

3 KOMBINAČNÉ OBVODY

3.1 TEÓRIA

Kombinačné obvody – sú logické obvody, ktorých výstup závisí len od kombinácie vstupov v danom časovom okamihu (obvody ktoré nemajú pamäť). Medzi takéto obvody môžeme zaradiť prevodníky kódov, kódery a dekódery, multiplexory a demultiplexory, komparátory binárnych čísel, generátory parity, sčítačky.

BCD kód (Binary Coded Decimal) – je to štvor bitový, váhový, binárny kód, kde každej desiatkovej číslici od 0 – 9 zodpovedajú štvor bitové binárne kódové slová. Zo 16 možných stavov sa využívajú 10. 6 nežiadúcich sa vylúči pomocou korelačných faktorov.

Gray – ov kód – je to n- bitový lineárny kód, u ktorého sa každá susedná kódová kombinácia líši len v 1 bite. Tým sa odstráni nebezpečenstvo viacnásobných zmien, čo je výhoda voči BCD kódu. Využíva sa pri číslicovom spracovaní obrazov. Dosahuje vyššiu kompresiu.

Kód +3 ku BCD – je 4 bitový, binárny, doplnkový kód. Vytvorí sa pripočítaním čísla 3 k desiatkovému číslu a vyjadrením v BCD kóde.

Syntézu kombinačných obvodov môžeme popísať pomocou nasledovných bodov:

- Podľa činnosti kombinačného obvodu zostavíme pravdivostnú tabuľku, v ktorej určíme vstupy a výstupy
- Z pravdivostnej tabuľky, alebo karnaughovej mapy určíme algebraické vyjadrenie výstupnej funkcie

- Na základe algebraického vyjadrenia výstupnej funkcie realizujeme navrhnutý obvod

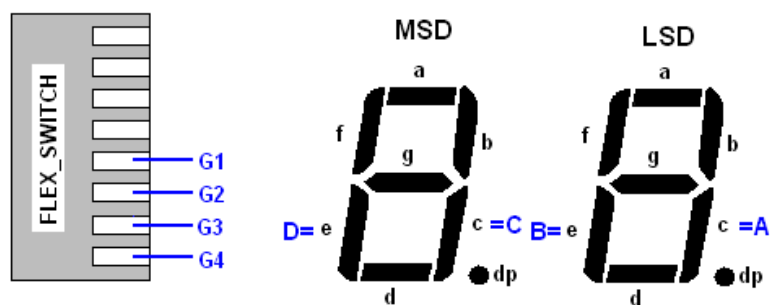
3.2 ZADANIE PRÍKLADU Č.1

Navrhnite prevodník z Grayovho kódu do kódu +3 ku BCD (0-9). Návrh realizujte pomocou hradiel NAND v grafickom editore vývojového prostredia Quartus II.

3.3 RIEŠENIE

3.3.1 ROZBOR

Po syntéze zadaného príklad, teda zostaveniu pravdivostnej tabuľky, karnaughových máp, určení algebraických funkcií nakreslíme schému logického obvodu. Tú potom realizujeme v grafickom editore Quartus II. Ako vstupy použijeme prepínače **FLEX_SWITCH** (obr.1). V hornej polohe reprezentujú logickú úroveň „1“ a v dolnej polohe reprezentujú logickú úroveň „0“. Výstupy budeme zobrazovať na sedem segmentovej LED, pričom použijeme segmenty **e, c** MSD sedem segmentovky a segmenty **e, c** LSD sedem segmentovky (obr.1). Displej reaguje na logickú úroveň „0“, preto musíme na každý výstup zaradiť jedno hradlo NOT.



Obr.1: Zobrazenie prepínačov **FLEX_SWITCH** a sedem segmentoviek **MSD** a **LSD**

Aby nesvietili nepoužívané segmenty pripojíme ich na VCC alebo príslušné piny necháme v stave vysokej impedancie. Všetky potrebné označenia pinov, typ, číslo a funkcia sú uvedené v tab.1.

Názov pinu	Typ pinu	Číslo pinu	Funkcia pinu
G1	Vstup	38	Prepínač (0 = poloha dole, 1 = poloha hore)
G2	Vstup	39	Prepínač (0 = poloha dole, 1 = poloha hore)
G3	Vstup	40	Prepínač (0 = poloha dole, 1 = poloha hore)
G4	Vstup	41	Prepínač (0 = poloha dole, 1 = poloha hore)
A	Výstup	19	Segment c LSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
B	Výstup	21	Segment e LSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
C	Výstup	8	Segment c MSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
D	Výstup	11	Segment e MSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other0	Výstup	17	Segment a LSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other1	Výstup	18	Segment b LSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other2	Výstup	20	Segment d LSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other3	Výstup	23	Segment f LSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other4	Výstup	24	Segment g LSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other5	Výstup	25	Segment dp LSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other6	Výstup	6	Segment a MSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other7	Výstup	7	Segment b MSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other8	Výstup	9	Segment d MSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other9	Výstup	12	Segment f MSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other10	Výstup	13	Segment g MSD (0=LED ON-svieti, 1=LED OFF-nesvieti)
Other11	Výstup	14	Segment dp MSD (0=LED ON-svieti, 1=LED OFF-nesvieti)

