Computer Systems 414 Tut 2 15 May 2021

Walt Deyzel: 21750793

Q1) Who are the people that must be present at a design review and what documentation do they need?

Team members discuss the design during the review. The team include: Designers, Review leader, Review Scribe, Other team members, sample clients or users etc.
Documents include: Code listings, flow charts, specifications, minuets etc.

Q2) What is a PDR and list four aspects that a PDR document should address?

PDR -> Preliminary Design Review
Must include the following:

1) Different designs (solutions) for a given problem.
2) Technical analysis/review of the project.
3) Risk assessment and review. Making changes if nessasery.
4) Project execution vs planned timeline review. Determine if on schedule to complete the design.

Q3) The most common port types in VHDL are 'in', 'out' and 'inout'. Describe briefly what each means.

IN, OUT, INOUT are different modes a port can take on within an entity. IN refers to an input port which can read data. OUT refers to an output mode that can update data. INOUT can read and update values.

Q4) When are latches or flip-flops generated by VHDL code?

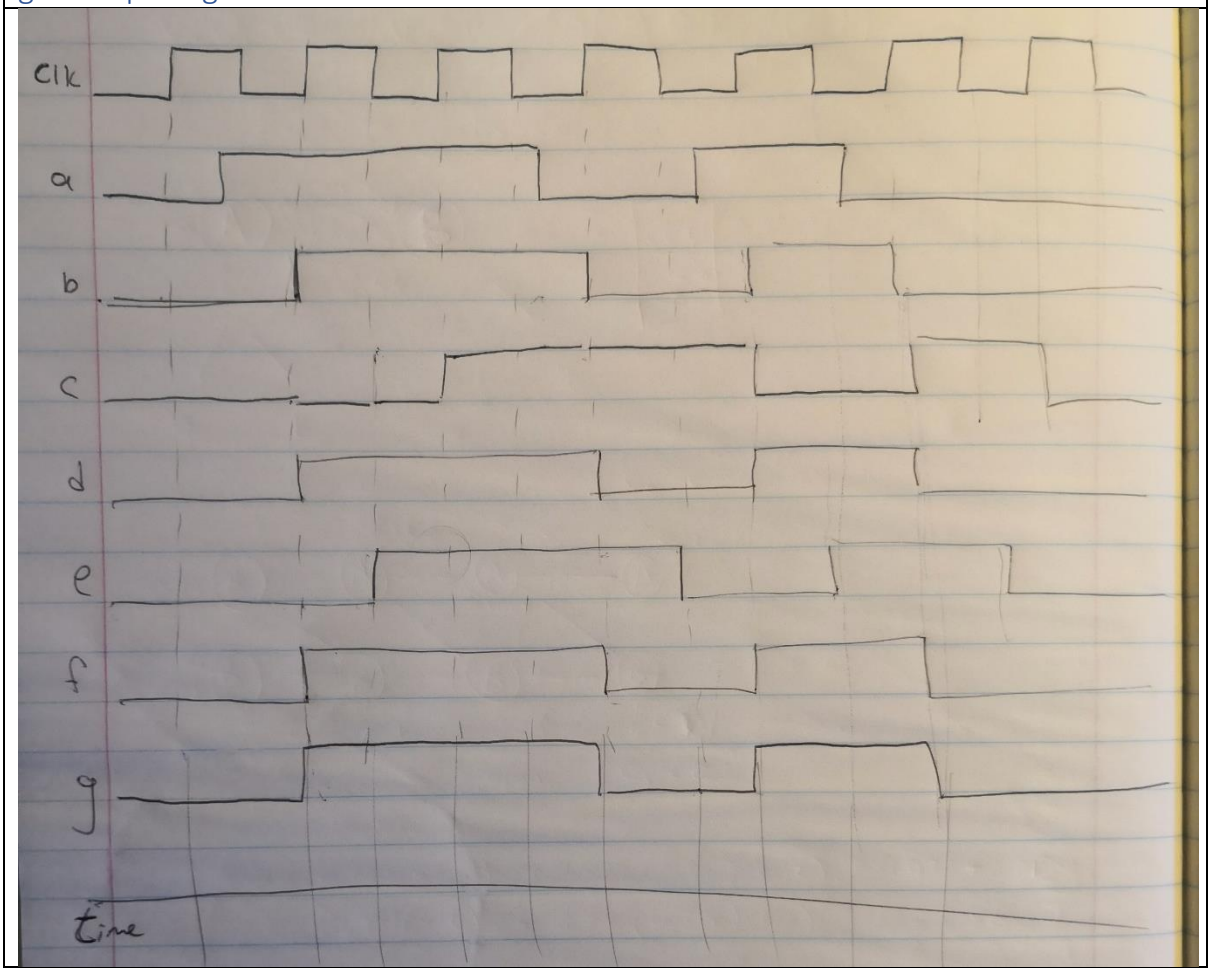
When the IF statement without its else is used a Latch is invered. If the conditional is true the output will latch to the condtiotional.
For a flip-flop we are looking for a conditional of a clock signal and a rising edged before a new output is generated otherwise it will keep the previous output.

