Architettura degli Elaboratori II

Corso di Laurea in Informatica

Università degli Studi di Milano

compilare in stampatello

Cognome		*Note consegna*
Nome		
Matricola		
e-mail	*@studenti.unimi.it*	

Appello del 17 giugno 2021

- Il presente plico include 5 esercizi da svolgersi nella durata massima di 3 ore.
- Le soluzioni vanno riportate negli spazi delimitati che seguono il testo di ciascun esercizio, la soluzione ad un esercizio non può occupare spazio riservato ad un altro esercizio, non possono essere consegnati fogli aggiuntivi.
- È possibile prendere appunti, ma è vietata la consultazione di qualsiasi tipo di materiale.
- È consentito l'uso della calcolatrice non scientifica.
- È vietato l'uso di qualsiasi dispositivo di comunicazione.
- È fornita una traccia di soluzione nell'ultima pagina.

parte riservata al docente

___ / 6
___ / 7
___ / 6
___ / 6
___ / 6
___ / 31

Esercizio 1 [6 punti]

Si assuma di avere a disposizione dei componenti hardware che consentano di svolgere le seguenti operazioni nella durata temporale indicata.

- Accesso a memoria: $4\ ns$
- Accesso al register file: $2\ ns$
- Operazione Aritmetico/Logica: $3\ ns$

Viene fornito un benchmark di 800 istruzioni caratterizzato dal profilo riportato in tabella.

Tipo di istruzione	Percentuale
Aritmetico/Logica	25%
Load word	9%
Store word	18%
Branch	32%
Jump	16%

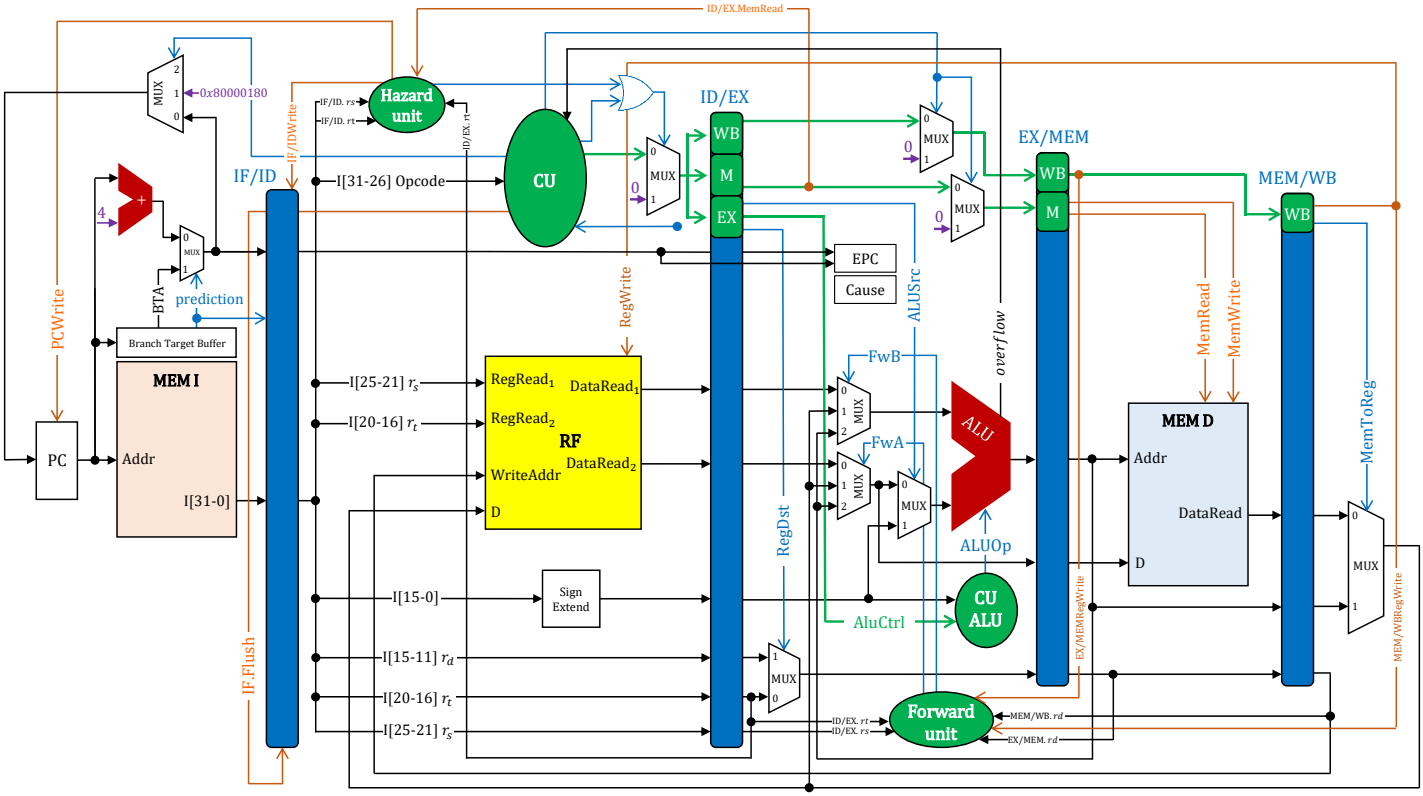
Rispetto al benchmark sopra riportato si calcolino, mostrando e motivando i passaggi, il tempo medio di esecuzione di una istruzione e il tempo totale di esecuzione del benchmark quando i componenti hardware sono organizzati in:

