Laboratory 5

(Due date: **011**: November 8th, **005**: November 9th, **007**: November 10th)

Objectives

- ✓ Describe synchronous circuits in VHDL.
- ✓ Learn Testbench generation for synchronous circuits.

VHDL Coding

- ✓ Refer to the Tutorial: VHDL for FPGAs for examples and parametric code for register.

First Activity (100/100)

Design Problem

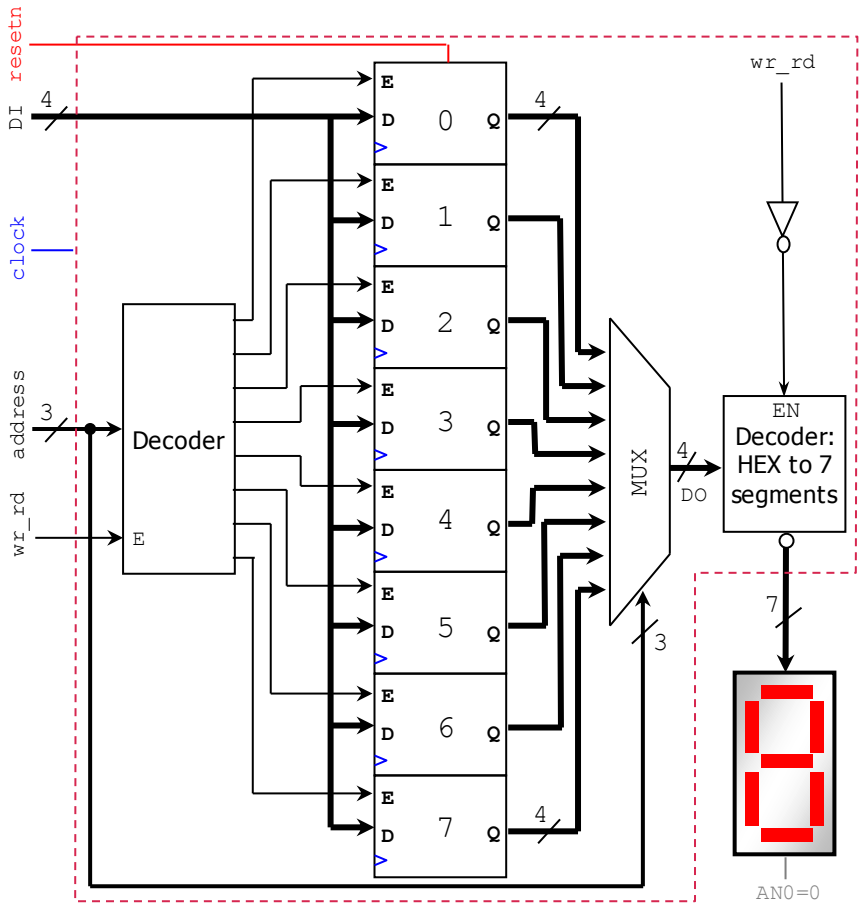
RANDOM MEMORY ACCESS (RAM) EMULATOR:

- The following circuit is a memory with 8 addresses, each address holding a 4-bit data. The memory positions are implemented by 4-bit registers. The *resetn* (active low) and *clock* signals are shared by all the registers. Data is written onto (or read from) one of the registers.

- **Memory Write** ($wr\_rd = 1$): The 4-bit input `DI` is written into one register. The `address[2..0]` signal selects the register to be written. Here, the 7-segment display must be OFF. Example: if `address= "101"`, then `DI` is written into register 5.

- **Memory Read** ($wr\_rd = 0$): The MUX output appears on the 7-segment display (hex. value). The `address[2..0]` signal selects the register from which data is read. For example, if `address= "010"`, then data in register 2 appears on the 7-segment display. If data in register 2 is `"1010"`, then the symbol '`A`' appears on the 7-segment display.

- Only one 7-segment display should be activated.



