Domande 4-5 Esonerato Architetture-Con Soluzione

DOMANDE 4

4) Descrivere le differenze tra i bus sincroni e quelli asincroni e descrivere un protocollo asincrono.

Le linee di controllo dei bus sincroni sono dotati di un segnale di sincronizzazione(clock), mentre i bus asincroni non lo sono e la comunicazione avviene tramite un protocollo di handshaking.

I bus sincroni sono molto veloci e non richiedono molta logica, poiché tutti gli eventi sono sincronizzati tramite clock, ma non possono avere lunghezze elevate altrimenti avremmo problemi di clock skew.

I bus asincroni invece possono avere lunghezza elevata e connettere molti dispositivi, ma sono più lenti dei bus sincroni.

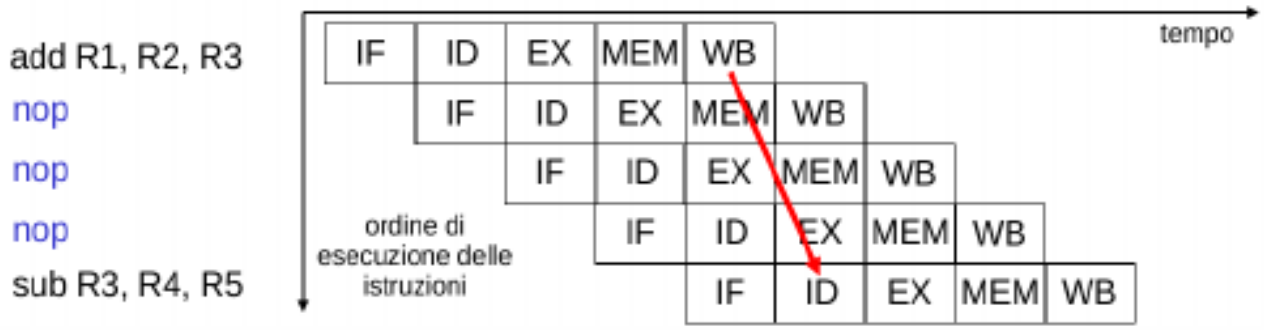
Protocollo di handshaking(sincrono):

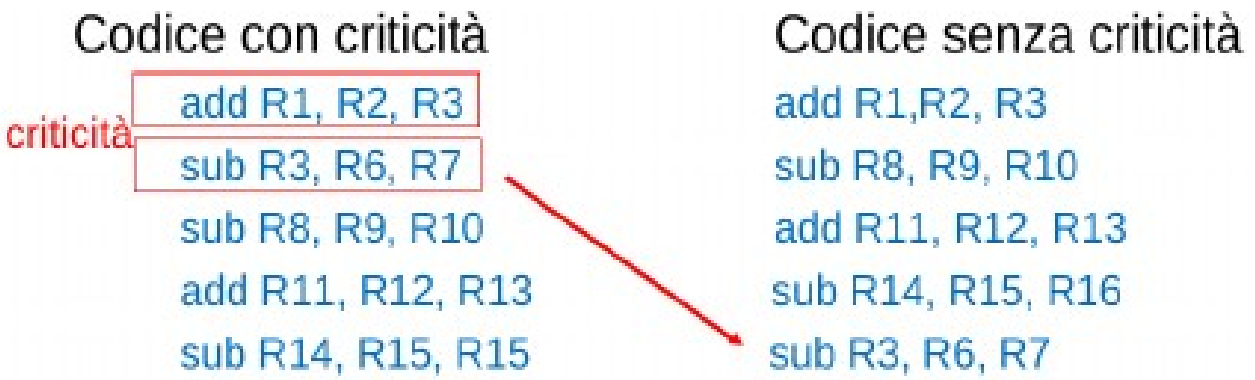
– ReadReq viene asserito
– Ack viene asserito in risposta a ReadReq
– ReadReq viene non asserito in risposta ad Ack
– Ack viene non asserito in risposta a ReadReq
– DataRdy viene asserito
– Ack viene asserito in risposta a DataRdy
– DataRdy viene non asserito in risposta ad Ack
– Ack viene non asserito in risposta a DataRdy

