Lab session Finite-State Machines

Group A: October 26, 2009
Group B: October 27, 2009

Work in the given groups of two. Submit your solutions to the respective assignment on Blackboard. The file name is:
`a04_s0XXXXX_s0XXXXX.tar.gz`
One of the group members commits your solution. Keep an eye on the deadline (see Blackboard)!

1 Project

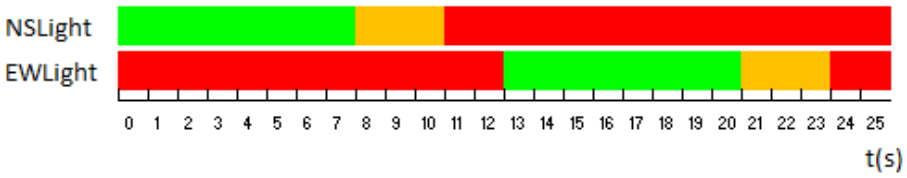
Read sections C.7, C.8 and C.10 of Appendix C.

1. Build an 8-bit register using D flip-flops. Inputs are 8-bit D (that denotes the input data) and C (the clock signal). 8-bit Q (that denotes the output data) is the only Output.

2. Build a register file made of 8 registers you just built. The register file must be able to read from and write to a specified register. The register file has (a) a register index number as input, denoting from/to which of the eight registers the user wants to read/write, (b) a bit selecting the operation (read or write), (c) an 8-bit word used as input for the write operation, and (d) a clock input. Additionally, the register file has an 8-bit word as output, that is used to return the value of a register in the read operation.

3. Build a finite-state machine that implements a traffic light system on a cross section. Finite-state machines use memory and a clock. Since finite-state machines are *synchronous*, a new state is computed every clock cycle. Use a 2 Hz clock (full clock cycle of 1 second). The two traffic lights behave like the following figure:

   Tip: use incrementors (adder that increments its input with 1) and registers. You can use the Logisim library components.


NSLight
EWLight
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
t(s)