

i UNIVERSITY OF OSLO
The Faculty of Mathematics and Natural Sciences
Written examination IN3160, IN4160
2022 SPRING
Duration: 2022-06-02, 09:00 to 2022-06-02, 13:00
Permitted aids: none

It is important that you read this front page before you start.

Multiple choice exercises

There is no penalty for answering wrong. The amount of boxes that can be checked cannot be greater than the number of correct answers.

Information about hand drawings(Scantron)

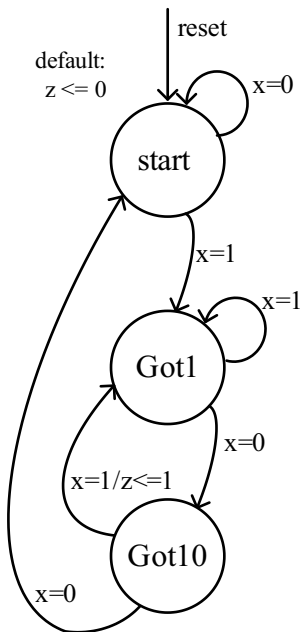
In this question set you have the opportunity to answer with hand drawings (question 8 and 10). You use the handed out sketch sheets. It is possible to use several sheets per assignment. See instructions on how to fill in the sketch sheets in the link below the assignment overview.

You are not supposed to submit hand drawings for any other questions than question 8 and 10.

You will NOT be given extra time to fill in the information boxes on the sketch sheets (assignment codes, candidate number, etc.)

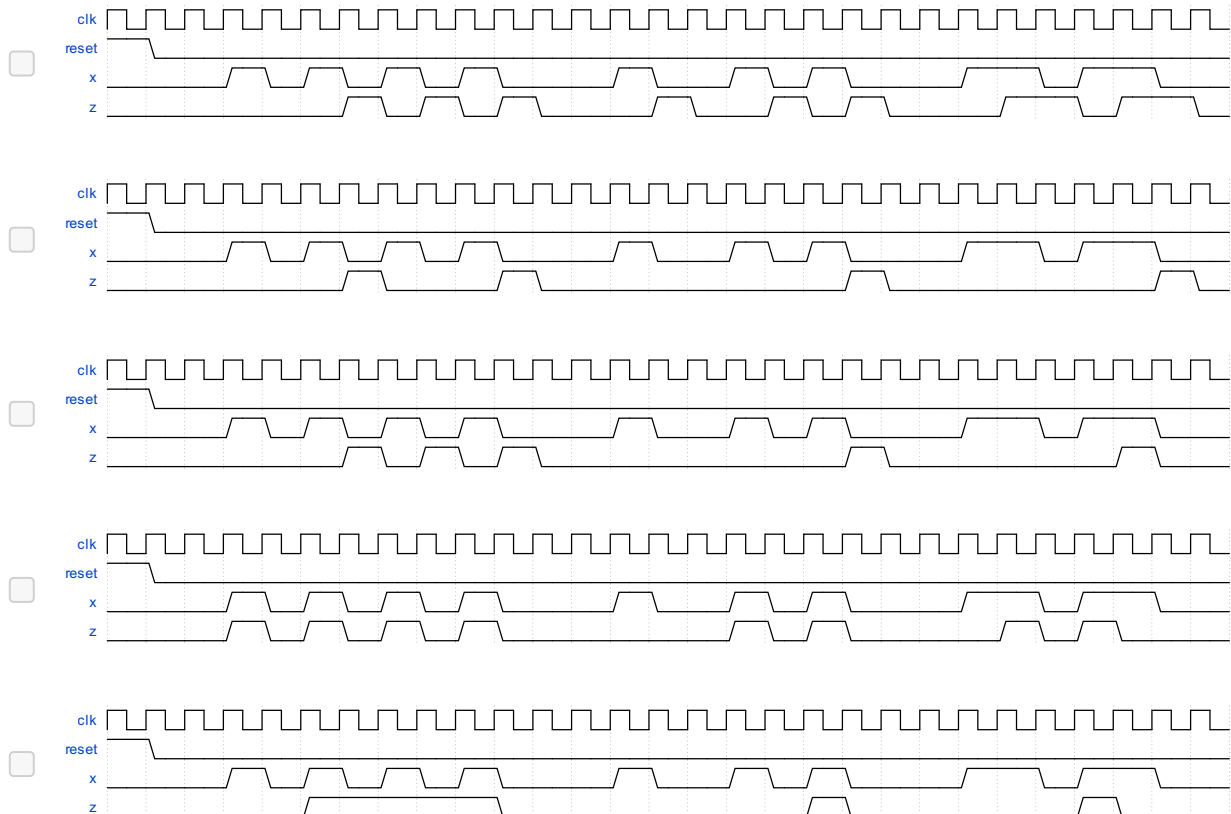
Hand drawings will be collected by the invigilator after the submission. Please do not leave your seat before your hand drawings has been collected. The process of collecting the hand drawings is not counted as a part of the exam duration, and may extend past the submission deadline.

