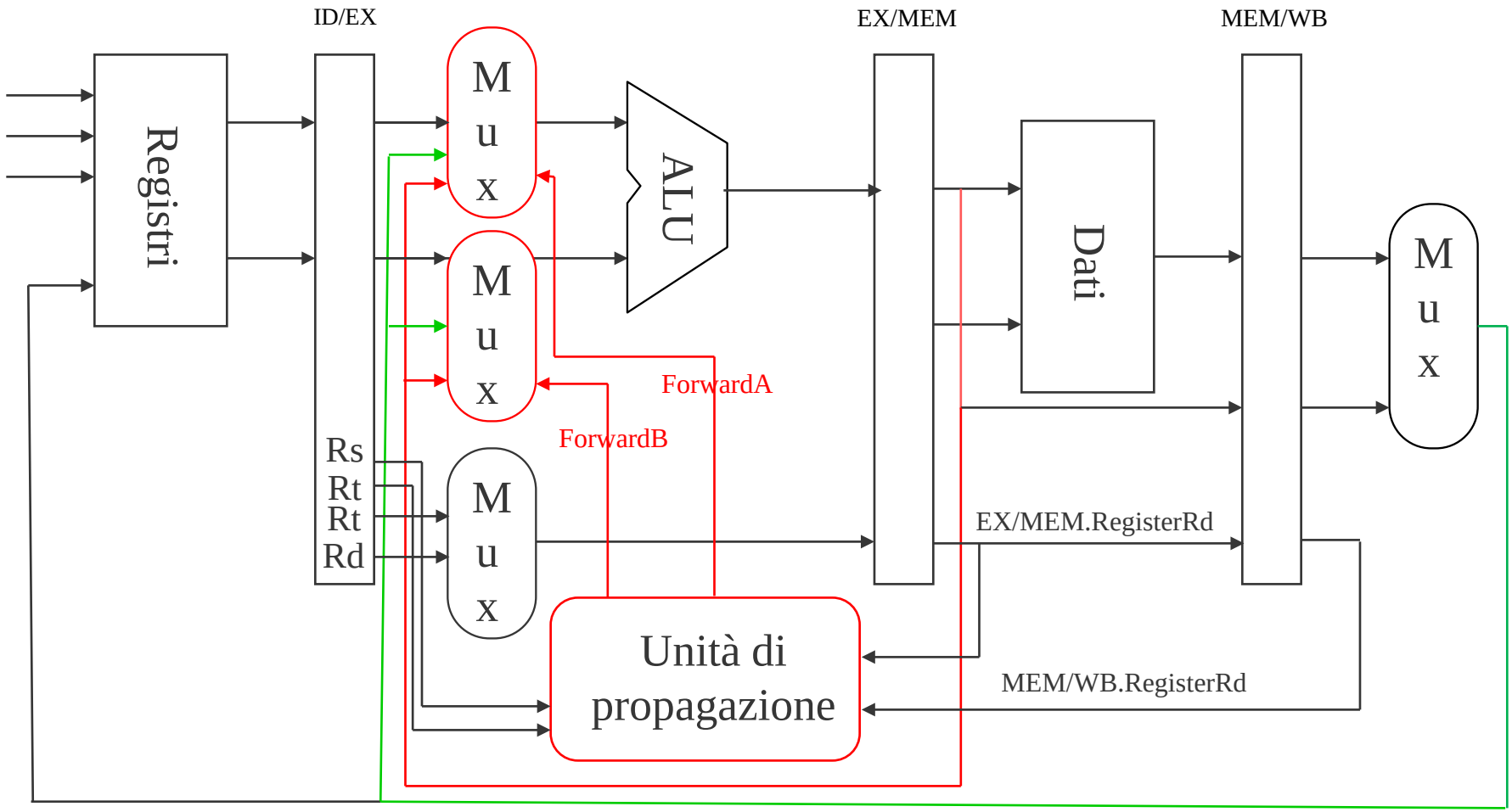
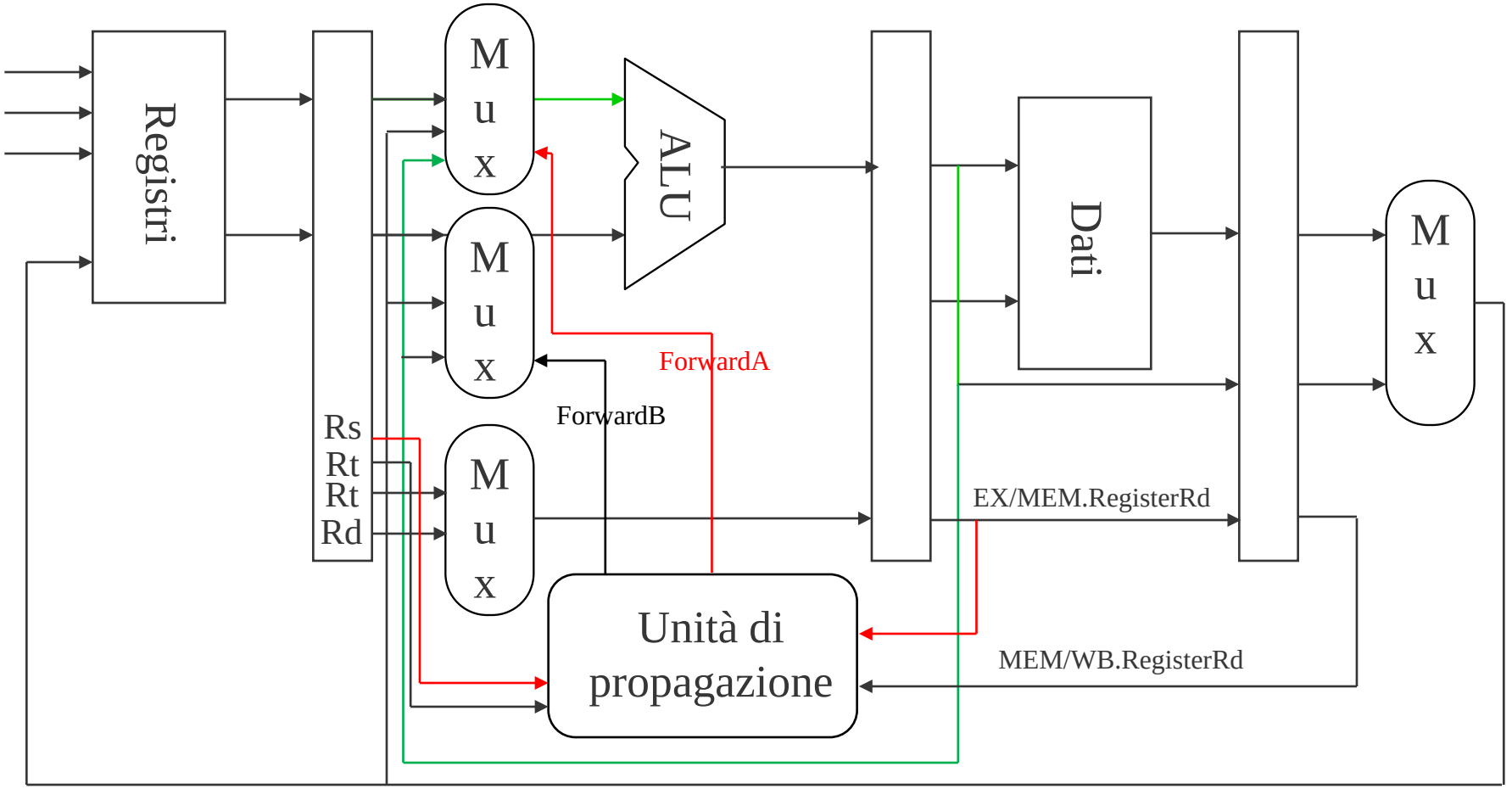
Il cammino dei dati rivisto (fase IF omessa)



