

Architettura degli elaboratori A.A. 2018-2019 Prima prova di verifica intermedia –
5/11/2019
Compito Tipo B

Indicare in alto a destra su tutti i fogli ricevuti il proprio nome, cognome, numero di matricola e tipo di compito (A o B).

Progettare la seguente unità funzionale U in modo da minimizzare **il numero di microistruzioni utilizzate**.

U riceve da un'unità esterna U1, prima un valore N ($N \leq 64$) e un bit OP che rappresenta un'operazione da svolgere (OP=0 moltiplicazione, OP=1 divisione intera) e successivamente N coppie di dati (tutti diversi da zero) di 32 bit, che costituiscono gli elementi $a[i]$ e $b[i]$ di due vettori a e b, e calcola gli elementi di un vettore c di $2N-1$ elementi, secondo la regola :

```
for i=0, i<N, i++
  {for j=i, j<N, j++
    {c[i+j]= c[i+j]+ a[i ] OP b[j ]}}
  }
```

I vettori a, b, e c sono memorizzati in tre memorie interne ad U, Ma ed Mb di 64 parole, Mc di 128 parole. Al termine del calcolo di c, U invia ad una terza unità, U2, il vettore c, un elemento alla volta, quando U2 è pronta a riceverlo.

Le porte logiche hanno un massimo di 8 ingressi, le ALU a disposizione richiedono un tempo pari a $6t_p$ per ogni operazione aritmetica che svolgono, è a disposizione una sola ALU che effettua sia moltiplicazioni che divisioni intere, mentre non c'è limite al numero di ALU che fanno addizioni o sottrazioni. Al termine della trasmissione di c ad U2, tutte le parole di Mc devono contenere zero.

Si chiede di fornire:

- 1) il microprogramma utilizzato
- 2) giustificazione della minimalità del numero di microistruzioni usate
- 3) le risorse di stato e di calcolo usate nella PO, e le loro connessioni
- 4) il calcolo della lunghezza del ciclo di clock, in termini di t_p
- 5) le espressioni booleane per σ_{PC}
- 6) la possibilità di usare il controllo residuo, e, in caso affermativo, dove può essere usato

Sfruttando tutte le possibilità di parallelizzazione, anche quelle che, con la tecnologia attuale, permettono di leggere e scrivere nello stesso registro o locazione di memoria durante la medesima microistruzione, ed eliminando tutte le NOP che non dipendono da comunicazione (queste sono ineliminabili), un possibile (tra i molti) **microprogramma** è il seguente:

```
// ricezione di N ed OP da U1
0. (RDY1=0) NOP, 0
(=1) INA → N, 0 → i, reset RDY1, set ACK1, 1
//ricezione dei vettori a e b
1. (RDY1, EQ(N,i)=0 -) NOP, 1
(=1 0) INA → Ma[i], INB → Mb[i], i+1 → i, reset RDY1, set ACK1, 1
(=1 1) 0 → i, 0 → j, 2
//controllo se fine calcolo o inizializzazione di c[i+j]
2. ( EQ(N,i), OP=0 0) Mc[i+j]+ Ma[i]*Mb[j] → Mc[i+j], j+1 → j, 3
( =0 1 ) Mc[i+j]+ Ma[i] div Mb[j] → Mc[i+j], j+1 → j, 3
(=1 -) 0 → i, SHL(N,1)-1 → N, 4
//calcolo di c o incremento di i e inizializzazione di j
3. ( EQ(N,j), OP=0 0) Mc[i+j]+ Ma[i]*Mb[j] → Mc[i+j], j+1 → j, 3
( =0 1 ) Mc[i+j]+ Ma[i] div Mb[j] → Mc[i+j], j+1 → j, 3
(=1 -) i+1 → i, i+1 → j, 2
// invio di c ad U2, ed azzeramento di c
4. ( EQ(N,i), ACK2=- 0 ) NOP, 4
(=0 1) Mc[i] → OUT, 0 → Mc[i-1], set RDY2, reset ACK2, 4
(=1 1) 0 → Mc[N-1], 0
```

Minimalità del numero di istruzioni utilizzate

Due istruzioni sono necessarie per l'ingresso dei dati: una per N, ed una per a e b; per l'uscita di c ce ne vuole una. Poi, ci vogliono due istruzioni, una per ogni ciclo for. Totale: 5 microistruzioni.

