



EEE4120F Class Test 2021



Date: 31 May 2021

ANSWER SHEET

STUDENT NUMBER: _____ STUDENT NAME: SAMPLE SOLUTION

Please make sure to legibly write both your student number and your name above.

Part 1 Answers : Multiple Choice

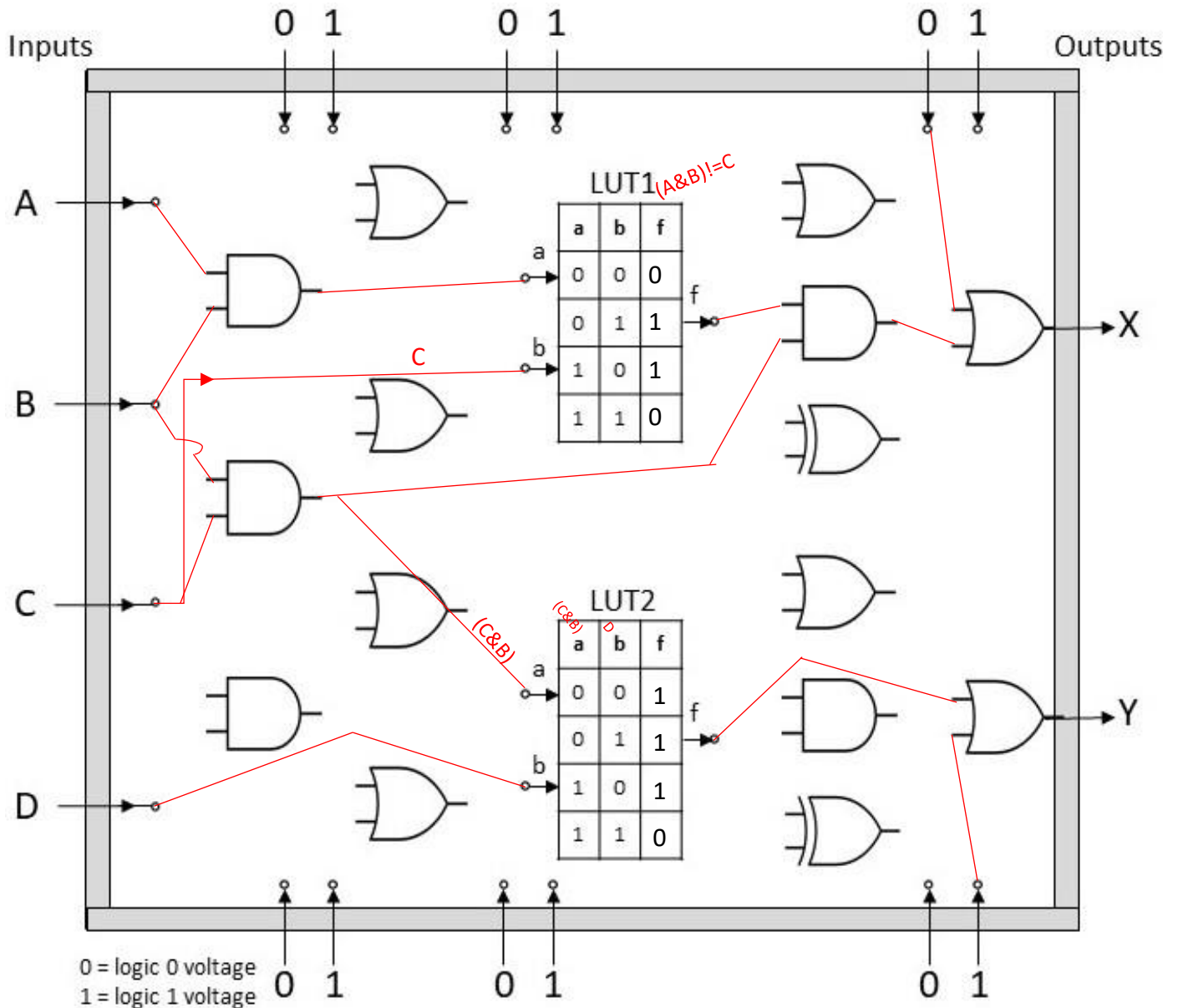
Please circuit your choice of answer below.



NOTE: please fill in your answers in pen by the end of the test so that it is a more permanent record of your answer selection (i.e. to avoid your selection is accidentally getting erased or having selected two options where you can only select one). Any answers in pen will be priority over pencil answer.

