Parallel Data Processing in Reconfigurable Systems

Lab nº 1

Objectives

- Familiarization with Vivado software
- Modeling in VHDL, synthesis and implementation of finite state machines
- Development of test benches and simulation
- Preparation of test data for experiments
- Synthesis, implementation and test on Nexys-4 prototyping board

Part I – Design of a reaction timer

A reaction timer circuit measures how fast a human hand can respond to a visual stimulus. This circuit operates as follows:

- The circuit has two input pushbuttons corresponding to reset and start_stop signals.
- It uses a LED as the visual stimulus.
- The measured time is displayed on four 7-segment displays in a format SSTT (where SS are seconds [0..99] and TT are hundredths of seconds [0..99]).
- When ready, the user presses the start_stop button to initiate the test. The 7-segment displays are off and the LED is off.
- After 3 seconds, the LED goes on and the timer starts to count. The value of the counter is shown on the displays.
- After the stimulus LED goes on, the user has to press the start_stop button as soon as possible. The timer stops counting and the displays show the reaction time.
- To start a new test, the user presses the start_stop button once again.
- The circuit can be reset at any time by pressing the reset button.

Figures 1 and 2 show a possible block diagram of the circuit (to be implemented on a Nexys-4 board). VHDL source code of all the components is available on the course site. The same BCD counter is reused to measure both the initial waiting time and the user reaction time.

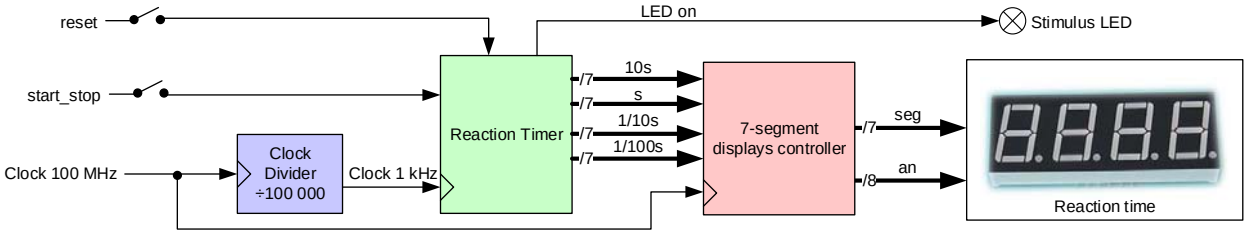

Figure 1 – Top-level architecture of the reaction timer.

