Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

SEMESTER 1 EXAMINATIONS 2022/2023

MODULE: EE496 - Computer Architecture and HDL

PROGRAMME(S):

ECE	BEng Electronic & Computer Engineering
ECSAO	Study Abroad (Engineering & Computing)
ECEI	BEng Electronic & Computer Engineering

YEAR OF STUDY: 4,O

EXAMINER(S):

Dr. Xiaojun Wang	(Internal)	(Ext:5808)
Prof. Martin Glavin	(External)	External

TIME ALLOWED: 2 Hours

INSTRUCTIONS: **Answer 4 questions. All questions carry equal marks.**

PLEASE DO NOT TURN OVER THIS PAGE UNTIL YOU ARE INSTRUCTED TO DO SO.
The use of programmable or text storing calculators is expressly forbidden.
Please note that where a candidate answers more than the required number of questions, the examiner will mark all questions attempted and then select the highest scoring ones.

There are no additional requirements for this paper.

QUESTION 1 ***[TOTAL MARKS: 25]***

Q 1(a) **[5 Marks]**

Briefly explain the underlying representation of a VHDL model and how VHDL handles concurrency.

Q 1(b) **[5 Marks]**

The following is the Truth table of a 4 to 2 *priority encoder*. Write a synthesisable VHDL model for it.

Inputs				Outputs		
Y3	Y2	Y1	Y0	A1	A0	V
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

Q 1(c) **[5 Marks]**

Sketch the logic circuit inferred from your VHDL model developed in Q1(b).

Q 4(d) **[10 Marks]**

Draw the logic circuit implied by the following VHDL model. Then, explain the inference of each of the logic components.

```
library ieee;
use ieee.std_logic_1164.all, ieee.numeric_std.all;
entity Q1d is
  port (d1, d2, clk, en : in  std_logic;  q1, q2 : out std_logic);
end entity Q1d;
architecture rtl of Q1d is
  signal s1, s2: std_logic;
begin
 p1: process(clk) begin
        if rising_edge(clk) then
          if en = '1' then
            s1 <= d1;
            s2 <= s1;
            q1 <= s2;
          end if;
        end if;
     end process p1;
p2: process
      variable v1, v2 : std_logic;
    begin
      wait until falling_edge(clk);
        if en = '1' then
          v1 := d2;
          v2 := v1;
          q2 <= v2;
        end if;
    end process p2;
end architecture rtl;
```

[End of Question 1]

QUESTION 2 ***[TOTAL MARKS: 25]***

Q 2(a) **[6 Marks]**

Give the range of decimal numbers that an N-bit two's complement number can represent.

[2 marks]

If M is an N-bit two's complement number and N > n, write a *hardware-efficient* VHDL expression without using multiplication and division operators that

i) implements *multiply* M by 2^n

[2 marks]

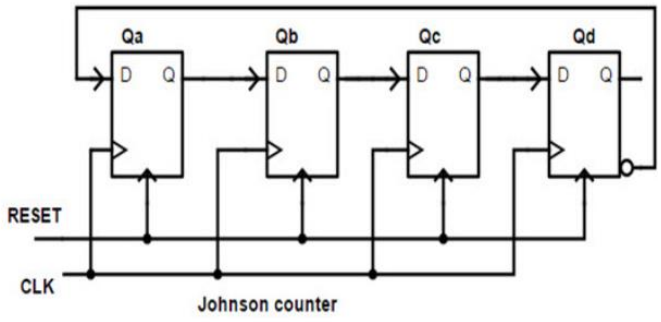
ii) implements *divide* M by 2^n

[2 marks]

Q 2(b) **[7 Marks]**

The following are the counting sequence and the logic circuit of a 4-bit Johnson counter. Write a VHDL model for the 4-bit Johnson counter. The reset input resets all DFFs to '0'. Clk is the clock input, and Qa, Qb, Qc, and Qd are the outputs.

Q_A	Q_B	Q_C	Q_D
0	0	0	0
1	0	0	0
1	1	0	0
1	1	1	0
1	1	1	1
0	1	1	1
0	0	1	1
0	0	0	1
repeat			



Q 2(c) **[7 Marks]**

Briefly describe VHDL signals, variables and shared variables, and their differences.

Q 2(d) **[5 Marks]**

If a signal `packet` is of subtype `std_logic_vector(N-1 downto 0)`, where `N` is an integer type generic constant declared in the corresponding entity declaration. Write a VHDL fragment that computes the *even* parity of the `packet`, placing the single bit result into a signal `prty` of subtype `std_logic`.(*note that `prty` should be `'1'` when there is an even number of bits with value `'1'` in the `packet`*).