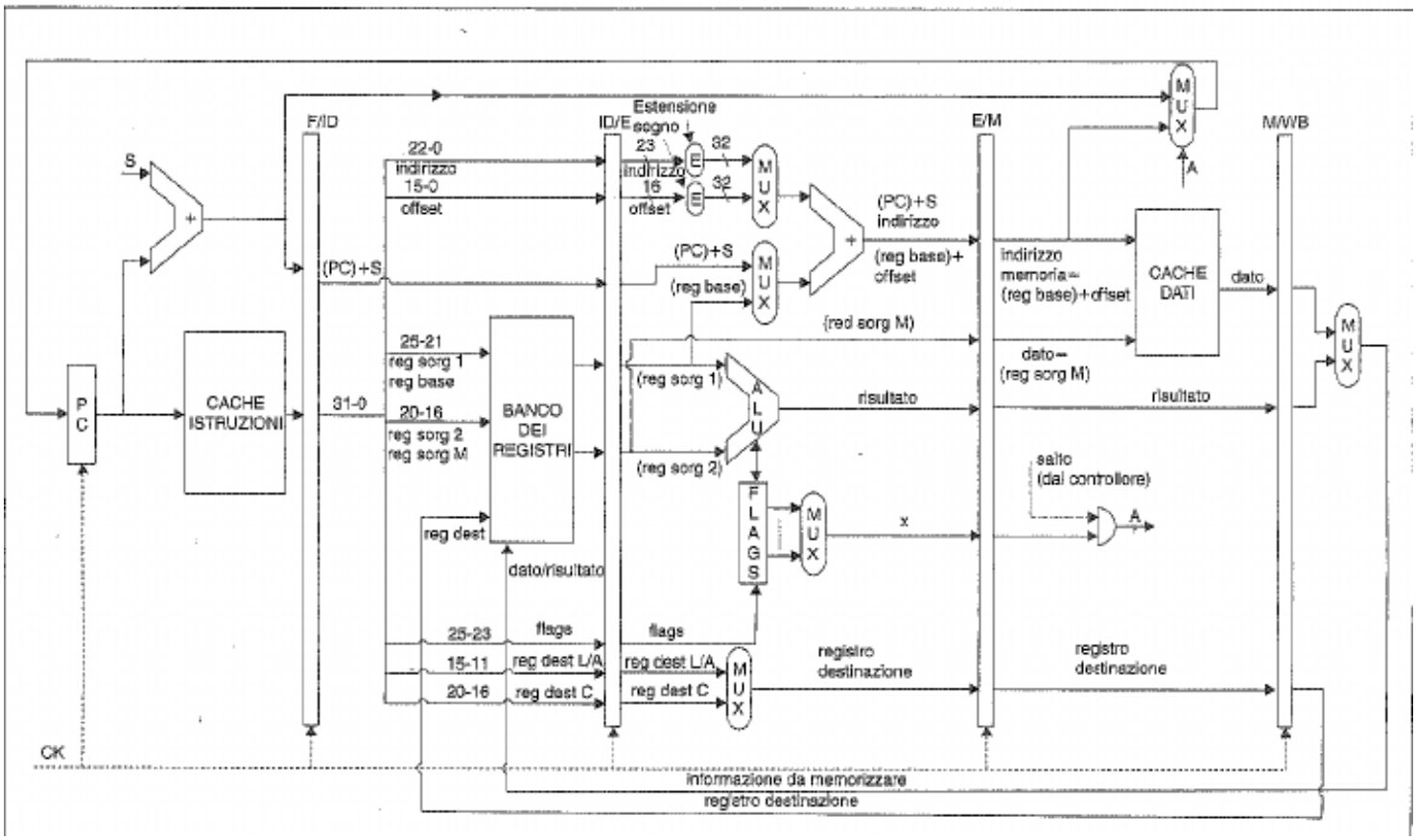
4) Dire cosa sono i conflitti di tipo define use e in che modo si risolvono via software (utilizzando a tal fine un disegno di massima dell'architettura pipeline RISC didattica vista a lezione), nell'esposizione si consiglia di scrivere frammenti di programmi per evidenziare tali fenomeni e come risolverli

I conflitti di tipo define use sono delle criticità dei dati in cui l'istruzione dipende dal risultato dell'istruzione precedente che è ancora nella pipeline.

