

**Slovenská technická univerzita v Bratislave**  
**FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLOGIÍ**  
Študijný program: Počítačové a komunikačné systémy a siete

---

Tím č. 8

**Univerzálny virtuálny verifikačný panel logických  
obvodov**

**Projektová dokumentácia**

Ročník: 1. Ing.  
Predmet: Tímový projekt  
Pedagogický vedúci tímu: Prof. Ing. M. Kolesár, CSc.  
Ak. rok: 2009/2010

Bc. Michal Kudlačák  
Bc. Stanislav Martinický  
Bc. Juraj Orságh  
Bc. Ján Pivarček  
Bc. Juraj Sebín  
Bc. Marek Sivák

# Obsah

<b>Obsah.....</b>	<b>2</b>
<b>1 Úvod.....</b>	<b>4</b>
1.1 Účel a rozsah dokumentu .....	4
1.2 Prehľad dokumentu .....	4
1.3 Zadanie projektu.....	4
1.4 Ciele projektu .....	5
1.5 Zoznam skratiek .....	5
1.6 Zoznam pojmov.....	5
<b>2 Analýza problémovej oblasti.....</b>	<b>6</b>
2.1 Teória .....	6
2.1.1 Výroky.....	6
2.1.2 Boolovská algebra a jej modely operácií .....	7
2.1.3 Úplný súbor logických členov.....	8
2.1.3.1 Logické obvody.....	14
2.2 Softvér na podporu modelovania logických funkcií .....	16
2.2.1 Log .....	16
2.2.2 LOGiX.....	17
2.2.3 ATANUA .....	18
2.2.4 SIMCIR .....	19
2.2.5 Porovnanie analyzovaného softvéru.....	20
<b>3 Špecifikácia riešenia.....</b>	<b>21</b>
3.1 Analýza a špecifikácia zadania .....	21
3.2 Funkcionálne požiadavky na softvér.....	22
3.2.1 Popis prípadov použitia.....	23
3.3 Nefunkcionálne požiadavky na softvér .....	24
3.4 Požiadavky na grafické rozhranie .....	24
<b>4 Návrh riešenia.....</b>	<b>26</b>
4.1 Hrubý návrh riešenia .....	26
4.1.1 Modul logických súčiastok.....	26
4.1.2 Modul grafického používateľského rozhrania.....	28

4.1.3	Modul podporných funkcií.....	29
<b>5</b>	<b>Prototyp.....</b>	<b>31</b>
5.1	Požiadavky na aplikáciu.....	31
5.2	Dosiahnuté výsledky .....	32
<b>6</b>	<b>Použitá literatúra.....</b>	<b>35</b>

# 1 Úvod

## 1.1 Účel a rozsah dokumentu

Dokument je výsledkom spoločnej práce šiestich študentov v predmete Tímový projekt v akademickom roku 2009/2010. Venuje sa najmä problematike logických obvodov a verifikačných programov pomocou, ktorých je možné modelovať rôzne logické obvody. Obsahuje teoretickú dokumentáciu, ktorá popisuje jednotlivé aspekty riešenia zadaného problému ako je analýza, špecifikácia a návrh riešenia verifikačného panelu. Tento dokument je určený hlavne študentom a pedagógom FIIT STU.

## 1.2 Prehľad dokumentu

Dokument je rozdelený do jednotlivých hlavných kapitol, ktoré zoskupujú danú časť riešenia problému. V prvej kapitole je uvedený účel a rozsah dokumentu, prehľad dokumentu, zadanie projektu, cieľ projektu a zoznam skratiek. Druhá kapitola analyzuje problematiku logických obvodov a poskytuje krátky prehľad existujúcich programov, ktoré slúžia na modelovanie logických obvodov. V tretej kapitole uvádzame špecifikáciu zadania a požiadaviek na výsledný produkt. V štvrtej kapitole je uvedený hrubý návrh riešenia, ktorý pozostáva z troch modulov. Piata kapitola opisuje vytvorený prototyp aplikácie, požiadavky na aplikáciu a dosiahnuté výsledky v tejto fáze riešenia. Posledná kapitola obsahuje zoznam použitej literatúry.

## 1.3 Zadanie projektu

Navrhnite a implementujte programový systém pre osobný počítač, pomocou ktorého možno zostaviť štruktúru a ručne overiť funkciu logického kombinačného obvodu s normálnou štruktúrou, ktorý má najviac štyri vstupy a štyri výstupy.

Programový systém má umožniť voľbu podľa možnosti čo najväčšieho počtu režimov činnosti na základe zadaných úplných súborov logických členov s konečným počtom vstupov.

Nastavovanie hodnôt vstupných premenných (vstupných vektorov) treba umožniť pomocou virtuálnych tlačidiel a hodnoty výstupných premenných (výstupných vektorov) majú byť signalizované virtuálnymi žiarovkami.

Programový systém treba navrhnuť tak, aby bol použiteľný v pedagogickom procese pre predmet Logické obvody.

## 1.4 Ciele projektu

Cieľom nášho projektu je vytvoriť programový systém spustiteľný na osobnom počítači, ktorý bude modelovať funkcionality logických obvodov. Dôraz je kladený na implementovanie úplného súboru členov, funkcionality a grafické používateľské rozhranie. Taktiež je dôležité, aby tento systém bol vyhovujúci na používanie v pedagogickom procese pre predmet Logické obvody. A preto je dôležité ho navrhnuť tak, aby bol jednoduchý, názorný a intuitívne ovládateľný. Aby výsledný produkt spĺňal všetky kritéria je najprv potrebné analyzovať teoretickú časť problematiky a existujúce softvérové riešenia používané na modelovanie logických obvodov. Analýza je potrebná pre následnú špecifikáciu zadania a požiadaviek systému, a taktiež na vytvorenie správneho návrhu riešenia.

## 1.5 Zoznam skratiek

- ČSN – Česká Technická Norma

## 1.6 Zoznam pojmov

- Drag & Drop – Technika, ktorá sa používa v grafickom používateľskom prostredí pri práci s objektmi. Pomocou myši je možné presúvať objekty jednoduchým stlačením tlačidla myši a jej súčasným presúvaním do cieľového miesta na obrazovke. Po pustení tlačidla sa objekt umiestni do cieľového bodu. Je to funkcia, ktorá umožňuje jednoduché ovládanie pomocou myši.

## 2 Analýza problémovej oblasti

Analýza problémovej oblasti je dôležitá na pochopenie samotnej problematiky a stanovanie cieľov a požiadaviek projektu. V tejto 4asti sa budeme venovať teoretickej časti problematiky a taktiež aj existujúcim softvérom, ktoré modelujú a simulujú logické obvody.

### 2.1 Teória

Na začiatok je veľmi dôležité uviesť teóriu, ktorá sa skrýva za celou problémovou oblasťou. Je to potrebné na lepšie pochopenia logiky, ktorá musí byť implementovaná vo výslednom produkte. Kapitola teória je spracovaná na základe literatúry [1][2][3][4].

#### 2.1.1 Výroky

Za výrok považujeme každú oznamovaciu vetu o ktorej má zmysel hovoriť či je pravdivá alebo nie. Výroky sa označujú veľkými písmenami A, B, ..., Z. Výrok je základným stavebným prvkom výrokovej logiky, ktorú využívajú logické obvody.

Informácia o tom, ktorý z dvoch možných prípadov pri vyhodnocovaní výroku nastal, predstavuje logickú hodnotu výroku. Logická funkcia je formálnym popisom výroku a nositeľkou pravdivostnej hodnoty výroku. Ide o zobrazenie z množiny výrokov do množiny  $\{0,1\}$ . Logická funkcia priradzuje výroku pravdivostnú hodnotu.

Logickými premennými (logická 1, logická 0) sú popisované pravdivostné hodnoty jednoduchých výrokov. Jednoduché výroky je možné spájať pomocou logických spojok, tým vznikajú nové zložené výroky. Ich pravdivostná hodnota závisí iba od pravdivostných hodnôt výrokov, z ktorých sú zložené a samozrejme od zvolenej (logickej) spojky.

Hlavné zložené výroky sú konjunkciu, disjunkciu, implikáciu a ekvivalenciu. Konjunkcia sa v bežnej reči vyjadruje ako "A a súčasne B", označuje sa  $A \wedge B$ . Disjunkcia sa vyjadruje ako "A alebo B" a označuje sa  $A \vee B$ . Implikácia, má tvar "ak A, potom B" a označuje sa  $A \Rightarrow B$ . Ekvivalencia má tvar "A práve vtedy, keď, B" a značí sa  $A \Leftrightarrow B$ . Je potrebné spomenúť

aj negáciu. Negácia sa vzťahuje iba na jeden výrok. Negácia výroku A je výrok A', popiera to, čo tvrdí pôvodný výrok. V hovorovej reči sa používa ako "neplatí A" alebo "nie je pravda, že A". Označuje sa aj  $\neg A$ , prípadne  $\bar{A}$ .

