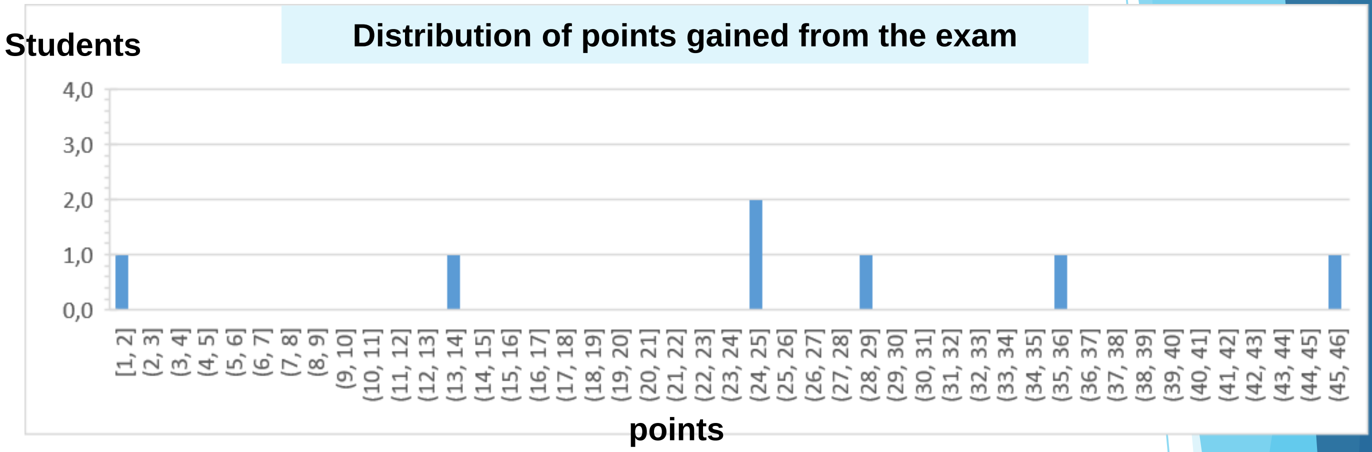
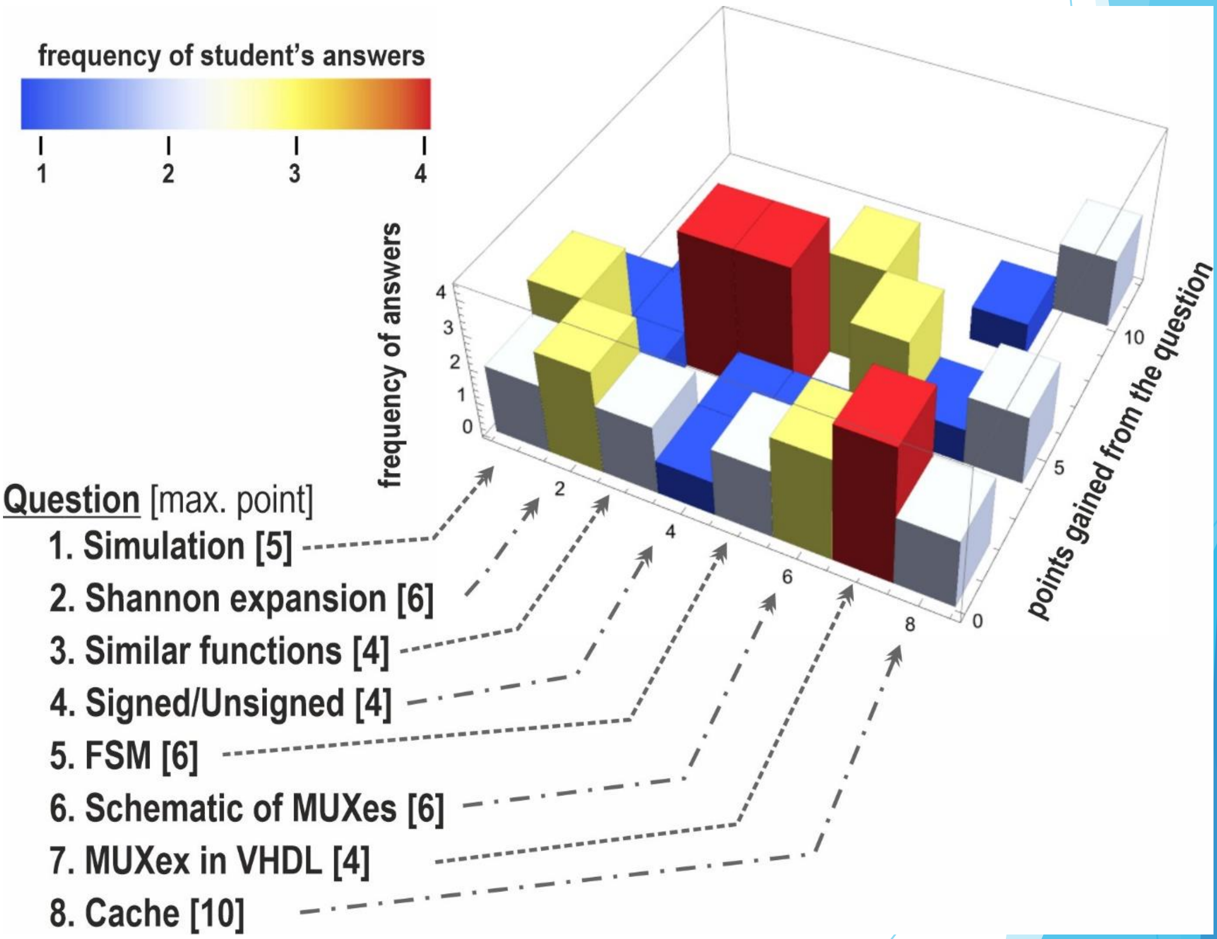
Statistical results of the LSP written exam on Friday 20 December 2024



