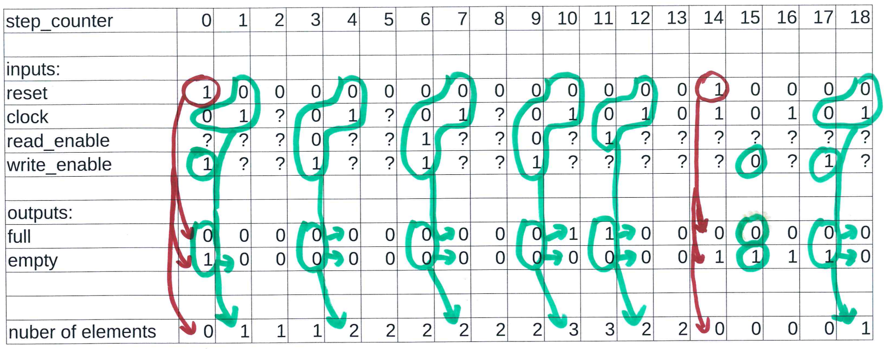
Homework 3 - buffer

Build an SMV module representing a hardware component of a buffer. For simplicity, we ignore the data and model only the control part. The buffer is parametrized by its size and has six wires. The first four wires serve for inputs (reset, clock, read_enable, and write_enable) and the last two are used for outputs (full and empty). On each wire, there is either low voltage or high voltage. Let us model the wires by boolean variables where FALSE stands for low voltage (i.e., 0) and TRUE represents the high voltage (i.e., 1). The first wire is an asynchronous (instant) reset. If there is a high voltage, the buffer goes instantly to the empty configuration. Other wires are operated in the synchronous mode triggered by a rising edge on the clock wire.

The expected behavior is demonstrated on the following trace for a buffer of size 3. We also explicitly print step-counter and the current number of elements stored in the buffer to increase readability. Question marks show positions where the value does not affect subsequent behavior.



step_counter	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
inputs:																			
reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
clock	0	1	?	0	1	?	0	1	?	0	1	0	1	0	1	0	1	0	1
read_enable	?	?	?	0	?	?	1	?	?	0	?	1	?	?	?	?	?	?	?
write_enable	1	?	?	1	?	?	1	?	?	1	?	?	?	?	?	0	?	1	?
outputs:																			
full	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
empty	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0
nuber of elements	0	1	1	1	2	2	2	2	2	2	3	3	2	2	0	0	0	0	1

The buffer is initialized by reset. Then a rising edge comes. Thanks to the write request expressed by high voltage in write_enable, a new element is added to the buffer, so it is no longer empty. The second element comes in the fourth step. There is one successful read and one successful write at step 7, so the number of the stored elements stays the same. The buffer gets full after step number 10. The subsequent successful read decreases the number of stored elements to 2. Note the question mark for write_enable at time 11. Even if it is 1, the write will fail due to the positive value in full at time 11. The buffer is reset at time 14, etc.

Write your own solution to file “hw03_buffer.smv” and configure the size by macro SIZE. You can test the specific behavior of the above-discussed picture by `nuXmv -pre cpp simple_test.smv`. Feel free to write other tests to be sure that your module works as expected also for other sizes and for other input sequences.