### 2.1.2 Boolovská algebra a jej modely operácií

Booleova algebra využíva symboliku analogickú ako v klasickej algebre. V booleovej algebre značíme premenné a, b, ..., z.

Unárne booleovské funkcie pracujú iba s jednou premennou. Najznámejšia unárna funkcia je logická negácia.

Binárne booleovské funkcie pracujú s dvomi premennými a, b. Najznámejšími binárnymi funkciami sú konjunkcia, disjunkcia, negácia konjunkcie, negácia disjunkcie a neekvivalencia.

V booleovej algebre platia nasledujúce zákony:

Zákon	Súčtový tvar	Súčinový tvar
Komutatívny zákon	$a+b=b+a$	$a.b=b.a$
Asociatívny zákon	$a+(b+c)=(a+b)+c$	$a.(b.c)=(a.b).c$
Distributívny zákon	$a+(b.c)=(a+b).(a+c)$	$a.(b+c)=(a.b)+(a.c)$
De Morganov zákon	$\overline{a+b} = \bar{a}.\bar{b}$	$\overline{a.b} = \bar{a} + \bar{b}$
Zákon absorpcie	$a+a.b=a$	$a.(a+b)=a$
Zákon absorpcie negácie	$a + \bar{a}.b = a + b$	$a.(\bar{a} + b) = a.b$
Zákon vylúčenia	$a + \bar{a} = 1$	$a.\bar{a} = 0$
Zákon neutrálnosti 0 a 1	$a+0=a$	$a.1=a$
Zákon agresívnosti 0 a 1	$a+1=1$	$a.0=0$
Zákon dvojitej negácie $\overline{\bar{A}} = A$	$\neg\neg a=a$	$\neg\neg a=a$

Tabuľka č. 1: Boolovská algebra

Booleovské funkcie je možné realizovať (modelovať) pomocou rôznych technických prostriedkov. Fyzikálne systémy, ktoré realizujú tieto booleovské funkcie budeme nazývať logické členy. Pri realizácii pomocou elektrických prostriedkov sa najčastejšie používajú tieto dva spôsoby:

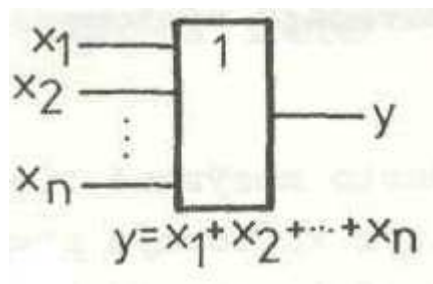
Využíva sa prítomnosť elektrického prúdu (napätia) na priradenie hodnoty 1 a jeho neprítomnosť na priradenie hodnoty 0. (V tomto prípade ide o tzv. pozitívnu logiku. Je možné uvažovať aj o opačnom priradení, vtedy hovoríme o negatívnej logike.

Využívajú sa technické zariadenia, ktoré sú schopné prevádzky pri dvoch (značne) odlišných hladinách elektrického napätia. Pre naše úvahy je výhodné jednu z nich označiť ako vysokú hladinu napätia (v praxi spravidla 5 V) a priradiť jej hodnotu 1 a druhú označiť ako nízku hladinu napätia (v praxi spravidla 0,5 V) a priradiť jej logickú hodnotu 0 (pozitívna logika).

### 2.1.3 Úplný súbor logických členov

Medzi základné logické členy patria :

- Logický člen alebo (OR) – realizuje operáciu logického súčtu,  $F(x_1, x_2) = x_1 + x_2$



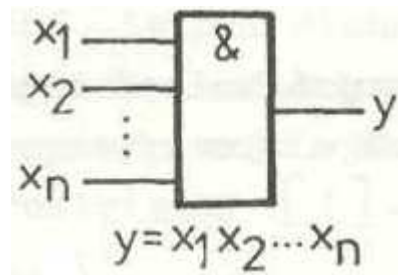
**Obrázok č.1:** Symbolická značka logického člena OR s viacerými vstupmi



$X_1$	$X_2$	$F(x_1, x_2)$
1	1	1
0	1	1
1	0	1
0	0	0

**Tabuľka č.2:** Pravdivostná tabuľka logického člena OR s 2 vstupmi

- Logický člen I (AND) – realizuje operáciu logického súčinu,  $F(x_1, x_2) = x_1 \cdot x_2$

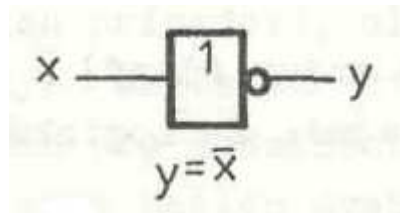


**Obrázok č.2:** Symbolická značka logického člena AND s viacerými vstupmi

$X_1$	$X_2$	$F(x_1, x_2)$
1	1	1
0	1	0
1	0	0
0	0	0

**Tabuľka č.3:** Pravdivostná tabuľka logického člena OR s 2 vstupmi

- Logický člen Invertor (NOT) – realizuje operáciu negácie,  $F(x) = \bar{x}$

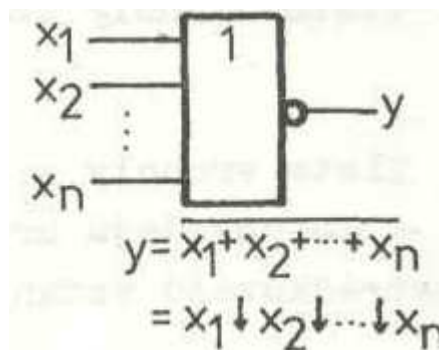


**Obrázok č.3:** Symbolická značka logického člena NOT

X	F(x)
1	0
0	1

**Tabuľka č.4:** Pravdivostná tabuľka logického člena NOT

- Piercov logický člen (NOR) – realizuje operáciu negácie logického súčtu,  $F(x_1, x_2) = \overline{x_1 + x_2} = x_1 \downarrow x_2$



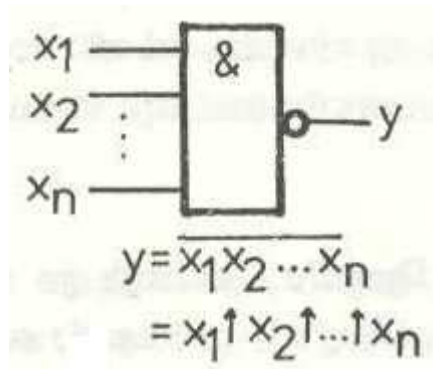
**Obrázok č.4:** Symbolická značka logického člena NOR s viacerými vstupmi

X <sub>1</sub>	X <sub>2</sub>	F(x <sub>1</sub> , x <sub>2</sub> )
1	1	0
0	1	0
1	0	0
0	0	1

**Tabuľka č.5:** Pravdivostná tabuľka logického člena NOR s 2 vstupmi

- Shefferov logický člen (NAND) – realizuje operáciu negácie logického súčinu,

$$F(x_1, x_2) = \overline{x_1 \cdot x_2} = x_1 \uparrow x_2$$



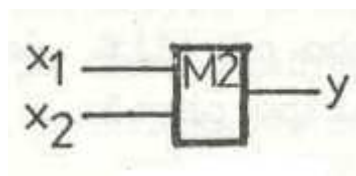
**Obrázok č.5:** Symbolická značka logického člena NAND s viacerými vstupmi

$X_1$	$X_2$	$F(x_1, x_2)$
1	1	0
0	1	1
1	0	1
0	0	1

**Tabuľka č.6:** Pravdivostná tabuľka logického člena NAND s 2 vstupmi

- Logický člen neekvivalencie (XOR) – realizuje operácie neekvivalencie,  $F(x_1, x_2) = \overline{x_1 \cdot x_2} + x_1 \cdot \overline{x_2} = x_1 \oplus x_2$ .

$$x_2 + x_1 \cdot \overline{x_2} = x_1 \oplus x_2$$

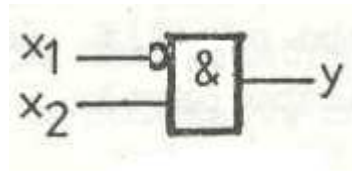


**Obrázok č.6:** Symbolická značka logického člena XOR

$X_1$	$X_2$	$F(x_1, x_2)$
1	1	0
0	1	1
1	0	1
0	0	0

**Tabuľka č.7:** Pravdivostná tabuľka logického člena XOR s 2 vstupmi

- Logický člen Zábrana – realizuje operáciu  $F(x_1, x_2) = \overline{x_1} \cdot x_2 = x_1 \rightarrow x_2$

