Handshake signaling for data transfer

Proposed problems

Oprițoiu Flavius
flavius.opritoiu@cs.upt.ro

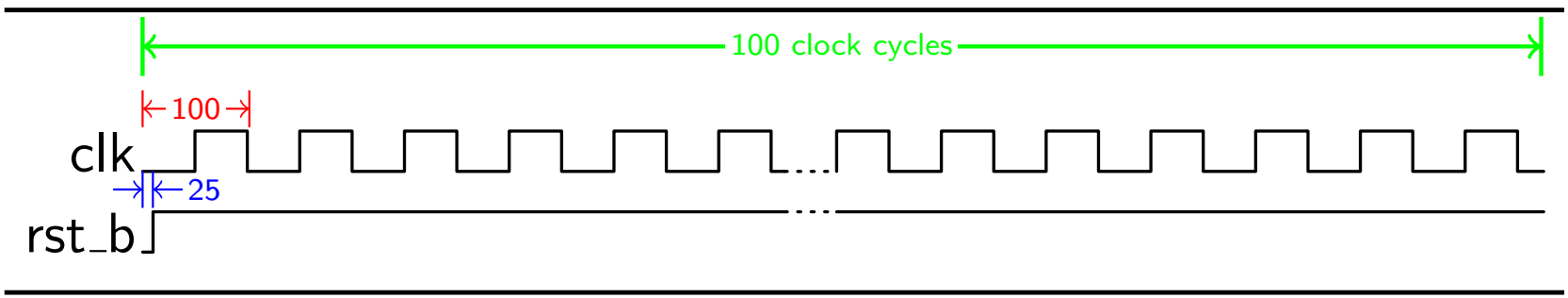
September 18, 2023

Problem 1

Build a module called *prod* (producer) having inputs *clk* (1-bit) and *rst_b* (1-bit) and outputs *val* (1-bit) and *data* (8-bits). The module produces random values between 0 and 5. Output *val* (*valid* handshaking signal) announces when the unit cannot produce new data ($val == 0$).

The module will generate data for at least 3 and at most 5 clock cycles after which it will wait at least 1 and at most 4 clock cycles without producing new data. Build:

1. the Verilog code
2. the script file, "run_prod.txt"
3. a testbench generating inputs as in the timing diagram bellow

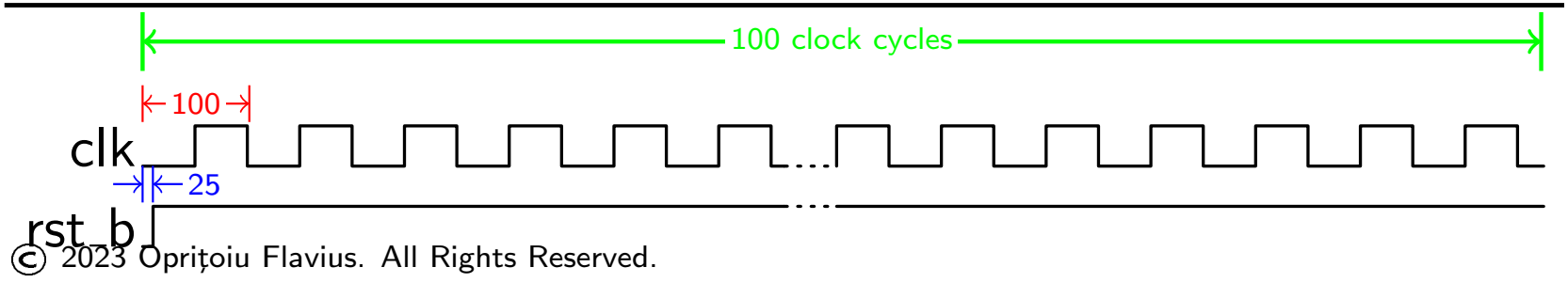


Problem 2

Construct a consumer module called *cons* (consumer) having inputs *clk* (1-bit), *rst_b* (1-bit), *val* (1-bit) and *data* (8-bits) and having one output *sum* (8-bit). The module consumes data generated by the producer. At the output *sum*, the consumer provides, at any moment, the sum of all monotonically increasing sequences of valid data (a valid data is part of a monotonically increasing sequence if it is greater or equal than the previous valid data). The testbench will instantiate the producer module from Exercise 1 and connects the 2 modules together.

Build:

1. the Verilog code
2. the script file, "run_cons.txt"
3. a testbench generating inputs as in the timing diagram bellow



Problem 3

Build the control unit described in the Solved problem of this week's material, called *sha2inctrl*. The testbench will instantiate the *sha2indpath* module constructed last week, which will be controlled by this control unit and connects the two modules together.

Build:

1. the Verilog code
2. the script file, "run_sha2inctrl.txt"
3. a testbench generating inputs as in the timing diagram bellow