Propagazione da EX/MEM (tipo 1a)

if (EX/MEM.RegWrite
and (EX/MEM.RegisterRd!=0)
and (EX/MEM.RegisterRd=ID/EX.RegisterRs))
    ForwardA = 10

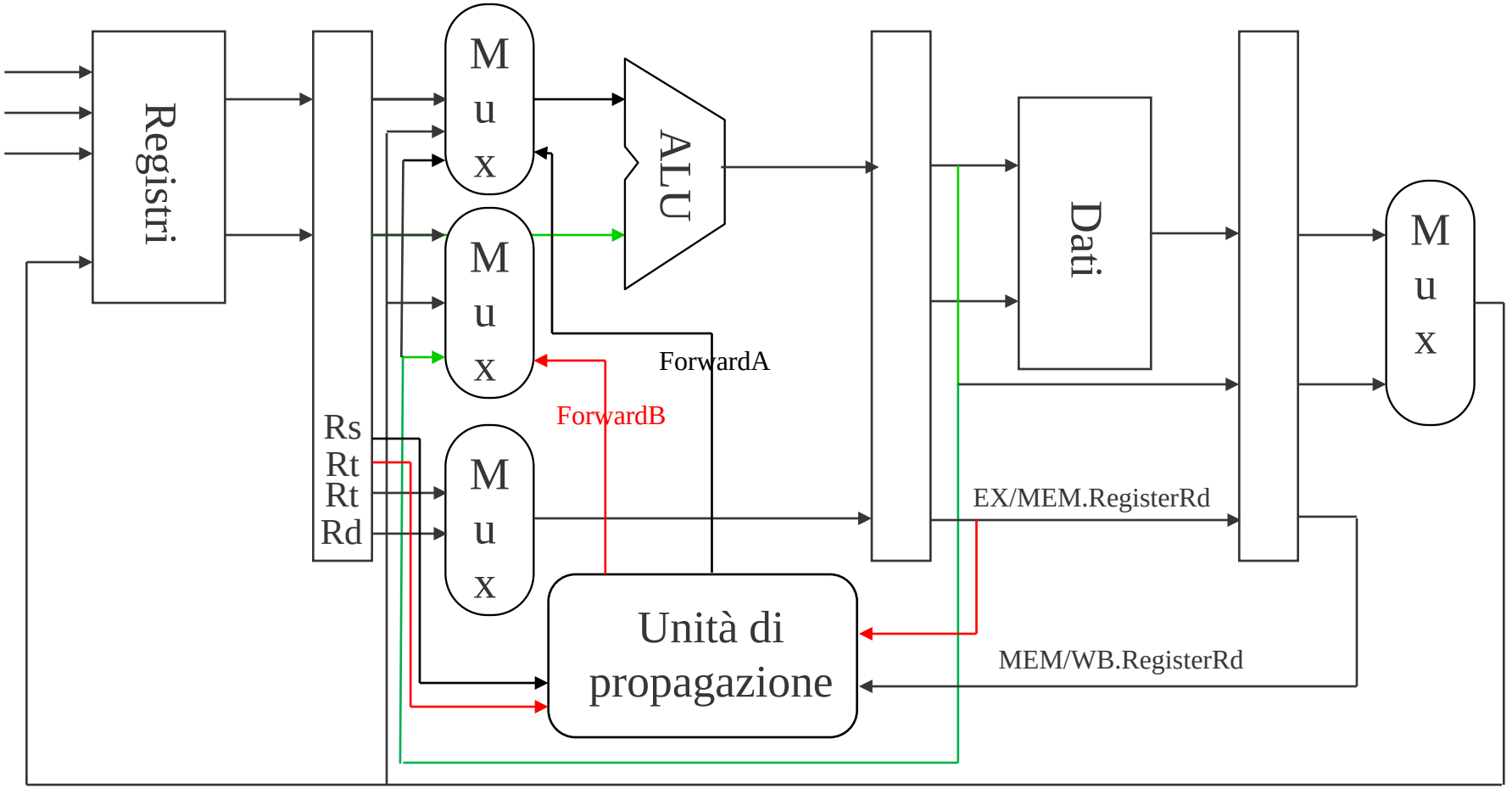
Il cammino dei dati rivisto



Propagazione da EX/MEM (tipo 1b)

if (EX/MEM.RegWrite
and (EX/MEM.RegisterRd!=0)