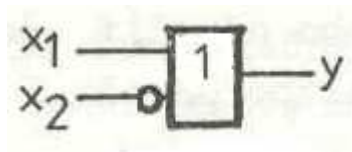


**Obrázok č.7:** Symbolická značka logického člena Zábrana

$X_1$	$X_2$	$F(x_1, x_2)$
1	1	0
0	1	1
1	0	0
0	0	0

**Tabuľka č.8:** Pravdivostná tabuľka logického člena Zábrana s 2 vstupmi

- Logický člen Implikácia – realizuje operáciu  $F(x_1, x_2) = \overline{x_2} + x_1 = x_1 \rightarrow x_2$

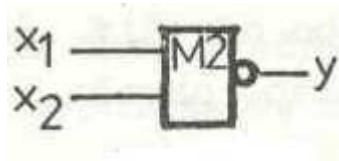


**Obrázok č.8:** Symbolická značka logického člena Implikácia

$X_1$	$X_2$	$F(x_1, x_2)$
1	1	1
0	1	0
1	0	1
0	0	1

**Tabuľka č.9:** Pravdivostná tabuľka logického člena Implikácia s 2 vstupmi

- Logický člen Ekvivalencia – realizuje operáciu  $F(x_1, x_2) = \overline{x_1} \cdot \overline{x_2} + x_1 \cdot x_2 = x_1 \equiv x_2$



**Obrázok č.9:** Symbolická značka logického člena Ekvivalencia

$X_1$	$X_2$	$F(x_1, x_2)$
1	1	1
0	1	0
1	0	1
0	0	1

**Tabuľka č.10:** Pravdivostná tabuľka logického člena Ekvivalencia s 2 vstupmi

Vezmime si niektorú množinu operácií  $G = \{ g_1, g_2, \dots, g_k \}$ ,  $k \geq 1$ . Množinu  $G$  nazývame funkčiou úplnou práve vtedy, ak pre každú Booleovu funkciu  $f$  existuje jej zodpovedajúci

výraz  $Z$ , t.j.  $f = |Z|$ , ktorý obsahuje iba operácie len z množiny  $G$ . Pre každý úplný súbor logických členov platí, že musí realizovať všetky operácie z niektorej funkčne úplnej množiny. Dá sa dokázať, že množina Booleových operácií je funkčne úplná. Existujú viaceré množiny operácií, ktoré sú funkčne úplné. Ďalej sú uvedené všetky takzvané minimálne úplné systémy, zostavené z jednej, dvoch, alebo troch operácií, pre ktoré platí, že z nich nie je možné bez porušenia ich funkčnej úplnosti odstrániť ani jednu operáciu.

- $\{\uparrow\}$
- $\{\downarrow\}$
- $\{., \bar{\phantom{x}}\}$
- $\{+, \bar{\phantom{x}}\}$
- $\{\rightarrow, \nrightarrow\}$
- $\{\rightarrow, \bar{\phantom{x}}\}$
- $\{\rightarrow, \oplus\}$
- $\{\nrightarrow, \equiv\}$
- $\{\nrightarrow, \bar{\phantom{x}}\}$
- $\{., \oplus, \equiv\}$
- $\{+, \oplus, \equiv\}$

Ak operácie môžu byť aj nulárne, t.j.  $0, I$ , potom existujú ešte nasledujúce minimálne úplné systémy:

- $\{\rightarrow, 0\}$
- $\{\nrightarrow, I\}$
- $\{\oplus, ., I\}$
- $\{\equiv, ., 0\}$
- $\{\equiv, +, 0\}$
- $\{\oplus, +, I\}$

### 2.1.3.1 Logické obvody

Logické obvody sa skladajú z logických členov.

Logický člen je elementárny číslicový systém, ktorý realizuje niektorú booleovskú funkciu nad vstupnými premennými a jej výsledok poskytuje na svojom výstupe. Na označovanie logických členov sa používajú schematické značky. Najznámejšie sú schematické značky vychádzajú z normy ČSN a z americkej normy MIL-STD-806B.

Logické obvody delíme na obvody kombinačné a obvody sekvenčné. Tieto dve skupiny obvodov sa opierajú o booleovu algebru. Obyčajne predpokladáme, že logický systém je definovaný na technickom zariadení a že jednotlivé premenné systému zodpovedajú určitým spojitým meniacim fyzikálnym veličinám, najčastejšie veličinám elektromagnetického poľa. Logické hodnoty sa vyjadrujú pomocou dvoch disjunktných intervalov hodnôt príslušnej fyzikálnej veličiny, napr. elektrického napätia, prúdu alebo náboja, resp. magnetického toku a podobne. ľubovoľnú  $n$ -ticu hodnôt vstupných premenných  $x_1, \dots, x_n$  nazývame vstupným vektorom a ľubovoľnú  $m$ -ticu hodnôt výstupných premenných  $y_1, \dots, y_m$  nazývame výstupným vektorom.

Pri kombinačných logických obvodoch výstupný vektor závisí od okamžitého vstupného vektora. Predchádzajúce kombinácie nemajú žiadny vplyv na výstupný vektor. Kombinačné logické obvody sú obvody, ktoré realizujú určitú logickú funkciu. Charakteristické je pre ne to, že výstup obvodu je jednoznačne daný jeho vstupom. Signál sa šíri postupne od vstupov k výstupu (neexistujú spätné väzby). Kombinačné obvody sú jednoduchšie ako sekvenčné obvody.

Pri sekvenčných logických obvodoch výstupný vektor závisí nielen od okamžitej kombinácie vstupov, ale taktiež od postupnosti vstupov v predchádzajúcom čase. Takto sa môže správať systém iba ak je schopný „pamätať si“, čo sa odohralo na jeho vstupoch v minulosti a podľa toho reagovať na súčasný vstupný vektor.

Kombinačné aj sekvenčné logické obvody môžu pracovať buď asynchrónne alebo synchronne. Pri asynchrónnych logických obvodoch vzniká zmena stavu a výstupného vektora okamžite po zmene stavu vstupov. Pri synchronných logických obvodoch vzniká zmena stavu a výstupného vektora až po prijatí synchronizačného (hodinového) signálu, kedy sa vyhodnocujú vstupné a stavové veličiny.

## 2.2 Softvér na podporu modelovania logických funkcií

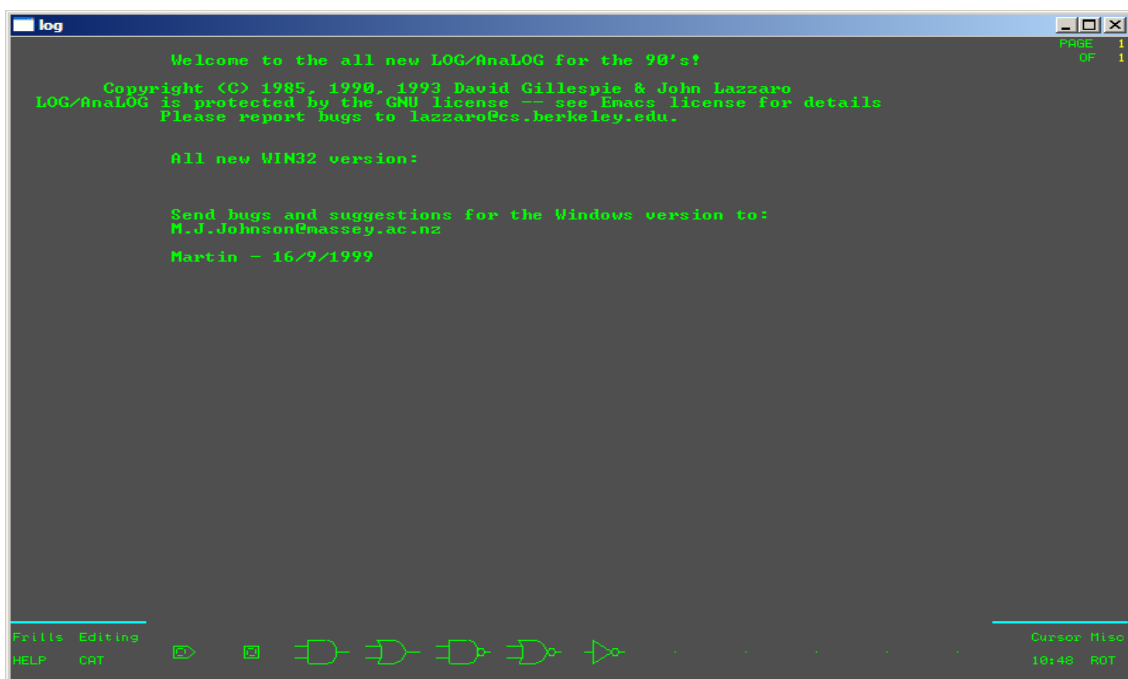
Existuje veľké množstvo komerčných a voľne dostupných softvérových riešení, ktoré sa zaoberajú problematikou modelovania logických funkcií a obvodov. V tejto kapitole opíšeme niektoré vybrané programy.