A. una architettura a singolo ciclo;
B. una architettura a ciclo variabile;
C. una architettura a ciclo multiplo.

D. In riferimento ai risultati ottenuti nei punti precedenti si indichino sinteticamente i pro e i contro di ciascuna architettura.

Esercizio 2 [7 punti]

Si consideri la CPU pipeline riportata in figura.



Il data path sopra riportato non include dei componenti che permettono la gestione di alcuni hazard.

A. Si dica quali sono questi componenti mancanti, si descriva il loro funzionamento e come dovrebbero essere collegati (non è necessario disegnare i componenti mancanti sul data path)

Si consideri successivamente questo programma eseguito dalla CPU

```
lw $t1 4($t0)
add $t2 $t2 $t1
sub $t2 $t2 $t5
sw $t2 48($t1)
slt $t2 $t6 $t2
```

B. Quanti cicli di clock sarebbero necessari se la pipeline potesse lavorare al massimo throughput ideale (che quindi non tiene conto di possibili hazard)?
C. Si indichino, per ogni ciclo di clock richiesto nel punto B, le possibili criticità strutturali che si verificherebbero se il data path fosse stato progettato senza replicazione di risorse.
D. Per ogni eventuale hazard di dato si indichino le istruzioni coinvolte e il tipo di hazard.
E. Si supponga che gli hazard dati vengano gestiti a *compile time* con l'inserimento di *nop*. Si riporti il programma risultante e si indichino quanti cicli di clock sarebbero richiesti alla CPU per eseguirlo.
F. Si supponga che gli hazard dati vengano gestiti tramite tecnica di forwarding. Si indichino, per ogni hazard, il valore dei segnali di controllo che determinano la sua gestione e si calcoli il numero di cicli di clock necessari alla CPU per eseguire il programma senza errori.

Esercizio 3 [6 punti]

I seguenti due blocchi di codice assembly fanno parte di un programma che viene eseguito su una CPU pipeline che implementa una predizione dinamica dei salti.

```
sub $s0 $s0 $t5          add $s0 $s0 $s0
and $s1 $s0 $t3          slt $s1 $t7 $s0
beq $s1 $zero A          beq $s1 $zero B
```

Il primo ciclo viene eseguito per 5 volte. In ogni esecuzione il valore iniziale dei registri t_i è pari ad i mentre il valore iniziale del registro s_0 cambia come riportato in tabella.

Esecuzione	Valore iniziale di s_0
1	9
2	71
3	21
4	-3
5	96

Anche il secondo ciclo viene eseguito per 5 volte con gli stessi valori iniziali delle esecuzioni considerate per il primo. Sapendo che il primo blocco inizia all'indirizzo $0x020D8022$ e il secondo all'indirizzo $0x02108022$ si riportino:

A. la tabella delle predizioni contenuta nel branch prediction buffer dopo le esecuzioni descritte sopra assumendo uno schema di predizione ad 1 bit;
B. la medesima tabella del punto A calcolata con uno schema di predizione a 2 bit (motivare i passaggi spiegando il funzionamento di tale meccanismo di predizione dei salti)

Esercizio 4 [6 punti]

Si consideri una cache set-associativa a 8 vie da $64KiB$ con blocchi da 64 bit. La cache lavora con uno spazio di indirizzamento a 32 bit.