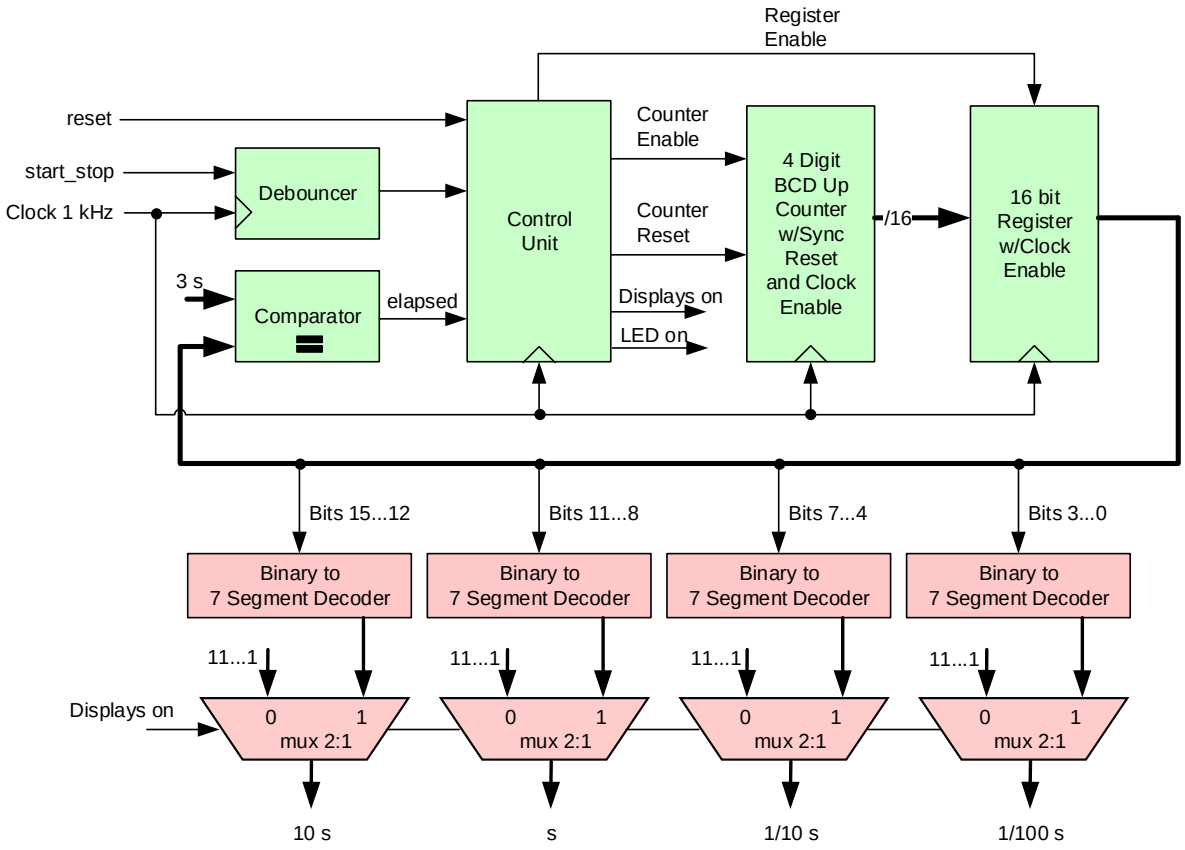

Figure 2 – Architecture of the block “Reaction Timer” from Fig 1.

1. Create a new project in Vivado for the FPGA of the Nexys-4 board.
2. Import the available VHD files of all the project components.
3. Set the file “ReactionTimer.vhd” as the top level of the project.
4. Import the constraints file “Nexys4_Master.xdc”.
5. Synthesize and implement the project and test it on the board. The following buttons control the reaction timer:
   - btnC – *reset*
   - btnU – *start_stop*

6. The reaction timer is controlled by the module *ReactionControl*, which implements a Moore finite state machine whose diagram is illustrated in Fig. 3. Analyze the given VHDL code and check if it corresponds to the state diagram from Fig. 3.

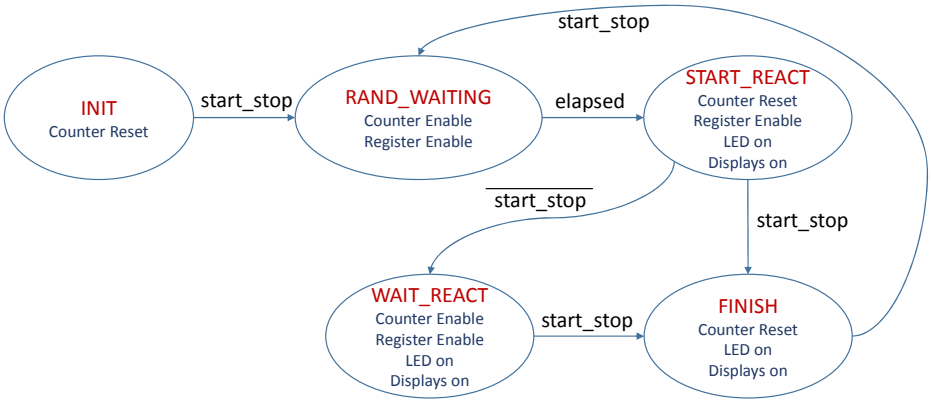

Figure 3 – State diagram of the reaction timer control module.

7. The implemented reaction timer has quite a limited functionality because it always switches on the stimulus LED after 3 seconds. It would be more interesting if the timer would activate the stimulus at random times. Modify the module *ReactionControl* and the architecture from Fig. 2 so that the LED would be switched on 0-9 seconds after pressing the button start_stop.

Part II – Generation of test data

The objective of this part is to model a RAM in VHDL and fill it in with some test data for experiments.

1. Describe in VHDL a single-port RAM for storing 16 8-bit words (with synchronous read and write). Follow the write-first synchronization mode, i.e. new content is immediately made available for reading. Initialize the RAM with data specified in the text file "Test.data" (shown below).

```
00000001
00000010
00000011
00000100
00000101
00000110
00000111
00001000
00001001
00001010
00001011
00001100
00001101
00001110
00001111
11111111
```

2. Write a test bench that would permit to test the functionality of the RAM. Execute simulation in Vivado and check the results.
3. Instantiate the RAM in a top-level VHDL file. Set the RAM clock frequency to 1 Hz (use a frequency divider provided in part I).
4. Synthesize and implement the project and test it on the board. Use the following I/O components to interact with the RAM. Do not forget to modify the file "Nexys4_Master.xdc" accordingly.
   - sw(11..8) – *read/write address*
   - sw(7..0) – *write data*
   - *btnC – write enable*
   - *led(7..0) – read data*
5. Check the utilization reports produced in Vivado and indicate whether the RAM was synthesized as a distributed memory or as a block RAM.
6. Change the implementation model through ram_style attribute and test once again the circuit.