Good luck!



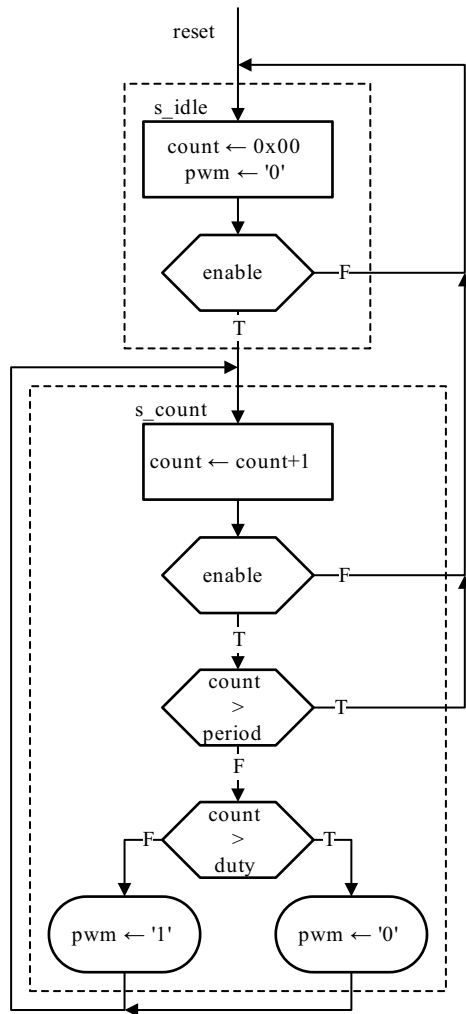
The bubble state diagram above describes the behavior of a 101 sequence detector circuit. Which of the following diagrams correspond to the behavior described in the state diagram?

Select one or more alternatives:



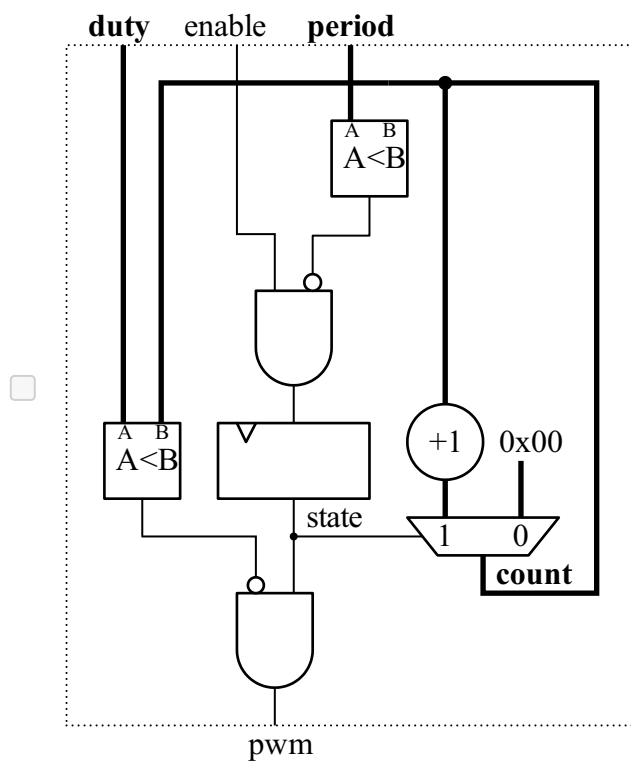
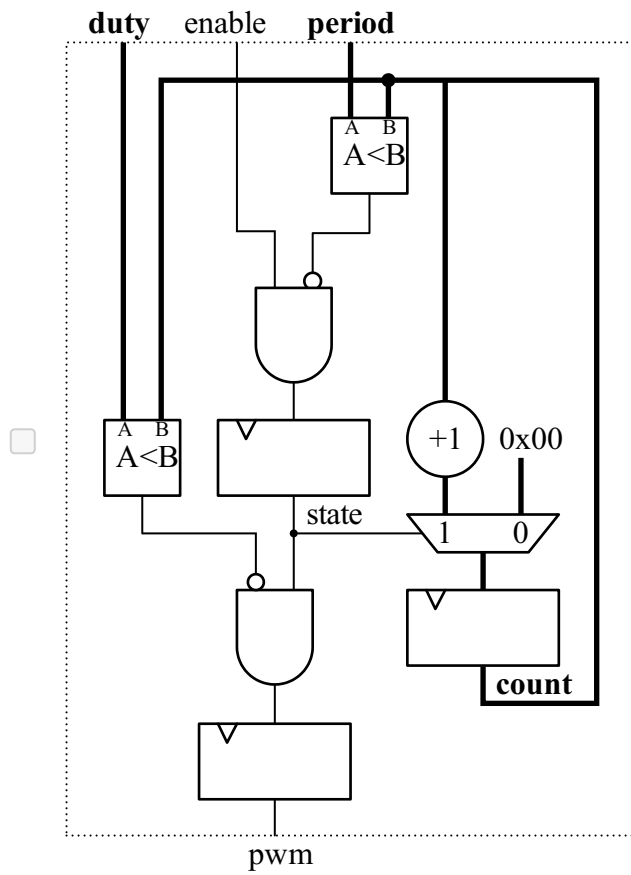
Maximum marks: 5

