il ciclo di clock del processore
è lo stesso nei due casi,
per come è costruito il processore
che interpreta il IFMEM =

d)



para realizarea salti

a 2^7 locuri inainte

o $2^7 - 1$ locuri inainte

la conector (max)

Domande 2

- 1) P va in esecuzione
- m blocco nello 1a send
 - va in esecuzione Q
 - esegue lo receive,
 - sblocca P che va in ready
 - Q m blocco nello 2a receive
 - P esegue lo 2a send
 - sblocca Q che va in ready
 - P m blocco nello send successivo
 - ... etc etc

2) se $K \geq N$

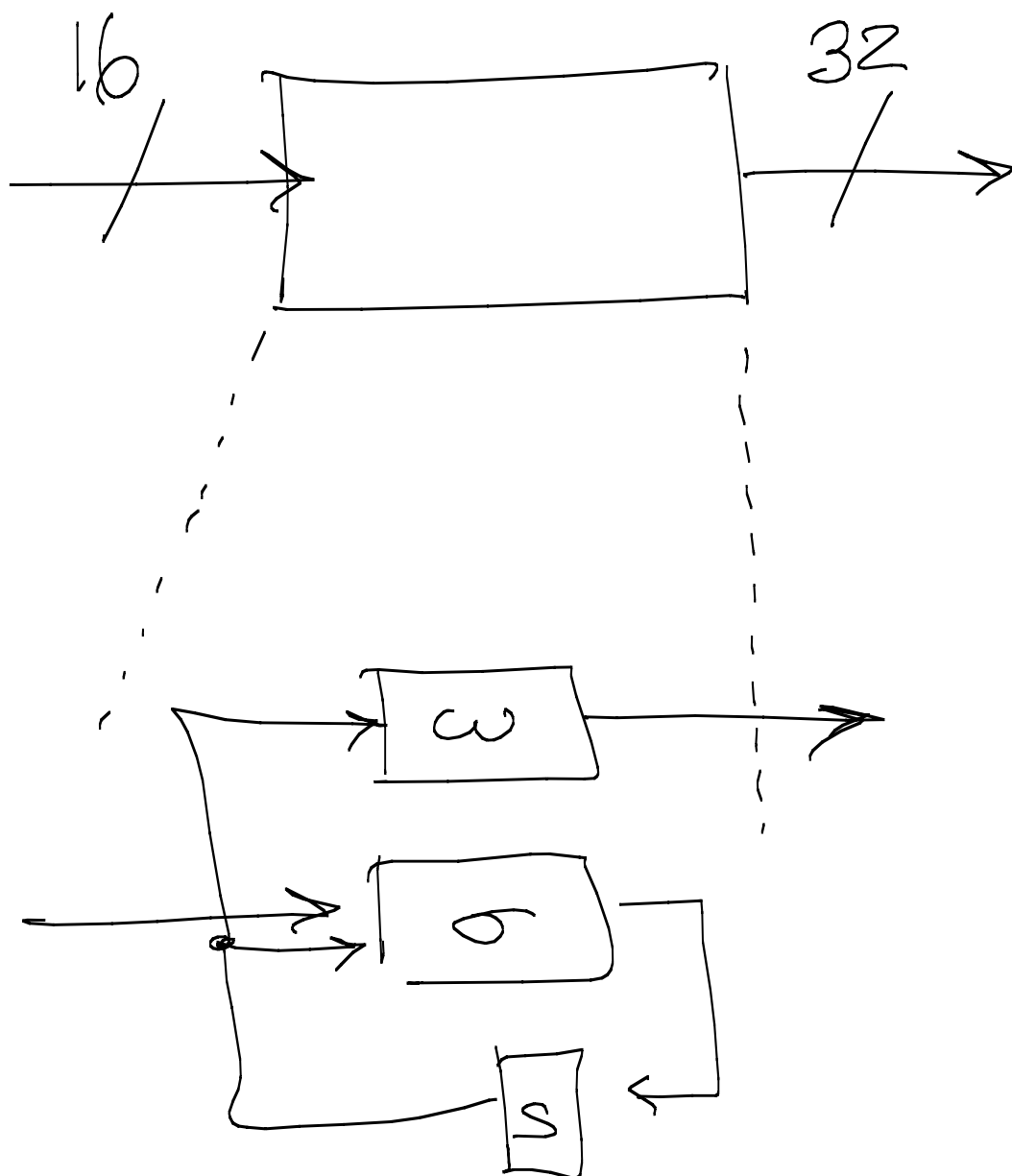
- \mathcal{P} esegue tutte le send (a più blocco nell'ultima se $K = N$)
- \mathcal{Q} va in esecuzione ed esegue tutte le receive.
- se $K = N$ l'ultima receive sblocca \mathcal{P}

Se invece $K < N$

- \mathcal{P} esegue K send poi si blocca

- Q esegue 1 receive e blocca P che va in pronto
- Q esegue le altre $k-1$ receive
- Q è bloccato nella $k+1$ esima receive
- va in esecuzione P esegue 1 send e blocca Q
- ...

Demanda 3



$$\omega = \text{id}$$

$$\sigma = \left[\begin{array}{c} \text{padding a } 32 \\ \text{bit} \\ \text{dell'ingresso} \\ \text{a } 16 \text{ bit} \end{array} \right]$$