A. Si calcoli la dimensione totale della cache in KiB riportando i passaggi di calcolo per ogni parametro che è necessario derivare.
B. Si fornisca lo schema circuitale di alto livello della cache limitandosi a riportare nel disegno un singolo banco. Si indichino quanti comparatori in totale sono necessari per la sua costruzione.
C. Si calcolino numero di blocco, indice di linea, tag, offset di blocco (quale parola dentro al blocco) e offset di parola (quale byte dentro alla parola) dei seguenti dati:
   - La parola di memoria che contiene il byte il cui indirizzo è $0x00000009$
   - La parola di memoria all'indirizzo $0x00001000$
   - Il byte all'indirizzo $0x00008205$
D. Si fornisca un esempio di programma che generi almeno un evento di *cache miss*.
E. Si indichi il numero di vie nel caso in cui, a parità degli altri parametri dati inizialmente, si progettasse la cache con uno schema direct-mapped e con uno fully-associative.

Esercizio 5 [6 punti]

Viene letta da memoria la seguente stringa di 14 bit

$$10011001110111$$

A. Si fornisca la rappresentazione esadecimale del dato rappresentato da questi 14 bit sapendo che, in fase di salvataggio in memoria, è stato usato un codice di Hamming a distanza $\delta = 4$.

Traccia[1] di soluzione

Esercizio 1

A. $T_{medio} = 15\ ns, T_{tot} = 12\ \mu s$ (con $T_{ck} = 15\ ns$, si poteva anche sovradimensionare a $20\ ns$)
B. $T_{medio} = 10.28\ ns, T_{tot} = 8.224\ \mu s$
C. $T_{medio} = 13.8\ ns, T_{tot} = 11.04\ \mu s$

Esercizio 2

A. Il datapath non include i componenti per la gestione degli hazard di controllo, in particolare il comparatore all'uscita del RF e il sommatore per il calcolo del BTA
B. 9
C. 2 criticità ALU in ciclo 2, 3 criticità ALU in ciclo 3, 3 criticità ALU e 2 MEM in ciclo 4,3 criticità ALU in ciclo 5, 2 criticità ALU in ciclo 6.
D. lw e add (tipo lw), sub e add (tipo A/L), sw e sub (tipo A/L), slt e sub (tipo A/L)
E. Vengono aggiunte 6 nop, la CPU impiega 15 cicli di clock per eseguire il programma risultante
F. Il forwarding genera uno stallo, la CPU impiega 10 cicli di clock. Segnali di forward nei 5 hazard: $FwA = 1, FwB = 2, FwA = 2, FwA = 1$.

Esercizio 3

Nelle 5 esecuzioni, per il ciclo 1 le branch sono: taken, not taken, taken, taken, not taken; per il ciclo 2 invece: not taken, not taken, not taken, taken, not taken

A. In tabella, per ciclo 1 $indirizzo = 0x020D802A, Target\ addr. = A, bit\ di\ salto = 0$; per ciclo 2 $indirizzo = 0x0200802A, Target\ addr. = B, bit\ di\ salto = 0$
B. In tabella, per ciclo 1 $indirizzo = 0x020D802A, Target\ addr. = A, stato = 10, bit\ di\ salto = 1$; per ciclo 2 $indirizzo = 0x0200802A, Target\ addr. = B, stato = 01, bit\ di\ salto = 0$

Esercizio 4

A. $84\ KiB$
B. Servono 8 comparatori
C. Per il primo dato: $N(d) = 1$, $I(d) = 1$, $offset_{blocco} = 0$, $offset_{parola} = 0$, $tag = 0$; per il secondo dato: $N(d) = 512, I(d) = 512, offset_{blocco} = 0, offset_{parola} = 0, tag = 0$; per il terzo dato: $N(d) = 4160, I(d) = 64, offset_{blocco} = 1, offset_{parola} = 1, tag = 4$;
D. In cold start qualsiasi programma, a regime un programma con più di 8 richieste di memoria sulla stessa linea
E. Direct-mapped $n = 1$, fully-associative $n = 8192$

Esercizio 5

A. Applicando il controllo di errore al dato letto si ottiene un codice di errore $ECC = 10$. Correggendo il bit errato si ottiene $0x093$

[1] Viene riportata solo una bozza di soluzione per gli esercizi numerici da usare come guida. Le soluzioni complete non verranno pubblicate, ma possono essere discusse a lezione o durante il ricevimento.