Tab.1: Tabuľka pinov

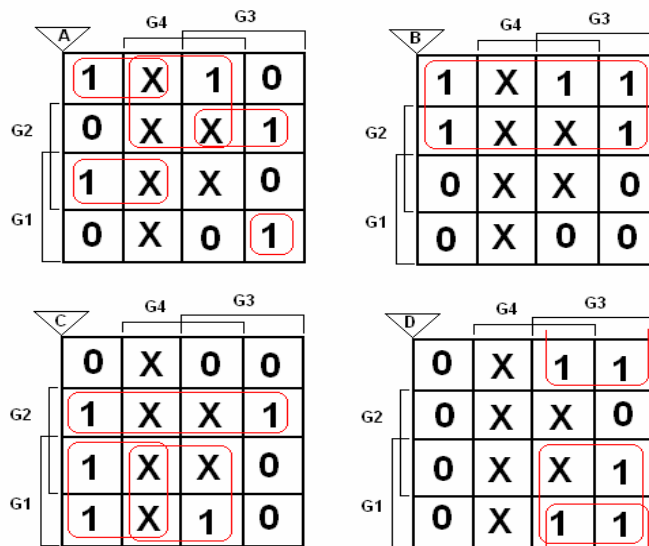
3.3.2 SYNTÉZA

Na začiatku návrhu podľa činnosti prevodníka zostavíme pravdivostnú tabuľku (tab.2), v ktorej určíme vstupy a výstupy.

dekadicky	vstupy				výstupy			
	Grayov kód				Kód +3 ku BCD			
	G4	G3	G2	G1	D	C	B	A
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	1	0	1	0	1
3	0	0	1	0	0	1	1	0
4	0	1	1	0	0	1	1	1
5	0	1	1	1	1	0	0	0
6	0	1	0	1	1	0	0	1
7	0	1	0	0	1	0	1	0
8	1	1	0	0	1	0	1	1
9	1	1	0	1	1	1	0	0

Tab.2. Pravdivostná tabuľka

Z pravdivostnej tabuľky zostavíme pre jednotlivé výstupné funkcie Karnaughove mapy (obr.2). Z Karnaughových máp určíme algebraické funkcie a zapíšeme ich pomocou Boolovej algebry do UDNF (úplná disjunktívna normálna forma) tvaru.



Obr.2: Karnaughove mapy pre jednotlivé výstupy

Z Karnaughových máp budú mať výstupné funkcie tvar:

$$A = \overline{G1}G4 + \overline{G1}G2\overline{G3} + \overline{G1}G2G3 + G1G2\overline{G3} + G1\overline{G2}G3\overline{G4} \quad (3.1)$$


$$B = \overline{G1} \quad (3.2)$$

$$C = \overline{G1}G2 + G1\overline{G3} + G1G4 \quad (3.3)$$

$$D = G1G3 + \overline{G2}G3 \quad (3.4)$$

Funkcie (3.1), (3.2), (3.3) a (3.4) prepíšeme pomocou dvojitej negácie a Boolovej algebry na súčin súčinov za účelom realizácie uvedených funkcií na základe zadania pomocou hradiel NAND.

3.3.3 OTVORENIE NOVÉHO PROJEKTU

Nový projekt otvoríme voľbou **New Project Wizard** z **File menu**. Ako prvé sa objaví úvodné okno, kde kliknutím na tlačidlo **Next**. Sa otvorí okno, v ktorom definujeme miesto uloženia, názov projektu a názov entity. Kliknutím na tlačidlo  sa objaví štruktúra adresárov, z ktorých si vyberieme ten, do ktorého chceme náš projekt ukladať. Pre tento projekt si vytvoríme adresár napr.: *Prevodnik_Grey*. V ďalších riadkoch definujeme názov projektu a názov najvyššej úrovne entity. V tomto prípade zvolíme rovnaké názov projektu a najvyššej úrovne entity ako názov adresára, kde ukladáme súbory nášho projektu - *prevodnik_gray*. Potom klikneme na tlačidlo **Next**.

V ďalšom okne môžeme priradiť súbory z iného projektu do tohto projektu, ak sú zhodné s tými, ktoré môžeme využiť pri návrhu. Nemusíme tú istú vec robiť dvakrát. Ak žiadne súbory nechceme priradiť, klikneme na tlačidlo **Next**. Pre náš projekt nepotrebuje priradiť iné súbory.

Stlačením **Next** sa objaví okno – tretie okno voľby **New project Wizard**, v ktorom definujeme rodinu obvodov, v našom prípade rodinu FLEX10K.