Si possono risolvere via software inserendo istruzioni nop oppure riordinando le istruzioni in modo da eliminare la criticità.







4) Descrivere le possibili modalità di arbitraggio dei bus.

L’arbitraggio del bus può essere centralizzato, ovvero un controllore decide a chi assegnare il bus utilizzando daisy chain e livelli multipli di priorità oppure schemi distribuiti, dove non è presente nessun controllore e i dispositivi seguono un algoritmo per il controllo d’accesso e cooperano per la condivisione del bus, utilizzando round-robin e rilevamento della collisione.

Con daisy chain ad ogni dispositivo è assegnata una priorità e viene scelto il dispositivo che richiede il bus e possiede priorità maggiore, il problema è che non viene garantita la fairness favorendo alcuni dispositivi rispetto ad altri.

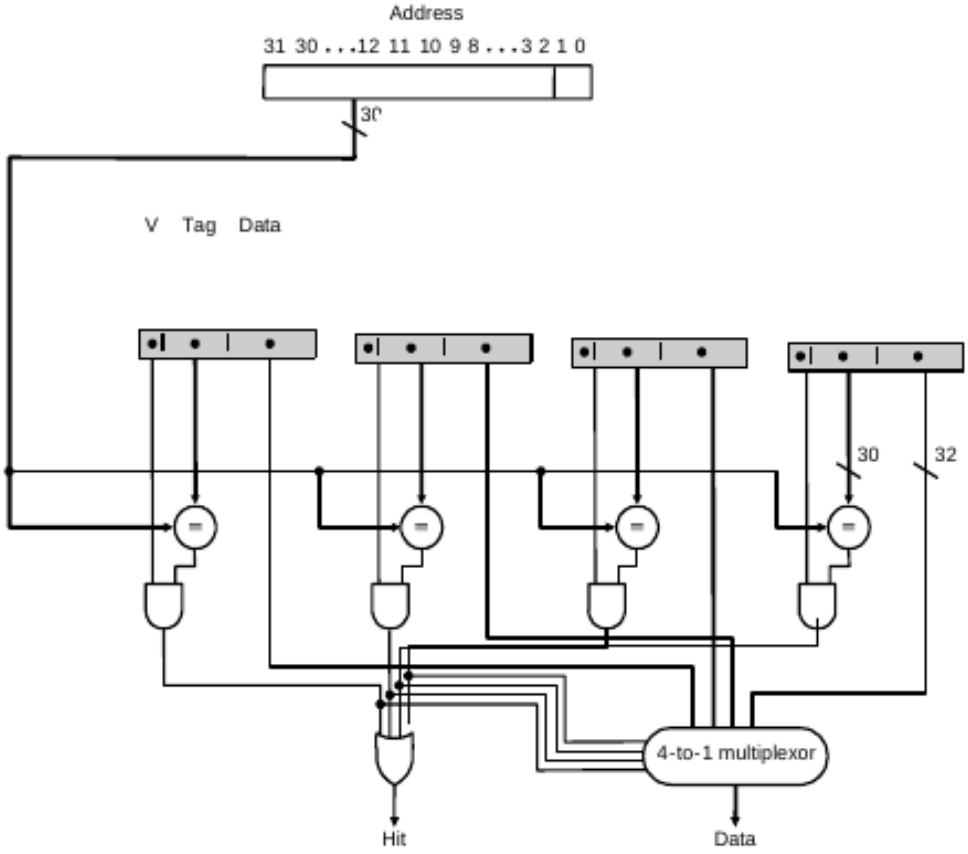
Con livelli multipli di priorità vengono associate diverse linee di richiesta a diversi livelli di priorità e in caso di conflitto vengono favorite le catene a priorità più alta.