Architettura degli Elaboratori II

Corso di Laurea in Informatica

Università degli Studi di Milano

compilare in stampatello

Cognome		*Note consegna*
Nome		
Matricola		
e-mail	*@studenti.unimi.it*	

Appello del 15 luglio 2021

- Il presente plico include 5 esercizi da svolgersi nella durata massima di 3 ore.
- Le soluzioni vanno riportate negli spazi delimitati che seguono il testo di ciascun esercizio, la soluzione ad un esercizio non può occupare spazio riservato ad un altro esercizio, non possono essere consegnati fogli aggiuntivi.
- È possibile prendere appunti, ma è vietata la consultazione di qualsiasi tipo di materiale.
- È consentito l'uso della calcolatrice non scientifica.
- È vietato l'uso di qualsiasi dispositivo di comunicazione.
- È fornita una traccia di soluzione nell'ultima pagina.

parte riservata al docente

__ / 6
__ / 8
__ / 6
__ / 6
__ / 5
__ / 31

Esercizio 1 [6 punti]

Si assuma di avere a disposizione dei componenti hardware che consentano di svolgere le seguenti operazioni nella durata temporale indicata.

- Accesso a memoria: $5\ ns$
- Accesso al register file: $2\ ns$
- Operazione Aritmetico/Logica svolta nella ALU: $2\ ns$
- Operazione Aritmetico/Logica svolta con hardware dedicato (non dalla ALU): trascurabile

Si consideri il programma nell'immagine a destra (si trattino le costanti come valori contenuti in registri e si assuma che $\$t0$ contenga inizialmente l'indirizzo di un array di più di 10 elementi).

A. Si indichi, per ogni **tipologia** di istruzione, il numero di volte che viene eseguita dal programma.

Assumendo che non venga svolta l'anticipazione del BTA, si calcolino tempo medio e tempo totale necessari per eseguire il programma su una CPU con architettura:

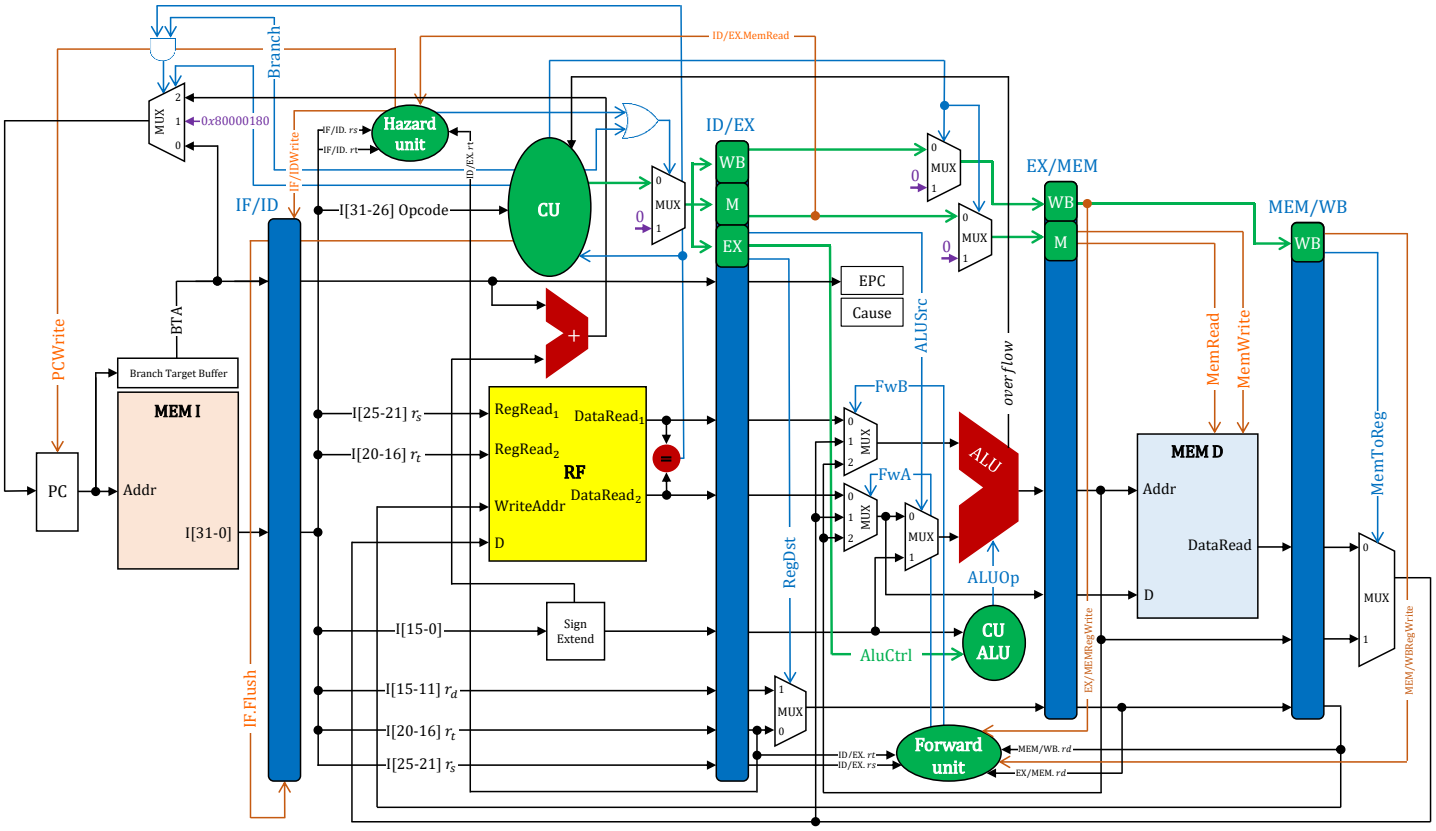
B. a singolo ciclo;
C. a ciclo variabile;
D. a ciclo multiplo.