V časti **Target device** zaškrtneme možnosť *Specific devices selected in 'Available devices' list*.

V spodnom okne vyberieme presný typ obvodu, s ktorým chceme pracovať. V našom prípade súčiastku EPF10K20RC240-4. Vo web verzii vývojového prostredia Quartus II ver.4.2 sa súčiastka EPF10K20RC240-4 nemusí nachádzať. Preto zvolíme súčiastku EPF10K20RC240-3. Rozdiel je len v rýchlosti logiky súčiastok.

Kliknutím na tlačidlo **Next** sa otvorí štvrté okno voľby **New Project Wizard**, v ktorom môžeme nastaviť nástroje EDA. V tomto okne nebudeme nič nastavovať. Stlačíme **Next**.

V poslednom piatom okne, sú zobrazené všetky naše nastavenia. Ak s nimi súhlasíme klikneme na tlačidlo **Finish**. Ak chceme niektoré údaje zmeniť, klikneme na tlačidlo **Back**.

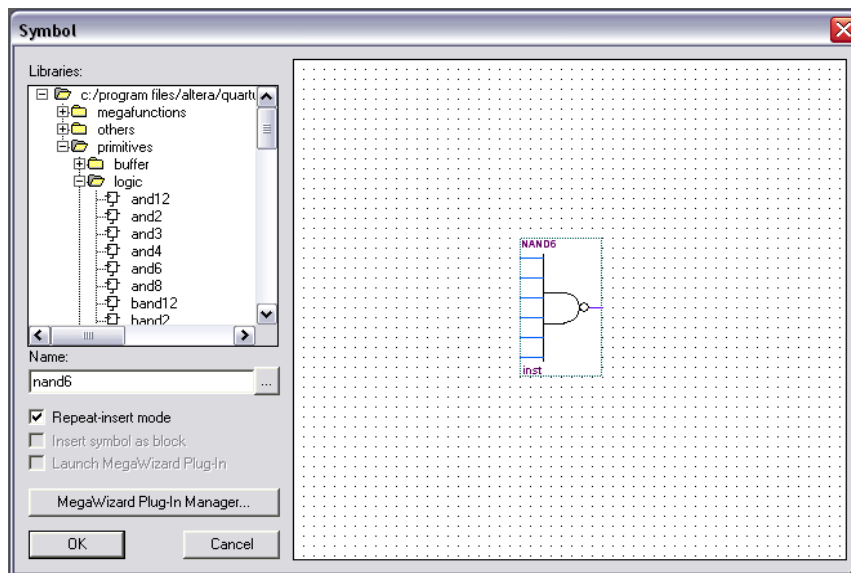
3.3.4 VYTVORENIE GRAFICKÉHO NÁVRHU PROJEKTU

Postupujeme podľa nasledovného postupu:

- Z **File Menu** vyberieme položku **New**
- V záložke **Design Files** zvolíme **Block Diagram/Schematic File**
- Kliknutím na tlačidlo **OK** sa otvorí okno grafického editora
- Z **File Menu** vyberieme položku **Save As**
- Vyberieme adresár *Prevodnik_Gray*, do ktorého uložíme náš súbor z názvom *prevodnik_gray.bdf*. Pod riadkom, kde sa definuje názov ukladaného súboru zaškrtneme voľbu **Add file to curent project** (pridať súbor do vlastného projektu).
- Kliknutím na tlačidlo **Save** uložíme a zároveň vložíme náš súbor do projektu.

3.3.5 VYTVORENIE SCHÉMY

Na panely nástrojov klikneme na **Symbol Tool**. V okne, ktoré sa objaví (obr.3) vyberieme *c:\quartus\libraries* → **primitives** → **logic** a potrebné hradlo. Voľbu potvrdíme kliknutím na **OK**. Symbol umiestnime na požadované miesto pohybom myšky a kliknutím ľavého tlačidla. Takto postupne vyberieme pre náš návrh všetky potrebné hradlá, vstupné a výstupné piny v knižnici *c:\quartus\libraries* → **primitives** → **pin**. Jednotlivé hradlá navzájom pospájame vodičmi podľa schémy, ktorú sme dostali syntézou. Klikneme na **Orthogonal Node Tool**, priblížime sa s myšou k vývodu symbolu a ak sa kurzor myši zmení na krížik klikneme a držíme ľavé tlačidlo myši a zároveň sa pohybujeme k vývodu ktorý chceme spojiť.



Obr. 3: Okno výberu prvkov

Názvy a hodnoty k vstupným/ výstupným pinom priradíme tak, že klikneme pravým tlačidlom na príslušnom pine a vyberieme možnosť **Properties**. Zobrazí sa okno, v ktorom v riadku **pin name** zadefinujeme názov pinu a v **Default value** zadefinujeme hodnotu vstupného pinu (VCC).

Výsledkom je schéma nášho projektu (obr.4).