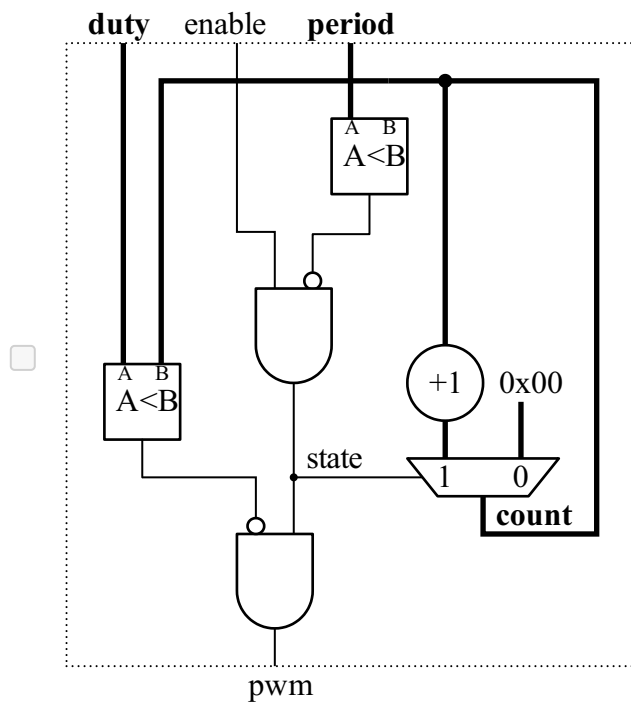
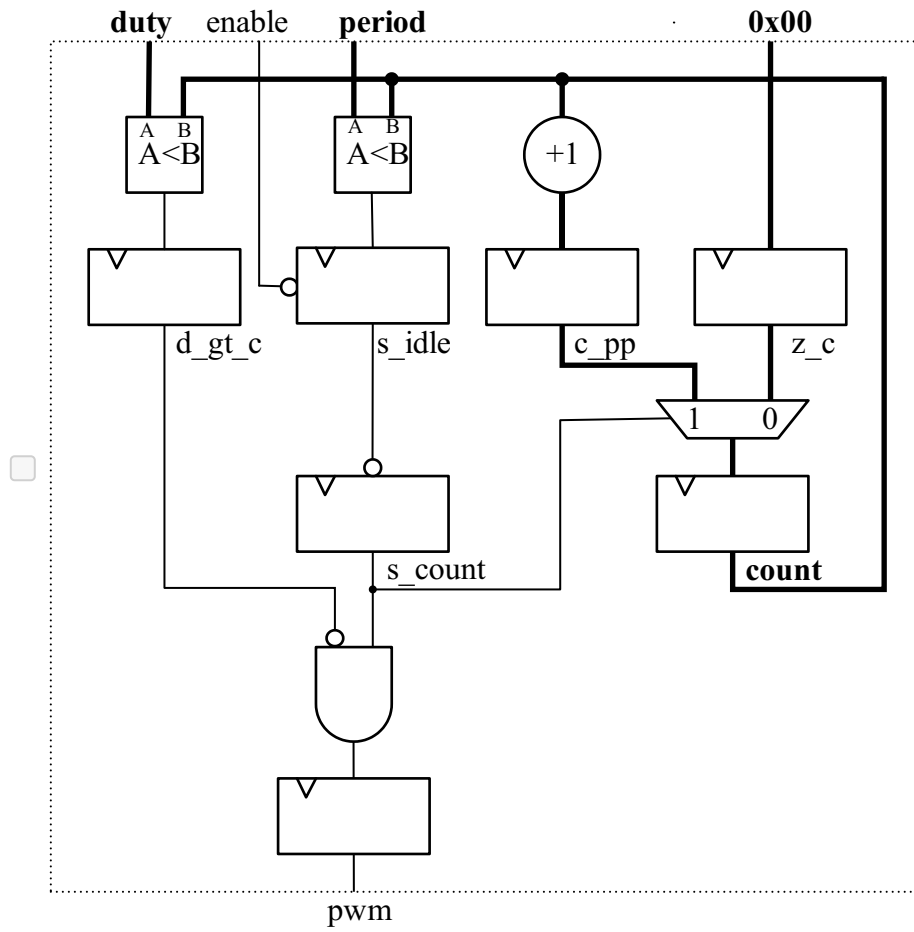
2

