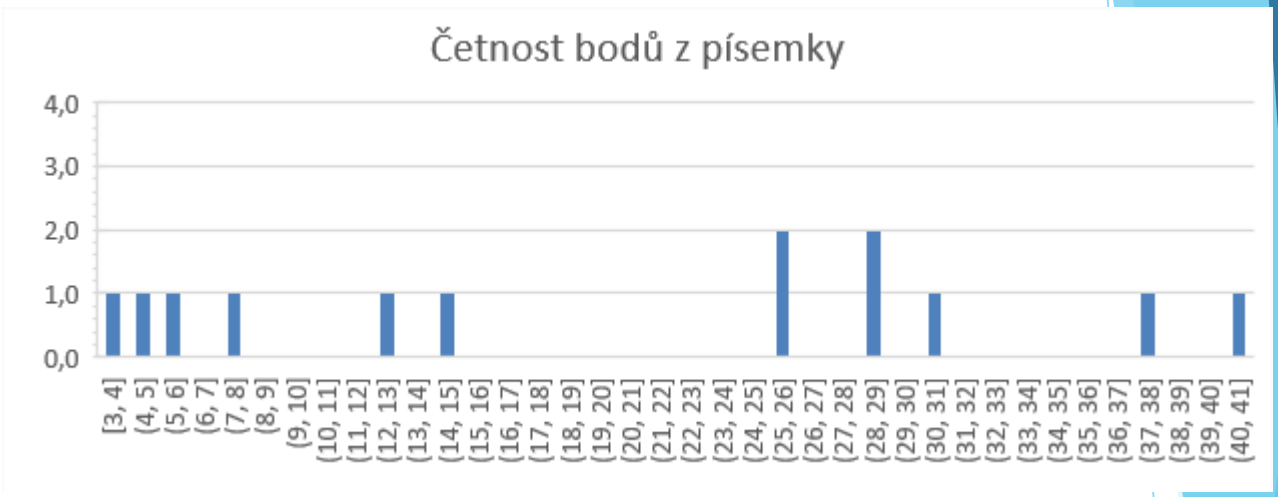
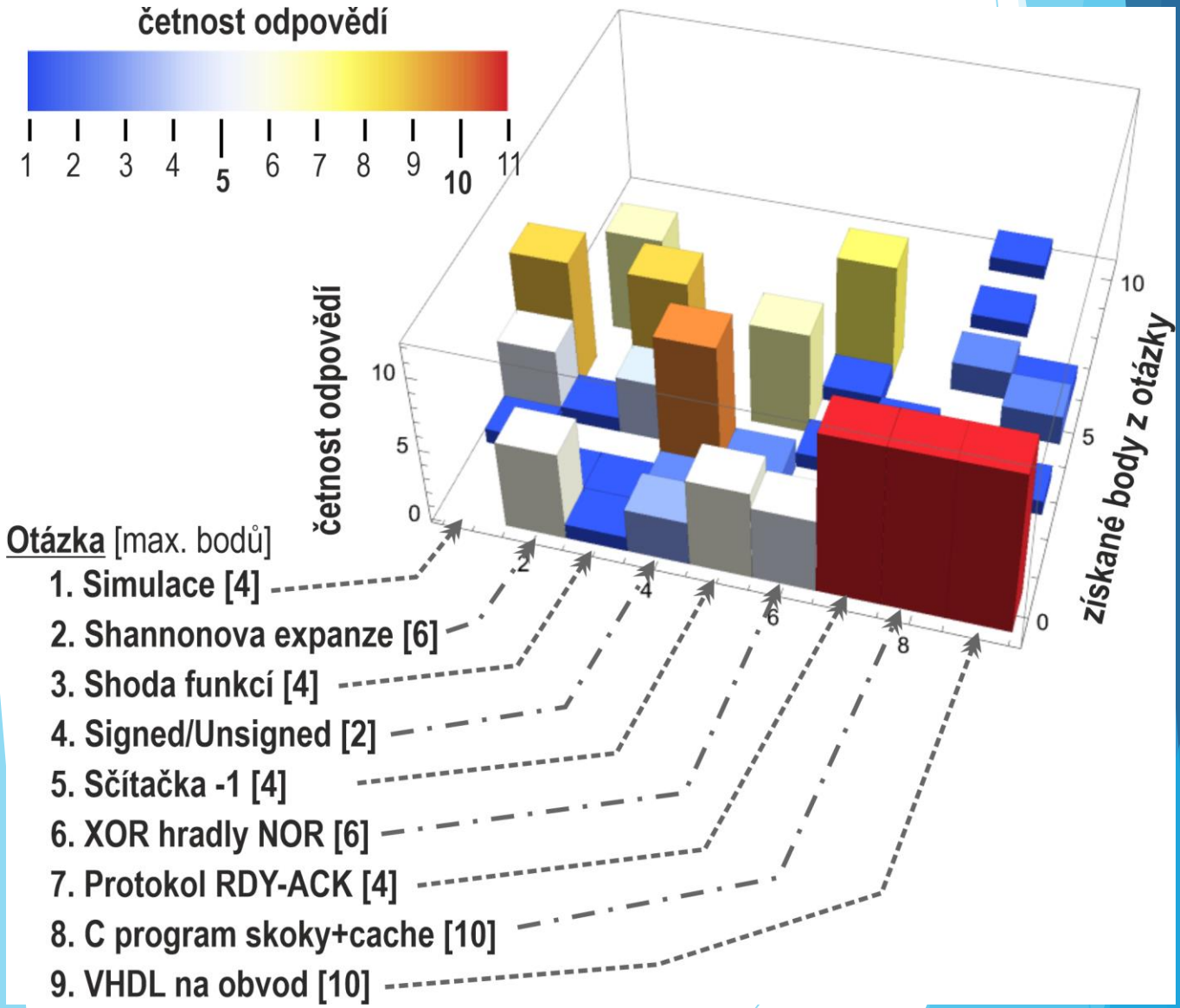
Statistické výsledky zkoušky LSP ve středu 14. června 2024