Distribution of points gained from the exam



Question [max. point]

1. Simulation [5]
2. Shannon expansion [6]
3. Similar functions [4]
4. Signed/Unsigned [4]
5. FSM [6]
6. Schematic of MUXes [6]
7. MUXex in VHDL [4]
8. Cache [10]

Surname and first name :..

Exam Dec 20, 2024 - write here only your answers

Do not write here

1. Inputs A, B and C have values shown in the figure for times t_0, t_1, t_2, t_3 , t_4. Write values of X and Y outputs. The intervals between input changes are long, so we can neglect the delays of gates. 5

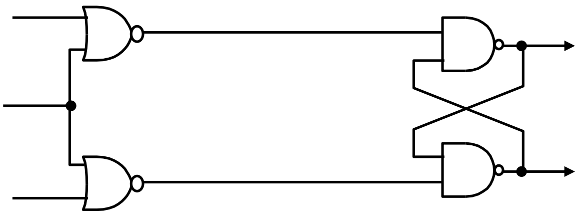
A = **..1..| ..0..|..1..|..0.. |..0..|**

t_0..|..t_1..|.. t_2..|..t_3..|..t_4..|..

B = **.0..|..1..|..0..|..0.. |..0..|**

t_0..|..t_1..|.. t_2..|..t_3..|..t_4..|..

C = **..0..|..1..|..0..|..0.. |..1..|**



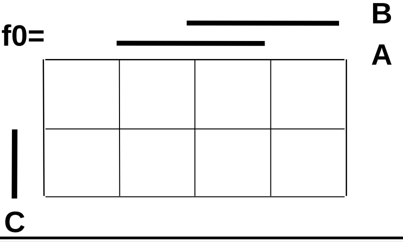
X = ...|... ....|... ....| ......|... ...|

t_0..|..t_1...|.. t_2..|..t_3..|..t_4..|..

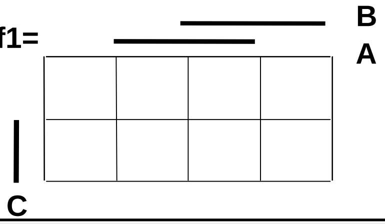
Y = ...|... ....|... ....| ......|.. ....|

2. Decompose function X=f(A,B,C, X) from question 1 with the aid of Shannon's expansion into X= (not X and f0(A, B,C)) or (X and f1(A,B,C)) . Write f0 and f1 functions as Karnaugh maps: 6

f0=



f1=



3. Mark all logic functions that have another equivalent logic function here*:* 4

x1<=(B and not A) or (A and not B);
x2<=(A and not C) xor (C and A);
x3<=(B or A) and (not B or not A);
x4<=(C xor A) or (B and not A);

y1 □
y2 □
y3 □
y4 □

4. . If we perform the addition **254+255+256+257** by an 9 bit adder, what is its 9-bit result converted to a decimal number as. 4

a) unsigned .... b) signed in two's-complement............. ...