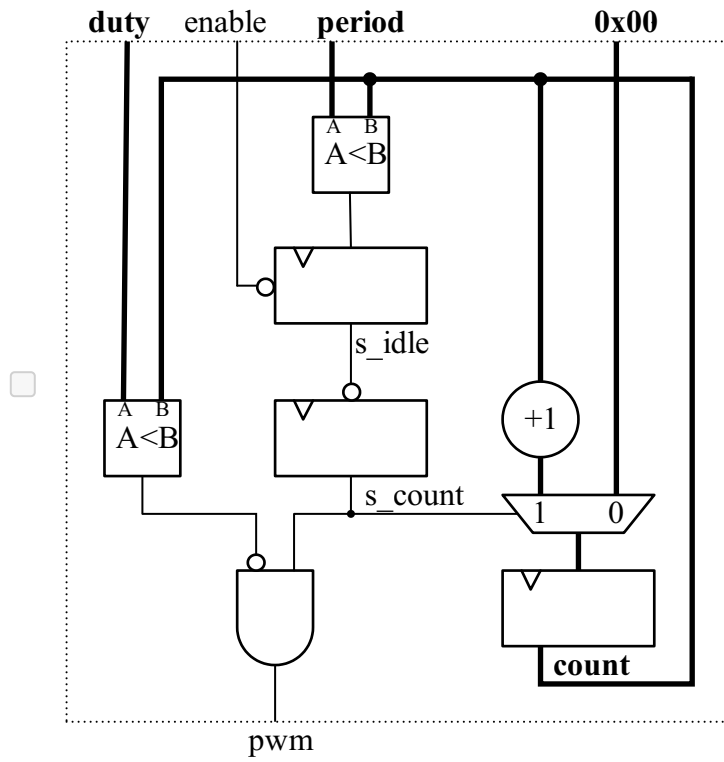


The ASMD diagram above depicts a simple pulse width modulator.
 Select the datapath diagram that corresponds best with the ASMD diagram.

Select one alternative:







Maximum marks: 5

- 3 In this exercise, there are several statements with various precision level presented in a randomized order. Your task here is to mark true statements.
Statements that are not always true, or does not make sense in the context of digital design, shall be considered false.

- Generic example: Assume we know Ole is the owner of a blue hat.
 - Which statement is true and which is false:
 - *Ole wears a blue hat* (False – owning one does not mean he wears it)
 - *Ole has a hat* (True, he may own several, but he does at least have one)
 - *Ole has hats* (False, although he may have several, we can only confirm that he has one)

Out of the following 50 statements, there are exactly 12 statements considered true, and you will only be able to select 12.