E. Che cosa cambia nei tempi calcolati ai punti precedenti qualora si introducesse l'anticipazione del BTA?

```
        add $t3 $zero 4
step:
        add $t1 $t0 36
loop:
        lw $t2 0($t1)
        add $t2 $t2 $t2
        sw $t2 0($t1)
        sub $t1 $t1 4
        beq $t0 $t1 break
        j loop
break:
        sub $t3 $t3 1
        beq $t3 $zero exit
        j step
exit:
        add $t0 $zero $zero
```

Esercizio 2 [8 punti]

Si consideri la CPU pipeline riportata in figura.



Il data path riportato in figura include 3 errori di progettazione.

A. Si indichino quali sono gli errori e si descriva su quali funzionalità della CPU essi impattano

La CPU (implementata correttamente) esegue il seguente programma per cui si assuma una situazione iniziale di memoria e register file come riportata in figura.

```
        add $t2 $t3 $t4
        add $t2 $t4 $t5
        slt $t2 $zero $t2
        beq $t2 $zero L
        lw $t5 4($s0)
        and $t5 $t2 $t5
        j E
L:
        lw $t4 4($s0)
        or $t5 $t2 $t5
E:
        sw $t5 4($s0)
```

Memoria

	⋮
	40
	30
	20
($s0) →	10
	⋮

Register file

Caso 1

($t3) = 3
($t4) = 4
($t5) = 5

Caso 2

($t3) = 3
($t4) = 4
($t5) = −5

Si assuma inoltre che la sovrascrittura del PC effettuata da una jump avvenga nella fase di decode.

B. Si determino le criticità che la CPU deve affrontare durante l'esecuzione del programma nei casi 1 e 2. Per ogni criticità si indichino il suo tipo e le istruzioni coinvolte.
C. Per ciascuna criticità identificata nel punto precedente (entrambi i casi) si specifichino i valori dei segnali di controllo necessari alla sua risoluzione.
D. Quanti cicli di clock impiega la CPU per eseguire il programma in ciascuno dei due casi?
E. Si supponga che gli hazard dati vengano gestiti a *compile time* con l'inserimento di nop. Si riporti il programma risultante e si indichino quanti cicli di clock impiega la CPU per eseguirlo (entrambi i casi).

Esercizio 3 [6 punti]

Una cache set-associativa a 8 vie utilizza un meccanismo di sostituzione dei blocchi di tipo pseudo-LRU. Si assuma che tutti i bit utilizzati dal meccanismo siano inizialmente pari a 0 su ogni set.

A. Qual è il significato del termine "pseudo-LRU"? Quanti bit sono necessari in ciascun set della cache considerata sopra per implementare questo metodo?