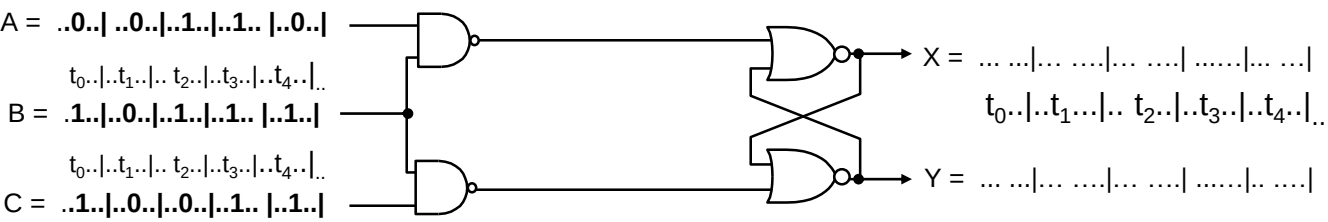


Příjmení:..... Jméno:.....

Odpovědní arch písemky LSP ze dne 5. června 2024 - pište sem jen Vaše odpovědi

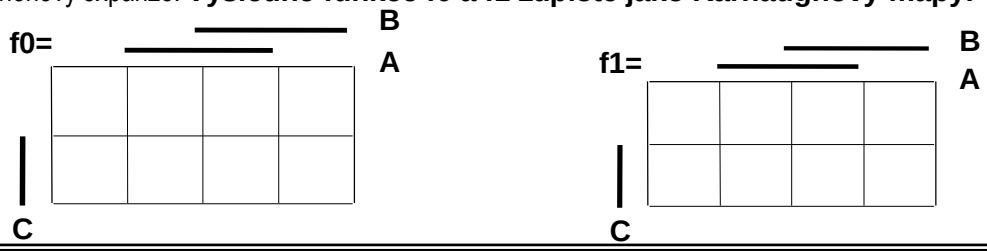
Zde nepište

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



4

2. Funkci X=f(A,B,C, X) obvodu z otázky 1, rozložte na tvar X= (not X and f0(A, B,C)) or (X and f1(A,B,C)) pomocí Shannonovy expanze. **Výsledné funkce f0 a f1 zapište jako Karnaughovy mapy:**