### 2.2.1 Log

V tejto aplikácii [4] sa dajú zostavovať jednoduché logické obvody s použitím základných logických členov ako AND, NAND, XOR, OR a iné. Táto aplikácia umožňuje, aby si používateľ sám navrhol schému a umiestnil logické členy ako uzná za vhodné. Logické členy sa dajú otáčať. LOG podporuje taktiež zvýraznenie vodičov v závislosti od logickej hodnoty, ktorá vodičom tečie. Nechýbajú mu ani základné editovanie funkcie ako mazanie, presúvanie a kopírovanie. V LOGu je možné svoju prácu uložiť alebo otvoriť už uloženú prácu.

Za najväčšiu výhodu považujem širokú škálu logických členov, ktoré LOG podporuje. LOG podporuje analógovú a taktiež digitálnu simuláciu.

Jediná vec voči ktorej by mohli byť výhrady, je azda trochu ťažkopádnejšie ovládanie a užívateľské rozhranie.



Obrázok č.10: LOG [4]

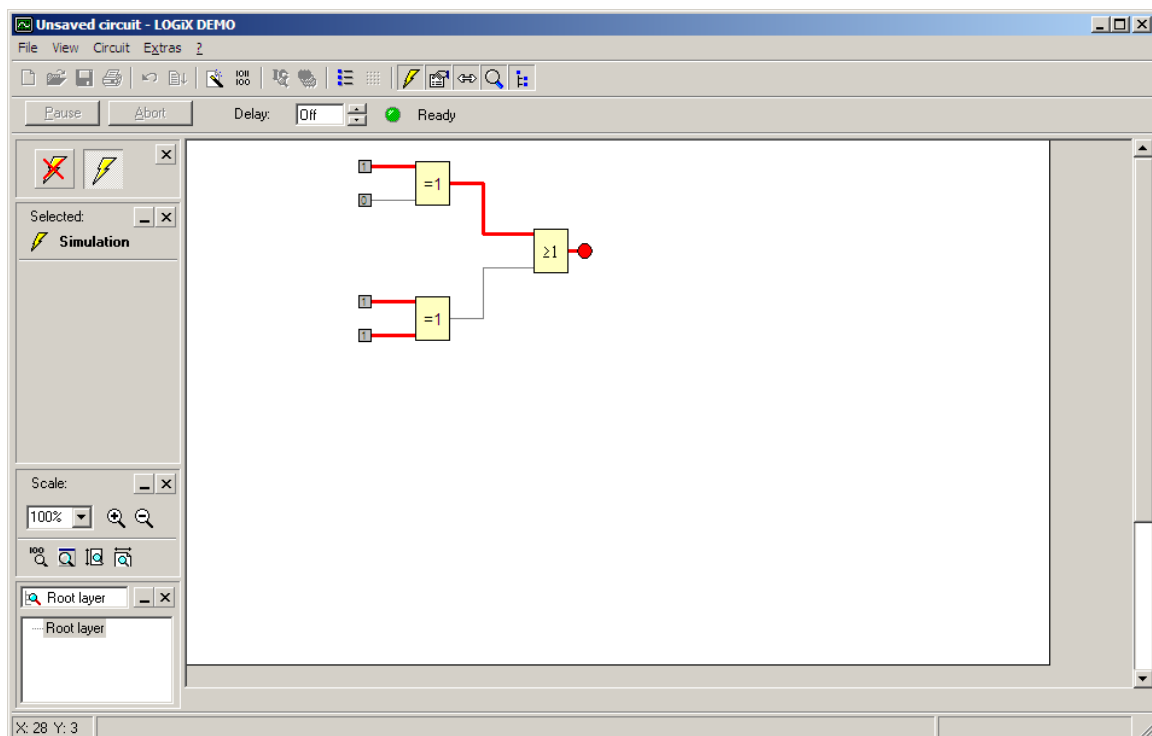


## 2.2.2 LOGiX

LOGiX [4] [9] je program pre tvorbu a simuláciu logických obvodov. Podobne ako LOG, LOGiX umožňuje umiestnenie logických členov na obrazovku a ich postupné spájanie vodičmi. O správnosti fungovania navrhnutého logického obvodu sa používateľ presvedčí pomocou simulácie, pri ktorej sú zvýraznené vodiče nesúce logickú 1.

LOGiX umožňuje generovanie logických obvodov na základe pravdivostnej tabuľky alebo booleovského výrazu. LOGiX umožňuje vytváranie vlastných elementov. Tieto elementy obsahujú obvod, ktorý používateľ navrhne a potom k nemu pristupuje ako k logickému členu. LOGiX tiež ponúka základné editovanie vlastností. Je možné svoju prácu uložiť a neskôr opäť otvoriť. Navrhnutý logický obvod je možné vytlačiť na papier.

Nevýhodou tohto programu je, že ak používateľ chce s ním pracovať viac ako 30 minút, musí si zakúpiť licenciu. LOGiX dáva k dispozícii menej logických členov ako vyššie spomenutý LOG.



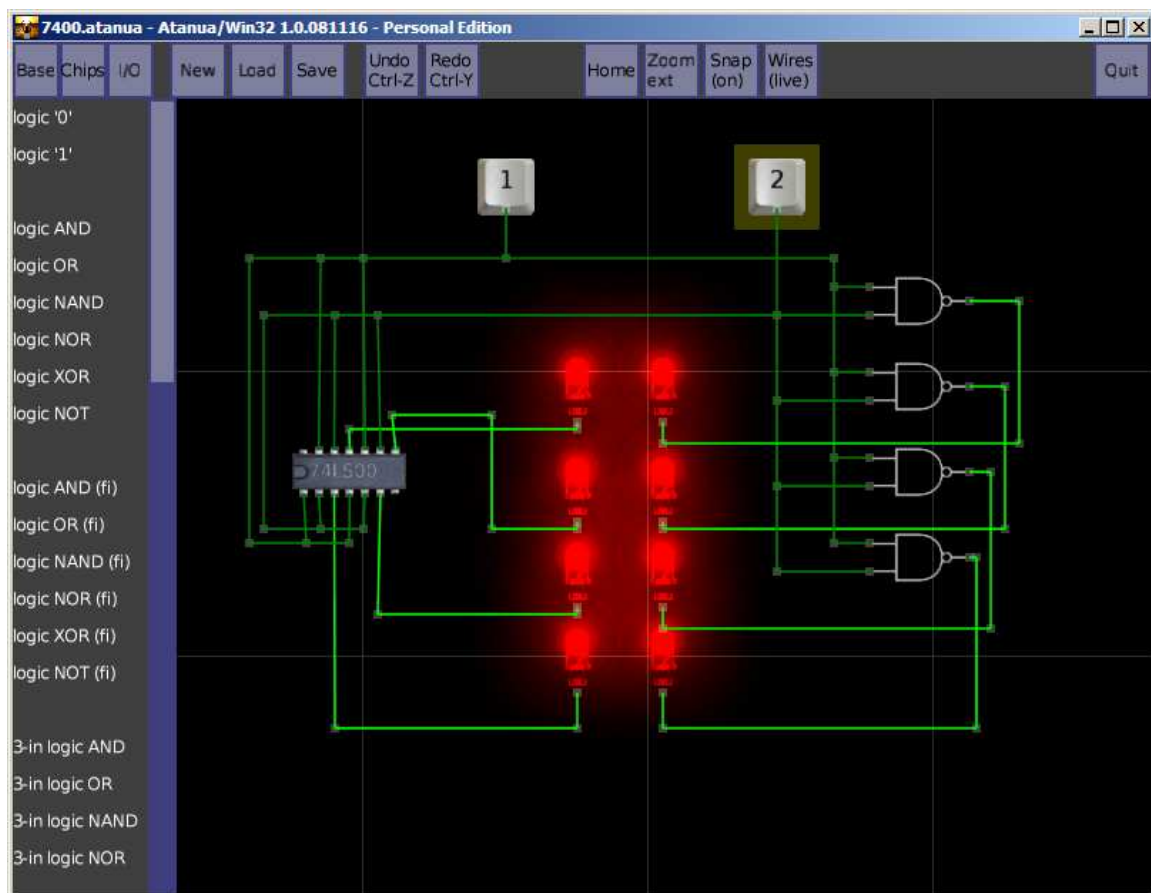
Obrázok č.11: LOGiX [9]

### 2.2.3 ATANUA

Atanua [4] [10] je logický simulátor navrhnutý na vyučovanie základov booleovskej logiky a elektroniky. Poskytuje príjemné používateľské rozhranie a jednoduché ovládanie. Umožňuje uložiť vytvorený obvod a opätovne ho otvoriť.