[End of Question 2]

QUESTION 3 ***[TOTAL MARKS: 25]***

Q 3(a) **[12 Marks]**

Figure Q 3(a) shows a black box view and the state transition diagram of a traffic light controller finite state machine (FSM). T_A and T_B are the traffic sensors inputs to the FSM, L_A and L_B are the traffic lights outputs of the FSM. Assume the clock input is `clk`, and the synchronous `reset` input is *active high*.

(i) Write a *synthesisable* VHDL model for the traffic light controller FSM *using one-hot state encoding*. [9 marks]

(ii) Is it a Moore machine or a Mealy machine? [3 marks]

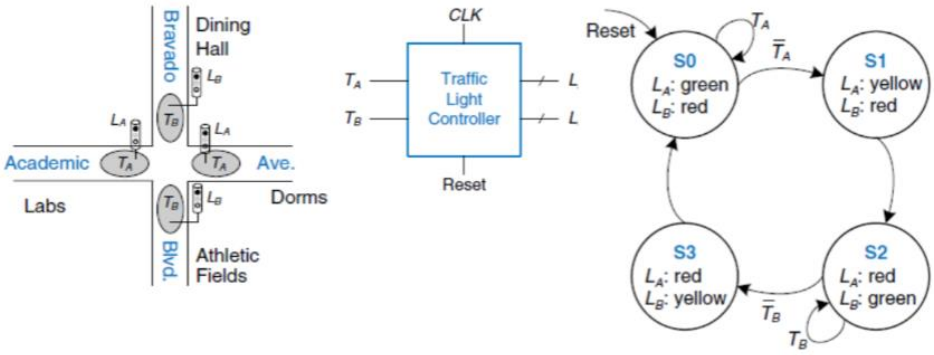


Figure Q3(a)

Q 3(b) **[5 Marks]**

Briefly describe the *memory hierarchy* of a modern computer system.

Q 3(c) **[8 Marks]**

Figure Q3(c) shows a 16-bit bidirectional IO buffer of a device connected to a *data_bus*. The device can write its output data (*data_out*) to the *data_bus* when its output is enabled (i.e. OE = '1'), and can read data (data_in) from the *data_bus* when its output is disabled (OE = '0', i.e $\overline{OE}$= '1'). Write a VHDL model that infers this bidirectional IO buffer.

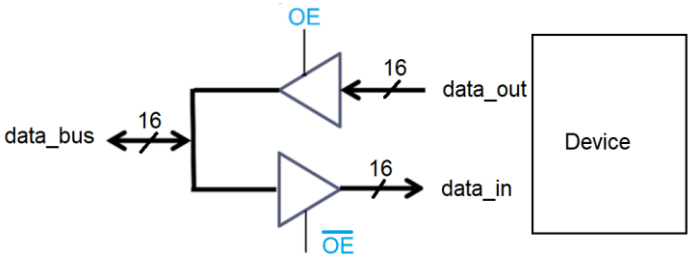


Figure Q 3(c) Bidirectional IO buffer

[End of Question 3]

QUESTION 4 ***[TOTAL MARKS: 25]***

Figure Q4(a) shows the timing constraints in sequential logic designs. In which t_{pcq} is the clock-to-Q propagation delay, t_{pd} is the combinational logic delay between two registers, and t_{setup} specifies the minimum time that the inputs to a register have to be stable before the rising edge of the clock for the register to sample the inputs correctly. The minimum clock period T_c can be represented as $T_c \geq t_{pcq} + t_{pd} + t_{setup}$. Moreover, t_{pcq} and t_{setup} are determined by the technology that will implement the design, so the only variable under the control of a design engineer is t_{pd}, which has to satisfy the condition $t_{pd} \leq T_c - (t_{pcq} + t_{setup})$.

