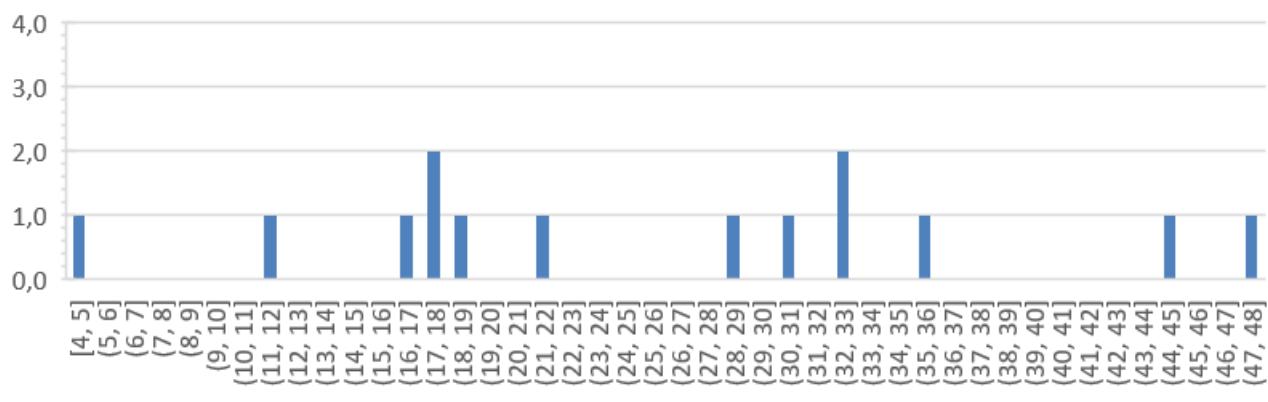
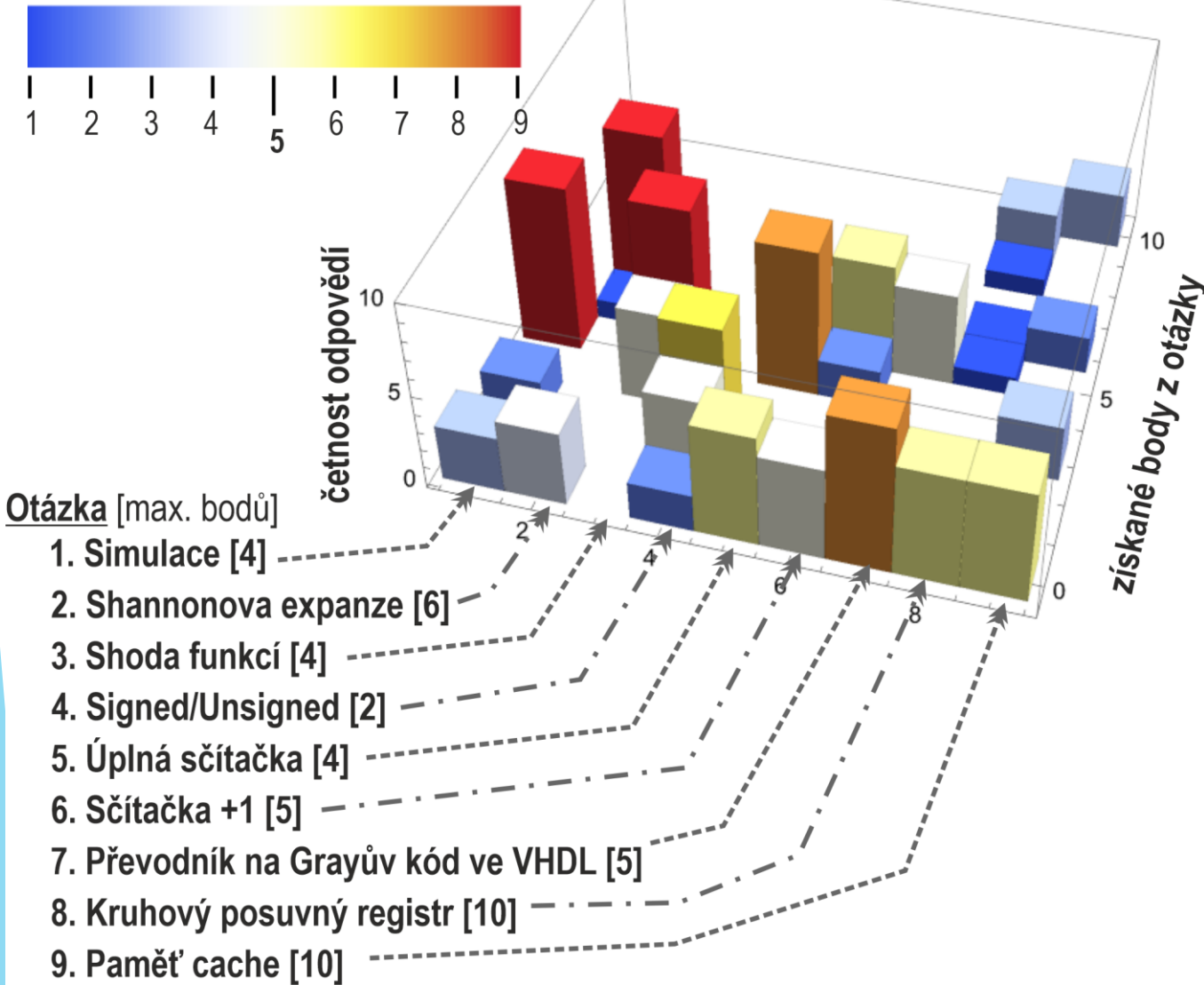