Tramite Round-Robin abbiamo uno scambio diclico di un segnale di disponibilità tra le unità utilizzatrici del bus.

Con il rilevamento delle collisioni esiste un’unica linea su cui è segnalato lo stato del bus, ovvero se libero o occupato, nel momento in cui più unità occupano contemporaneamente il bus, occorre rilevare la collisione ed annullare la trasmissione, la quale verrà ripetuta dopo un intervallo di tempo.

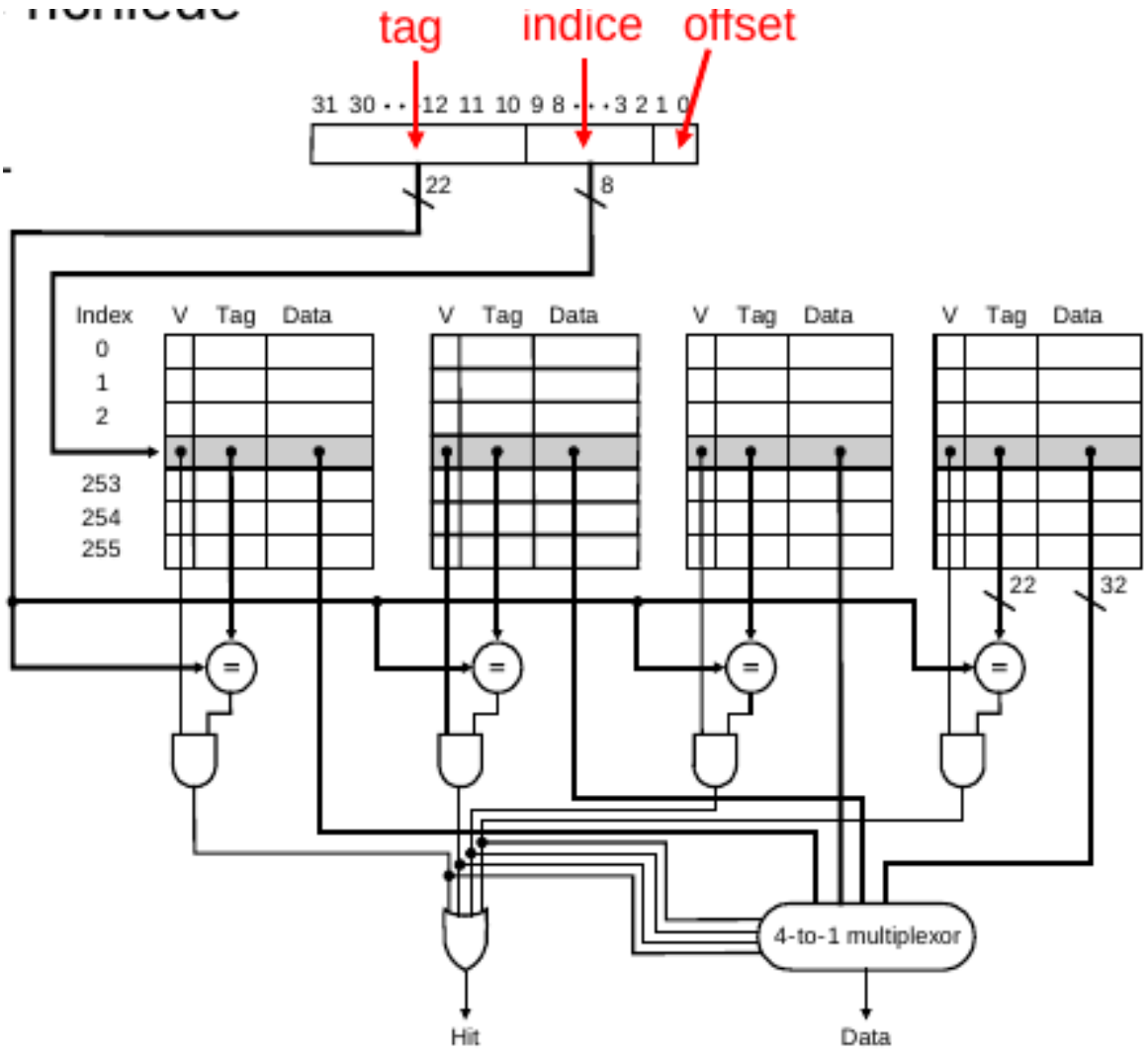
4) Disegnare una memoria cache completamente associativa ed una set associativa a quattro vie e confrontarle tra di loro dal punto di vista dei loro vantaggi e svantaggi