6

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

y1<=((not A or C) and B and not D) or (A and D); y1 ☐
y2<=((B and (not A or C)) or D) and (A or not D); y2 ☐
y3<= (A or C or D) and (A or B) and (not D or C); y3 ☐
y4<=(((not A and not D) or (A and C)) and B) or (A and D); y4 ☐

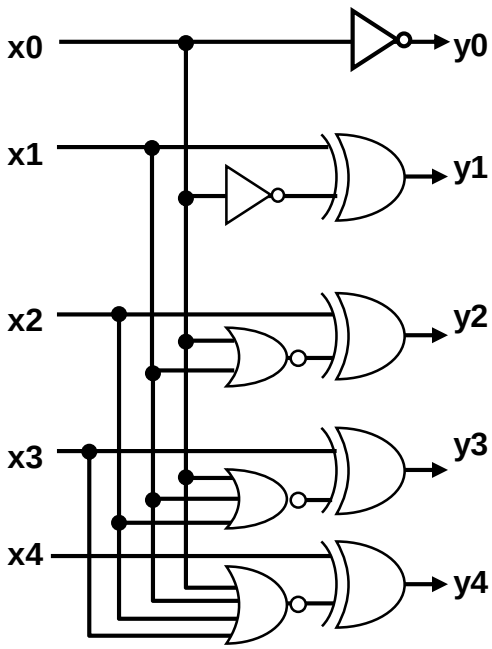
4

4. Jaký bude výsledek součtu 255+253+251 na 8bitové sčítačce, bereme-li ho jako 8bitové dekadické číslo

a) bez znaménka (unsigned)..... *b) se znaménkem ve druhém doplňku (signed).....*

2

5. Nechť x0 a y0 označuje nejnižší bity 5 bitových signed signálů x a y, jejich nejvyšší bity y4 a y4. Identifikujte operaci obvodu a popište ji maximálně úsporným příkazem VHDL:




```
library ieee; use ieee.std_logic_1164.all; use ieee.numeric_std.all;
entity Test20240614q5 is
    port( x: in signed(4 downto 0); y: out signed(4 downto 0));
end entity;
architecture rtl of Test20240614q5 is
begin
.....
.....
.....
.....
.....
.....
.....
end architecture;
```

4

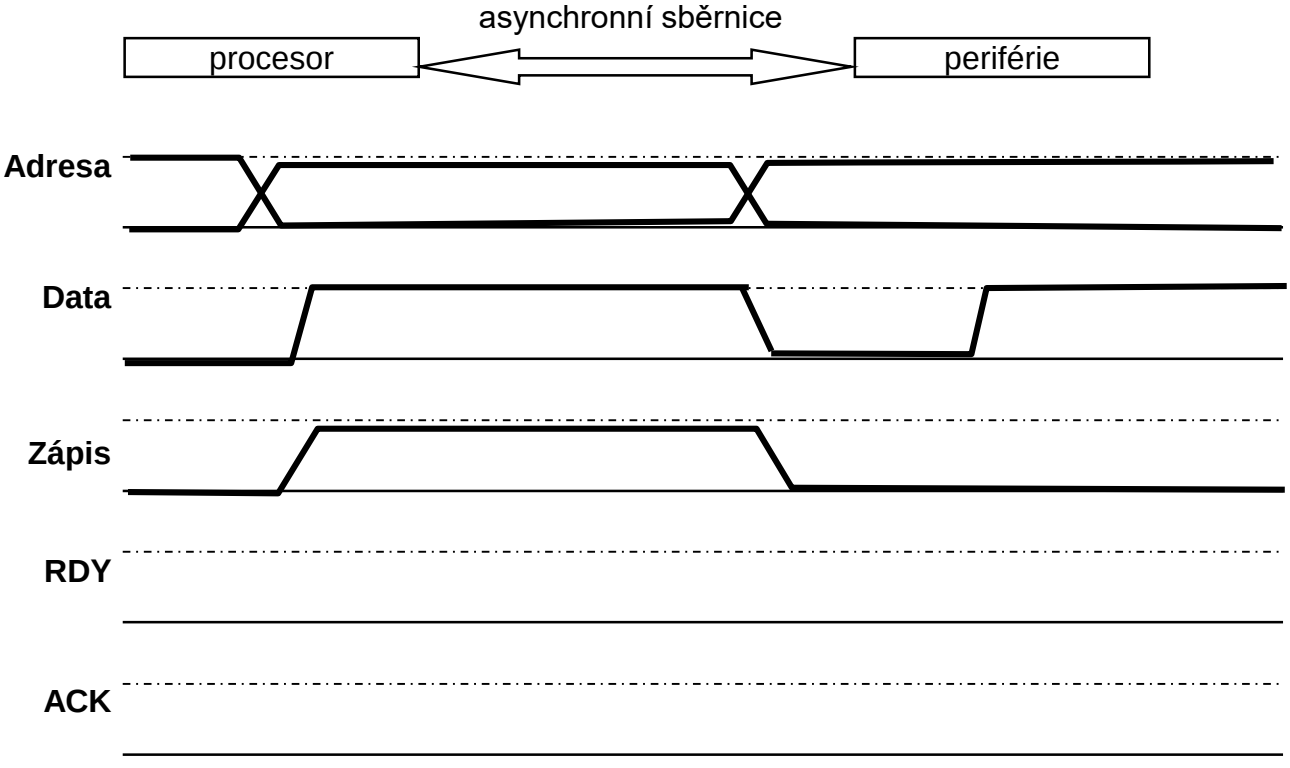
6. Máte k dispozici pouze 2vstupová hradla NOR, a nic jiného. Vytvořte XOR hradlo jen pomocí nich a drátů.