Select one or more alternatives:

- ☐ Divide and conquer (or partitioning) in digital system design is the process of dividing a system into manageable modules
- ☐ Look up tables can be implemented using SRAM cells and transmission gates
- ☐ A Moore machine does not create hazards
- ☐ A self testing test bench is equivalent to a self test
- ☐ Structural components are written with data-flow code style
- ☐ Push flow control assumes receiver always is ready to receive data
- ☐ A Mealy machine does not contain registers
- ☐ It is good practice to rather use a function than a procedure to create combinational logic
- ☐ Math_real.uniform() is used to synthesize clock jitter
- ☐ VHDL simulation is purely cycle based
- ☐ Programmable logic is well suited for keeping component cost low compared to microcontroller systems
- ☐ RTL simulation can be used to verify logic function of a digital design
- ☐ VHDL requires block layout and indentation
- ☐ A variable can have several values within a process
- ☐ INOUT should be used to code efficiently
- ☐ If statements is best suited when there is a single output target
- ☐ Synchronous combinational logic has feedback
- ☐ A process can have multiple subprograms
- ☐ Selected statements (with...select) cannot be used to infer latches
- ☐ Linear feedback shift registers must be used for parity checks
- ☐ Flow control is not needed for clock domain crossing
- ☐ ASICs are made of programmable logic

- ☐ A procedure must return a `std_logic_vector`
- ☐ A signal is always updated immediately before next line is read
- ☐ Case statements is best suited for single input vector and multiple output targets
- ☐ A counter is normally synchronous
- ☐ A subprogram in VHDL can only have one parameter
- ☐ A package should not contain procedures
- ☐ VHDL code cannot be ported between compilers
- ☐ Static timing simulation requires RTL simulation to be performed
- ☐ File IO is prohibited in VHDL
- ☐ A built-in self-test is better than using a test bench
- ☐ The physical implementation of a `std_logic_vector` is one wire
- ☐ VHDL processes uses position for creating priority
- ☐ Pipelining does not work with control signals
- ☐ Modern FPGAs contains only one primitive, the logic cell
- ☐ `Rising_edge()` and `falling_edge()` should be used together
- ☐ A shifter is the same as a shift register
- ☐ VHDL is case sensitive
- ☐ Apples and pears are valid entity names in VHDL
- ☐ When `.. else` statements is best suited for multiple output targets
- ☐ VHDL loops cannot be used to synthesize hardware
- ☐ Registers can be created in functions
- ☐ An encoder is a multiplexer paired with a register
- ☐ HDL defines a sequence of instructions that can be read from the FPGA memory

- ☐ VHDL simulation uses event queues
- ☐ A decoder is a look up table with recursive elements
- ☐ A single VHDL architecture can be used with several different entities
- ☐ Designing VHDL modules is the first step in a typical system design process
- ☐ A test bench must be written in synthesizable code

Maximum marks: 24

For a random (asynchronous) signal being stored in a register, the probability of error (P_{error}) is given by the following formula:

$$P_{\text{error}} = (t_s + t_h) / t_{\text{clk2}}$$

where t_s is the setup time and t_h is the hold time for the register used to store the value, and t_{clk2} is the period of the clock in the receiving domain.

Note on entering numbers:

Inspira cannot automatically read exponents. Entering a number normally best expressed with an exponent, will have to be done using leading zeros when there is a negative exponent. Example: $4,30 \cdot 10^{-5}$ has three significant digits. This number can only be entered as 0,000043. Both comma and period (dot) is accepted as decimal separator. Numbers normally expressed with a positive exponent will have to be written using trailing zeros down to the decimal mark. Example $5,6 \cdot 10^3$ (which has two significant digits) will have to be entered as 5600.

The windows below are set up to accept 8 digits which should be more than sufficient for each task. For each task, two significant digits precision should be sufficient (not counting leading/trailing zeros).

- 4 We assume the system used to sample signals has a clock frequency of 50 MHz, and the registers have 100ps setup- and 100ps hold time. $1\text{ps} = 10^{-12}\text{s}$.

What is the probability of error P_{error} when this system is used to sample a random, asynchronous signal?

Enter the value for P_{error} : .

No more than two significant digits is required. (Leading zeros are not counted as significant digits).

Maximum marks: 2

- 5 We use the system to read a signal from a quadrature encoder. The quadrature encoder has 1000 states per round.

What will be the average error frequency (f_{error}) reading the quadrature signal when the motor spins at 3000 rounds per minute?

Enter the value of f_{error} : Hz.

No more than two significant digits is required.