	To	Location	General Function	Special Function	Reserved
1	G1	PIN_38	Row I/O		
2	G2	PIN_39	Row I/O		
3	G3	PIN_40	Row I/O		
4	G4	PIN_41	Row I/O		
5	A	PIN_19	Row I/O		
6	B	PIN_21	Row I/O		
7	C	PIN_8	Row I/O		
8	D	PIN_11	Row I/O	CLKUSR	
9	other1	PIN_18	Row I/O		
10	other2	PIN_20	Row I/O		
11	other3	PIN_23	Row I/O	RDYnBSY	
12	other4	PIN_24	Row I/O		
13	other5	PIN_25	Row I/O		
14	other6	PIN_6	Row I/O		
15	other7	PIN_7	Row I/O		
16	other8	PIN_9	Row I/O		
17	other9	PIN_12	Row I/O		
18	other10	PIN_13	Row I/O		
19	other11	PIN_14	Row I/O		
20	other0	PIN_17	Row I/O		
21	<<new>>	<<new>>			

Obr. 5: Okno priradenia pinov

Zadeinovaním všetkých pinov zavrieme okno **Pins z Assignments menu** a zmeny uložíme.

3.3.6 KOMPILÁCIA

Kompiláciu spustíme voľbou **Start Compilation** z **Processing menu**, ikonou na panely nástrojov alebo voľbou **Compiler Tool** z **Tools menu**.

3.3.7 SIMULÁCIA

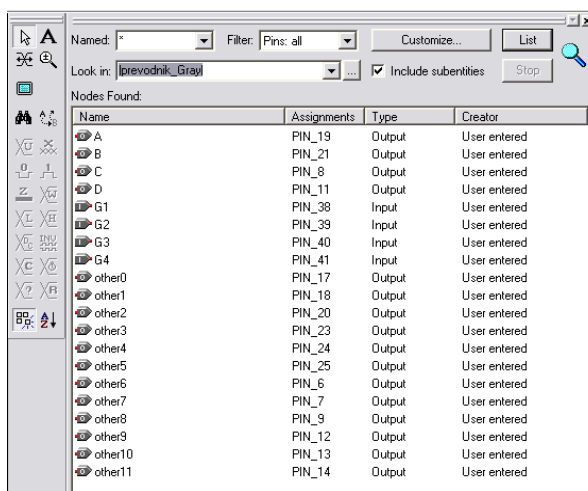
3.3.7.1 ČASOVÁ SIMULÁCIA

Po úspešnej kompilácii môžeme prejsť na simuláciu projektu. Simuláciu urobíme podľa nasledujúceho postupu:

- Vytvoríme vektorový súbor priebehu signálov – **Vector Waveform file (.vwf)**, voľbou **New** z **File menu**
- Pomocou voľby **Save As** z **File menu** uložíme tento vektorový súbor v tomto prípade s názvom *prevodnik_gray.vwf*
- Pomocou **Node Finder** (obr.6) vložíme do tohto súboru všetky vstupy a výstupy (stimuly), ktoré chceme simulovať. **Node Finder** otvoríme nasledujúcim postupom: **View**→**UtilityWindows**→**Node Finder**. Označíme požadované piny (Shift + ľavé

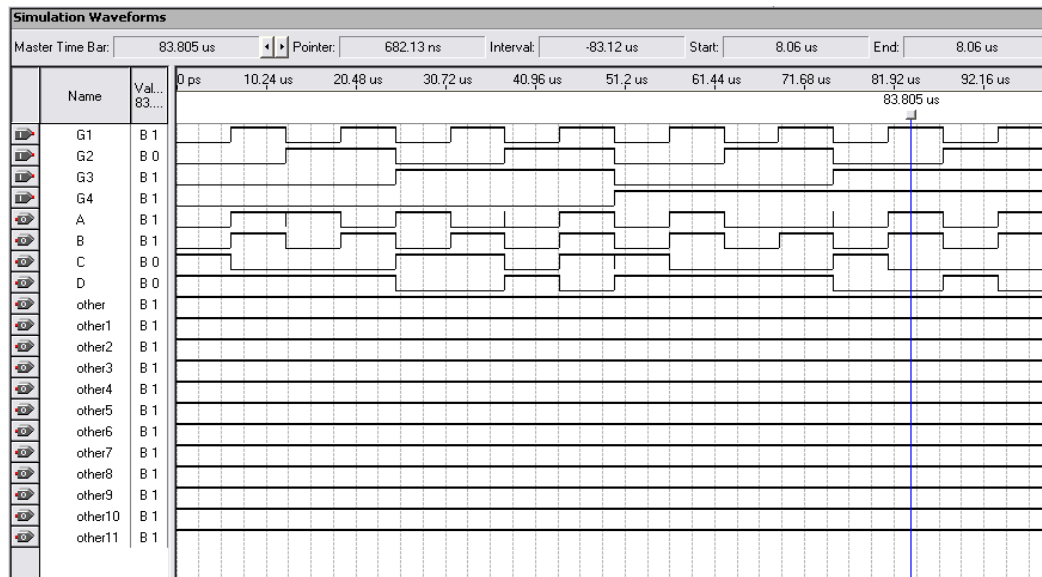
tlačidlo myši), skopírujeme ich (Ctrl+C) a vložíme do vektorového súboru (Ctrl+V). Druhou možnosťou je chytiť myšou požadovaný pin a jednoducho ho presunúť do vektorového súboru (.vwf). V našom prípade presunieme piny: G1, G2, G3, G4, A, B, C, D.