1) Cache completamente Associativa



Nella memoria cache completamente associativa ogni blocco di memoria può essere mappato in una qualsiasi linea di cache.
Il contenuto di un blocco di cache è identificato mediante l'indirizzo completo di memoria: il tag è costituito dall'indirizzo completo della parola e l'accesso è indipendente dall'indice di cache.
La ricerca viene effettuata mediante confronto in parallelo dell'indirizzo cercato con tutti i tag.
Il problema è che l'hardware è molto complesso e utilizzabile con un piccolo numero di blocchi.

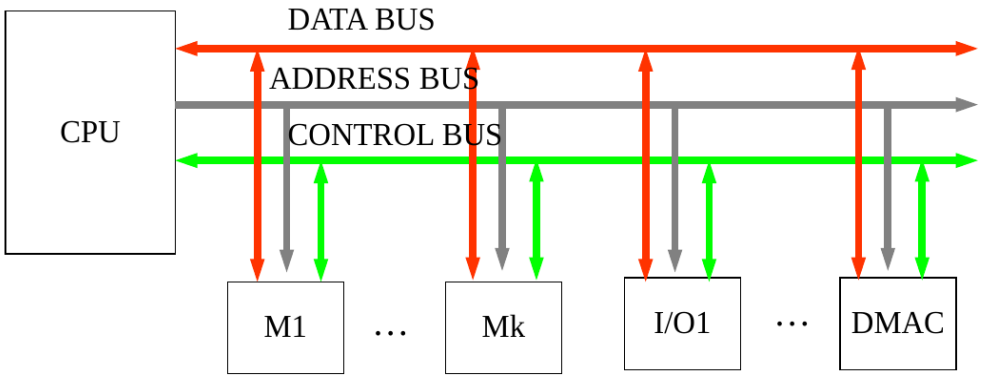
2) Cache set-associativa a N vie



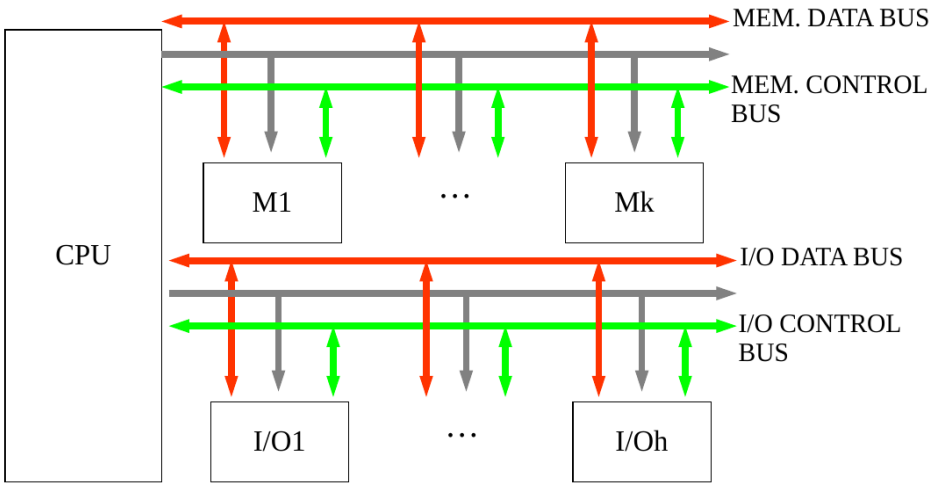
Nella memoria cache set-associativa a 4(N) vie la cache è organizzata come insieme di set, si attenua così il problema della collisione di più blocchi sulla stessa linea di cache.
Aumentando quindi l'associatività diminuiamo il miss rate ma abbiamo un incremento dell'hit rate e un maggior costo implementativo.