Nápověda: Využijte De Morganovo pravidlo.

Zde nepište

6

7. Doplňte chybějící signály a průběhy RDY-ACK komunikace po asynchronní sběrnici, kde signál Zápis='1' určuje zápis dat do periférie a Zápis='0' znamená čtení dat z periférie. Data je obousměrným signálem a "Adresa" jen vybírá pokaždé nějaký registr periférie.



4

Příjmení a jméno:.....

Odpovědní arch písemky LPS dne 5. června 2024 - pište sem jen Vaše odpovědi

Zde nepište

8. Mějme jednoduchý C program určený k testu výkonu procesoru:

```
int i, j; double pole[2000];
for (i=0; i<5; i++) {  for (j=0; j<2000; j++) pole[j]++;
                    }
```

A) Kolik v něm bude chybných skokových predikcí za předpokladu, že for-loop s konstantními mezemi se úsporně přeložila cyklem do-while a procesor používá pouze: 6

1bitové prediktory, které měly výchozí stav Not-Taken, NT , misses=……

2bitové prediktory, které měly výchozí stav WT, Weakly Taken, misses=……

B) Nechť program běží na 64bitovém procesoru, který má přímo mapovanou datovou cache velkou 32 kilobytů a s bloky o délce 4 slova a pole začíná na adrese 0x100000. Kolik bude minimální počet cache miss vyvolaných přístupy do pole, jestliže na začátku programu není v cache ani jeden jeho prvek. 4

……..

9. Nakreslete schéma obvodu, který vznikne z VHDL kódu dole.

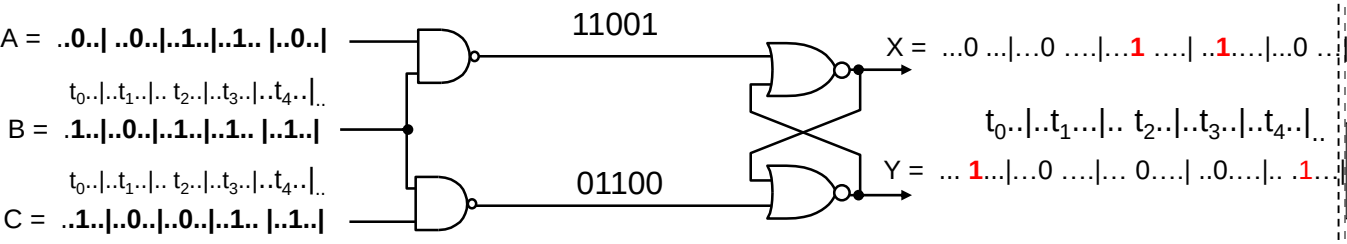
Nápověda: Nevytvoří se v něm ani loop a ani latch.

```
library ieee; use ieee.std_logic_1164.all;
entity SomeName is
    port( clk:in std_logic; X1: in std_logic; Y1, Y2: out std_logic); end entity;
architecture rtl of SomeName is
begin
 px:process(X1)
 variable v1 : std_logic;
 begin
        Y1<=v1; v1:= not X1;
end process;
pex: process(clk)
 variable v1 : std_logic;
 begin
     if rising_edge(clk) then  Y2<=v1; v1:= not X1;  end if;
  end process;
end architecture;
```