and (EX/MEM.RegisterRd=ID/EX.RegisterRt))
    ForwardB = 10

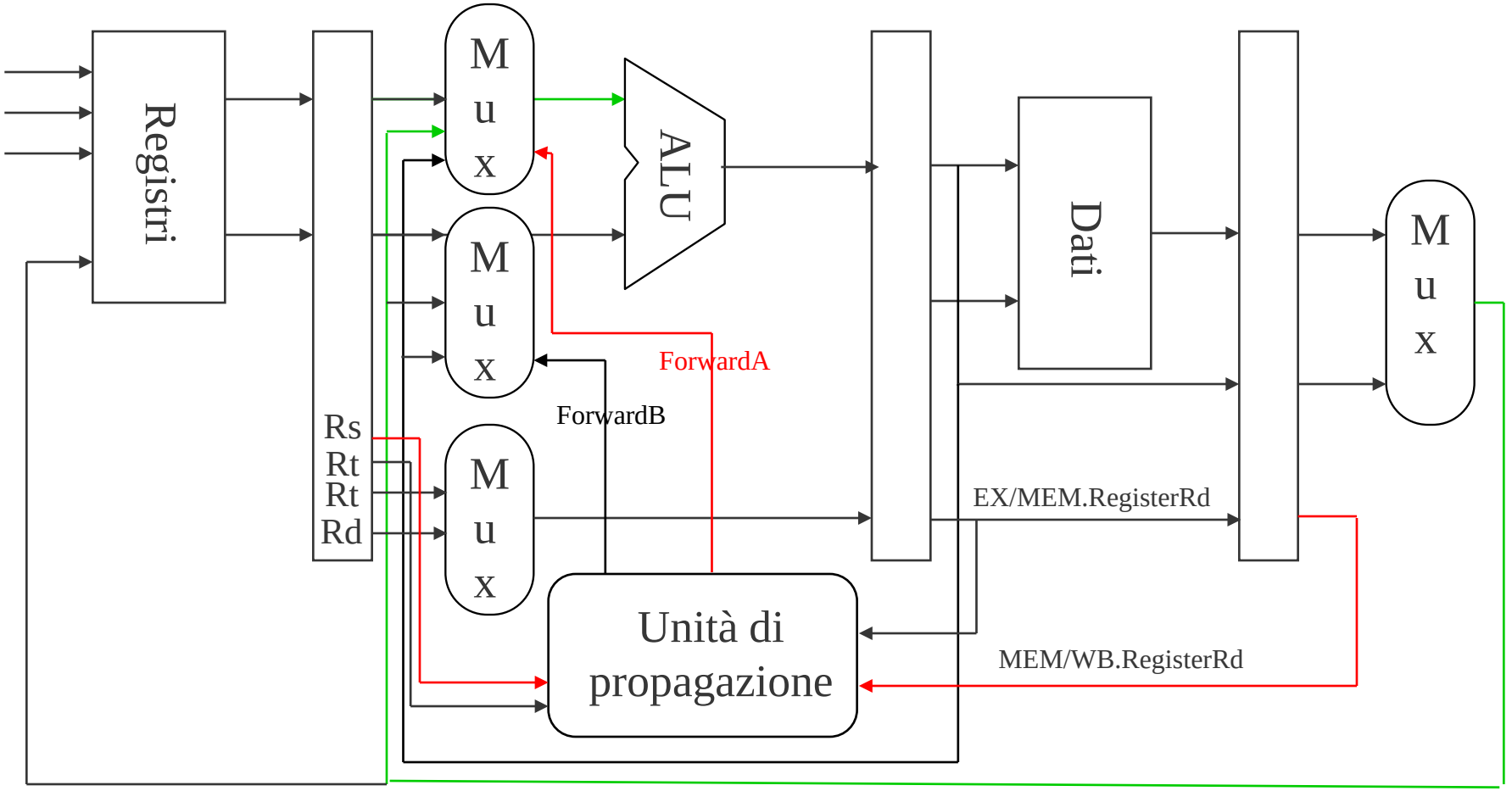
Il cammino dei dati rivisto



Propagazione da MEM/WB (tipo 2a)

if (MEM/WB.RegWrite
and (MEM/WB.RegisterRd!=0)
and (MEM/WB.RegisterRd=ID/EX.RegisterRs))
    ForwardA = 01

