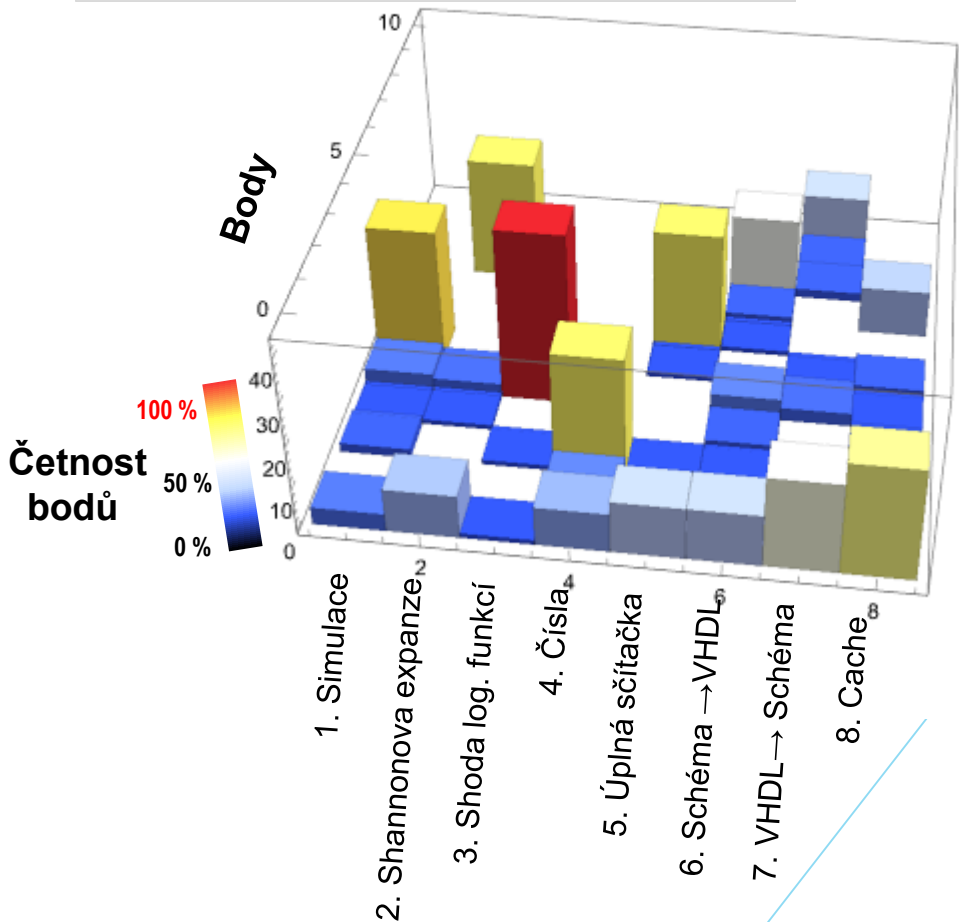
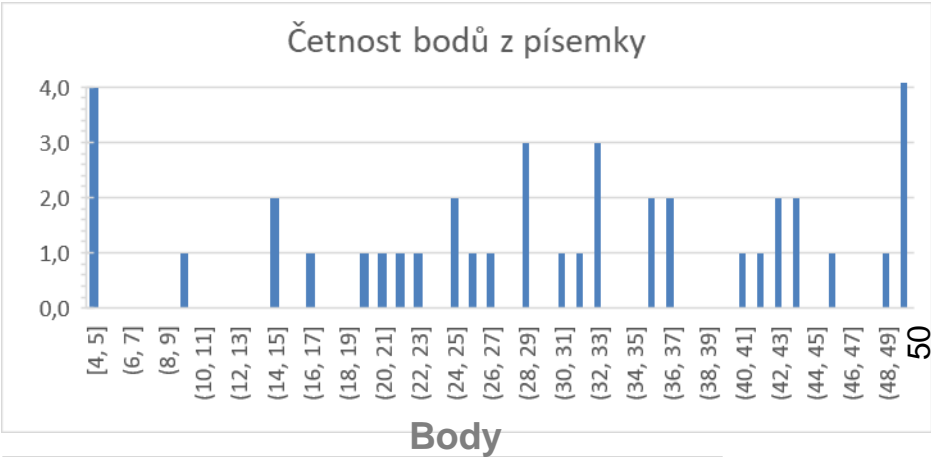


Statistické výsledky zkoušky

LSP ve středu 4. června 2025

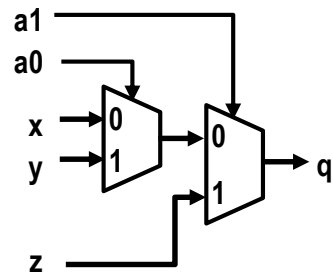
- verze V1.1 - opraven histogram

Získalo studentů



6. Obvod na obrázku popište ve VHDL jak **jedním** souběžným, tak i **jedním** sekvenčním příkazem

```
library ieee; use ieee.std_logic_1164.all;use ieee.numeric_std.all;
entity Test20250604q6 is
    port( x, y ,z, a1,a0: in std_logic:= '0';
          qcon, qseq: out std_logic := '0')
end entity;
architecture rtl of Test20250604q6 is
begin
```



-- qcon souběžným (concurrent) příkazem.....

