Slovenská technická univerzita v Bratislave

FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Študijný program: Počítačové a komunikačné systémy a siete

Tím č. 8

Univerzálny virtuálny verifikačný panel logických obvodov

Projektová dokumentácia

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

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

Obsah

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.

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 navrhnúť 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 funkcionalitu logických obvodov. Dôraz je kladený na implementovanie úplného súboru členov, funkcionalitu 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 navrhnúť 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 – 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.

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 priraďuje 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 Λ B. Disjunkcia sa vyjadruje ako "A alebo B" a označuje sa A V B. Implikácia, má tvar "ak A, potom B" a označuje sa A => B. Ekvivalencia má tvar "A práve vtedy, keď, B" a značí sa A <=> 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 ¬A, prípadne $\overline{A}$.

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	$\overline{a+b} = \overline{a}.\overline{b}$	$\overline{a.b} = \overline{a} + \overline{b}$
Zákon absorbcie	a+a.b=a	a.(a+b)=a
Zákon absorbcie negácie	$a + \overline{a}.b = a + b$	$a.(\overline{a} + b) = a.b$
Zákon vylúčenia	$a + \overline{a} = 1$	$a.\overline{a} = 0$
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 $\overline{A} = A$	¬¬a=a	¬¬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

Vezmime si niektorú množinu operácií $G=\{ g_1, g_2 \ldots. g_k \}$, $k \geq 1$. Množinu G nazývame funkčou 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 funčne 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 úplnnosti odstrániť ani jednu operáciu.

- $\{ \uparrow \}$
- $\{ \downarrow \}$
- $\{ \cdot , \overline{\ } \}$
- $\{ + , \overline{\ } \}$
- $\{ \rightarrow , \nrightarrow \}$
- $\{ \rightarrow , \overline{\ } \}$
- $\{ \rightarrow , \oplus \}$
- $\{ \nrightarrow , \equiv \}$

- { ↛ , ¯ }
- { . , ⊕ , ≡ }
- { + , ⊕ , ≡ }

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

- { → , 0 }
- { ↛ , I }
- { ⊕ , . , I }
- { ≡ , . , 0 }
- { ≡ , + , 0 }
- { ⊕ , + , 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 spojito sa 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 x1, ..., xn nazývame vstupným vektorom a ľubovoľnú m-ticu hodnôt výstupných premenných y1, ..., ym 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 synchrónne. Pri asynchrónnych logických obvodoch vzniká zmena stavu a výstupného vektora okamžite po zmene stavu vstupov. Pri synchrónnych logických obvodoch vzniká zmena stavu a výstupného vektora až po prijatí synchronizačného (hodinového) signálu, kedy sa vyhodnocujú vstupné a stavové veličiny.

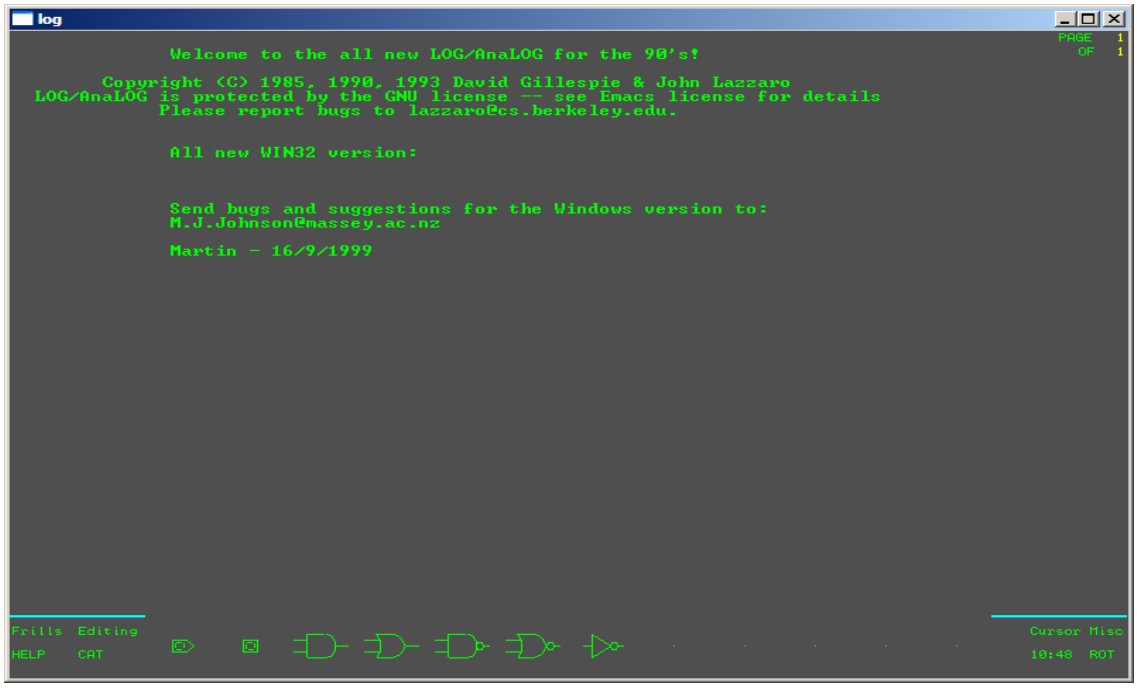
2.2 Softvér na podporu modelovania logických funkcií