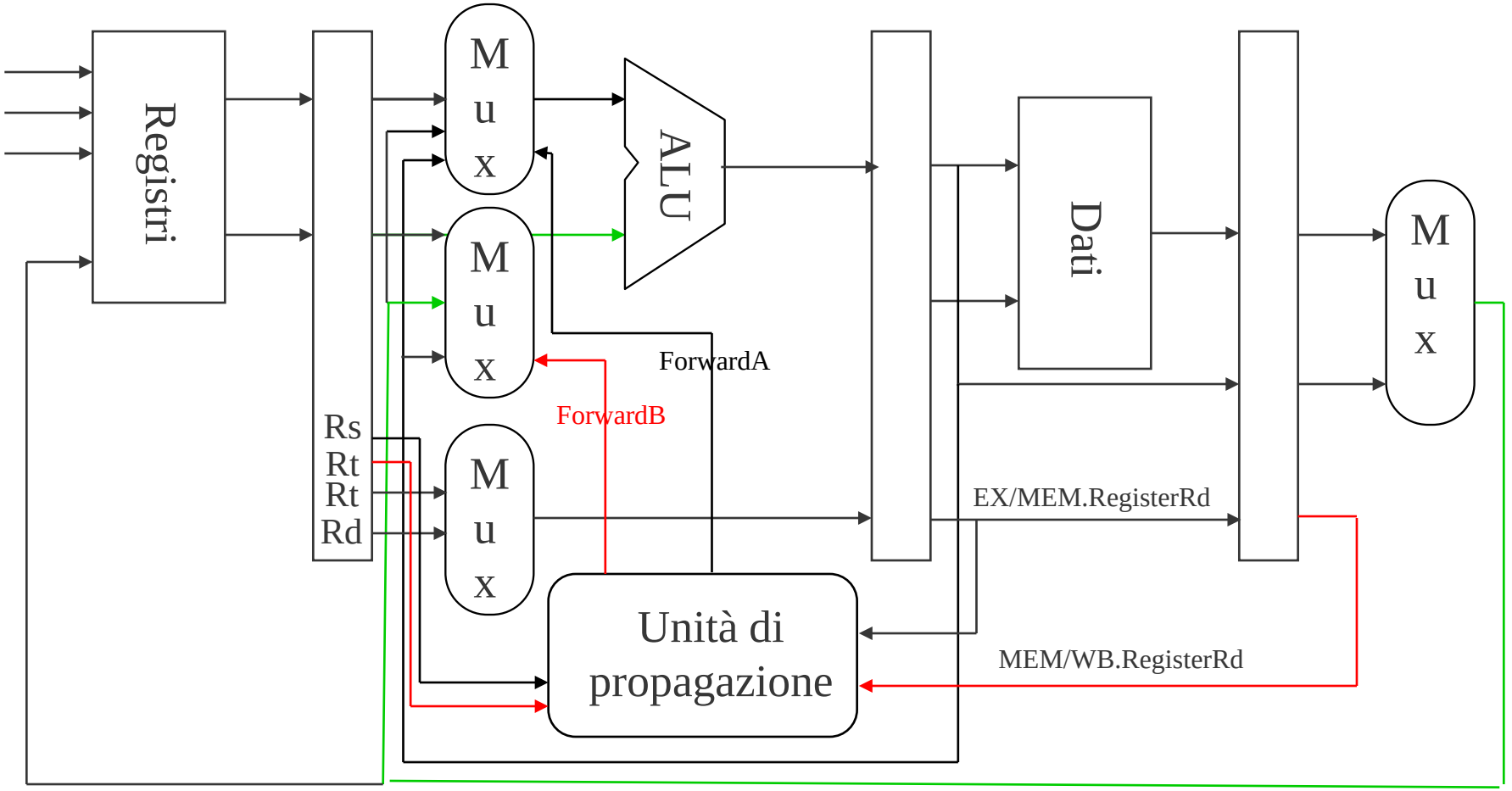
Il cammino dei dati rivisto (tipo 2a)



Propagazione da MEM/WB (tipo 2b)

if (MEM/WB.RegWrite
and (MEM/WB.RegisterRd!=0)
and (MEM/WB.RegisterRd=ID/EX.RegisterRt))
    ForwardB = 01

Il cammino dei dati rivisto (tipo 2b)



Problema: conflitto tra tipo 1 e 2

- Criticità tra risultato di istruzione in WB, risultato di istruzione in MEM ed operando sorgente di istruzione in EX
- Esempio:

```
add   $s1, $s1, $s2
add   $s1, $s1, $s3
add   $s1, $s1, $s4
```

- Quando si fa EX della 3^ add, avrei abilitata sia la propagazione dallo stadio MEM della 2^ add, sia la propagazione dallo stadio WB della 1^ add → conflitto!