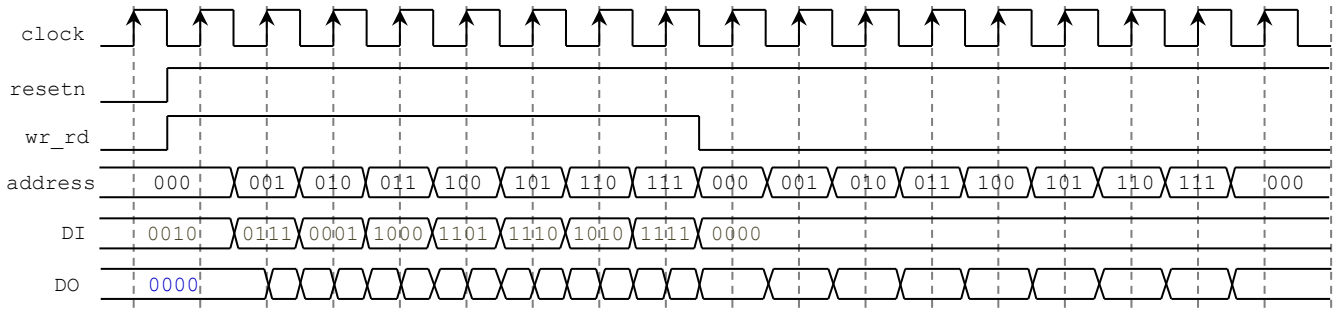
Procedure

- **Vivado: Complete the following steps:**

  - ✓ Create a new Vivado Project. Select the corresponding Artix-7 FPGA device (e.g.: the XC7A50T-1CSG324 FPGA device for the Nexys A7-50T).
  - ✓ Write the VHDL code for the given circuit. Synthesize your circuit to clear syntax errors.
    - Use the Structural Description: Create a separate `.vhd` file for i) Register with enable, ii) Bus MUX, iii) decoder with enable, iv) HEX-to-7 segments decoder (with enable and active-low outputs), and v) top file.

  - ✓ Write the VHDL testbench to simulate your circuit.
    - The testbench should be written according to the timing diagram shown in the figure. There are 8 writes (each on a different memory address), and then 8 reads (each from a different memory address). You must generate a 100 MHz input clock with 50% duty cycle.

Instructor: Daniel Llamocca
TAs: Michael Bowers, Abd Alrahman Al Nounou



- ✓ Perform Functional Simulation and Timing Simulation of your design. **Demonstrate this to your TA**.
  - Behavioral Simulation: Add the internal signal `DO` the waveform view. Go to: SCOPE window: testbench → UUT. Then go to Objects Window → Signal(s) → Add to Wave Window. Then, re-run the simulation.
  - If your circuit works correctly, the data written on `DI` will appear on `DO` (when wr_rd=0) for the corresponding address.
    - Complete the timing diagram in the figure based on your Vivado simulation (you can use hexadecimals for the cases when wr_rd=1).

- ✓ I/O Assignment: Generate the XDC file associated with your board.
  - Suggestion (for Basys 3, use `SW15` instead of `CPU_RESETN` for *resetn* input)

Board pin names	CLK100MHZ	CPU_RESETN	SW7	SW6-SW4	SW3-SW0	CA-CG	AN7-AN0
Signal names in code	*clock*	*resetn*	wr_rd	$address_2$-$address_0$	DI_3-DI_0	CA-CG	AN_7-AN_0

  - The board pin names (except `CPU_RESETN`) are used by all the listed boards (Nexys A7-50T/A7-100T, Nexys 4/DDR, Basys 3). I/Os: Note that `CA-CG` and `AN7-AN0` are active low; whereas the switches are active high.

  - Note: synchronous circuits always require a clock and reset signal.
    - **Reset signal**: As a convention in this class, we use active-low reset (*resetn*). As a result, ensure that *resetn* is tied to the proper board resource:
      - Nexys A7-50T/A7-100T, Nexys 4/DDR: For *resetn*, use `CPU_RESETN` pin. This is an active-low push button.
      - Basys 3: There is no active low push button. Thus, for *resetn*, use `SW15`. Even though `SW15` is active high, we can still think of it as active-low *resetn*, where the circuit is reset when the switch position is OFF ('0').
    - **Clock signal**: Like other signals in the XDC file, you need to uncomment the lines associated with the clock signal and replace the signal label with name used in your code. In addition, there is parameter `-period` that is set by default to `10.00`. This is the period (in ns) that your circuit should support.
      - Nexys A7-50T: In these lines, replace the label `CLK100MHZ` with the signal name you use in your code (`clock`):

```
set_property -dict { PACKAGE_PIN E3    IOSTANDARD LVCMOS33 } [get_ports { CLK100MHZ }];
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK100MHZ}];
```

      - Basis 3: In these lines, replace the label `clk` with the signal name used in your code (`clock`):

```
set_property PACKAGE_PIN W5 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
```

- ✓ Generate and download the bitstream on the FPGA and test. **Demonstrate this to your TA**.

SUBMISSION

- Submit to Moodle (an assignment will be created):
  - ✓ This lab sheet (as a .pdf) completed and signed off by the TA (or instructor)
  - ✓ (As a .zip file) all the generated files: VHDL code files, VHDL testbench, and XDC file. **DO NOT submit the whole Vivado Project**.
    - Your .zip file should only include one folder. Do not include subdirectories.
    - It is strongly recommended that all your design files, testbench, and constraints file be located in a single directory. This will allow for a smooth experience with Vivado.
    - You should only submit your source files AFTER you have demoed your work. Submission of work files without demoing will be assigned NO CREDIT.

TA signature: _____ **Date:** _____

Instructor: Daniel Llamocca
TAs: Michael Bowers, Abd Alrahman Al Nounou