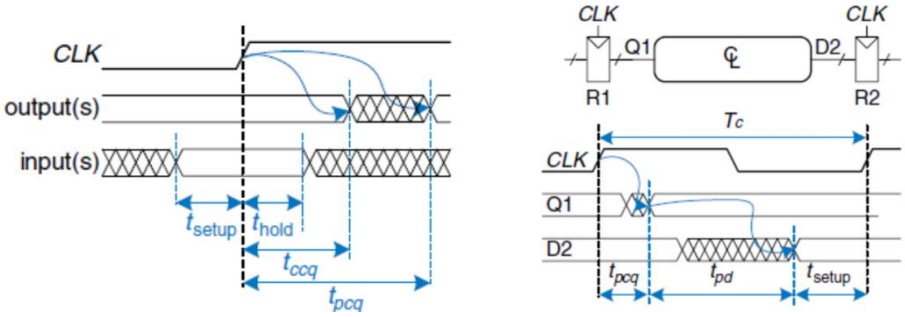


Figure Q4(a) Time constraints of sequential circuits

The following VHDL model adds four 64-bit numbers, and Figure Q4(b) shows the Vivado synthesised circuit. Assume that the clock-to-Q propagation delay t_{pcq} is 50 ps, the register setup time t_{setup} is 40 ps, and a 64-bit addition takes 2 ns.

```
library ieee;
use ieee.std_logic_1164.all,ieee.numeric_std.all;
use ieee.numeric_std_unsigned.all;
entity addition is
generic (width : positive := 64);
port( clk : in std_logic;
      di0 : in std_logic_vector(width-1 downto 0);
      di1 : in std_logic_vector(width-1 downto 0);
      di2 : in std_logic_vector(width-1 downto 0);
      di3 : in std_logic_vector(width-1 downto 0);
       do : out std_logic_vector(width-1 downto 0));
end addition;

architecture syn of addition is
   type reg_file is array (3 downto 0) of
                    std_logic_vector(width-1 downto 0);
   signal REG : reg_file;
begin
   -- register inputs at the rising edge of clk
   REG <= (di3, di2, di1, di0) when rising_edge(clk);

   -- addition and register results at the rising edge of clk
   do <= REG(0)+REG(1)+REG(2)+REG(3) when rising_edge(clk);
end syn;
```

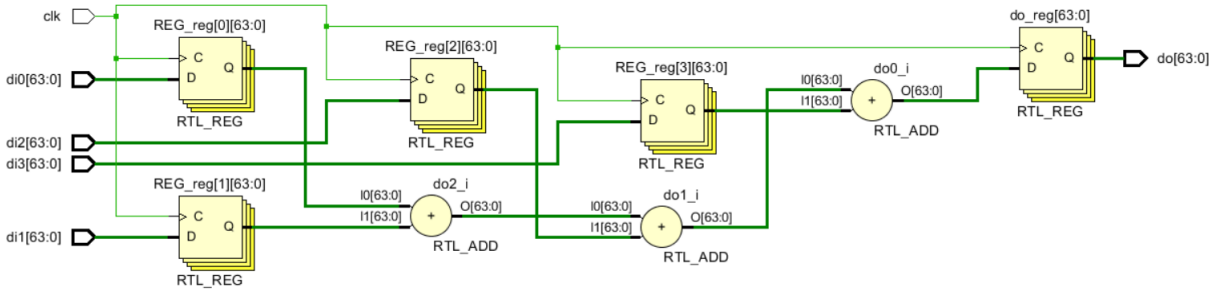


Figure Q4(b) Vivado synthesised circuit

Q 4(a) **[5 Marks]**

What is the minimum clock period T_c of the synthesised circuit in Figure Q4b?

Q 4(b) **[8 Marks]**

(i) Write an improved VHDL model, so the synthesised circuit is faster than the circuit in Figure Q4(b) without additional hardware resources. (*You only need to write the modified part, no need to write the full model*).

[3 marks]

(ii) Draw the synthesised logic circuit of your improved VHDL model.

[3 marks]

(iii) What's the minimum clock period T_c of this improved design?

[2 marks]

Q 4(c) **[12 Marks]**

(i) Improve the VHDL model to infer a pipelined implementation to increase the system throughput significantly. (*You only need to write the modified part, no need to write the full model*).

[6 marks]

(ii) Draw the synthesised logic circuit of your improved VHDL model.

[3 marks]

(iii) What's the maximum throughput of this pipelined design?

[3 marks]

[End of Question 4]

Q 5(a) **[5 Marks]**

Assuming a multi-cycle computer datapath has five pipeline stages:

1. Hardware to support an instruction *fetch*
2. Hardware to support an instruction *decode*
3. Hardware to support instruction *execution* (i.e. the ALU)
4. Hardware to support a *memory* read or write
5. Hardware to support the *write-back* of the ALU result to the register-file

Assume that each of the above stages takes the amount of time specified in the table below.

Fetch Instruction from Memory	**Decode** & read operands from reg files	Instruction **Execution** ALU	**Memory** Read/Write	Results **write-back** to Reg files
200 ps	120 ps	160 ps	200 ps	80 ps

Given the times for the *datapath* stages listed above, what would the *clock period* be for the entire *datapath*?

Q 5(b) **[5 Marks]**

For the pipelined *datapath* in Q5(a), assuming no hazards or stalls, how long does it take to execute *one* instruction, or what is the instruction *latency*?

Q 5(c) **[5 Marks]**

What is the *throughput* of the pipelined datapath in Q5(a)?

5(d) **[10 Marks]**

For **a single-cycle processor** with the same data path time specification as in the table of Q5(a), What would be the clock period, the instruction latency and the throughput of the single-cycle processor?

[End of Question 5]

[END OF EXAM]