# Statistické výsledky zkoušky LSP ve středu 5. června 2024

Četnost bodů z písemky



četnost odpovědí

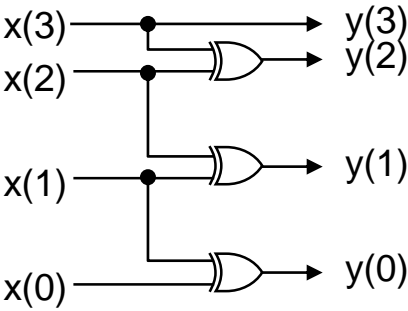




6. Sčítačka +1 se dá zapojit bez užití úplné sčítačky mnohem jednodušeji pomocí hradel.  
Nakreslete její schéma:



7. Obvod na obrázku optimálně popište souběžným (concurrent) VHDL kódem bez užití sekvenčních příkazů. Plný počet bodů dostanete jen tehdy, popíšete-li celé zapojení v bloku architektury pouze jediným příkazem. Za každý další v ní napsaný Vám bude stržen 1 bod.



```
library ieee; use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity Test20240605q7 is port(
```

```
.....
.....);
end entity;
architecture rtl of Test20240605q7 is

begin
.....
.....
.....
.....
.....
.....
end architecture rtl;
```

Zde  
nepište

5

5

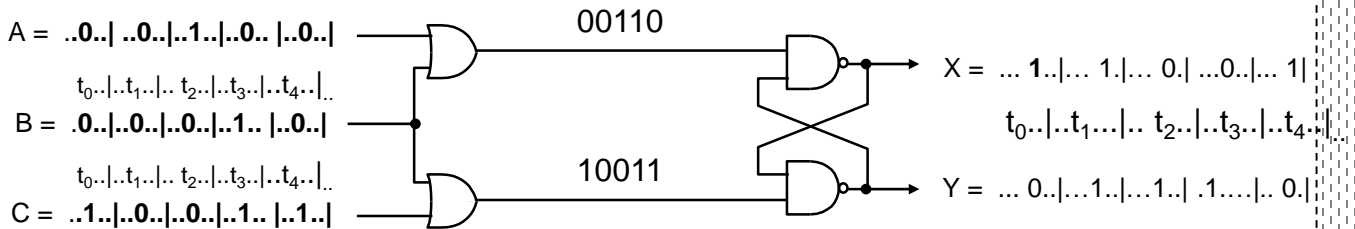
<div>Příjmení a jméno:.....</div> <div>Odpovědní arch písemky LPS dne 5. června 2024 - pište sem jen Vaše odpovědi</div>	<div>Zde nepište</div>
<div>8. Nakreslete schéma obvodu, který je popsán následujícím kódem, a uveďte, jak se nazývá:</div> <div>library ieee; use ieee.std_logic_1164.all;</div> <div>entity Test20240605q8 is port (A, B : in std_logic; C : in std_logic_vector(3 downto 0); D: out std_logic);</div> <div>end entity;</div> <div>architecture rtl of Test20240605q8 is</div> <div>begin process(A)</div> <div>    variable rg:std_logic_vector(C'RANGE);</div> <div>    begin</div> <div>        if rising_edge(A) then</div> <div>            if B='1' then rg:=C; else rg:= rg(0) &amp; rg(3 downto 1); end if;</div> <div>        end if;</div> <div>        D&lt;=rg(0);</div> <div>    end process;</div> <div>end architecture;</div>	
<div></div>	<div>schéma</div> <div>8</div> <div></div> <div>název</div> <div>2</div> <div></div>

9. Malý 32-bitový procesor má cache dlouhou jen **256 bytů** a organizovanou jako **přímo mapovanou** o délce řádku **4 slova**. Označte přístupy do paměti, které budou mít cache hit, budou-li provedené přesně v pořadí uvedeném v tabulce dole:

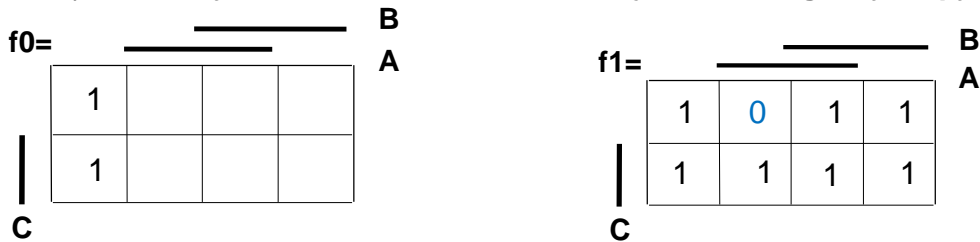
hex. adresa	cache hit
0x14	<input type="checkbox"/>
0x18	<input type="checkbox"/>
0x2C	<input type="checkbox"/>
0x24	<input type="checkbox"/>
0x118	<input type="checkbox"/>
0x10	<input type="checkbox"/>
0x11C	<input type="checkbox"/>
0x110	<input type="checkbox"/>