.....

.....

.....

.....

.....

.....

qcon
4

-- qseq sekvenčním příkazem.....

```
iseq: process.....
.....
.....begin.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....end process;.....
```

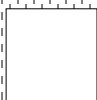
qseq
4

end architecture;

7. Pro obvod popsáný následujícím kódem nakreslete jeho **blokové schéma** s využitím značek multiplexorů a registrů DFF.

A obvod správně pojmenujte!

```
library ieee; use ieee.std_logic_1164.all;
entity Test20250604q7 is
  port (A, B : in std_logic;
        C : in std_logic_vector(3 downto 0);
        D : out std_logic);
end entity;
architecture rtl of Test20250604q7 is
begin
  process(A)
    variable rg:std_logic_vector(C'RANGE);
    begin if rising_edge(A) then
      if B='1' then rg:=C;
        else rg:= rg(2 downto 0) & not rg(3);
      end if;
    end if;
    D<=rg(0);
  end process;
end architecture;
```



8. Malý 32-bitový procesor má cache dlouhou jen **128 bytů a** organizovanou jako **přímo mapovanou** o délce řádku **2 slova**.

Vyplňte tabulku mapování adres.

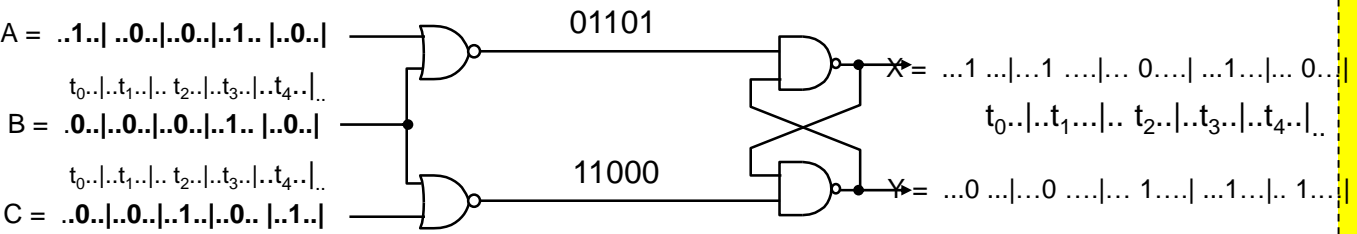
	binárně	tag	set	blok-offset
0x10				
0x14				
0x28				
0x2C				
0x94				
0xA8				
0xAC				

Předpokládejme, že cache je prázdná. při spuštění programu.
Označte přístupy do paměti, které budou mít cache hit,
provedou-li se přesně v pořadí uvedeném v tabulce dole.

hex. adresa	cache hit
0x10	<input type="checkbox"/>
0x14	<input type="checkbox"/>
0x28	<input type="checkbox"/>
0x94	<input type="checkbox"/>
0x2C	<input type="checkbox"/>
0x10	<input type="checkbox"/>
0xA8	<input type="checkbox"/>
0xAC	<input type="checkbox"/>



1. Vstupy A, B, C měly v časech t_0, t_1, t_2, t_3 hodnoty uvedené v obrázku. Napište hodnoty X a Y výstupů. Předpokládejte, že intervaly mezi změnami vstupů jsou tak dlouhé, že lze zanedbat zpoždění hradel.



2. Funkci $X=f(A,B,C, X)$ obvodu z otázky 1, rozložte na tvar $X=(\text{not } X \text{ and } f_0(A, B, C)) \text{ or } (X \text{ and } f_1(A, B, C))$ pomocí Shannonovy expanze. **Výsledné funkce f_0 a f_1 zapište jako Karnaughovy mapy:**

$f_0 =$

	B			
	A			
	0	1	1	1
C	0	1	1	1

$X = (X \text{ and not } C) \text{ or } (A) \text{ or } (B)$

$f_1 =$

	B			
	A			
	1	1	1	1
C	0	1	1	1

3. Zaškrtnutím označte všechny logické funkce, které zde mají jinou funkci s nimi **shodnou**:

$x_1 \leftarrow (B \text{ and not } A) \text{ or } (A \text{ and not } B);$
 $x_2 \leftarrow (A \text{ and not } C) \text{ xor } (C \text{ and } A);$
 $x_3 \leftarrow (B \text{ or } A) \text{ and } (\text{not } B \text{ or not } A);$
 $x_4 \leftarrow (C \text{ xor } A) \text{ or } (B \text{ and not } A);$