- Nastavíme koncový čas simulácie. V položke **Edit→End Time** nastavíme hodnotu time na 100 μ s.
- Klikneme pravým tlačidlom myši na pin G1, v tabuľke ktorá sa objaví vyberieme **Value→Count Value**. V zobrazenom okne v záložke **Timing** zadefinujeme **End Time**: 100 μ s a **Count Event**: 6,25 μ s. Pre ostatné piny postup opakujeme pričom pre pin G2 zadefinujeme **End Time**: 100 μ s a **Count Event**: 12,5 μ s, pre pin G3 zadefinujeme **End Time**: 100 μ s a **Count Event**: 25 μ s, pre pin G4 zadefinujeme **End Time**: 100 μ s a **Count Event**: 50 μ s. Volili sme dosť dlhé časy, aby hradlá stíhali preklápať. (Môžeme zadefinovať aj iné hodnoty).



Obr.6: Node Finder

- Uložíme nastavenia voľbou **Save** z **File menu**, alebo prostredníctvom ikony na panely nástrojov. Potom spustíme simuláciu voľbou **Start Simulation** z **Processing menu**, alebo prostredníctvom ikony na panely nástrojov.
- Výsledkom je okno **Simulation Report** (obr.7), v ktorom sa zobrazí časová analýza prevodníka.

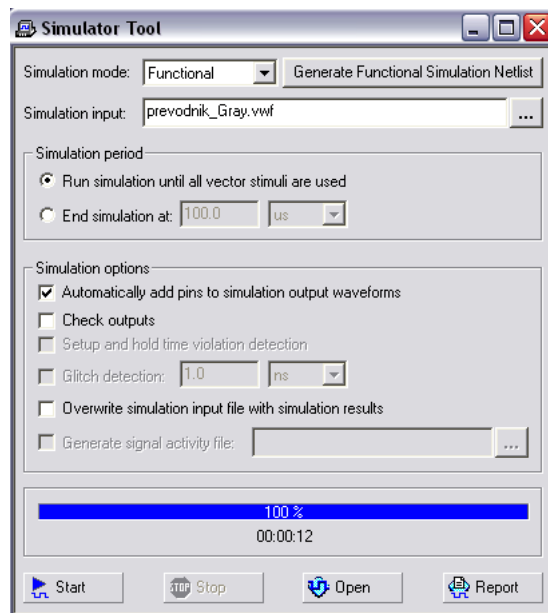


Obr.7: Výsledky časovej simulácie

Zo zobrazených priebehov a pravdivostnej tabuľky môžeme zistiť, či prevodník pracuje správne. Musíme však uvažovať s tým, že na každý výstup sme umiestnili hradlo NOT (invertor) z dôvodu, že LED svieti pri logickej úrovni „0“. Preto pri vyhodnocovaní výstupov a pri porovnaní s výstupmi uvedenými v Tab.2 logickú nulu berieme ako „1“ a logickú jednotku ako logickú „0“.

3.3.7.2 FUNKČNÁ SIMULÁCIA

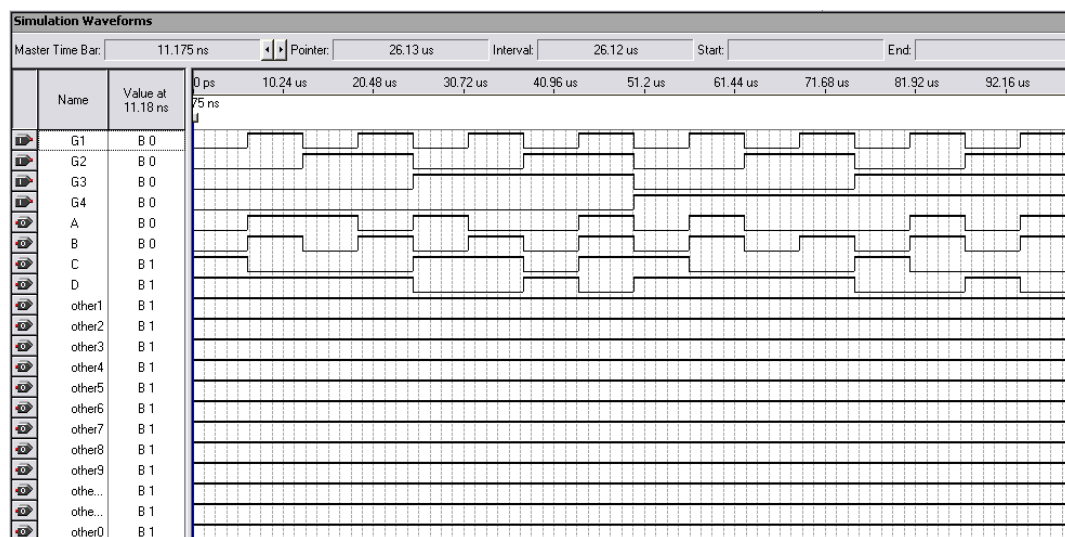
Pre funkčnú simuláciu platia tie isté nastavenia ako pre časovú simuláciu. Funkčnú simuláciu spustíme voľbou z **Tools menu, Simulator Tool** (obr.8).



