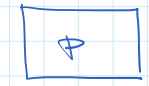
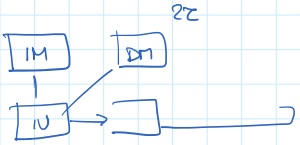


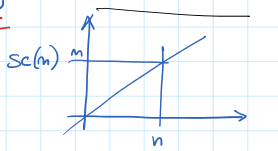
monolitico



pipeline

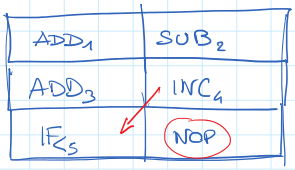
forma replicazione funzionale

"superscalare"

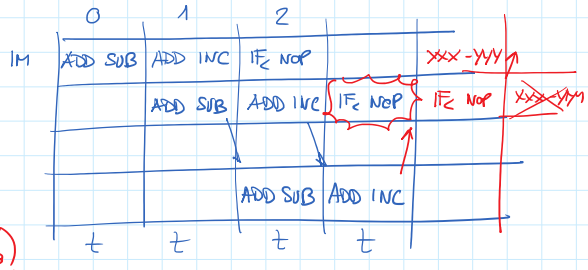


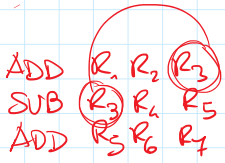
$ADD_1\ R_1\ R_2\ R_3$
$SUB_2\ R_4\ R_5\ R_6$
$ADD_3\ R_3\ R_6\ R_7$
$INC_4\ R_i$
$IF_{<5}\ R_i, R_N, —$

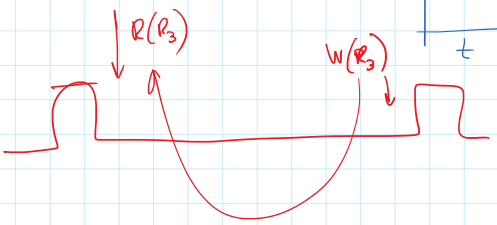
n "slot"

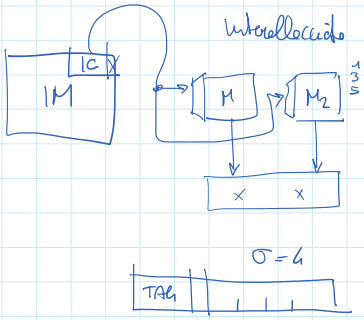


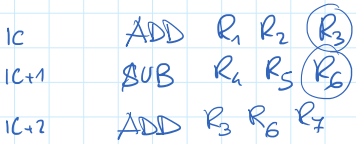
"VLIW" very long instruction WORD

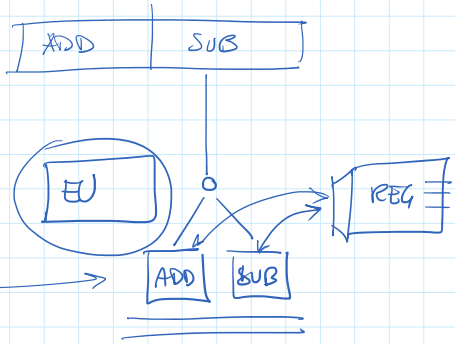


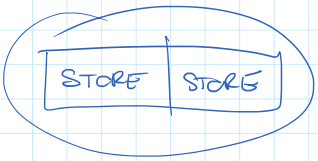


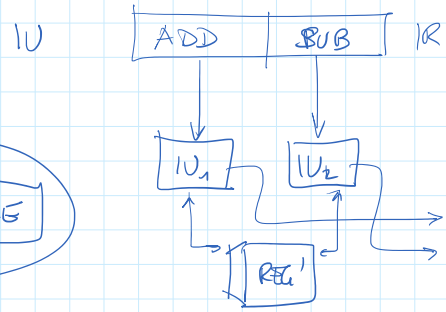




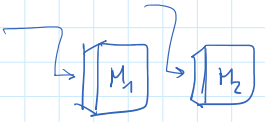




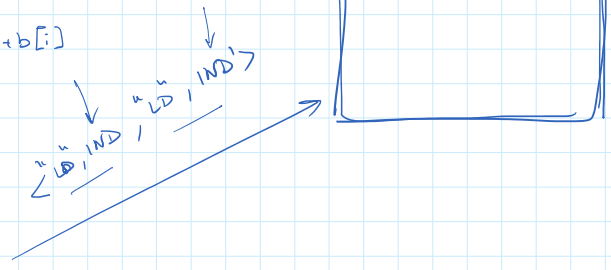


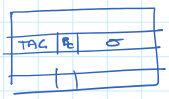


6 accessi in lettura



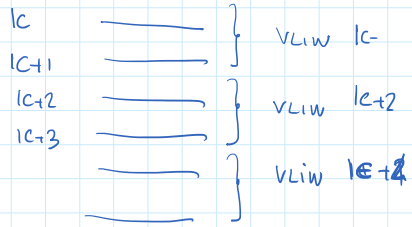
for( )
x[:] = a[:] + b[:]