Un set della cache riceve questa sequenza di accessi (B_i indica il blocco i all'interno del set).

Lettura di B_0
Lettura di B_2
Lettura di B_4
Scrittura I
Lettura di B_6
Scrittura II

B. Si rappresenti l'albero binario nel set tra il secondo e il terzo accesso.
C. Si indichino in quali blocchi vengono effettuati il primo e il secondo accesso in scrittura.

Esercizio 4 [6 punti]

Si consideri una cache set-associativa a 4 vie da $128KiB$, 2048 linee e indirizzamento a 32 bit.

A. Quanti bit sono necessari per l'indice di linea, l'offset di blocco e il tag? Quanti comparatori sono necessari nella sua implementazione?
B. Si calcoli la dimensione totale della cache in MB riportando i passaggi per ogni parametro ricavato.
C. Si calcolino numero di blocco, indice di linea, tag, offset di blocco (quale parola dentro al blocco) e offset di parola (quale byte dentro alla parola) dei seguenti dati:
   - La parola di memoria che contiene il byte il cui indirizzo è $0x00000027$
   - La parola di memoria successiva a quella del punto precedente
   - Il terzo byte della parola di memoria con indirizzo $0x00008204$
D. Si fornisca un esempio di programma che generi una cache miss senza considerare il cold start.
E. Si indichi il numero di vie nel caso in cui, a parità degli altri parametri dati inizialmente, si progettasse la cache con uno schema direct-mapped e con uno fully-associative.

Esercizio 5 [5 punti]

Si consideri il seguente dato

$$0xAFA$$

A. Si fornisca la codifica esadecimale di tale dato utilizzando un codice a ripetizione a distanza 3.
B. Si fornisca la codifica esadecimale di tale dato utilizzando un codice di Hamming a distanza 4.

Traccia[1] di soluzione

Esercizio 1

A. A/L: 82, lw: 36, sw: 36, branch: 40, jump: 35
B. $T_{medio} = 1.6 \times 10^{-08}$, $T_{tot} = 3.664 \times 10^{-06}$ (in secondi)
C. $T_{medio} = 1.13 \times 10^{-08}$, $T_{tot} = 2.5877 \times 10^{-06}$ (in secondi)
D. $T_{medio} = 1.835 \times 10^{-08}$, $T_{tot} = 4.2022 \times 10^{-06}$ (in secondi)

Esercizio 2

A. Manca la logica di sovrascrittura PC (PC+4 e prediction), manca comando $RegWrite$ nel RF, manca $<< 2$ nel calcolo BTA. (Ci sono altre parti mancanti, come il forwarding per le branch, che però non abbiamo mai rappresentato esplicitamente sul datapath. L'indicazione di queste parti è comunque considerata risposa corretta.)
B. Assumendo speculazione su *not taken*:
   a. Caso 1: add/slt (A/L), slt/beq (A/L), lw/and (lw), j/lw (controllo), sw/and (A/L)
   b. Caso 2: add/slt (A/L), slt/beq (A/L), beq/lw (controllo), sw/or (A/L)
C. Nel caso 1: forwarding per le A/L, una bolla per la criticità lw, un flush per la criticità di controllo; nel caso 2: forwarding per le A/L, un flush per la criticità di controllo (speculazione errata)
D. Nel calcolo ricordare che: la pipeline deve riempirsi all'inizio, le speculazioni errate o i salti incondizionati si rilevano sempre nella fase ID.
E. L'inserimento di nop va fatto una volta sola, a compile time.

Esercizio 3

A. Servono 7 bit.
B. Assumendo che 1 indichi il ramo destro e procedendo per livelli crescenti e da sinistra a destra: $1,0,0,1,1,0,0$
C. Prima scrittura in B_1, seconda scrittura in B_3

Esercizio 4

A. Indice di linea su 11 bit, offset di blocco su 4 bit (2 per la parola, 2 per il byte), tag su 17 bit. Servono 4 comparatori.
B. $0.149504\ MB$
C. $M(d) = 0x00000024$: $N(d) = 2, I(d) = 2, tag = 0, offset_{blocco} = 1, offset_{parola} = 0$;
   $M(d) = 0x00000028$: $N(d) = 2, I(d) = 2, tag = 2, offset_{blocco} = 2, offset_{parola} = 0$;
   $M(d) = 0x00008206$: $N(d) = 2080, I(d) = 32, tag = 1, offset_{blocco} = 1, offset_{parola} = 2$.
D. TBD
E. Direct-mapped $n = 1$, fully-associative $n = 8192$

Esercizio 5

A. $0xE38FFFE38$
B. $0x393E8$

[1] Viene riportata solo una bozza di soluzione per gli esercizi numerici da usare come guida. Le soluzioni complete non verranno pubblicate, ma possono essere discusse a lezione o durante il ricevimento.