Zde nepište

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.



4

2. Funkci X=f(A,B,C, X) obvodu z otázky 1, rozložte na tvar X= (not X and f0(A, B,C)) or (X and f1(A,B,C)) pomocí Shannonovy expanze. **Výsledné funkce f0 a f1 zapište jako Karnaughovy mapy:**

f0=

C \ B A				
	0	0	1	0
	0	0	0	0

f1=

C \ B A				
	0	0	1	0
	0	0	1	0

6

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

y1<=((not A or C) and B and not D) or (A and D); y1 ■
y2<=((B and (not A or C)) or D) and (A or not D); y2 ■
y3<= (A or C or D) and (A or B) and (not D or C); y3 □
y4<=(((not A and not D) or (A and C)) and B) or (A and D); y4 ■

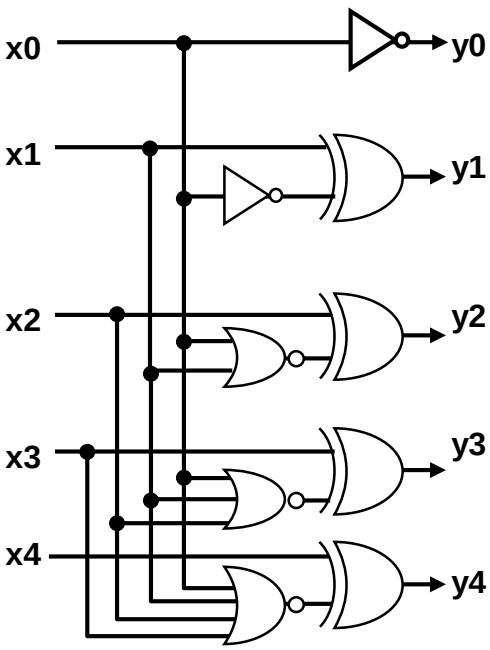
4

4. Jaký bude výsledek součtu 255+253+251 na 8bitové sčítačce, bereme-li ho jako 8bitové dekadické číslo

*a) bez znaménka (unsigned)....247= 256-**1-3-5**. b) se znaménkem ve druhém doplňku (signed). -9 = -1 - 8*

2

5. Nechť x0 a y0 ozna4uje nejnižší bity 5 bitových signed signálů x a y a y4 a y4 naopak jejich nejvyšší bity. Identifitujte obvod dole a zapište ho maximálně úsporným příkazem VHDL:




```
library ieee; use ieee.std_logic_1164.all; use ieee.numeric_std.all;
entity Test20240614q5 is
    port(  x: in signed(4 downto 0); y: out signed(4 downto 0));
end entity;
architecture rtl of Test20240614q5 is
begin
    y<=x-1;
end architecturel;
```

Sčítačka -1

Logické obvody na FPGA str. 97

4

Řešení

6. Máte k dispozici pouze 2vstupová hradla NOR, a nic jiného. Vytvořte XOR hradlo jen pomocí nich a drátů.
Nápověda: Využijte De Morganovo pravidlo