Používateľovi poskytuje základné logické obvody, ale taktiež integrované obvody (napr. IO 7400 obsahujúci NAND členy), s ktorými sa dá stretnúť v reálnom svete. Simulácia beží neustále a preto používateľ už pri návrhu vidí ako sa obvod správa. Atanua poskytuje množstvo vstupných a výstupných elementov. Z výstupných sú to hlavne rôznofarebné LED diódy a 7 segmentový displej. Ako vstupy sa dajú nadefinovať rôzne tlačidlá na klávesnici. Napríklad tlačidlo 1. Pri stlačení vygeneruje logickú 1 a pri pustení logickú 0.

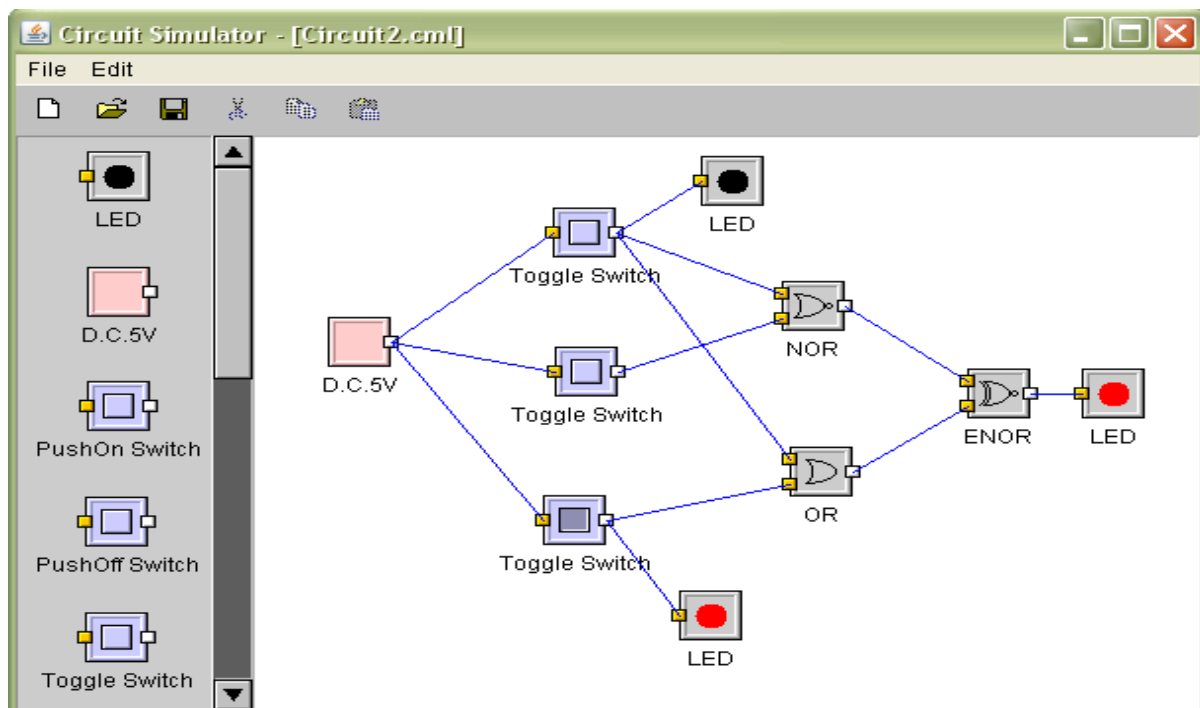
Zaujímavá je aj vlastnosť nazvaná Anti-cheating tool, ktorá je určená pre učiteľov na skontrolovanie domácej úlohy študentov. Atanua je k dispozícii pre platformy Windows, MAC OS a Linux. Pre nekomerčné využitie je program zadarmo.



Obrázok č.12: Atanua [10]

## 2.2.4 SIMCIR

Simcir 1.2.1 [5] [8] umožňuje používateľovi zapájať ponúknuté súčiastky a na ich vstup vysielat' kombinácie signálov. Takto si používateľ môže overiť priechodnosť jednotlivých logických obvodov. Na ľavej strane okna aplikácie sa nachádza konečný súbor súčiastok, ktoré si používateľ môže ľubovoľne zapájať do logických štruktúr. V ponuke sa nachádza LED, zdroj napätia, tri druhy vypínačov a sedem druhov logických členov. Všetky logické členy s výnimkou invertora majú dva vstupné póly a jeden výstupný. Dané komponenty je možné pohybom myši preniesť na pracovnú plochu a tam ich ľubovoľne zapájať. Zmeny v zapojení alebo v signáloch sa prejaví hneď ako sa uskutočnia.



Obrázok č.13: Simcir 1.2.1 [8]

## 2.2.5 Porovnanie analyzovaného softvéru

V tejto časti dokumentu sa nachádza porovnanie výhod a nevýhod jednotlivých softvérových produktov, ktoré boli analyzované.

Názov produktu	Výhody	Nevýhody
<b>Log</b>	široký výber logických členov možnosť editovania modelu možnosť zvýraznenia logických hodnôt vo vodičoch	ťažkopádne ovládanie neprívetivé grafické rozhranie
<b>LOGiX</b>	možnosť vytvárania vlastných elementov možnosť generovania logických obvodov na základe pravidlostnej tabuľky možnosť editovania modelu	menší výber logických členov
<b>ATANUA</b>	prehľadné grafické rozhranie množstvo vstupných a výstupných členov anti-cheating tool	
<b>SIMCIR</b>	prehľadné grafické rozhranie	malý počet logických členov

**Tabuľka č. 11:** Porovnanie dostupných softvérových riešení

## 3 Špecifikácia riešenia

Táto kapitola obsahuje analýzu a špecifikáciu zadania, z ktorej vychádzajú požiadavky na výsledný softvér. Táto časť je dôležitá na správne navrhnutie riešenia.

### 3.1 Analýza a špecifikácia zadania

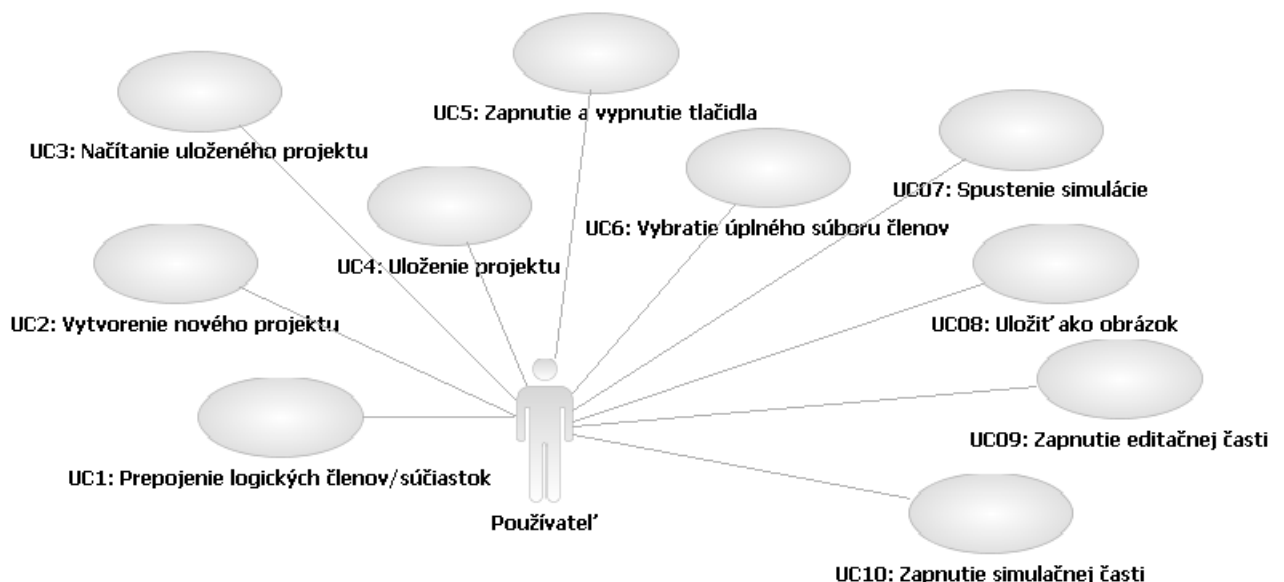
Navrhovaný programový systém by mal spĺňať zadanie v plnom rozsahu. Výsledný produkt má mať formu programu, ktorý je ľahko prevádzkovateľný na školských osobných počítačoch. Keďže tento programový systém bude používaný v pedagogickom procese, finálny produkt musí byť prívetivý pre používateľa, jednoducho a intuitívne ovládateľný a názorný. Ďalej musí umožniť modelovať a simulovať kombinačné logické obvody s normálnou štruktúrou, ktoré majú najviac štyri vstupy a štyri výstupy. Musí umožniť uloženie navrhnutého logického obvodu, následné otvorenie a upravenie. Logický obvod bude reprezentovaný graficky, jednotlivé členy budú napevno pridelené na plochu. Spojenie členov bude možné pomocou myši. Bude podporovaná veľká škála logických členov s rôznym počtom vstupov, ktoré budú uložené v knižnici logických členov do úplných súborov členov. Knižnica bude obsahovať všetky úplne súbory členov. Na zobrazenie vstupnej a výstupnej hodnoty budú použité žiarovky, ktoré budú farebne rozlišovať logickú nulu a jednotku. Jednotlivé časti modelovaného logického obvodu budú prepájané spojovníkmi, ktoré budú zobrazovať prechodovú hodnotu. Hodnota bude farebne rozlišovať logickú nulu a jednotku. Výsledný produkt sa bude skladať z dvoch častí, editačnej a simulačnej. V editačnej časti bude možné navrhnuť a upraviť logický obvod. V simulačnej časti bude možné odsimulovať jeho správanie. Výsledný produkt má dávať správne výstupy. Pomocou simulácie bude možné overiť funkciu logického kombinačného obvodu.