# Řešení

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:**



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

$y_1 \leq (\text{not } A \text{ and not } C) \text{ or } (A \text{ and } C \text{ and not } D)$

$y_2 \leq (\text{not } A \text{ or } C) \text{ and } (\text{not } A \text{ or not } D) \text{ and } (A \text{ or not } C)$

$y_3 \leq (\text{not } A \text{ and not } C) \text{ xor } (A \text{ and not } D)$

$y_4 \leq (A \text{ xnor } C) \text{ and } (\text{not } C \text{ or not } D)$

$y_1$  ☒

$y_2$  ☒

$y_3$  ☐

$y_4$  ☒

4. Uložíme-li dolní bity výsledku operace s čísly **4\*1023** do 10bitového registru,

jaká hodnota v něm bude, bereme-li ji jako 10bitové dekadické číslo

a) bez znaménka (unsigned).  $4 * (2^{10} - 1) = (2^{12} - 4) \rightarrow$  uloží se jen dolní bity jako  $(2^{10} - 4) = 1020$

b) se znaménkem ve druhém doplňku (signed) 1023 je -1, a tak  $4 * -1 = -4$

5. Plná sčítačka Logické obvody na FPGA str. 92 či přednáška 11

6. Sčítačka +1 Logické obvody na FPGA str. 97 a určitě i někde v přednáškách

7. Správné řešení

```
library ieee; use ieee.std_logic_1164.all; use ieee.numeric_std.all;
entity Test20240605q7 is port( x: in std_logic_vector(3 downto 0); y: out std_logic_vector(3 downto 0)); end entity;
architecture rtl of Test20240605q7 is
begin
    y<= ('0' & x(3 downto 1)) xor x;
end architecture rtl;
```

Tolerované varianty, pokud měly i správné pořadí operací, u sloučení & na něm záleží:

$y \leq x(3) \& (x(3 \text{ downto } 1) \text{ xor } x(2 \text{ downto } 0));$

$y \leq x(3) \& (x(3) \text{ xor } x(2)) \& (x(2) \text{ xor } x(1)) \& (x(1) \text{ xor } x(0));$

8. Posuvný registr - či přednáška 11 snímek 29 a 30

# Řešení

9. Malý 32-bitový procesor má cache dlouhou jen **256 bytů a** organizovanou jako **přímo mapovaná** o délce řádku **4 slova**. Označte přístupy do paměti, které budou mít cache hit, budou-li provedené přesně v pořadí uvedeném v tabulce

hex. adresa	cache hit	hex. adresa	cache hit
0x14	<input type="checkbox"/>	0x14	miss-první v setu 1 - načte se řádek, tag=0
0x18	<input checked="" type="checkbox"/>	0x18	hit
0x2C	<input type="checkbox"/>	0x2C	miss - první v setu 2, načte se řádek, tag=0
0x24	<input checked="" type="checkbox"/>	0x24	hit - blok v setu 2
0x118	<input type="checkbox"/>	0x118	miss - nesouhlasí tag, naplněn set 1, tag=1
0x10	<input type="checkbox"/>	0x10	miss - opět přepsaný set 1, tag=0
0x11C	<input type="checkbox"/>	0x11C	miss - zase přepsaný set 1, tag=1
0x110	<input checked="" type="checkbox"/>	0x110	hit - čtení ze setu 1 data, tag=1

**Vysvětlení:** Slovo 32bitového procesoru má 4 byty.

Řádek načítá (dle zadání) 4 slova, má tedy 16 bytů a vždy se naplní celý.

$256/16 = 16$ , cache má tedy 16 setů. U přímo mapované cache má každý set právě 1 řádek.

Adresy se mapují takto: Dolní 2 bity (1 a 0) indexují byte ve slově.

Další 2 bity (3 a 2) patří indexu slova v řádku. Následující 4 bity (7 až 4) definují index setu.

Zbýlé horní bity označují tag rozlišující obsah setu.

**Metodika hodnocení:** Za správné označení +3 body, za chybné -3 body.

Kdo neoznačil nic, má 0. Při všech 3 správných +1 prémiový;

Mapování adres:		přímo mapovaná		
		tag	3-bity set	1 bit blok
0x10	00...0001 0000	0	1	0
0x14	00...0001 0100	0	1	1
0x18	00...0001 1000	0	1	2
0x24	00...0010 1100	0	2	3
0x2C	00...0010 0100	0	2	1
0x110	001 0001 0000	1	1	0
0x118	001 0001 1000	1	1	2
0x11C	001 0001 1100	1	1	3