  SOLUZIONE: quando si fa EX della 3^ add, l'istruzione in MEM (2^ add) ha *priorità maggiore* di quella in WB (1^ add) poiché quel risultato è il più recente

Criticità da WB corretta

if (MEM/WB.RegWrite
and (MEM/WB.RegisterRd!=0)
and (EX/MEM.RegisterRd!=ID/EX.RegisterRs)
and (MEM/WB.RegisterRd=ID/EX.RegisterRs))
ForwardA = 01

if (MEM/WB.RegWrite
and (MEM/WB.RegisterRd!=0)
and (EX/MEM.RegisterRd!=ID/EX.RegisterRt)
and (MEM/WB.RegisterRd=ID/EX.RegisterRt))
ForwardB = 01

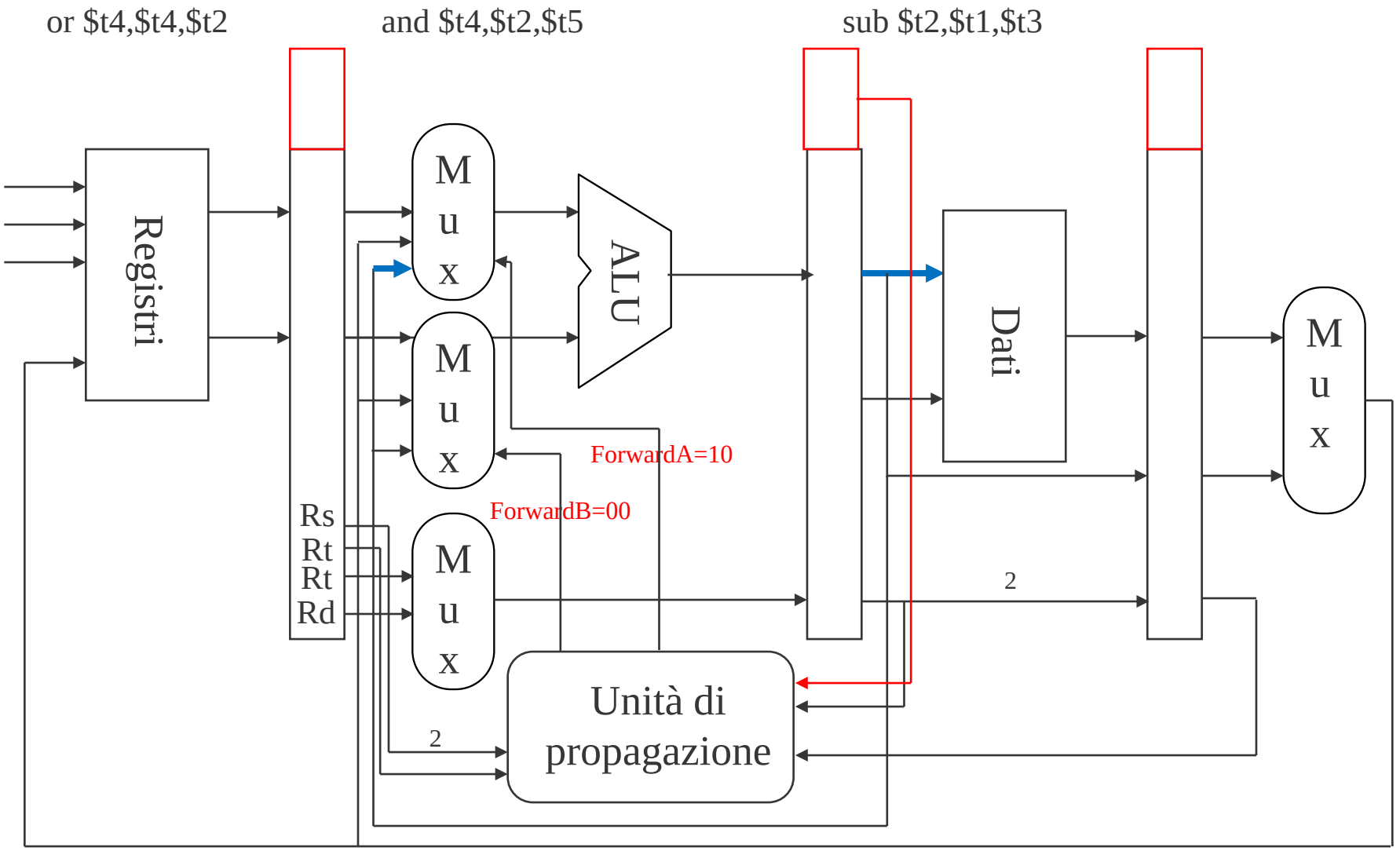
Esempio

- Consideriamo le seguenti quattro istruzioni

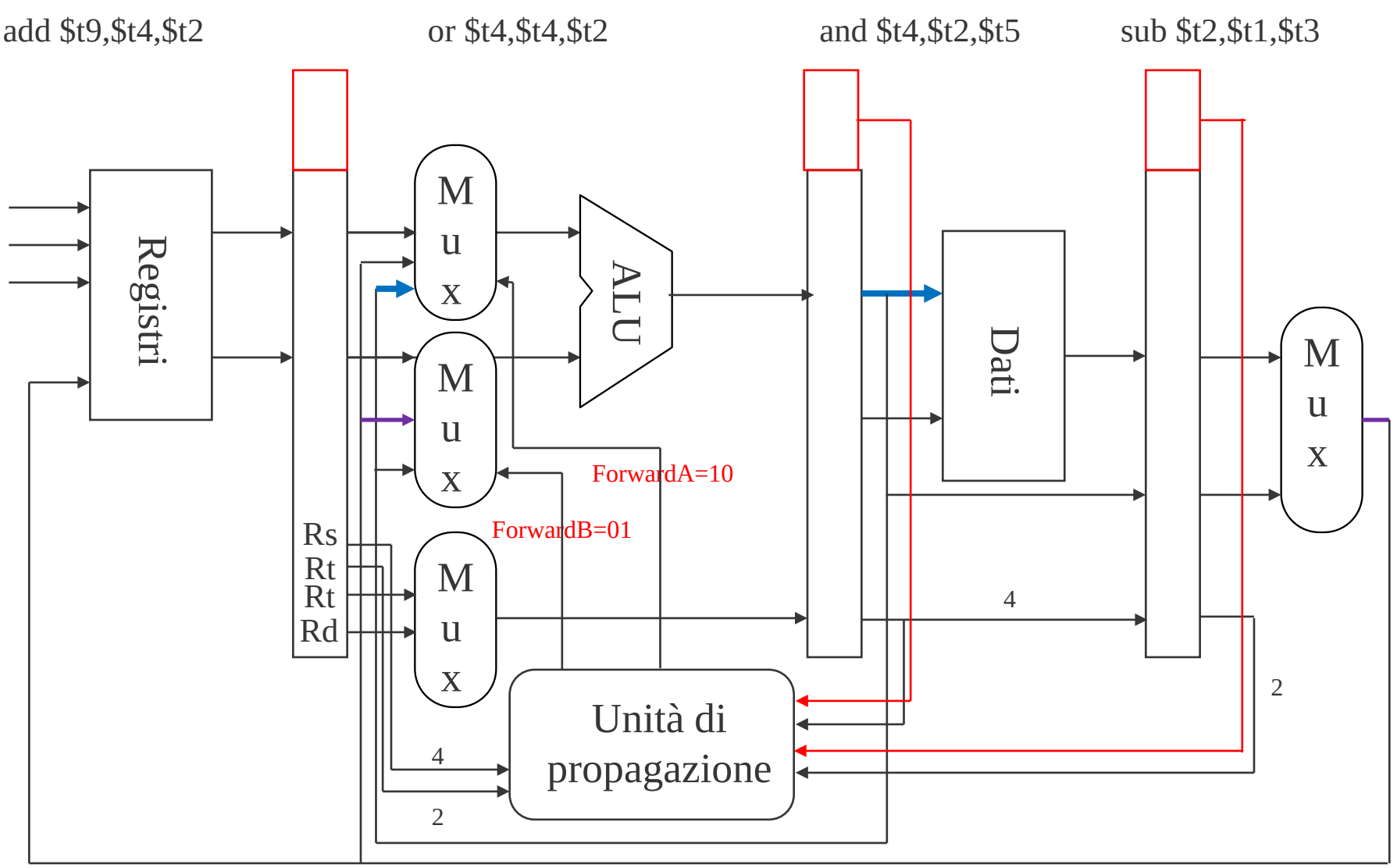
```
sub   $t2,  $t1,  $t3
and   $t4,  $t2,  $t5
or    $t4,  $t4,  $t2
add   $t9,  $t4,  $t2
```