Pri samotnom vývoji softvéru bude kladená pozornosť najmä na použiteľnosť, príjemné užívateľské prostredie a modulárnosť, aby sa v prípade potreby dal ľahko rozšíriť o ďalšie časti. Podľa nás by určite bola zaujímavá funkcia *Drag & Drop*, ktorá by poskytovala jednoduché a intuitívne ovládanie celej aplikácie.

## 3.2 Funkcionálne požiadavky na softvér

Uvažujeme o jednom používateľovi, ktorý bude mať prístup k nasledujúcim funkciám programu. Diagram prípadov použitia, ktorý zobrazuje funkcie nášho riešenia je na obrázku číslo 14. Bude poskytovaná nasledovná funkcionalita systému:

- prepojenie logických členov alebo súčiastok
- vytvorenie nového projektu
- načítanie uloženého projektu
- uloženie projektu
- zapnutie a vypnutie tlačidla
- vybratie úplného súboru členov
- spustenie simulácie
- uložiť ako obrázok
- zapnutie editačnej časti
- zapnutie simulačnej časti



Obrázok č.14: Prípady použitia.

### 3.2.1 Popis prípadov použitia

Táto časť obsahuje popis jednotlivých prípadov použitia, ktoré sú zobrazené na obrázku 14.

**Prepojenie logických členov alebo súčiastok** – Táto funkcia umožní priamo interakciu používateľa s logickými členmi. Bude možné prepojiť jednotlivé súčiastky medzi sebou a tak vytvoriť logickú schému.

**Vytvorenie nového projektu** – Jedna z možností na ovládacom paneli (panel podporných funkcií) bude ponúkať možnosť vytvoriť nový projekt, ktorý bude obsahovať prázdnu pracovnú plochu.

**Načítanie uloženého projektu** – Jedna z možností na ovládacom paneli (panel podporných funkcií) bude ponúkať možnosť otvoriť uložený projekt, ktorý bude obsahovať schému logického obvodu už predtým vytvorenú.

**Uloženie projektu** – Jedna z možností na ovládacom paneli (panel podporných funkcií) bude ponúkať možnosť uložiť si vytvorenú schému logického zapojenia, a tak mať túto schému k dispozícii neskôr.

**Zapnutie a vypnutie tlačidla** – Táto funkcia umožní meniť používateľovi logické hodnoty (log. 0 a log. 1) na vstupe. Stlačením tlačidla sa zmení vstupná hodnota.

**Vybratie úplného súboru členov** – Používateľ si bude môcť vybrať úplný súbor členov zo zoznamu. Každý súbor bude obsahovať obmedzený počet a typ preddefinovaných členov, s ktorými bude musieť používateľ pracovať.

**Spustenie simulácie** – Jedna z možností na ovládacom paneli (panel podporných funkcií) bude ponúkať možnosť spustenia simulácie zapojeného logického obvodu. To umožní zobrazenie výsledných logických hodnôt na výstupe.

**Uložiť ako obrázok** – Jedna z možností na ovládacom paneli (panel podporných funkcií) bude ponúkať možnosť uložiť zapojenú schému ako obrázok vo formáte .jpg, .gif alebo .bmp.

**Zapnutie editačnej časti** – Používateľ bude môcť prepínať medzi dvomi stavmi aplikácie. V tomto stave bude môcť editovať a zapájať logickú schému.

**Zapnutie simulačnej časti** – Používateľ bude môcť prepínať medzi dvomi stavmi aplikácie. V tomto stave bude môcť simulovať zapojenú logickú schému.

### 3.3 Nefunkcionálne požiadavky na softvér

Medzi ďalšie požiadavky, ktoré nemajú funkcionálny charakter by sme mohli zaradiť:

- Používateľsky prívetivé a prehľadné grafické rozhranie.
- Názorné a prehľadné zobrazenie výstupu a celého programu (farby a rozlíšenie).
- Farebné rozlíšenie spojov podľa vedenej logickej hodnoty.
- Jednoduché a intuitívne ovládanie.
- Prenositeľnosť a multiplatformový produkt.
- Nie je nutná inštalácia programu.
- Čo najmenšia náročnosť na počítačový systém (procesor a pamäť).
- Modulárnosť programu.

### 3.4 Požiadavky na grafické rozhranie

Je potrebné navrhnuť rozhranie, ktoré bude jednoduché, prehľadné a prívetivé. Preto je potrebné ho rozdeliť na rovnocenné časti. Grafické používateľské rozhranie sa bude skladať z nasledovných častí:

- pracovná plocha
- knižnica logických členov
- riadiaca lišta (panel podporných funkcií).



**Pracovná plocha** – Bude slúžiť na prepájanie a modelovanie logickej schémy.

**Knižnica logických členov** – Bude obsahovať úplný súbor logických členov, ktoré bude možné používať pri modelovaní schémy. Knižnica bude obsahovať úplné súbory členov.

**Riadiaca lišta** – Bude ponúkať používateľovi jednoduché ovládanie aplikácie pomocou podporných funkcií.

## 4 Návrh riešenia

Pri návrhu výsledného produktu budeme vychádzať zo zadania a našej špecifikácie. Program budeme vyvíjať v jazyku C# (platforma Microsoft .Net Framework) v prostredí Microsoft Visual Studio. Keďže výsledný produkt bude použitý vo výučbe na školských počítačoch, kde je nainštalovaný operačný systém Microsoft Windows, tak sme sa rozhodli pre toto prostredie a programovací jazyk. Taktiež by sme sa radi zdokonalili vo vývoji aplikácií v prostredí .Net. Pri samotnom návrhu riešenia problému sa bude vychádzať z vlastností objektovo orientovaného prístupu. Projekt rozložíme na niekoľko menších častí a ich riešenie si rozdelíme v rámci tímu. Budeme sa usilovať o čo najpresnejšie splnenie požiadaviek objednávateľa.

### 4.1 Hrubý návrh riešenia

System bude zložený z viacerých podsystemov, ktoré sú medzi sebou navzájom závislé v určitých spoločných bodoch. Toto zaručí, že jednotlivé podsystemy, alebo moduly sa budú môcť vytvárať súčasne, pričom bude dopredu známe rozhranie pomocou ktorého si budú medzi sebou vymieňať dáta. Vysoko abstraktný model architektúry celého systému je znázornený na obrázku číslo 15.



Obrázok č.15: Moduly systému.

#### 4.1.1 Modul logických súčiastok

Tento modul bude obsahovať abstraktnú reprezentáciu jednotlivých súčiastok logických obvodov. Výsledný produkt bude podporovať použitie hlavne nasledovných logických členov:

- AND – vynásobenie vstupných hodnôt.

- NAND – znegovanie vynásobených vstupných hodnôt.
- OR – sčítanie vstupných hodnôt.
- NOR – znegovanie sčítaných vstupných hodnôt.
- XOR – logická neekvivalencia.
- NOT – negovanie vstupnej hodnoty.

Ďalšie objekty, s ktorými bude možné pracovať sú:

- Žiarovka – indikácia 1 alebo 0. Bude sa používať na indikovanie vstupu a aj výstupu.
- Spoj – prechod medzi jednotlivými súčiastkami.
- A iné (pozn.: bude určené po dohode s vedúcim).

Členy budú zaradené do úplných súborov členov. Každý takýto súbor bude obsahovať obmedzený počet logických členov a len určitý typ členov. Jednotlivé členy budú napevno pridelené na plochu. Definície logických súčiastok respektíve členov budú udávať ich funkcionalitu. Všetky logické členy budú dediť od triedy *Gate*, ktorá bude vyzeráť približne takto:

```
public class Gate
```

```
{
```

```
    int id = 0;
```

```
    int inputs;
```

```
    public boolean input1 = false; // logicka hodnota na prvom vstupe
```

```
    public boolean input2 = false; // logicka hodnota na druhom vstupe
```

```
    public boolean input3 = false; // logicka hodnota na tretom vstupe
```

```
    public boolean input4 = false; // logicka hodnota na stvrtem vstupe
```

```
    public boolean output = false; // logicka hodnota na vystupe
```

```
    public int input1ID = 0; // id prveho vstupu
```

```
    public int input2ID = 0; // id druhého vstupu
```

```
    public int input3ID = 0; // id tretieho vstupu
```

```
    public int input4ID = 0; // id stvrteho vstupu
```

```

    public int outputID = 0; // id vystupu
}

```

Každý člen bude mať maximálne štyri vstupy. Koľko ich naozaj bude, sa určí pri vytváraní objektu triedy reprezentujúcej logický člen nastavením premennej *inputs*. Bude tu aj možnosť upravovať počet vstupov aj vtedy, keď člen už bude vytvorený. Bude to prebiehať zmenou hodnoty premennej *inputs*.

