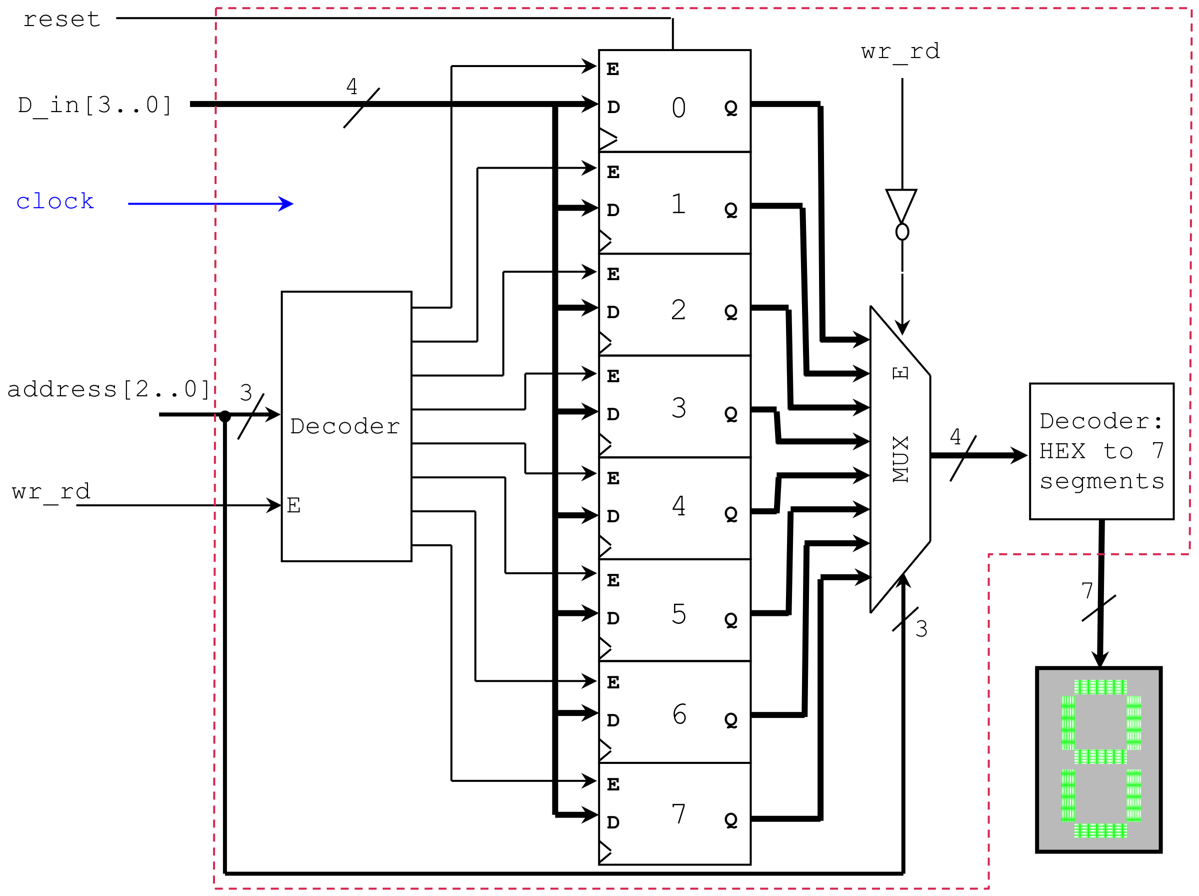
Extra points (+12) - Midterm

(Due date: October 31st @ 9:30 am)

RANDOM-ACCESS MEMORY EMULATOR

- The following sequential circuit represents a memory with 8 addresses, where each address holds a 4-bit data. The memory positions are implemented by 4-bit registers. The `reset` and `clock` signals are shared by all the registers. Data is written or read onto/from one of the registers (selected by the signal '`address`').
- **Writing onto memory** (`wr_rd = 1`): The 4-bit input data (`D_in`) is written into one of the 8 registers. The `address` signal selects which register is to be written. Here, the 7-segment display must show `0`. For example: if `address = "101"`, then `D_in` is written into register 5.
- **Reading from memory** (`wr_rd = 0`): The MUX output appears on the 7-segment display (hexadecimal value). The `address` signal selects the register from which data is read.
  For example: If `address = "010"`, then data in register 2 must appear on the 7-segment display. If data in register 2 is '`1010`', then the symbol '`A`' appears on the 7-segment display.



- ✓ Write the VHDL code for this circuit. Use the ***Structural description***: create a different file for: i) register with enable, ii) MUX with enable, iii) decoder with enable, iv) HEX to 7-segments decoder, and v) top file. (7 pts)
- ✓ Testbench code generation and simulation: Verify that your circuit works in both Behavioral and Post-Route Simulation. Your testbench must verify the operation of the memory for at least two writes onto memory and two reads from memory. You must generate an input clock of 100 MHz (3 pts).
- ✓ Implement your circuit on the NEXYS3 Board. Use any 7-segment display as an output. (2 pts).
  - `D_in [3..0]`: Connect these input bits to `SW3|SW2|SW1|SW0`
  - `address[2..0]`: Connect these input bits to `SW6|SW5|SW4`
  - `wr_rd`: Connect this input to `SW7` `reset`: Connect this active-high input to `BTNS`
  - `clock`: Connect this input to the 100 MHz clock (connected to pin `V10`)