Architettura degli Elaboratori II

Corso di Laurea in Informatica

Università degli Studi di Milano

compilare in stampatello

Cognome		*Note consegna*
Nome		
Matricola		
e-mail	*@studenti.unimi.it*	

Esempio di tema d'esame

- Il presente plico di x[1] pagine include 4 esercizi da svolgersi nella durata massima di 3 ore.
- Le soluzioni vanno riportate negli spazi delimitati che seguono il testo di ciascun esercizio, la soluzione ad un esercizio non può occupare spazio riservato ad un altro esercizio, non possono essere consegnati fogli aggiuntivi.
- È possibile prendere appunti, ma è vietata la consultazione di qualsiasi tipo di materiale.
- È consentito l'uso della calcolatrice non scientifica.
- È vietato l'uso di qualsiasi dispositivo di comunicazione.

[1] Questo documento non riporta gli spazi aggiuntivi per le risposte che invece saranno presenti durante l'esame vero e proprio.

Esercizio 1

Una CPU a ciclo multiplo esegue un programma definito dal blocco di codice riportato sotto. Si assuma che, prima di eseguire il blocco, il valore di un registro t_i sia pari a i e che nel registro s_0 vi sia un indirizzo di memoria dati valido.

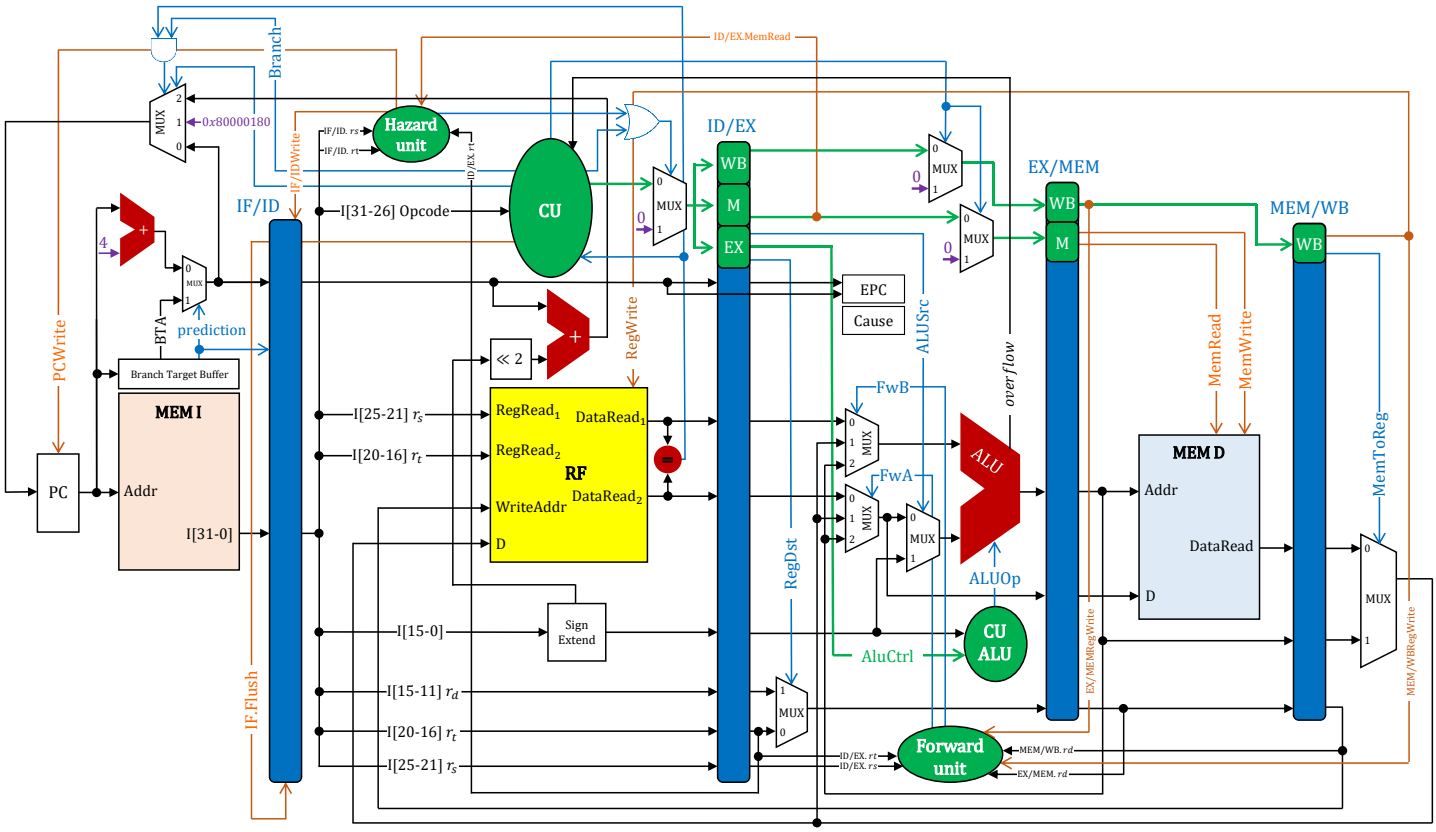
```
        add $t7 $t7 $t1
loop:
        beq $t7 $t0 continue
        lw $t3 0($s0)
        add $t3 $t3 $t3
        sw $t3 0($s0)
        add $s0 $s0 $t4
        sub $t7 $t7 $t1
        j loop

continue:
        add $v0 $t0 $t5
        add $v0 $v0 $t5
```