Maximum marks: 2

- 6 A quadrature signal consist of two bits where only one is supposed to change at any given time. A quadrature signal that is 00_{bin} will for example only transition to 01_{bin} or 10_{bin} , never to 11_{bin} . When reading a quadrature signal, such as from the motor system described above, what is the best way of managing metastability?

Select one alternative:

- ☐ Handshake (push-pull) flow control, brute force synchronizers for control signals
- ☐ Pull flow control, brute force synchronizers for control signals
- ☐ Adding a FIFO for reading the signal
- ☐ Brute force synchronizers (“double flopping”) for both quadrature signals
- ☐ Push flow control, brute force synchronizers for control signals

Maximum marks: 2

- 7 In this assignment, the module `compute_seq` which computes $a+b+c+d$ shall be implemented. The signals a, b, c and d are synchronous to the clock signal. The entity `compute_seq` is listed below.

The computation shall be using unsigned arithmetic operation on the operands a, b, c and d . The output result shall have the required number of bits such that the result is always correct, and overflows do not occur.

Implement the architecture to the module `compute_seq` entity as listed under as a sequential process in synthesizable VHDL. The implementation shall use one clock cycle to compute the result. Change the signal `result` in the entity `compute_seq` such that the declaration of `std_logic_vector` has the correct number of bits.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity compute_seq is
  port
    (clk : in std_logic;
     a : in std_logic_vector(15 downto 0);
     b : in std_logic_vector(15 downto 0);
     c : in std_logic_vector(15 downto 0);
     d : in std_logic_vector(15 downto 0);
     result : out std_logic_vector(XX downto 0)); -- XX shall be
changed
end entity compute_seq;

architecture rtl of compute_seq is
  -- implement the compute_seq process
end architecture rtl;
```

Fill in your answer here

1	
---	--

Maximum marks: 10

- 8 The `compute_seq` implementation does not meet timing closure with the selected technology and the required clock frequency. Therefore the design has to be pipelined.

In addition, the signal `vdata` determines when the inputs `a`, `b`, `c` and `d` has valid data, i.e.. `vdata='1'` when `a,b,c` and `d` are valid. The output signal `vresult` shall be '0' when the output result is not valid and '1' when the output signal result is valid. When the output signal result is not valid the value shall be '0'. The pipelined architecture shall allow new data each clock cycle.

Draw a datapath diagram of the of two stage pipeline architecture as according the the entity listed below.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity compute_seq is
  port
    (clk : in std_logic;
     a : in std_logic_vector(15 downto 0);
     b : in std_logic_vector(15 downto 0);
     c : in std_logic_vector(15 downto 0);
     d : in std_logic_vector(15 downto 0);
     vdata : in std_logic;
     vresult : out std_logic;
     result : out std_logic_vector(XX downto 0)); -- XX shall be
changed
end entity compute_seq;

architecture rtl of compute_seq is
  -- implement the compute_seq process
end architecture rtl;
```

In this exercise you can answer with digital hand drawing. Use your own sketch sheet (distributed). See instructions for filling in the sketch sheet in the link below the task bar.

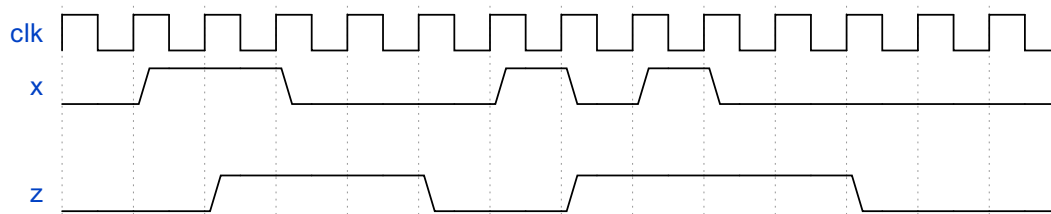
Maximum marks: 10

- 9 Implement the architecture from the previous assignment in synthesizable VHDL.

Fill in your answer here

1	
---	--

Maximum marks: 10



In this assignment, a Moore type finite state machine shall be implemented. The state machine is called an pulse stretcher.

The state machine works as follows:

The output signal z is high for N clock cycles after a pulse is detected on the input signal x.

N shall be an generic integer specified in the entity. The input signal x is synchronous to the clock signal clk.

The timing diagram depicted above shows an example of input and output values of the state machine when N is 2.