5. *Add:* Moore (Mealy) finite state machine is the tuple **M = < X, S, Z, ω, δ, s_0** ∈ **S >,** where

X is.. ....

S is..

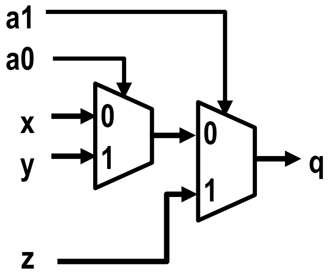
Z is...

δ is a mapping for Moore.. for Mealy..............................

ω is a mapping for Moore..for Mealy..............................

s0 is 6

6. Use only AND, NAND, OR, NOR gates a inverters NOT, and wires to connect the schematic at the figure right.



Do not write here

5

7. Describe the circuit from question 6 using both VHDL concurrent and sequential statements:

```
library ieee; use ieee.std_logic_1164.all;use ieee.numeric_std.all;
entity Test20241220q7 is port( x, y ,z, a1,a0: in std_logic:= '0';
                                          qcon, qseq: out std_logic := '0')
end entity;
architecture rtl of Test20241220q7 is
begin
-- qcon concurrent statement.............................................................
..................................................................................................................................
   ...............................................................................................................................
   ...............................................................................................................................
   ...............................................................................................................................
 -- qseq sequential statement..................................................................................
 iseq: process.................................................................................................................
. ..............................................................................................................................
.........begin...................................................................................................................
 ...............................................................................................................................
 ...............................................................................................................................
 ...............................................................................................................................
 ...............................................................................................................................
 ...............................................................................................................................
 ...............................................................................................................................
..............................................................................................................................
 ...............................................................................................................................
 ........end process;.....................................................................................................
end architecture;
```

qcon
5

qseq
5

8.The small 32-bit processor has a cache of only **256 bytes** that is organized as a **directly mapped 2-word** line length.
1/ Fill the table with indexes of tag, set and block for used addresses.
2/ Mark the memory accesses that will have a cache hit if the addresses are executed in the order shown in the table below and the cache was empty at the beginning.

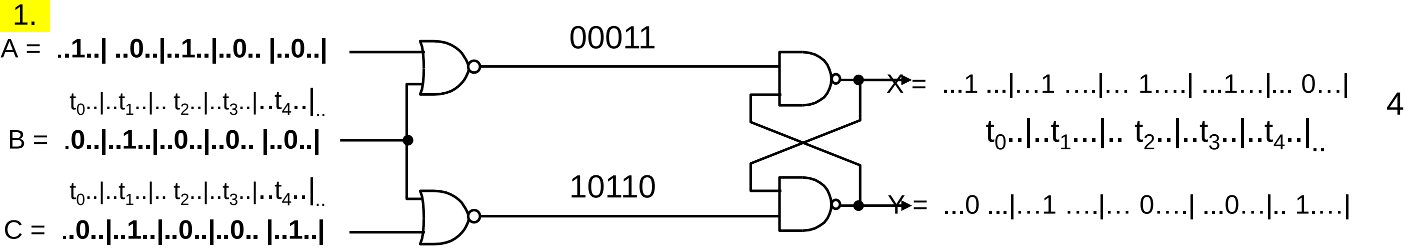
		Direct Mapped Cache		
	Address as binary code	**tag**	**set**	**blok**
0x10				
0x14				
0x28				
0x2C				
0x94				
0xA8				
0xAC				

hex. adresa	**cache hit**
0x10	□
0x14	□
0x28	□
0x94	□
0x2C	□
0x10	□
0xA8	□
0xAC	□

10

Results

1.