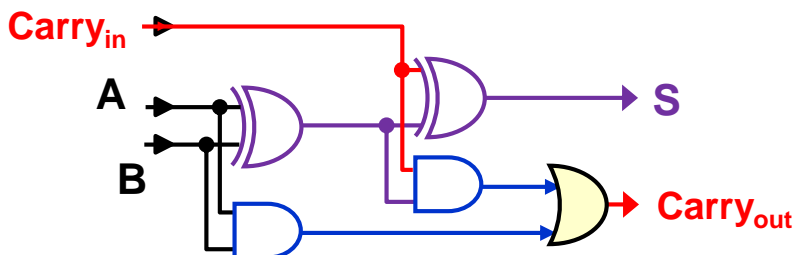
y_1 ☒
 y_2 ☐
 y_3 ☒
 y_4 ☐

4. Uložíme-li dolní bity výsledku operace s čísly **4*510** do **9-bitového** registru,

jaká hodnota v něm bude, bereme-li ji jako 9bitové dekadické číslo $4 * (2^{**9-2}) = 2^{**11-8} \equiv 2^{**9-8} = 504 \mid -8$

a) bez znaménka (unsigned).....504..... b) se znaménkem ve druhém doplňku (signed).....-8.....

5. Nakreslete schéma úplné jednobitové sčítačky.



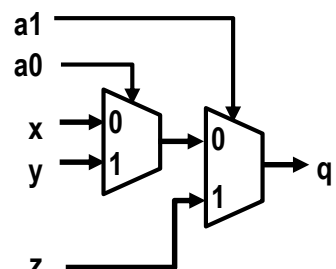
6. Obvod na obrázku popište ve VHDL jak souběžným, tak sekvenčním příkazem

$Qcon \leftarrow z \text{ when } a1='1' \text{ else } y \text{ when } a0='1' \text{ else } x;$

```

process(all)
begin
  if a1='1' then Qseq<=z; elsif a0='1' then Qseq<=y; else Qseq<=x;
end if;
end process;

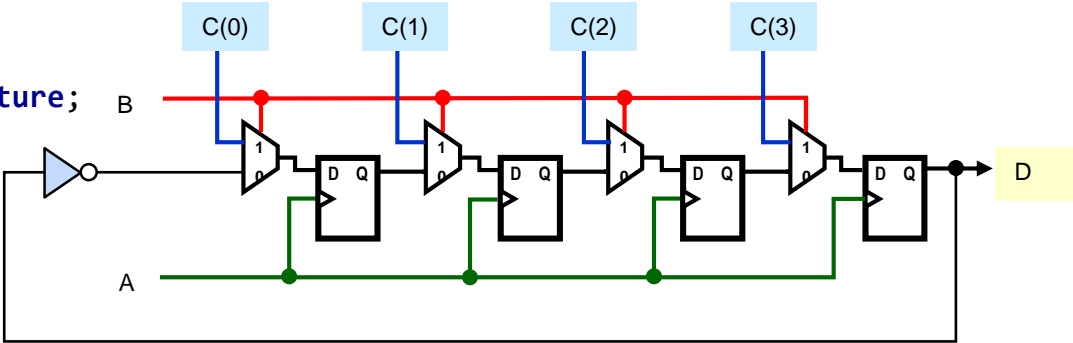
```



7. Nakreslete schéma obvodu, který je popsán následujícím kódem, a uveďte, jak se nazývá:

```
library ieee; use ieee.std_logic_1164.all;
entity Test20250604q7 is port (A, B : in std_logic; C : in std_logic_vector(3 downto 0); D: out std_logic);
end entity;
architecture rtl of Test20250604q7 is
begin
  process(A)
    variable rg:std_logic_vector(C'Range);
    begin if rising_edge(A) then
      if B='1' then rg:=C; else rg:= rg(2 downto 0) & not rg(3); end if;
    end if;
    D<=rg(0);
  end process;
end architecture;
```

Kruhový posuvný registr doleva ← dle váhy bitů
resp. Johnsonův čítač, obé s předvolbou



8. Malý 32-bitový procesor má cache dlouhou jen **128 bytů** a organizovanou jako **přímo mapovanou** o délce řádku **2 slova**. Vyplňte tabulku mapování adres a cache hit

			4-bity	1 bit
		tag	set	blok
0x10	00...0001 0000	0	2	0
0x14	00...0001 0100	0	2	1
0x28	00...0010 1000	0	5	0
0x2C	00...0010 1100	0	5	1
0x94	00...1001 0100	1	2	1
0xA8	000 1010 1000	1	5	0
0xAC	000 1010 1100	1	5	1

128/(2 slova*4 byty)=16
- cache má 16 setů

hex. adresa	cache hit
0x10	<input type="checkbox"/>
0x14	<input checked="" type="checkbox"/>
0x28	<input type="checkbox"/>
0x94	<input type="checkbox"/>
0x2C	<input checked="" type="checkbox"/>
0x10	<input type="checkbox"/>
0xA8	<input type="checkbox"/>
0xAC	<input checked="" type="checkbox"/>

blok se vždy načte celý

přepíše set 2