INF3430 - Lab assignment 1
Design flow and VHDL

Magnus Andersen
magnuand@ifi.uio.no

September 20, 2013

1 Design flow

a)

Worked fine. Did not elaborate since the assignment text said that nothing needed to be shown.

b)

See enclosed files.

2 Decoder

See enclosed files.

3 Signals vs. variables

a)

By looking at the waveform, we see that the output changes its value at time $450ns$. While variables update immediately, signals on the other hand update when the process terminates. With this in mind, we can set up a table as shown in Table 1 to get a detailed peek at the value propagation.

time	rst_n	mclk	indata	data1*	data2	data3*	data4	data5*	outdata
$< 200ns$	...	....	0000	0000	0000	0000	0000	0000	0000
$200ns$	1	fall	1111	0000	0000	0000	0000	0000	0000
$250ns$	1	rise	1111	0000*	0000	0000	0000	0000	0000
$300ns$	1	fall	1111	1111	0000	0000	0000	0000	0000
$350ns$	1	rise	1111	1111	1111	0000*	0000	0000	0000
$400ns$	1	fall	1111	1111	1111	1111	1111	0000	0000
$450ns$	1	rise	1111	1111	1111	1111	1111	0000*	0000
$450ns + \delta$	1	....	1111	1111	1111	1111	1111	1111	0000*
$450ns + 2\delta$	1	....	1111	1111	1111	1111	1111	1111	1111

Table 1: Propagation from input to output

Here, $data1$, $data2$ and $data3$ are marked with a * to indicate that they are signals. $data2$ and $data4$ are variables (no *). I only wrote the four leftmost bits in the various datas, since the four rightmost ones never changes (to save space). When $mclk$ rises, an update are scheduled on the signals. The signal that will get a new value after the process terminates is marked with a * next to its current value. Finally, the δ-symbol means a delta cycle, i.e. an "inifinitely" small time delay. In conclusion we see that the output changes its value at time $450ns + 2\delta \approx 450ns$, as per the aforementioned waveform.

b)

By looking at the waveform, we see that the output changes its value from "UUUUUUUU" to "00000000" at time $100ns$, and from there to "11110000" at time $750ns$, and finally to "00001111" at time $850ns$. Just as in a), we can set up a table as shown in Table 2 to get a more detailed look at what's going on. Here $X = 00000000$. We also write U for "UUUUUUUU". I did this in order for Table 2 to stay within reasonable dimensions.

time	rst_n	mclk	indata	data1*	data2*	data3*	data4*	data5*	outdata
$0ns$	1	...	X	U	U	U	U	U	U
$50ns$	1	rise	X	$U*$	U	U	U	U	U
$100ns$	0	fall	X	X	$U*$	$U*$	$U*$	$U*$	U
$100ns + \delta$	0	...	X	X	X	X	X	X	$U*$
$100ns + 2\delta$	0	...	X	X	X	X	X	X	X

Table 2: Propagation from input to output

As we can see from Table 2, the reason $outdata = UUUUUUUU$ at time $50ns$ is that $data5$ remains uninitialized up until rst_n changes from 1 to 0 at time $100ns$ (after one delta cycle). Then it takes another delta cycle before $outdata$ syncs up with $data5 == indata$. We could easily have expanded Table 2 to show additional changes on $outdata$, but the assignment text simply asked for temporal information with regards to value changes, not *how* the values get there.

c)

To sum up what's happening here (in order):

1. $var1, var2 := indata$
2. $sig1 <= var1 = indata$ (scheduled)
3. $sig2 <= var2 = indata$ (scheduled)
4. $outdata(1\,downto\,0) <= var2 \& var1 = indata \& indata$ (scheduled)
5. $outdata(3\,downto\,2) <= sig2 \& sig1 = indata \& indata$ (scheduled)
6. $var1, var2 := \text{not}\, indata$
7. $sig1 <= \text{not}\, var1 = indata$ (scheduled)
8. $sig2 <= indata$ (scheduled)
9. $outdata(5\,downto\,4) <= var2 \& var1 = \text{not}\, indata \& \text{not}\, indata$ (scheduled)
10. $outdata(7\,downto\,6) <= sig1 \& sig2 = indata \& indata$ (scheduled)

From the numbered list above we see that $sig1$ and $sig2$ will both always get the value of the previous $indata$. This means that $outdata(1\,downto\,0)$ and $outdata(3\,downto\,2)$ always gets scheduled for the same value, since the order in which $sig1$ and $sig2$ get concatenated becomes irrelevant. Furthermore, since the variables get flipped after the first variable assignment, $outdata(5\,downto\,0)$ will always be the exact inversion of $outdata(7\,downto\,6)$ and vice versa.

d)

The reason is that $outdata(3\,downto\,2)$ and $outdata(7\,downto\,6)$ are based off of the signals $sig1$ and $sig2$. Since said signals have been removed from the sensitivity list, the process won't run again until $indata$ changes value (at time $100ns$). This in turn means that the $outdata$ parts in question will be scheduled for undefined values during the first process evaluation, and will remain undefined until $indata$ changes value. This obviously makes their values differ from c).

4 Component instantiation

a)

See enclosed files.

b)

See enclosed files.

c)

See enclosed files.