Každý člen bude mať svoj identifikátor *id*. Bude to celé číslo väčšie ako nula. Keď výstup pripojíme na vstup nejakého člena, tak do premennej *outputID* sa zapíše *id* člena, do ktorého vstupu sme sa pripojili. Podobne to bude fungovať aj so vstupmi.

Každý člen bude mať metódu *processInputs*, ktorá podľa hodnôt na vstupe priradí hodnotu na výstupe. Táto metóda sa zavolá pre každý člen logického obvodu vždy, keď sa udeje nejaká zmena. Príkladom zmeny môže byť odstránenie člena, prepojenie členov, zmena vstupnej premennej obvodu. Metóda *processInputs* pre trojvstupový člen AND vyzerá nasledovne:

```

public void processInputs()
{
    if (input1 && input2 && input3) output = true;
    else output = false;
}

```

#### 4.1.2 Modul grafického používateľského rozhrania

Tento modul bude obsahovať reprezentáciu grafického rozhrania ako jediný prístupový bod používateľa k aplikácii. Modul bude obsahovať tri časti:

- pracovná plocha,
- knižnica logických členov,
- riadiaca lišta (panel podporných funkcií).

Pracovná plocha bude slúžiť na prepájanie a modelovanie logickej schémy. Knižnica logických členov bude obsahovať úplný súbor logických členov, ktoré bude možné používať

pri modelovaní schémy. Riadiaca lišta bude ponúkať používateľovi jednoduché ovládanie aplikácie. Jej súčasťou bude menu, ktoré bude umožňovať uloženie a načítanie schémy, otvorenie novej pracovnej plochy a iné podporné funkcie.

Jednotlivé členy budú napevno pridelené na plochu na základe výberu úplného logického súboru členov. Počet a typ členov bude obmedzený. Keď sa pridá nový logický člen na plochu, tak sa vytvorí objekt triedy, ktorá ho opisuje, ktorý sa pridá do poľa *gates*. Je to objekt triedy *ArrayList*, ktorá umožňuje robiť s užitočné operácie so svojimi členmi. Deklarácia a inicializácia *ArrayListu gates* vyzerá takto:

```
public static ArrayList<Gate> gates;
```

```
...
```

```
gates = new ArrayList();
```

Ak sa zmení hodnota výstupu nejakého člena, tak sa tak sa zavolajú metódy, ktoré ju prenesú na ostatné pripojené členy a okrem toho overia, či nie je potrebné zmeniť vizuálnu reprezentáciu členov (obrázky na pracovnej ploche).

### 4.1.3 Modul podporných funkcií

Modul podporných funkcií bude obsahovať ďalšiu funkcionality programu, ktorá priamo nesúvisí so zadaním, ako možnosť uloženia a načítania namodelovaného logického obvodu, ukladanie a načítavanie zmien v modely a podobne. Tieto funkcie bude možné ovládať pomocou riadiacej lišty, ktorá bude v hornej časti grafického používateľského rozhrania.

Základná funkcionality programu bude ovládaná nasledovnými úkonmi na pracovnej ploche:

- **Prepojenie logických členov alebo súčiastok** – prepojenie logických členov a súčiastok na ploche.
- **Zapnutie a vypnutie tlačidla** – nastavovanie logickej hodnoty na vstupe.
- **Vybratie úplného súboru členov** – vybratie počtu a typu členov na modelovanie.
- **Zapnutie editačnej časti** – editovanie a zapájanie schémy.
- **Zapnutie simulačnej časti** – simulovanie zapojenej schémy.

Riadiaca lišta bude obsahovať nasledovné voľby:

- **Súbor**
  - **Nový** – vytvorenie nového projektu.
  - **Otvoriť** – načítanie uloženého projektu.
  - **Uložiť** – uloženie projektu.
  - **Uložiť ako obrázok** – schému na pracovnej ploche uloží ako jpg súbor.
  - **Ukončiť** – vypnutie programu.
- **Knižnica**
  - **Zobrazenie úplného súboru členov** – Zobrazí úplný súbor členov.
  - **Vybratie úplného súboru členov** – Vybratie úplného súboru členov.
- **Spustenie simulácie** – Umožní začatie simulovania toku logických hodnôt.
- **O programe**
  - **Autori** – Informácie o autoroch programu.
  - **Program** – Informácie o programe.
  - **Pomoc** – Krátky návod ako používať program.

Ďalšie funkcie môžu byť pridané po dohode s pedagogickým vedúcim.

## 5 Prototyp

Táto kapitola popisuje vytvorený prototyp aplikácie. Delí sa na dve hlavné časti. Požiadavky na aplikáciu opisuje funkcie a vlastnosti, ktoré sú zahrnuté v prototypu. Časť dosiahnuté výsledky sa zaoberá výstupom z fázy tvorby prototypu. Opisuje možnosti aplikácie a všetky funkcie, ktoré boli v tejto fáze implementované.

### 5.1 Požiadavky na aplikáciu

Cieľom je navrhnuť a implementovať aplikáciu, ktorá by spĺňala požiadavky dohodnuté s vedúcim projektu. Požiadavky boli do značnej miery všeobecne formulované, preto sme často museli hľadať zhodu na spôsobe ich realizácie na spoločných stretnutiach celého tímu.

Aplikácia má umožniť zostaviť a ručne overiť štruktúru logického obvodu s normálnou štruktúrou. Používateľ mal k dispozícii konečný počet členov zo zvoleného minimálneho úplného súboru logických členov. Na výber majú byť všetky možné minimálne úplné súbory logických členov. Spomenuté sú v kapitole 2.1.3. Medzi nimi má byť možné ľubovoľne prepínať, vždy je však možné pracovať len s jedným. Je potrebné ponúknuť taký počet jednotlivých členov, ktorý sa ukáže ako optimálny. Cieľom tohto je prinútiť používateľa pracovať len s tým, čo má k dispozícii. To môže znamenať, že počet členov a ich vstupov môže ovplyvniť výslednú štruktúru obvodu. Je na nás, či členy budú vo verifikačnom paneli umiestnené napevno alebo bude možné s nimi hýbať. Čo sa týka počtu vstupov a výstupov obvodu, požadované je, aby to bol tiež konečný počet, odporúčajú sa štyri. Pri každom člene má byť napísané, akú logickú hodnotu má na svojom výstupe.