- 10** Draw an ASM-diagram which depicts the finite state machine described above.

In this exercise you can answer with digital hand drawing. Use your own sketch sheet (distributed). See instructions for filling in the sketch sheet in the link below the task bar.

Maximum marks: 10

- 11** Implement the finite state machine as a two or three process state machine in synthesizable VHDL.

The signals x and z shall be of type `std_logic`. N shall be an integer. The implementation shall use synchronous reset.

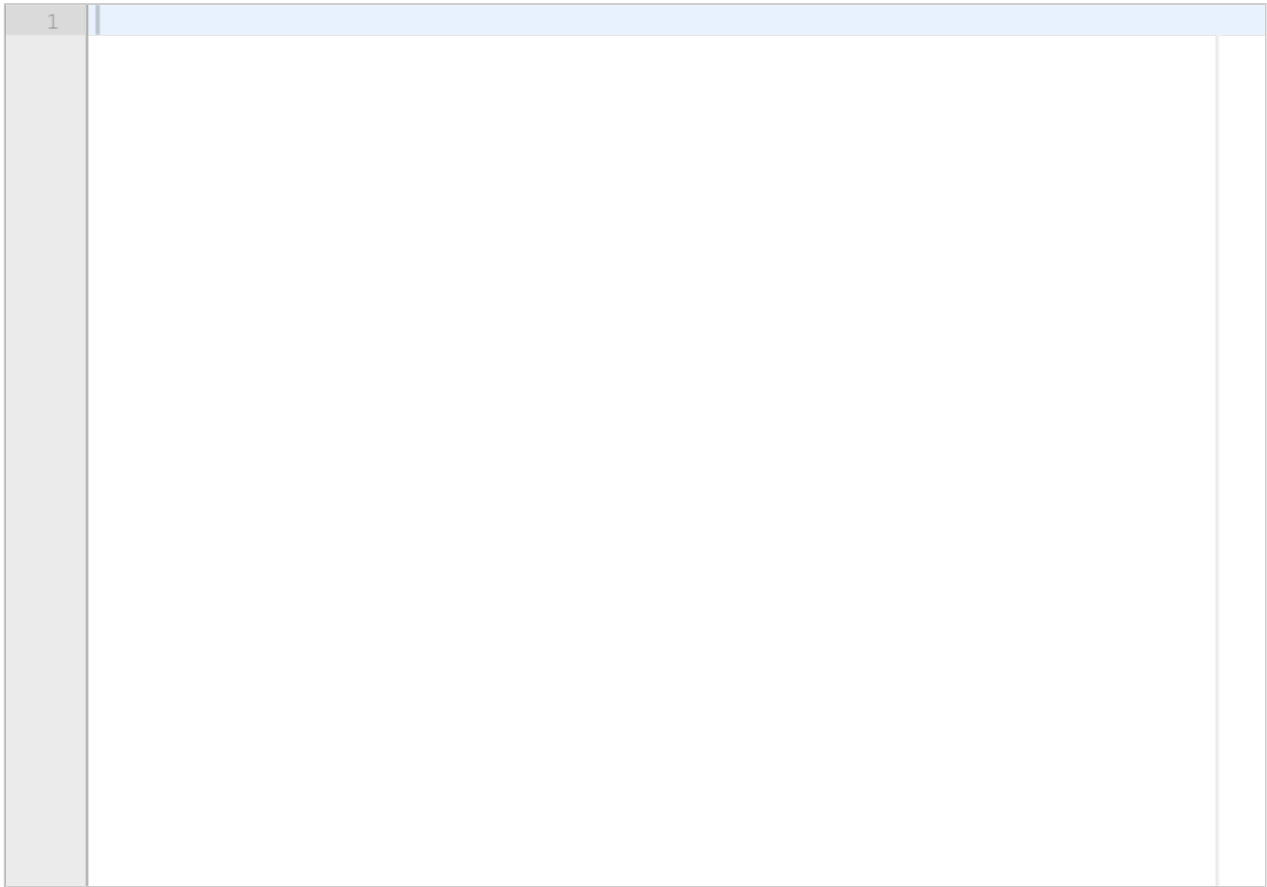
Fill in your answer here

1

Maximum marks: 10

- 12** Implement a VHDL test bench that verifies and reports correct behaviour according to the waveform figure when N is set to 2.

Fill in your answer here



Maximum marks: 10