A. Sapendo che un accesso a memoria richiede $3\,ns$, un accesso al register file $0.5\,ns$ e un'operazione aritmetico/logica $1\,ns$, calcolare, mostrando e motivando i passaggi, il tempo medio di esecuzione di una istruzione e il tempo totale di esecuzione in riferimento al programma sopra riportato.
B. Rispetto al tempo medio sopra calcolato si indichi il guadagno o perdita percentuale che si avrebbe rispetto a una CPU a singolo ciclo descrivendo come le diverse architetture incidono su tale numero.

Esercizio 2

Si consideri la CPU pipeline riportata in figura.



Si consideri il seguente programma:

```
loop:   addi $s0 $zero 8
        lw $s1 0($s0)
        lw $t2 0($s1)
        add $t2 $t2 $t2
        sw $t2 0($s1)
        subi $s0 $s0 4
        beq $s0 $zero exit
        j loop
exit:
```

A. Si elenchino tutte le criticità che emergono dall'esecuzione del programma nella CPU sopra schematizzata, per ogni criticità si indichino le istruzioni coinvolte e il tipo di criticità.
B. Per ogni criticità identificata nel punto precedente si descriva come viene gestita dalla CPU indicando quali segnali e componenti del data path sono coinvolti.
C. Si determini una versione equivalente del programma mediante la tecnica del *loop unrolling*.
D. Si applichi al programma ottenuto nel punto C il *register renaming* e si dica quali vantaggi si ottengono con questa operazione.

Si consideri uno scheduling dinamico della pipeline in una CPU a lancio multiplo dove sono presenti 3 unità funzionali per istruzioni aritmetico-logiche e branch e 2 unità funzionali di tipo load/store.

E. Si schematizzi l'esecuzione del programma del punto D, indicando l'ordine con cui le istruzioni arrivano e diventano pronte nella commit unit.

Esercizio 3

Si consideri una cache a mappatura diretta da $2\ MiB$ composta da 512 linee.

A. Si calcoli dimensione di blocco, numero di bit necessari per l'indice di colonna e per il tag
B. Si calcoli la dimensione totale della cache
C. Si determinino indice di linea e offset dei seguenti dati
   - Parola di memoria all'indirizzo $0x000017B0$
   - Byte all'indirizzo $0x00003075$
D. Si ripetano i punti A e B considerando una cache completamente associativa, si descriva come funzionerebbe in questo caso l'accesso ad un dato

Esercizio 4

Si consideri il seguente dato codificato su 8 bit

$$0x4E$$

A. Si fornisca la codifica binaria di tale dato utilizzando un codice a ripetizione la cui distanza di Hamming sia pari a 4.
B. Si descriva la procedura di rilevazione e correzione degli errori per tale codice. Quanti e quali errori possono essere riconosciuti o corretti?