4) Descrivere le possibili organizzazioni dei bus di connessione tra un processore, la memoria di lavoro e i dispositivi di ingresso/uscita.

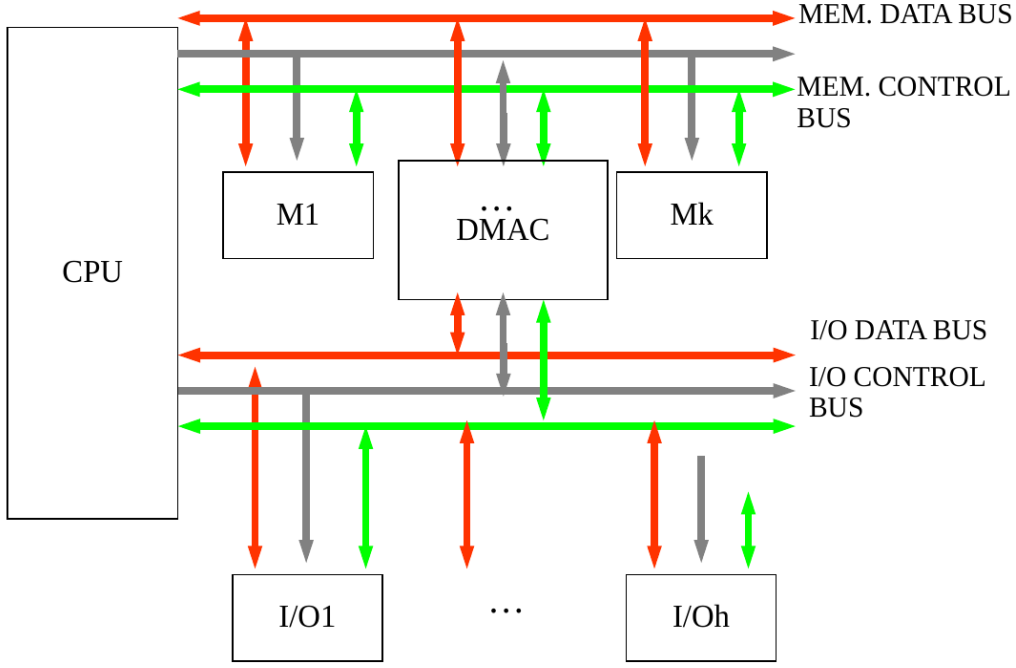
Nell'organizzazione ad un bus processore,memoria e i vari dispositivi sono collegati allo stesso bus



Nell'organizzazione a due bus il processore condivide due bus separati tra la memoria e i dispositivi di I/O



Nella variane con DMAC controller



4) Dire cosa è la banda passante di un bus e far vedere come è possibile incrementarne la capacità.

