Esercizi per l'orale

1 Raddoppio in base 7, inserire cin e cout. Sintesi porte nor cout

Tabella di verità

x_2	x_1	x_0	c_{in}	z_2	z_1	z_0	cout
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	0	1	0	0	0
0	1	1	0	1	1	0	0
1	0	0	0	0	0	1	1
1	0	1	0	0	1	1	1
1	1	0	0	1	0	1	1
0	0	0	1	0	0	1	0
0	0	1	1	0	1	1	0
0	1	0	1	1	0	1	0
0	1	1	1	0	0	0	1
1	0	0	1	0	1	0	1
1	0	1	1	1	0	0	1
1	1	0	1	1	1	0	1

x_2x_0 \ $c_{in}x_1$	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	-	-	1
10	0	1	1	0

Sintesi NOR:

	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	1	-	-	0
10	1	0	0	1

$\rightarrow \overline{Cout} = \overline{(\overline{x_2}\cdot\overline{x_1}) + (\overline{c_{in}}\cdot\overline{x_2}) + (\overline{x_2}\cdot\overline{x_0})}$

$= Cout = \overline{(\overline{x_2}\cdot\overline{x_1})}\cdot\overline{(\overline{c_{in}}\cdot\overline{x_2})}\cdot\overline{(\overline{x_2}\cdot\overline{x_0})}$

$= (x_2+x_1)\cdot(c_{in}+x_2)\cdot(x_2+x_0)$

forma NOR

$\overline{\overline{(x_2+x_1)}+\overline{(c_{in}+x_2)}+\overline{(x_2+x_0)}}$

2) Montare 2 chip di ram 128K x 8bit in modo che rispondano agli indirizzi più alti di un calcolatore con spazio di indirizzamento di 2^{20} locazioni

Per accedere alle locazioni + alte implica la seguente cosa per il select

a_{19}	a_{18}	a_{17}	/s chip 1	/s chip 2
1	1	0	0	1
1	1	1	1	0

$\rightarrow /s\,chip1 = \overline{a_{19}\cdot a_{18}\cdot\overline{a_{17}}}$

$/s\,chip2 = \overline{a_{19}\cdot a_{18}\cdot a_{17}}$

tenere a_{19} e a_{18} a 1 mi assicura che sia agli indirizzi + alti

il valore di a_{17} mi discrimina quale dei 2 chip sto usando

da a_{16}-a_0 sono gli spazi dei chip inseriti

3 Dato A naturale di n cifre in base β, dimostrare che $|A|_m=0$ se e solo se $|A_0|_m=0$, con $m=\alpha*\beta$ $(\alpha\geq 1, \beta\geq 1)$

$$\cancel{|A_{mod}|_m} = |A|_{\alpha*\beta} = \left|A_0 + \sum_{i=1}^{n-1} A_i \beta^i\right|_{\alpha*\beta}$$

Se $\alpha=1 \rightarrow \sum_{i=1}^{n-1} A_i \cdot \beta^i$ è multiplo di m per cui diventa $|A|_m = |A_0|_m$

$$= \left||A_0|_{\alpha\beta} + |\Sigma A_i \cdot \beta^i|_{\alpha\beta}\right|_{\alpha\beta}$$

↳ è sicuramente divisibile per β $\rightarrow |\beta \Sigma A_i \cdot \beta^{i-1}|_{\alpha\beta}$

↓

$|\Sigma A_i \cdot \beta^{i-1}|_\alpha$.

Quindi se $\alpha>1$, $|A|_m=0=|A_0|_m$ se e solo se $\sum_{i=1}^{n-1} A_i \cdot \beta^{i-1}$ è multiplo di α

4 Si supponga di fornire un registro $AX=\{AH,AL\}$...

La Fase di fetch può essere ricondotta al formato FO in quanto l'unico operatore è un registro, quindi non è necessario fare alcuna lettura in memoria

Fase esecuzione

```
IncAx: Begin
  AH<=(AL==8'b11111111)?AH+1:AH;
  AL<=AL+1;
end

ShL: Begin
  AH<={AH[6:0],AL[7]};
  AL<={AL[6:0],1'b0};
end

ShR: Begin
  AH<={1'b0,AH[7:1]};
  AL<={AH[0],AL[7:1]};
end

Sar: Begin
  AH<={AH[7],AH[7:1]};
  AL<={AH[0],AL[7:1]};
```