INF2270, equivalence of Boolean expressions and making a decoder from a demultiplexer: example solution

P. Häfliger

May 3, 2010

Equivalence of two boolean expressions

$$
\begin{aligned}
F &= (a \vee \bar{d}) \wedge (a \vee \bar{c}) \wedge (a \vee \bar{b}) \;\wedge (\bar{c} \vee \bar{d}) &(1)\\
&= [a \vee (\bar{d} \wedge \bar{c} \wedge \bar{b})] \;\wedge (\bar{c} \vee \bar{d}) &(2)\\
&= [a \wedge (\bar{c} \vee \bar{d})] \vee [(\bar{d} \wedge \bar{c} \wedge \bar{b}) \wedge (\bar{c} \vee \bar{d})] &(3)\\
&= (a \wedge \bar{c}) \vee (a \wedge \bar{d}) \vee [(\bar{d} \wedge \bar{c} \wedge \bar{b}) \wedge \bar{c}] \vee [(\bar{d} \wedge \bar{c} \wedge \bar{b}) \wedge \bar{d}] &(4)\\
&= (a \wedge \bar{c}) \vee (a \wedge \bar{d}) \vee (\bar{d} \wedge \bar{c} \wedge \bar{b}) \vee (\bar{d} \wedge \bar{c} \wedge \bar{b}) &(5)\\
&= a \wedge \bar{c} \vee a \wedge \bar{d} \vee \bar{b} \wedge \bar{c} \wedge \bar{d} &(6)
\end{aligned}
$$

I chose to start with the expression derived by grouping the zeros in the K-map (1). In step (2), a is factored out from the first three brackets. That is the 'distributive' rule in the second column of the rules table in the compendium, used from the left to the right. In step (3) the term $(\bar{c} \vee \bar{d})$ is factored in, using the distributive rule in the first column from left to right. By factoring in a in the term $[a \wedge (\bar{c} \vee \bar{d})]$ in step (4) we obtain the first two minterm experessions of the expression derived from the ones. Also in step (4), the entire term $(\bar{d} \wedge \bar{c} \wedge \bar{b})$ has been factored into $(\bar{c} \vee \bar{d})$. We can now apply the rule $a \wedge a = a$ in step (5) to get rid of the extra $\bar{c}$ and $\bar{d}$ and finally use the rule $a \vee a = a$ to obtain the final minterm expression of the function derived by grouping the ones in step (6).

Building a decoder from a demultiplexer

The solution here is really simple. If you consider the similarity of the truth tables of decoder and demultiplexer it might become apparent: the only difference is that where the output is 1 for the decoder it is I for the demultiplexer. Thus, simply by setting the input of a demultiplexer $I = 1$ and renaming the input S to I, makes a decoder, as shown in figure 1

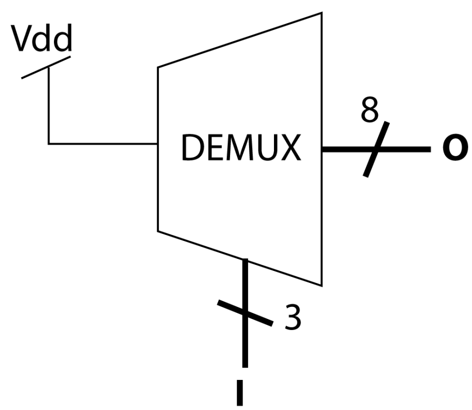

Figure 1: 3-bit decoder, possible implementation using a demultiplexer