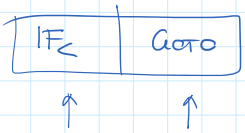




martedì 13 dicembre 2016 11:05

D - RISC





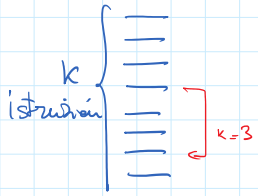
1) non devo avere dip IU-EU fra le due istruzioni
2) non devo avere dip EU-EU " " " "
3) non devo avere 2 istruzioni di salto

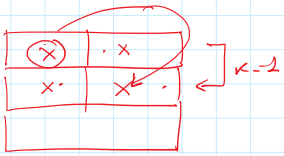
se valgono queste condizioni

⇒ posso raggruppare le ISTR D-RISC a due a due negli slot

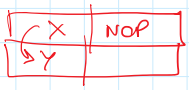
se non valgono : riduco NOP

" $\varepsilon = 1$ non ho NOP nel codice "

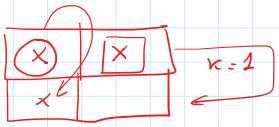




dip K=1



dip K=2



```
for(i=0; i<N; i++) {
    a[i] = b[i] * c;
    sum = sum + a[i];
}
```

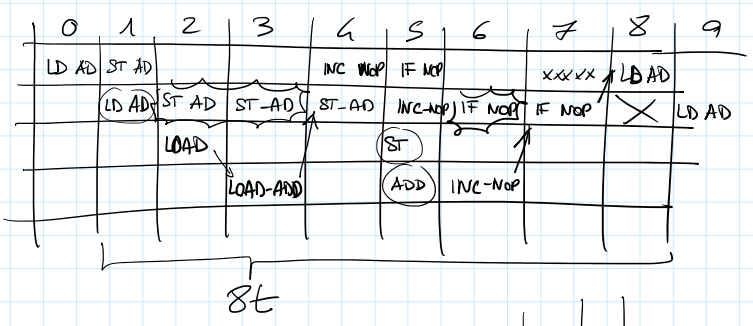
```
loop: LOAD  Rb, Ri, R0
      ADD   Rc, R1, R1
      STORE Ra, Ri, R2
      ADD   R0, Rsum, Rsum
      INC   Ri
      IF<   Ri, RN, loop
```

IU EU

LOAD	ADD
STORE	ADD
INC	NOP
IF<	NOP

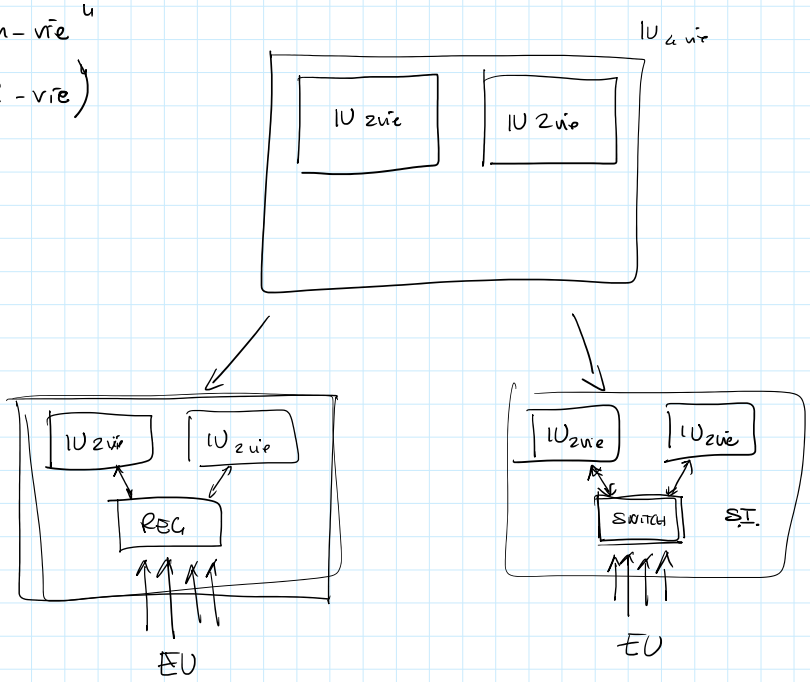
M[Rb+Ri] → R1

<LOAD, ADD, >



$T = \frac{8t}{6}$ $\frac{8t}{4}$

Superscalare a "n-vie" (2-vie)