A = ..1..| ..0..|..1..|..0.. |..0..|
t_0..|..t_1..|.. t_2..|..t_3..|..t_4..|..
B = .0..|..1..|..0..|..0.. |..0..|
t_0..|..t_1..|.. t_2..|..t_3..|..t_4..|..
C = ..0..|..1..|..0..|..0.. |..1..|
00011
10110
X= ...1 ...|...1 ....|... 1....| ...1...|... 0...|
t_0..|..t_1...|.. t_2..|..t_3..|..t_4..|..
Y= ...0 ...|...1 ....|... 0....| ...0...|.. 1....|
4

2.

X<=(X and not C) or (A) or (B)

f0=

B

A

0	1	1	1
0	1	1	1

C

f1=

B

A

1	1	1	1
0	1	1	1

C

3.

x1<=(B and not A) or (A and not B);
x2<=(A and not C) xor (C and A);
x3<=(B or A) and (not B or not A);
x4<=(C xor A) or (B and not A);

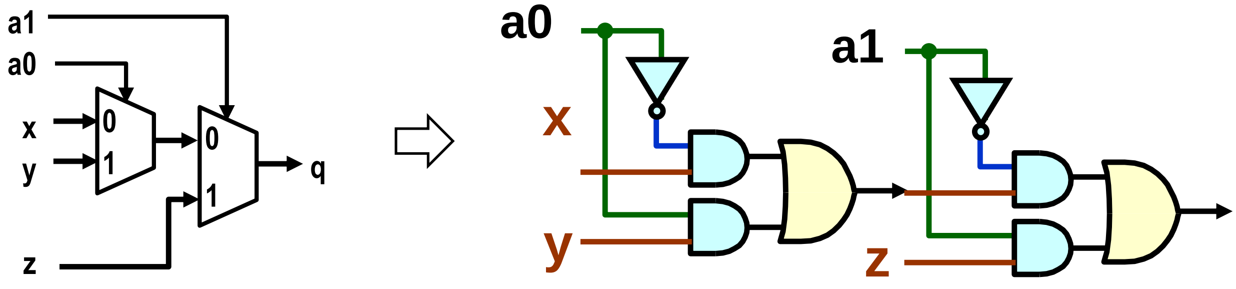
y1 ■
y2 □
y3 ■
y4 □

4. . If we add 254+255+256+257 by an 9 bit adder, what will be its 9-bit result converted to a decimal number.

a) unsigned ....510.................................... b) signed in two's-complement.....-2......... ...

5. *lecture 9, slide 12*

6.



7.

```
library ieee; use ieee.std_logic_1164.all;use ieee.numeric_std.all;
entity Test20240528q6 is port( x, y ,z, a1,a0: in std_logic:= '0';
                                        qcon, qseq: out std_logic := '0')
end entity;
architecture rtl of Test20240624q7 is
begin
  Qcon<= z when a1='1' else y when a0='1' else x;
.iseq: process(x, y ,z, a1,a0)
        begin   if a1='1' then Qseq<=z;
                elsif a0='1' then Qseq<=y; else Qseq<=x;
                end if;
        end process;
end architecture;
```

8. The small 32-bit processor has a cache of only **256 bytes** that is organized as a **directly mapped 2-word** line length.
1/ Fill the table with indexes of tag, set and block for used addresses.
2/ Mark the memory accesses that will have a cache hit if the addresses are executed in the order shown in the table below and the cache was empty at the beginning.

The simple example specifies 256/8=32- cache with **32 sets**
and all addresses have **tag 0**, thus, the cache hit occurs if we read again a set.

Address
as binary code

		direct mapped		
			5-bits	1 or 3 bits
		tag	**set**	**blok**
0x10	00…0 0001 0000	**0**	2	**0** *00*
0x14	00…0 0001 0100	**0**	2	**1** *00*
0x28	00…0 0010 1000	**0**	5	**0** *00*
0x2C	00…0 0010 1100	**0**	5	**1** *00*
0x94	00…0 1001 0100	**0**	18	**1** *00*
0xA8	000 0 1010 1000	**0**	21	**0** *00*
0xAC	000 0 1010 1100	**0**	21	**1** *00*

hex. adresa	**cache hit**
0x10	□
0x14	■
0x28	□
0x94	□
0x2C	■
0x10	■
0xA8	□
0xAC	■