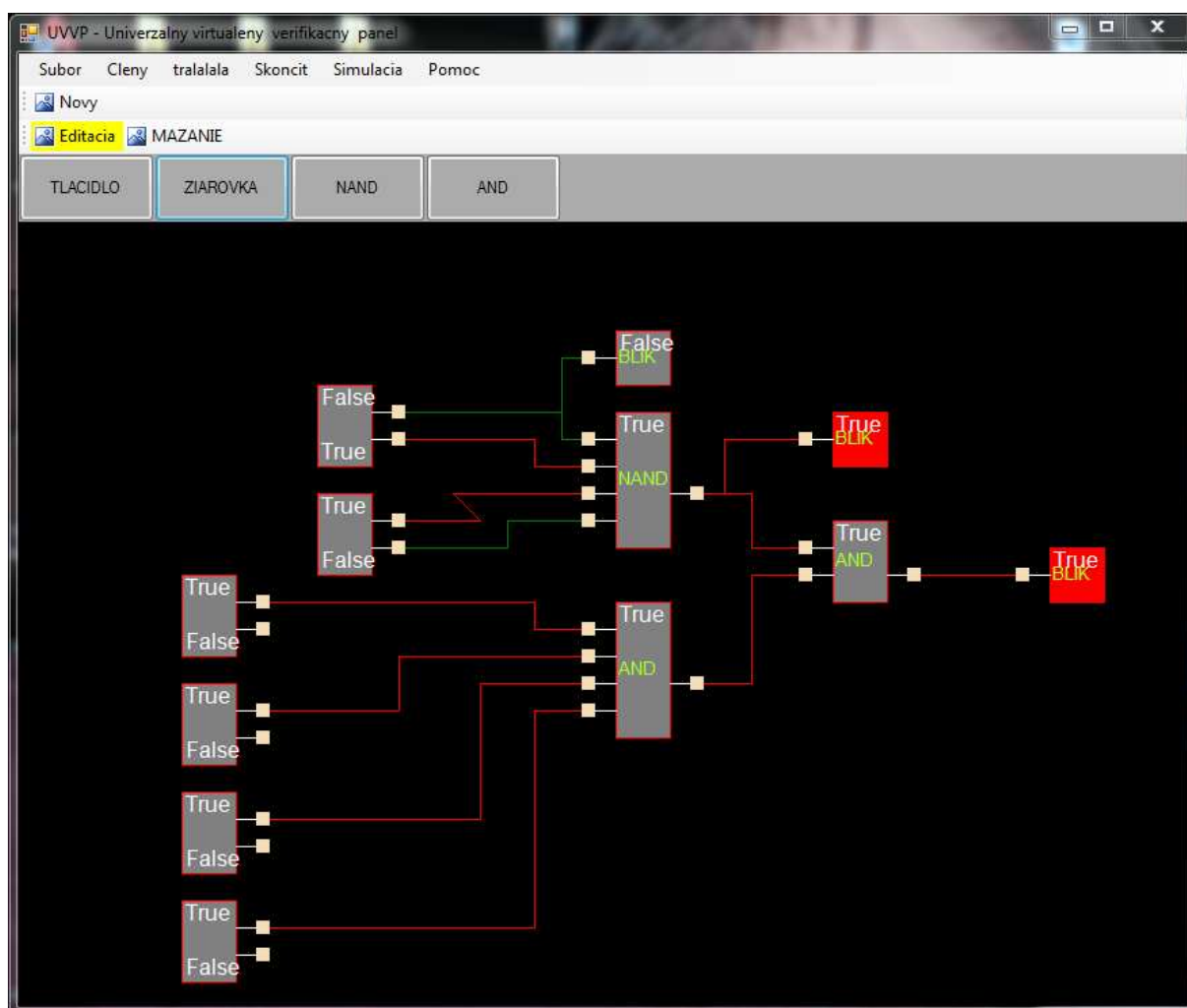
Ďalej je potrebné do štruktúry pridať žiarovky, ktoré by signalizovali výstupnú hodnotu zostaveného logického obvodu a tlačidlá, ktoré by do obvodu privádzali požadované logické hodnoty. Tlačidlo má mať dva výstupy, na jednom má byť logická 1, na druhom 0. Po stlačení sa majú oba výstupy prehodiť.

Bolo treba vymyslieť vhodný spôsob, ktorým by sa členy obvodu prepájali. Pre uľahčenie hľadania prípadných chýb na zostavenom logickom obvode malo byť umožnené sledovať

hodnoty aj na čiarach prepájajúcich členy. To sa dá dosiahnuť napríklad zmenou farieb, ktorými sa vykresľujú.

## 5.2 Dosiahnuté výsledky

Prototyp aplikácie vychádza z požiadaviek, ktoré boli naňho od začiatku kladené. Ukážka okna aplikácie je na obrázku 16.



Obrázok č.16: Ukážka prototypu.

Po spustení aplikácie sa zobrazí dialógové okno, ktoré vyzve používateľa, aby si vybral súbor logických členov. Na výber budú všetky možné. Údaje o nich sú uložené v xml súbore, z ktorého sa načítajú. Logický súbor, s ktorým pracujeme, je možné kedykoľvek zmeniť



kliknutím na tlačidlo Nový. Vtedy sa do ponuky, z ktorej môžeme vyberať logické členy nahrajú členy nového súboru a obsah pracovnej plochy, na ktorej zapájame logický obvod, sa vyčistí.

Prototyp zatiaľ dovoľuje pracovať s neobmedzeným počtom členov, vstupov a výstupov. Nie je to chyba, ale v ďalších fázach vývoja aplikácie sa všetko obmedzí na konečný počet. Budú k dispozícii členy úplného súboru, ktoré budú umiestnené priamo na paneli a budú pozostávať z konečného počtu vstupov.

Prototyp umožňuje prácu v dvoch režimoch: editácie a mazania. V režime editácie je možné pridávať, premiestňovať a prepájať členy a meniť hodnoty na tlačidlách. Režim mazania umožňuje vymazať ľubovoľnú časť logického obvodu, ktorú označíme kliknutím. Možné je teda vymazávať nielen logické členy, žiarovky, virtuálne tlačidlá, ale aj čiary.

Výber člena prebieha kliknutím na tlačidlo z ponuky. Pri niektorých členoch je možnosť zvoliť počet vstupov. Maximálne môžeme nastaviť osem vstupov. Po potvrdení sa člen zobrazí v ľavom hornom okraji pracovnej plochy, odkiaľ je možné ho presunúť na ľubovoľné miesto plochy. Hodnota výstupu člena je napísaná v jeho hornej časti.

Okrem členov, ktoré tvoria logický súbor, sú v ponuke aj členy Tlačidlo a Žiarovka. Tlačidlo privádza do obvodu vstupné hodnoty. Ako bolo požadované, má dva výstupy, na každom je iná logická hodnota. Po kliknutí naňho sa vymenia. Žiarovka má len jeden vstup, ak je na ňu prevedená logická 1, rozsvieti sa načerveno, inak bude signalizovať logickú 0. Neskôr budú členy Tlačidlo a Žiarovka pevne umiestnené na paneli.

Členy sa prepájajú čiarami, ktoré sa kreslia režime editácie. Na prepojenie dvoch členov je potrebné kliknúť myšou na výstup jedného člena, preniesť ju na vstup druhého člena a pustiť. Po kliknutí sa pri kreslení čiary zobrazí biela mriežka, ktorá uľahčí kreslenie. Mriežka sa zobrazí aj pri presúvaní členov logického obvodu. Ak cez čiary prechádza hodnota logická 0, je znázornená zelenou farbou. Ak 1, tak je červená.

Vzhľadom k tomu, že sa jedná o prototyp aplikácie, mnohé veci na nej budú prerobené, pridané alebo vymazané, za účelom zvýšenia jej efektívnosti alebo uľahčenia ovládania.

Snahou bolo, aby bola aplikácia jednoduchá, názorná a intuitívne a ľahko ovládateľná. Je to dôležité zvlášť preto, lebo má byť použiteľná v pedagogickom procese. Zostavený obvod je možné exportovať ako výstup vo forme obrázku.

## 6 Použitá literatúra

- [1] Doc. Ing. N. Frištacký, CSc.; Doc. Ing. J. Kolenička, CSc.; Doc. Ing. M. Kolesár, CSc.: *Logické systémy - Kombinačné obvody*. júl 1986. Bratislava. Editačné stredisko SVŠT. [cit. 2009-10-24].
- [2] Doc. RNDr. Jana Galanová, PhD.; RNDr. Peter Kaprálik, PhD.; Mgr. Marcel Polakovič, PhD.: *Logické systémy*. [cit. 2009-10-31].
- [3] Frištacký N., Kolesár M., Kolenička J., Hlavatý J: *Logické systémy*, 1986. Alfa – Vydavateľstvo technickej a ekonomickej literatúry Bratislava. [cit. 2009-10-24].
- [4] Pivarček Ján: Virtuálny verifikačný panel s členmi XOR a OR, Bakalárska práca, FIIT STU 2009.
- [5] Sivák Marek: Hradlá a hradlové štruktúry, podpora výuky – aplikácia pod OS Windows, Bakalárska práca, FIIT STU 2009.
- [6] Palaj Tomáš: Virtuálny verifikačný panel s členmi AND-NOR a NAND, Bakalárska práca, FIIT STU 2009.
- [7] Pilarčík Viliam: Virtuálny verifikačný panel s členmi NOR, Bakalárska práca, FIIT STU 2009.
- [8] Arase, Kazuhiko: Simcir the circuit simulator. [Online]. 6. júla 2000. [cit. 2009-11-07]. Dostupný z WWW: <http://www.d-project.com/simcir/index.html>.
- [9] CommTec: LOGiX - Simulation of Logic Circuits. [Online]. 04. júl 2005. [cit. 2009-11-07]. Dostupný z WWW: <http://www.simtel.net/product.php%5Bid%5D90288%5Bcid%5D86%5BsiteID%5Dsimtel.net>.

- [10] Komppa: Atanua Logic Simulator. [Online]. [cit. 2009-11-07]. Dostupný z WWW: <http://sol.gfxile.net/atanua/>.
- [11] Knaian, Ara. Digital Simulator. [Online]. Massachusetts Institute of Technology, 1995. [cit. 2009-11-07]. Dostupný z WWW: <http://www.mit.edu/people/ara/ds.html>.
- [12] Tetzl, Andreas: Logic Simulator. [Online]. 22. novembra 2008. [cit. 2009-11-07]. Dostupný z WWW: [http://www.tetzl.de/java\\_logic\\_simulator.html](http://www.tetzl.de/java_logic_simulator.html).