Obr. 8: Okno Simulator Tool

V riadku **Simulation Mode** vyberieme možnosť **Functional**. Pred začatím funkčnej simulácie musíme vygenerovať **Netlist** pre simuláciu. Ten vygenerujeme stlačením tlačidla **Generate Functional Simulation Netlist**. Ak máme vygenerovaný **Netlist** začneme simuláciu kliknutím na tlačidlo **Start**.

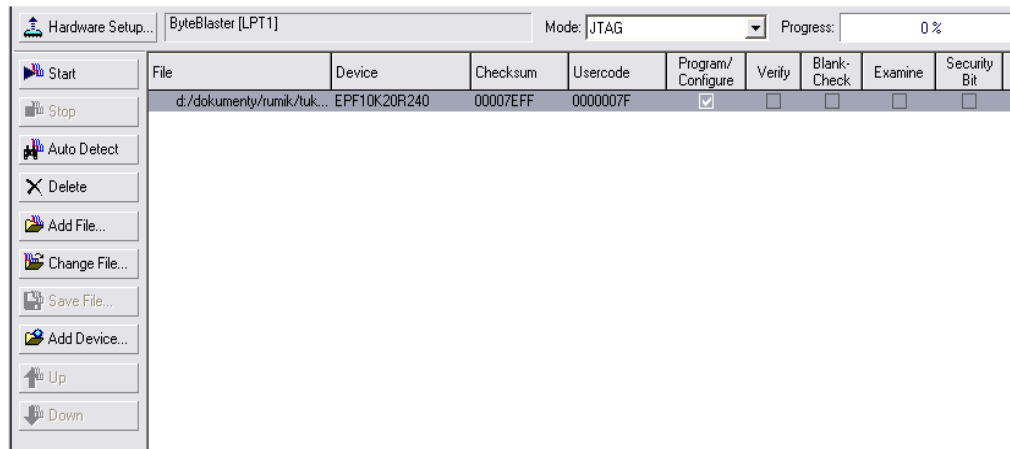
Výsledky simulácie si môžeme pozrieť po stlačení tlačidla **Report** (obr.9).



Obr.9: Výsledky funkčnej simulácie

3.3.8 PROGRAMOVANIE/KONFIGURÁCIA

Po bezchybnej kompilácii, a správnymi výsledkami simulácií, môžeme prejsť k programovaniu obvodu. Programovanie spustíme voľbou **Tools→Programmer**. Po spustení sa otvorí okno programátora (obr.10).



Obr.10: Okno programovania/konfigurácie

Proces programovania spustíme zaškrtnutím políčka **Program/Configure** a kliknutím na tlačidlo **Start**. Nastavenie **Hardware Setup** vid' cvičenie č.2 – 2.3.6 Konfigurácia. Od tohto okamihu je prevodník nakonfigurovaný v súčiastke, na doske UP1 CPLD.

3.4 ZADANIE PRÍKLAD Č.2

Navrhnite prevodník z BCD kódu do kódu +3 ku BCD (0-9). Návrh realizujte pomocou hradieľ NAND, v grafickom editore vývojového prostredia Quartus II.

3.5 RIEŠENIE

3.5.1 ROZBOR

V tomto príklade postupujeme analogicky ako v príklade č.1. Po syntéze zadaného príkladu, otvoríme nový projekt, zhotovíme v grafickom editore zapojenie, vykonáme

kompiláciu a simuláciu. Použijeme tie isté prepínače a sedem segmentovky (obr.1) ako v príklade č.1, platí teda tá istá tabuľka pinov (tab.1).