Question	Answer Selection (circle one unless otherwise instructed)					
Q 1.1	A	B	C	D	E	5 marks
Q 1.2	A	B	C	D	E	5 marks
Q 1.3	A	B	C	D	E	5 marks
Q 1.4	A	B	C	D	E	5 marks
Q 1.5	A	B	C	D	E	5 marks
Q 1.6	A	B	C	D	E	5 marks
Q 1.7	A	B	C	D	E	5 marks
Q 1.8	A	B	C	D	E	5 marks
Q 1.9	A	B	C	D	E	5 marks
Q 1.10	A	B	C	D	E	5 marks

50 marks total for this part

STUDENT NUMBER: _____ **SAMPLE SOLUTION**



Comments: you just need to draw lines indicating the wires that are connecting the parts. You might want to use hops  to emphasise wires are not connected and dots  to show that they are.

Workspace / explanations / assumptions you want to add:

This space provided to allow student to motivate / explain solution or assumptions made.

Marking tips: Award full 15 marks for an accurate solution that looks to get the right result.

Award reasonable marks (e.g. 10/15) for answer that at least connects things up properly, no missing connections (open inputs) for an essential path. It obviously doesn't matter if the gates inputs are not all tied to a value (e.g. 0V), as per this solution gates not in a critical path are just open circuit. (anyone being silly and tying an output to 0V or 1V should get a -1 penalty)

Part 2 Answers Question 2.2 – 2.3

STUDENT NUMBER: SAMPLE SOLUTION

Question 2.2:

Please indicate the calculated propagation delay here:

Calculated Propagation delay = 26ns ← fill in answer here!

Tick if it is a speedup or slowdown :

8 marks, but needs WORKING and decent speedup or slowdown shown!! If a correct-looking answer but no working then max 4/8 given.

SPEEDUP

SLOWDOWN

← tick / circle one option

but should check, i.e. micro/propdel. Should be > 4.

Indicated value of speedup or slowdown of FPGA over CPU: 7.69 ← fill in!

Please show your working below: (you can use your exam answer book for rough work)

YOUR WORKING:

Longest back-tracing path is the one from X to A, has XOR, AND, LUT and AND

→ delays of $7\text{ns} + 5\text{ns} + 9\text{ns} + 5\text{ns} = 26\text{ns}$

CPU is needing 200ns. So speedup is $200/26 = 7.69$

Question 2.3:

Answer to what would make the FPGA reprogrammable, so that connections from inputs to specific LEs can be changeable. (you can continue to answer on back of this page)

You would need to add multiplexers so as to choose which of the inputs

A – D to link to which of the gates or LUTs. If you can connect any input to any gate, these multiplexers may be big. If you can connect any output of a gate to the input of any other gate, that is going to be massive multiplexers.

Somewhere around one gate output hooking up to possible $2 \times 16 = 32$ inputs of other gates. To program that one multiplexer for an input would require a register of 5 bits to just select which input to hook up. With 17 gates with

two inputs each that would need $17 \times 2 \times 5 \text{ bits} = 170 \text{ bits}$. Which is not so

much, but this is a rather small FPGA. More practical reprogrammable support

would be having slices, like a slice 1 for the first set of AND and OR gates, a slice 2

for the LUTs and slice 3 for LUT outputs (and perhaps inputs) hooking up those gates to the outputs.

7 marks for decent explanation showing the student knows a thing or two of what would be needed. If a short superficial response max 3 unless it is a really clever and concise response that explains what is needed very briefly.

Part 3 Answer

STUDENT NUMBER: ANSWER

Question 3: the module interface has been provided as a starting point, you can add to the interface if you want to. Assume that one loop of the C++ while loop provided equates to the time it takes between positive clock edges of the clk input.

```
module flywheel (reset, clk, blip, period
                );
    // input and outputs for the module: (add more if you need to)
    input reset, clk, blip;
    output reg [31:0] period;

    // add your code solution here:

    // EEE4120F Class Test 2021 Verilog placing and routing
    // decider module

    module flywheel (reset, clk, blip, period);
        // input and outputs for the module
        input reset, clk, blip;
        output reg [31:0] period;
        reg [32:0] counter;

        always@ (negedge (reset)) // i.e. assume below done on reset
            begin
                counter = 0;
            end // always@

        always@ (posedge (clk)) // below happens on clk positive edge
            begin
                counter = counter + 1;
            end // always@

        always@ (posedge (blip)) // period is set to counter and
        counter resets
            begin
                period = counter;
                counter= 0;
            end // always@

    endmodule
```

View on EDAPlayground at:

<https://www.edaplayground.com/x/R7sZ>

[20 marks]

20 marks for a good and correct effort that includes some useful comments.

This is a rather easy problem, especially if one is just solving it with a bunch of always statements in this way. A partial, but incomplete attempt can get around 10, up to max 15 for something really close but not quite there.