Risorse di stato e di calcolo

Oltre a RDY1 e ACK1 (per comunicare con U1), RDY2 e ACK2 (per comunicare con U2), l'interfaccia di uscita prevede due registri a 32 bit, INA e INB, ed un registro ad 1 bit, OP, per comunicare, rispettivamente, N ed il vettore a, il vettore b, e il tipo di operazione da effettuare. Quella con U2 prevede un registro a 32 bit, OUT, per inviare ad U2 il vettore c. Saranno inoltre necessari i vettori i ed N, di 7 bit, e j, di 6 bit, tutti dotati di un commutatore sugli ingressi: quello di i per selezionare tra 0 ed i+1, quello di N per selezionare tra INA o 2N-1, e quello di j per selezionare tra 0, j+1 e i+1. Inoltre, sono utilizzate tre memorie Ma, Mb, Mc come descritte nel testo. Mc sarà una memoria a doppia porta che permette di leggere e scrivere (anche nella stessa parola) nel medesimo ciclo di clock. Si useranno 6 ALU per addizioni o sottrazioni, ed una ALU per moltiplicazione o divisione intera (come richiesto dal testo): questo riduce il numero di commutatori necessari. Per il calcolo di 2N, si farà uno shift sinistro di una posizione: si inserisce direttamente nel registro N il suo bit i-esimo come (i+1)-esimo, si scarta il più significativo, e si inserisce uno 0 come bit meno significativo

(per effettuare l'operazione più velocemente). Infine, per testare l'uguaglianza o meno di due valori, (EQ), si userà un circuito costituito da confrontatori ad un bit in parallelo, le cui uscite saranno gli ingressi di un gate OR, con uscita negata, per un totale di 3tp.

Lunghezza del ciclo di clock

Il tempo di accesso alle memorie è dominato dalla lettura. Le memorie Ma ed Mb hanno al massimo $64=2^6$ parole, quindi indirizzi a 6 bit ed un bit ad 1 in ogni riga della tabella di verità il commutatore di lettura, quindi un solo livello di AND. Per l'OR, notiamo che l'uscita Z ha 64 valori ad 1, e quindi bastano due livelli di gate OR, con t_{aa} e t_{ab} (i due tempi di accesso)=3tp. Per Mc, invece, avendo $128=2^7$ parole, saranno necessari 3 livelli di gate OR, con $t_{ac}=4tp$.

Per la parte controllo, notiamo che ci sono 5 microistruzioni (quindi i bit di stato saranno 3), e 4 variabili di condizionamento, di cui al più 2 testate contemporaneamente. Inoltre, abbiamo 14 frasi: se i bit ad "1" di una uscita sono più di 8, codifichiamo gli "0" e complementiamo il risultato. Pertanto, sia per ωPC che per σPC bastano un livello di AND ed uno di OR, che porta ad avere $t_{\omega PC} = t_{\sigma PC} = 2tp$. L'unica variabile di condizionamento complessa è EQ, e richiede 3tp. Per quanto riguarda $t_{\sigma PO}$, la microoperazione più lenta è: $Mc[i+j] + Ma[i] * Mb[j] \rightarrow Mc[i+j]$, che richiede una ALU per calcolare i+j, seguita da un accesso ad Mc, e da un'altra ALU per la somma con il prodotto (o la divisione intera) tra Ma ed Mb, quest'ultimo effettuato in parallelo con le due operazioni precedenti. Segue la memorizzazione in Mc del risultato, per un totale di $2t_{alu} + 2t_{ac} = 20tp$. Pertanto, essendo $\tau = t_{\omega PO} + \max\{(t_{\omega PC} + t_{\sigma PO}), t_{\sigma PC}\} + \delta$, si ha che:

$$\tau = 3tp + 2tp + 20tp + 1tp = 26tp$$

Espressioni booleane per σPC

La tabella di verità per il nuovo stato sarà:

$s_2 \ s_1 \ s_0 \ RDY1 \ EQ \ OP \ ACK2 \ | \ s_2' \ s_1' \ s_0'$

0	0	0	0	-	-	-		0	0	0
0	0	0	1	-	-	-		0	0	1
0	0	1	0	-	-	-		0	0	1
0	0	1	1	0	-	-		0	0	1
0	0	1	1	1	-	-		0	1	0
0	1	0	-	0	-	-		0	1	1
0	1	0	-	1	-	-		1	0	0
0	1	1	-	0	0	-		0	1	1
0	1	1	-	0	1	-		0	1	1
0	1	1	0	1	-	-		0	1	0
1	0	0	-	-	-	0		1	0	0
1	0	0	-	0	-	1		1	0	0
1	0	0	-	1	-	1		0	0	0

da cui, ad esempio, $s_2' = \underline{s_2 \ s_1 \ s_0 \ EQ} + \underline{s_2 \ s_1 \ s_0 \ ACK2} + \underline{s_2 \ s_1 \ s_0 \ EQ \ ACK2}$.

Controllo residuo: Il controllo residuo si può avere usando OP come α per la ALU che fa moltiplicazione o divisione intera.