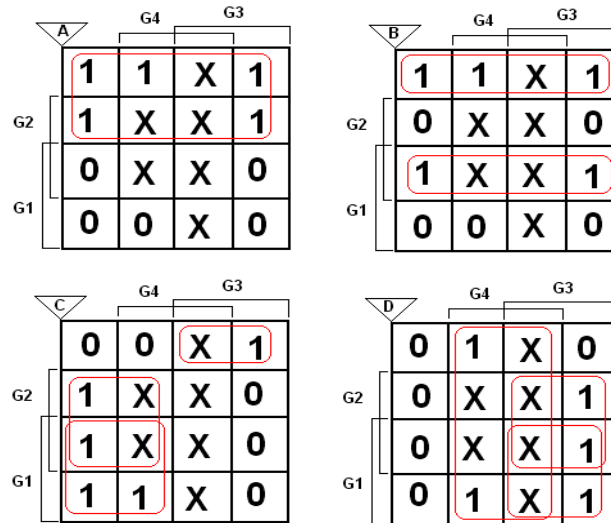
3.5.2 SYNTÉZA

Ako prvé si zostavíme pravdivostnú tabuľku, v ktorej si označíme vstupy, výstupy (tab.3).

dekadicky	vstupy				výstupy			
	BCD kód				Kód +3 ku BCD			
	G4	G3	G2	G1	D	C	B	A
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

Tab. 3: Pravdivostná tabuľka prevodníka

Z pravdivostnej tabuľky zostavíme Karnaughove mapy (obr.11) z ktorých určíme algebraické funkcie pre každý výstup. Funkcie potom upravíme podľa Boolovej algebry na tvar, ktorý môžeme realizovať pomocou hradiel NAND.



Obr.11: Karnaughove mapy prevodníka

Z Karnaughových máp dostávame výstupné funkcie v tvare:

$$A = \overline{G1} \quad (3.5)$$

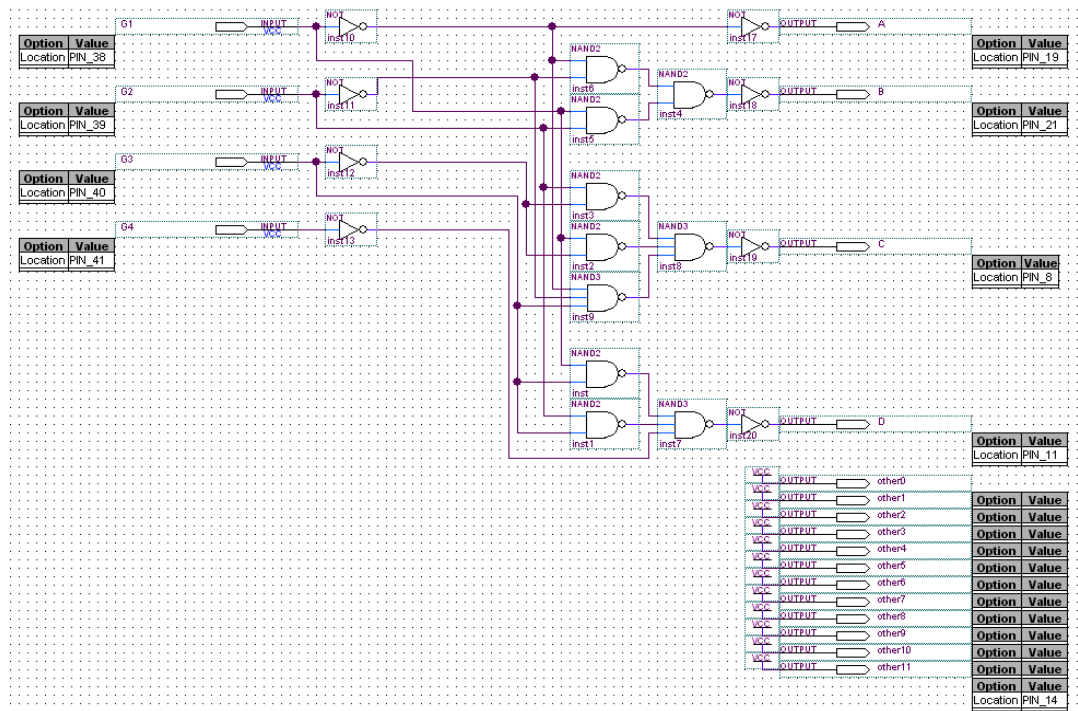
$$B = \overline{G1G2} + G1G2 \quad (3.6)$$

$$C = G2\overline{G3} + G1\overline{G3} + \overline{G1}G2G3 \quad (3.7)$$

$$D = G4 + G1G3 + G2G3 \quad (3.8)$$

Funkcie (3.5), (3.6), (3.7) a (3.8) prepíšeme pomocou dvojitej negácie a Boolovej algebry na súčin súčinov za účelom, aby sme mohli uvedené funkcie realizovať podľa zadania pomocou hradiel NAND.

Na základe predchádzajúcej syntézy a použitím grafického editora (po otvorení nového projektu, v tomto prípade s názvom *prevodník_BCD*) realizujeme zapojenie prevodníka BCD kódu na kód +3 ku BCD (obr.12).