MULTITHREADING

martedì 13 dicembre 2016 11:06

MONOLITICO → PIPELINE (4 stadi, 1 istr ∀ 2τ) → SUPERSCALARE (PIPELINE) a 2 via (2 istr ∀ 2τ)

→ 1 flusso (flussi) di istruzioni dello stesso processo

→ esecuzione di più flussi di istruzioni (diversi) dello stesso processo

Thread 1 $[0, N/2]$ Thread 2 $[N/2+1, N-1]$

$\forall i \quad x[i] = y[i] + 1 \qquad i \in [0, N-1]$

```
loop: LOAD Ry, Ri, R1
      INC R1
      STORE Rx, Ri, R1
      INC Ri
      IFc Ri, Rbound, loop
```

Stesso codice

3 Modalità di esecuzione

PROC PIPELINE — BLOCKING

PROC PIPELINE — INTERLEAVING

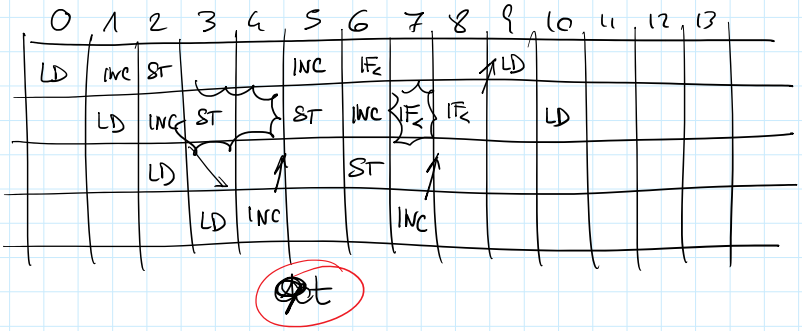
PROC SUPERSCALARE — SIMMETRICA (HYPERTHREADING)

Condividere MV del processo ma hanno un proprio STATO

↙ IC ↘ REG

BLOCKING

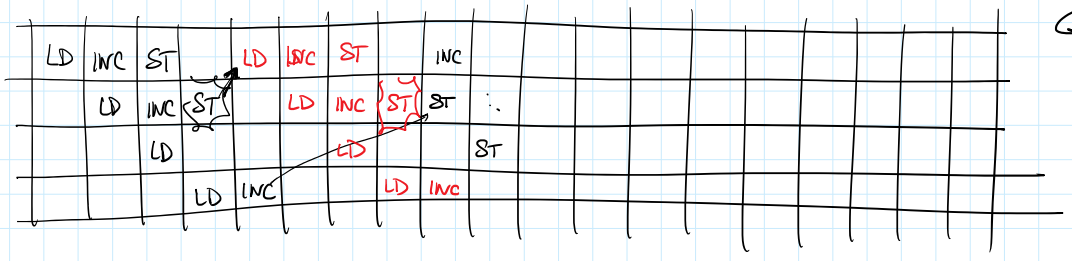
esegue istruzioni del th_i quando trova un bello di dip logica passa ad eseguire le istruzioni del th_{i+1}

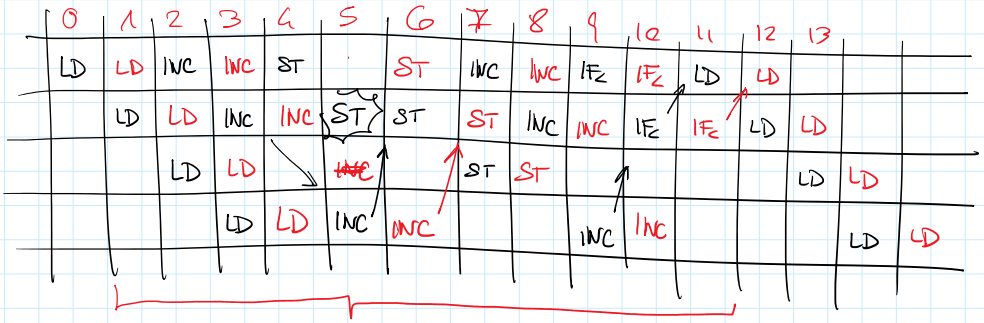


esecuzione "normale" di 1 unica thread

INTERLEAVING

→ esegue 1 istr del th_1 e 1 istr del th_2





11 t

2 iterazioni

SIMMETRICO

utilizzare gli slot di una istruzione VLIW x istruzioni di entrambi i thread

LD	LD
INC	INC
ST	ST
INC	INC
IF_2	IF_2