Src: http://web.engr.oregonstate.edu/~traylor/ece474/vhdl_lectures/inferring_storge_elements.pdf

Q5) Why do VHDL state machines need the clock signal in the sensitivity list?

The Process will only execute if it is triggered. This happens when one of the signals on the sensitivity list changes value. The clock signal changes value at a rate of the clock speed making it a convenient option.

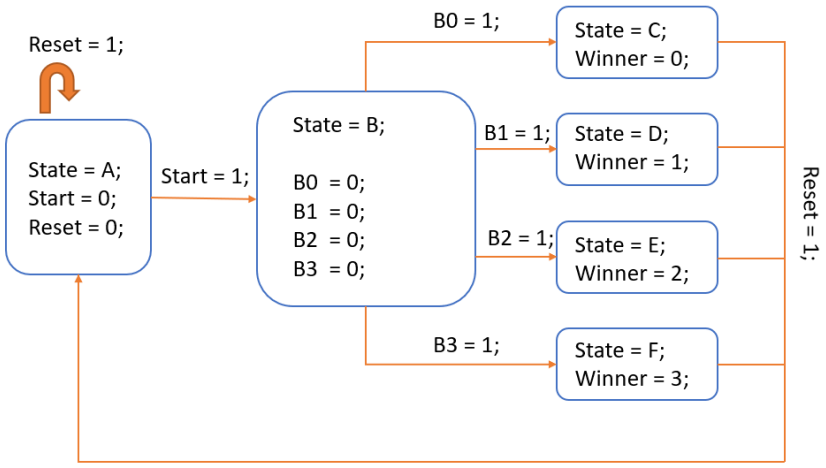
Q6) Sketch the signals and variable (b, c, d, e, f and g) for the design below and given input signals.



Q7) The VHDL code below is supposed to generate a slow clock but contains functional and syntax errors. Correct the errors.

```
1   LIBRARY ieee;
2   USE ieee.std_logic_arith.ALL;
3   USE ieee.std_logic_1164.all;
4   use ieee.numeric_std.all;
5
6   ENTITY vraag3 IS
7       PORT( clk : IN STD_LOGIC;
8       slow_clock : OUT STD_LOGIC );
9   END vraag3;
10
11  ARCHITECTURE logic OF vrg3 IS
12      --constant divider : integer range 0 to 255 := 55;
13      Integer div:
14      constant divider : div := 55;
15      VARIABLE counter : integer range 0 to 255;
16      signal slow_clock := '0';
17  BEGIN
18      PROCESS( clk )
19      BEGIN
20          IF rising_edge(clk) then
21              IF counter = divider THEN
22                  slow_clock <= not slow_clock;
23              ELSE
24                  counter := counter + 1;
25              end if;
26          END IF;
27      END process;
28  END logic;
```

Q8.a) Draw the state machine diagram of the design for the reaction-time game. Indicate all states and the conditions which determine the states.



Q8.b) Show the full VHDL code for the design.

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity gameblock is
    port(start, reset, clk: IN std_logic;
    -- Game buttons 0-3; start 4 and reset 5
    BUTTON : IN STD_LOGIC_VECTOR(0 TO 5);
    winner : OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
end gameblock;
architecture structure of gameblock is
    type states is (A, B, C, D, E, F);
    signal current: states := A;
begin
    process(clk)
    begin
        if clk 'EVENT' and clk ='1' then
            case current is
                -- Game buttons 0-3; start 4 and reset 5
                when A=> if buttons(4)='1' then current <= B;
                    elsif buttons(5) = '1' then current <= A;

                when B=> if buttons(0)='1' then current <= C;
                    elsif buttons(1) = '1' then current <= D;
                    elsif buttons(2) = '1' then current <= E;
                    elsif buttons(3) = '1' then current <= F;

                when C=> if buttons(5)='1' then current <= A;
                when D=> if buttons(5)='1' then current <= A;
                when E=> if buttons(5)='1' then current <= A;
                when F=> if buttons(5)='1' then current <= A;
            end case;
            end if;
    end process;
    winner(0) <= '1' when current = C else '0';
    winner(1) <= '1' when current = D else '0';
    winner(2) <= '1' when current = E else '0';
    winner(3) <= '1' when current = F else '0';
 end structure
```