Si hanno criticità su $t2 e $t4

Quarto ciclo di clock

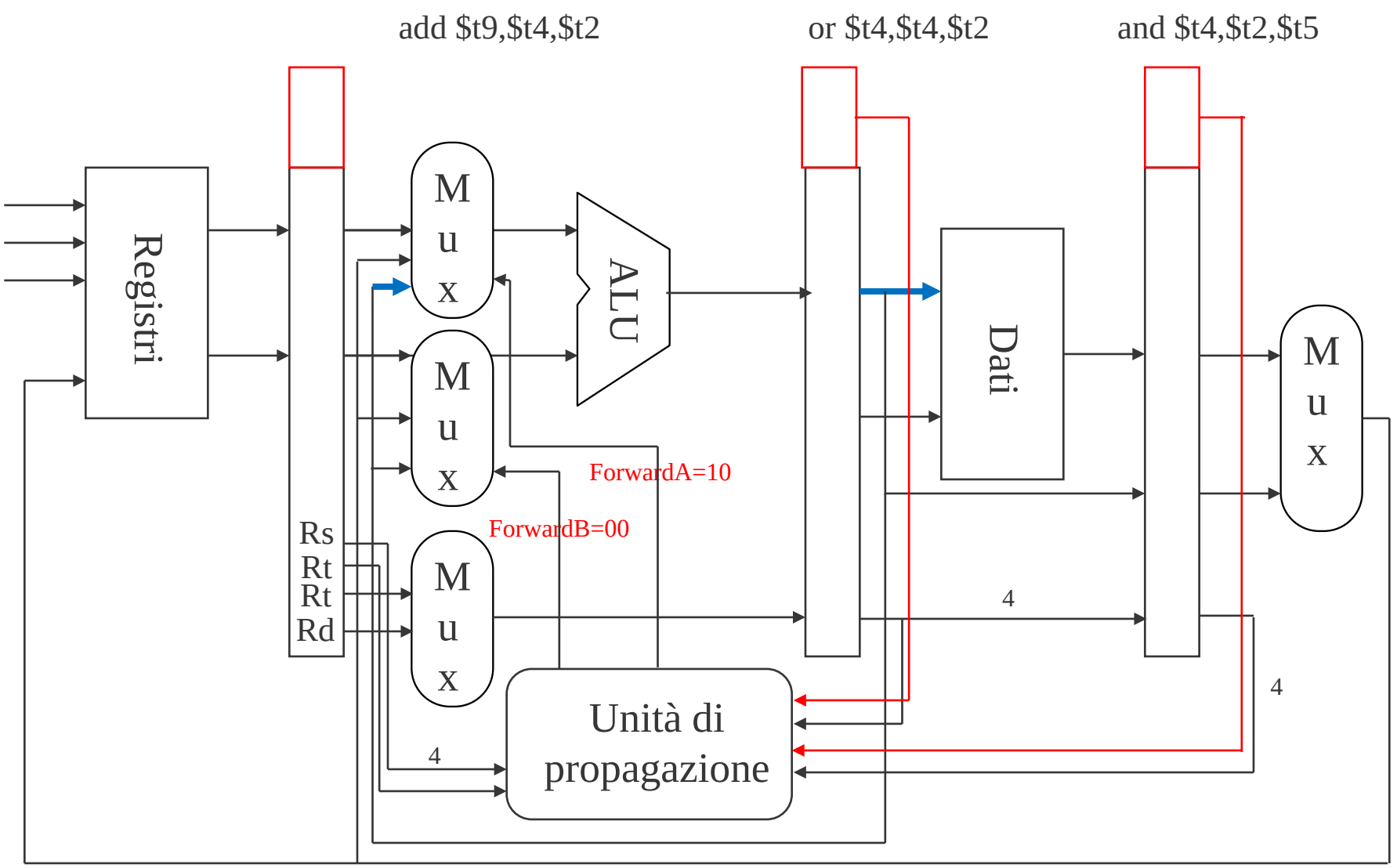


Quinto ciclo di clock

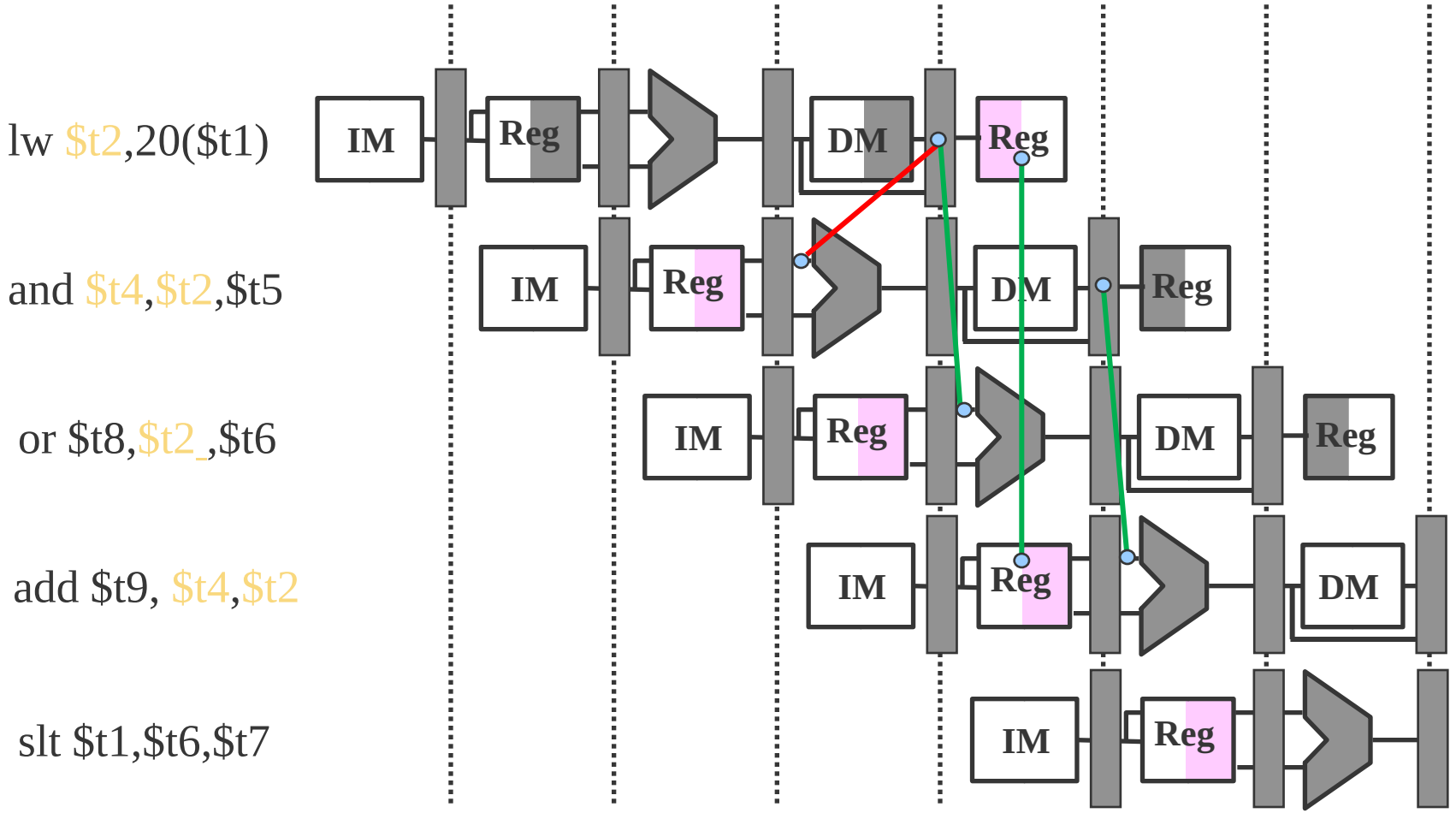


Sesto ciclo di clock

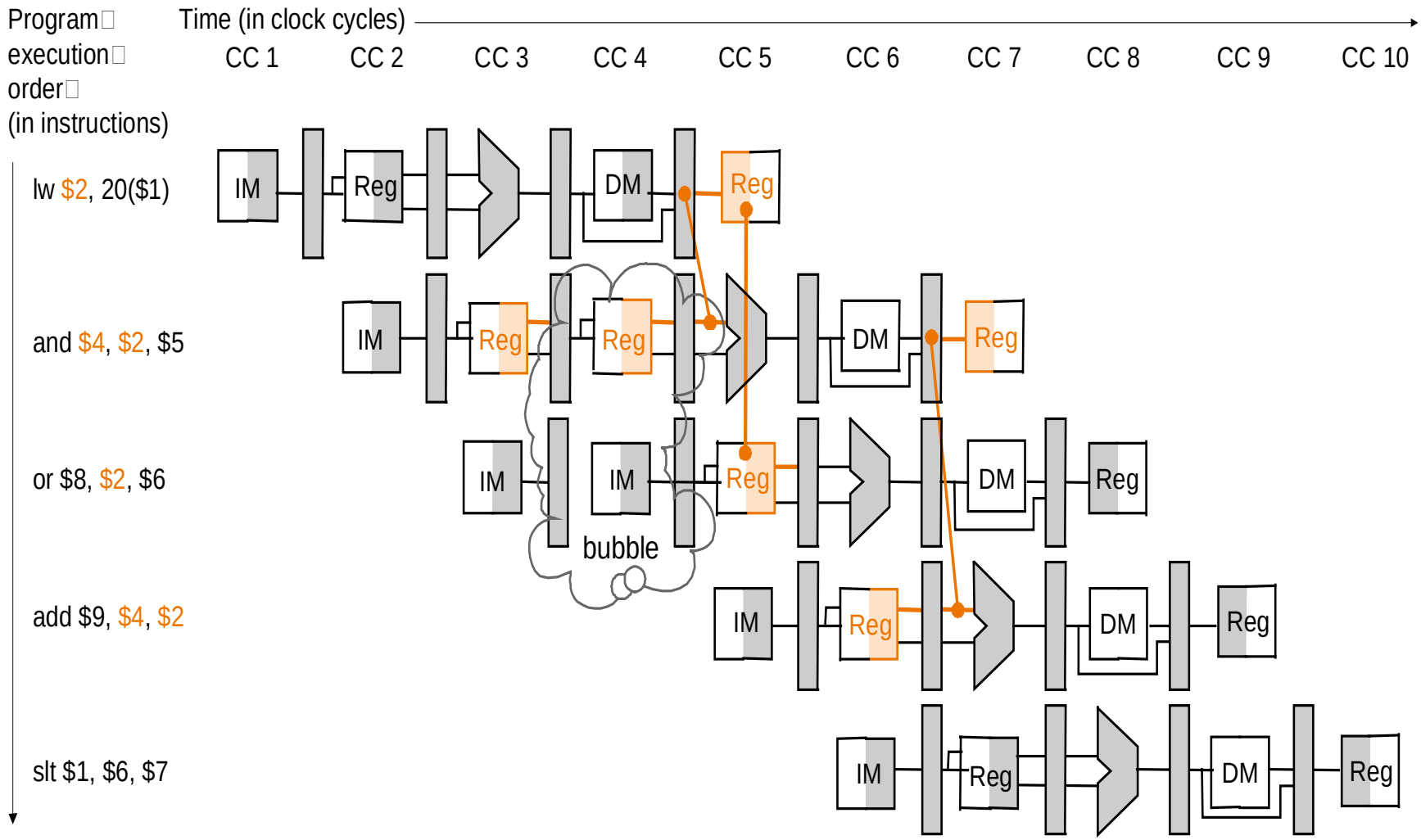
NOTA: Ho dato priorità al risultato proveniente da EX/MEM, poichè più recente!



Criticità di dati dopo lw (load-use)



Soluzione: stallo in fase EX



Stalli

- Unità di rilevazione delle criticità
  - Durante la fase ID
  - Inserisce uno stallo tra la load ed il suo uso
- Condizione:
  - if (ID/EX.MemRead
    and ((ID/EX.RegisterRt=IF/ID.RegisterRs)
    or (ID/EX.RegisterRt= IF/ID.RegisterRt)))
    metti in stallo la pipeline

se ho una istruzione load nello stadio EX

se il registro di scrittura della load coincide con uno dei registri sorgente dell'istruzione nello stadio ID

Realizzazione

- Istruzioni in IF e ID messe in stallo per 1 ciclo
  - Impedire la modifica di PC
  - Impedire la modifica di IF/ID

  - Le istruzioni vengono semplicemente ripetute
- Inserire una "bolla" (nop) da EX in poi
  - Porre a 0 i campi EX, MEM e WB del registro di pipeline **ID/EX**, che contengono i segnali di controllo
    - nessun registro o locazione di memoria viene modificato

- Come conseguenza, la bolla è propagata nella pipeline

Unità di Rilevazione Criticità (in ID)