Obr.12: Zapojenie prevodníka

Po zhotovení schémy prevodníka v blokovom editore návrhového softvéru Quartus II, priradíme vstupom a výstupom názvy a čísla pinov podľa postupu uvedenom v predchádzajúcom príklade. Čísla pinov priradíme podľa zoznamu uvedeného v tab.1. Výslednú schému uložíme voľbou **File**→**Save** (Ctrl + S). Po priradení pinov projekt skompilujeme (**Processing**→**Start Compilation**), podľa postupu uvedenom v predchádzajúcom príklade.

Po bezchybnej kompilácii projektu, vykonáme simuláciu. Voľbou **New** z **File menu** otvoríme nové okno waveform. V záložke **Other files** výberom na položku **Vector Waveform File** a potvrdením na **OK**. Uložíme ho voľbou **File**→**Save As** v tomto prípade pod názvom *prevodnik_BCD.vwf*. Postupom v predchádzajúcom príklade a pomocou **Node Finder** pridáme do vektorového okna všetky potrebné vstupné a výstupné piny (G1, G2, G3, G4, A, B, C, D). V položke **Edit**→**End Time** nastavíme koncový čas simulácie na hodnotu 100µs. Potom kliknutím pravého tlačidla na príslušný vstupný pin a výbere položky **Value**→**Count Value** nastavíme v záložke **Timing** tieto hodnoty pre jednotlivé piny:

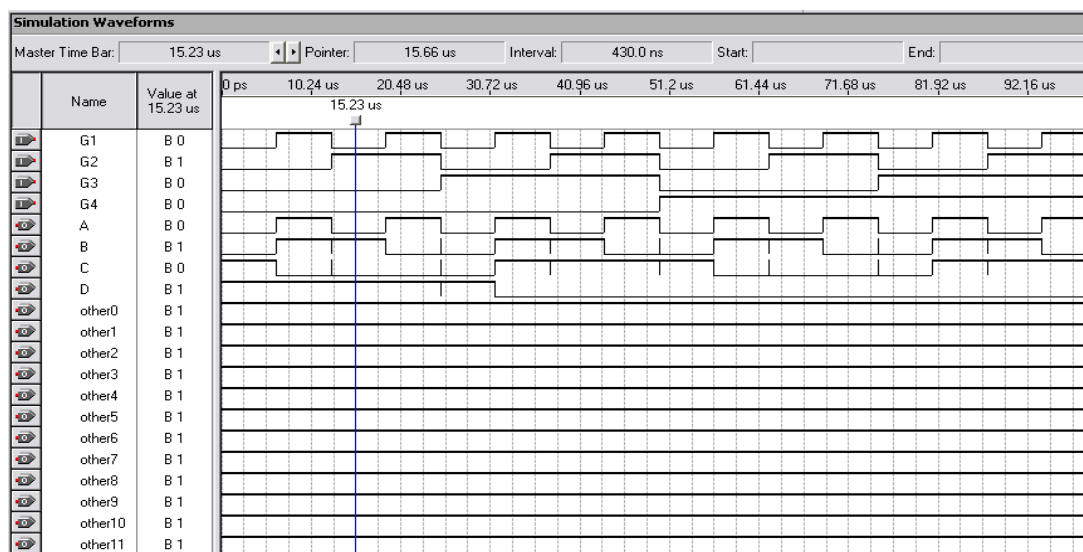
G1 - End Time: 100 μ s a **Cout Event:** 6,25 μ s,

G2 - End Time: 100 μ s a **Cout Event:** 12,5 μ s,

G3 - End Time: 100 μ s a **Cout Event:** 25 μ s,

G4 - End Time: 100 μ s a **Cout Event:** 50 μ s.

Po uložení nastavení spustíme simuláciu voľbou v **Processing menu**, **Start Simulation**. Po úspešnej simulácii sa zobrazia priebehy výstupov na základe vstupných časových priebehov (obr.13).



Obr.13: Výsledky simulácie

Z grafických priebehov a na základe pravdivostnej tabuľky, zhotovenej v etape syntézy, môžeme zistiť, či by navrhnutý prevodník pracoval správne. Musíme brať do úvahy tú skutočnosť, že v návrhu je na každom výstupe ešte zaradený invertor z dôvodu ktorý bol popísaný v príklade č.1. Preto, pre správne vyhodnotenie simulácie musíme logickú úroveň „0“ brať ako jednotku a logickú úroveň „1“ ako nulu.

Ak sú výsledky simulácie správne, môžeme prejsť k programovaniu/konfigurácii projektu do obvodu. Použijeme postup, ktorý bol opísaný v príklade číslo 1. Spustíme programovacie okno voľbou **Programmer** z **Tools menu**. Zaškrtneme políčko **Program/Configure**, nastavíme výstupné zariadenie vid' cvičenie č.2 – 2.3.6 Konfigurácia a klikneme na tlačidlo **Start**. Od tohto okamihu je projekt naprogramovaný/nakonfigurovaný do súčiastky na doske UP1 CPLD.