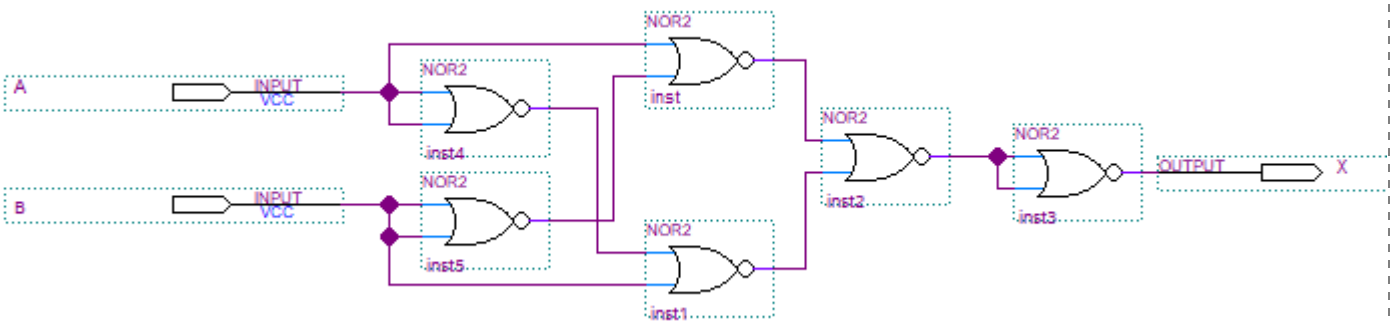
XOR je v Logické obvody na FPGA str. 22. Zadání vede na dvě možná řešení, lze vyjít jak z xor tak z xnor

A xor B= (not A and B) or (A and not B)

= not not (not not (not A and B) or not not (A and not B))

= not (not (A or not B) **nor** not (not A or B))

= not ((A **nor** not B) **nor** (not A **nor** B))

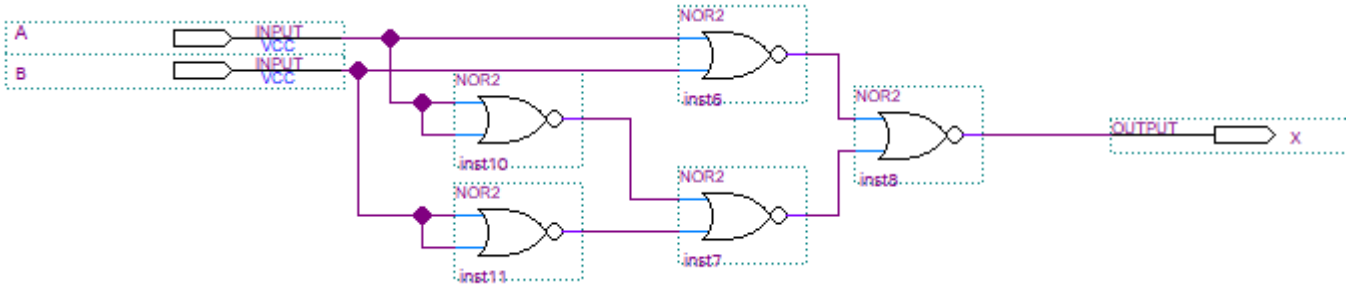


A xor B= not (A ≡ B) = not (A xnor B)

= not ((not A and not B) or (A and B)) = (not A and not B) **nor** (A and B)

= not not (not A and not B) **nor** not not (A and B)

= not (A or B) **nor** not (not A or not B)) = (A **nor** B) **nor** (not A **nor** not B))



---

7. Moodle: Vybrané příklady ze zkoušek str. 7 a 8

8. for-loop: 1bitové prediktory v Not-Taken, NT 2 chybné predikce: 2(vnější) + 5*2(vnitřní) = **12**,

2bitové prediktory WT 1 chybná predikce: 1(vnější) + 5*1(vnitřní) = **6**

2000 prvků pole * **8** (double) = **16000** bytů → **vejde se celé do cache**

Bude se do ní načítat po blocích, každý vždy celý, a tak jeho načtení má 1 miss, 1 blok obsahuje dle zadání 4 slova, 1 slovo má u 64 bitového procesoru 8 bytů, tedy celé double.
První blok začíná na adrese dělitelné délkou bloku, není tedy chybný memory align - ten by počet miss zvýšil o 1. Minimální počet cache miss je tedy jen počet prvků pole / délkou bloku = **2000/4 = 500**

9. Moodle: 6. přednáška snímek 45