La banda passante di un bus corrsponde alle prestazioni del bus, per ottenere la massima banda passante teorica dobbiamo effettuare il prodotto tra la frequenza e il numero di linee.
Per incrementare la banda passante possiamo: aumentare il numero di linee, separare linee dati da quelle per gli indirizzi, trasferire i dati a blocchi riducendo il tempo di risposta, utilizzare un protocollo split transaction dove dividiamo la transazione sul bus in due parti: transazione di richiesta e transazione di risposta

DOMANDE 5

5) Disegnare l'architettura PIPELINE del processore didattico, dopodiché dato il programma sottostante, da modificare via software nel miglior modo possibile per evitare conflitti sui dati, dire in quanto tempo viene eseguito ipotizzando che il processore possa lavorare con un clock con periodo di 100 psec.

```
sub R5, R5, R7
sub R5, R7, R7
load R5, 300(R2)
sub R9, R9, R9
add R9, R7, R8
sub R11, R12,R13
sub R1,R2, R3
```

5) Disegnare l'architettura PIPELINE del processore didattico, dopodiché dato il programma sottostante, da modificare via software nel miglior modo possibile per evitare conflitti sui dati, dire in quanto tempo viene eseguito ipotizzando che il processore possa lavorare con un clock con periodo di 200 psec.

```
load R3, 120(R1)
add R3, R3, R6
sub R4, R3, R6
add R4, R7, R3
sub R3, R9, R10
add R12, R13, R13
```

5) Disegnare l'architettura PIPELINE del processore didattico, dopodiché dato il programma sottostante, da modificare via software nel miglior modo possibile per evitare conflitti sui dati, dire in quanto tempo viene eseguito ipotizzando che il processore possa lavorare con un clock con periodo di 100 psec.

```
load R4, 100(R2)
sub R4, R4, R6
sub R4, R6, R6
add R9, R7, R8
sub R6, R7, R10
add R11, R12,R13
sub R1,R2, R3
```

5) Disegnare l'architettura PIPELINE del processore didattico, dopodiché dato il programma sottostante, da modificare via software nel miglior modo possibile per evitare conflitti sui dati, dire in quanto tempo viene eseguito ipotizzando che il processore possa lavorare con un clock con periodo di 200 psec.

```
load R4, 125(R1)
sub R4, R5, R6
sub R4, R6, R6
add R4, R7, R8
sub R4, R9, R10
add R11, R12,R13
```

5) Scrivere un segmento di programma in linguaggio assembly del processore didattico a 32 bit di 6 istruzioni in cui ci sia almeno un conflitto define use ed uno load use. Modificarlo per evitare i conflitti oppure, nel caso in cui non sia possibile evitare tutti i conflitti, inserire il minor numero di NOP. Inoltre dare le linee guida per risolvere via hardware tali tipi di conflitti tramite la propagazione in avanti. Per comodità degli esaminandi si riporta la sintassi delle istruzioni:

- nop - operL/A regsorg1, regsorg2, regdest
- load regdest, offset(regbase)
- store regsorgM, offset(regbase)
- jumpX indirizzo

5) Descrivere cosa sono le criticità strutturali, sui dati e sul controllo di un processore RISC e come si possono risolvere, a tal fine si utilizzi l'architettura di riferimento introdotta a lezione.

Criticità strutturali: tentativo di usare la stessa risorsa hardware da parte di diverse istruzioni in modi diversi nello stesso ciclo di clock, si potrebbe risolvere evitando che il banco dei registri venisse progettato per evitare conflitti tra la lettura e la scrittura nello stesso ciclo, esempio: scrittura del banco dei registri nella prima metà del ciclo di clock e lettura nella seconda metà.

Criticità sui dati: tentativo di usare un risultato prima che sia disponibile e può essere di due tipi: define-use prodotto da un'operazione logico aritmetica e load-use prodotta da una load.

Soluzioni software: inserimento di istruzioni nop, peggiorando il throughput o con il riordino delle istruzioni

Soluzioni hardware: inserimento di bolle o stalli nella pipeline(si inseriscono dei tempi morti), oppure tramite propagazione o scavalcamento.

Criticità sul controllo: nel caso di salti condizionati, decidere quale sia la prossima istruzione da eseguire prima che la condizione sia valutata

Due approcci, pessimistico: non si eseguono istruzioni significative fino al completamento della scelta (perdita di tempo), ottimistico: si prospetta che non venga effettuato il sato(necessita di annullare gli effetti delle istruzioni eseguite nel caso di salto)