2.2.1 Log

V tejto aplikácii 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.



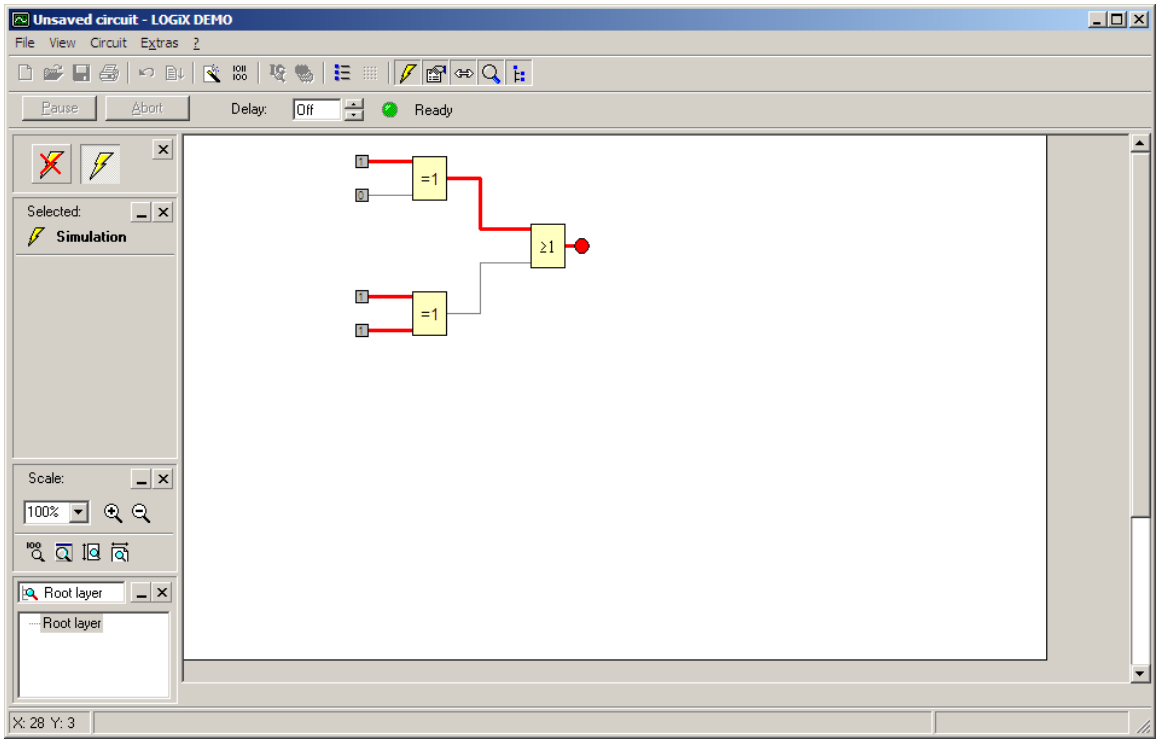
Obrázok č.1: LOG

2.2.2 LOGiX

LOGiX 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 požívateľ navrhne a potom k nemu pristupuje ako k logickému členu. LOGiX tiež ponúka základné editovanie vlastnosti. 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.



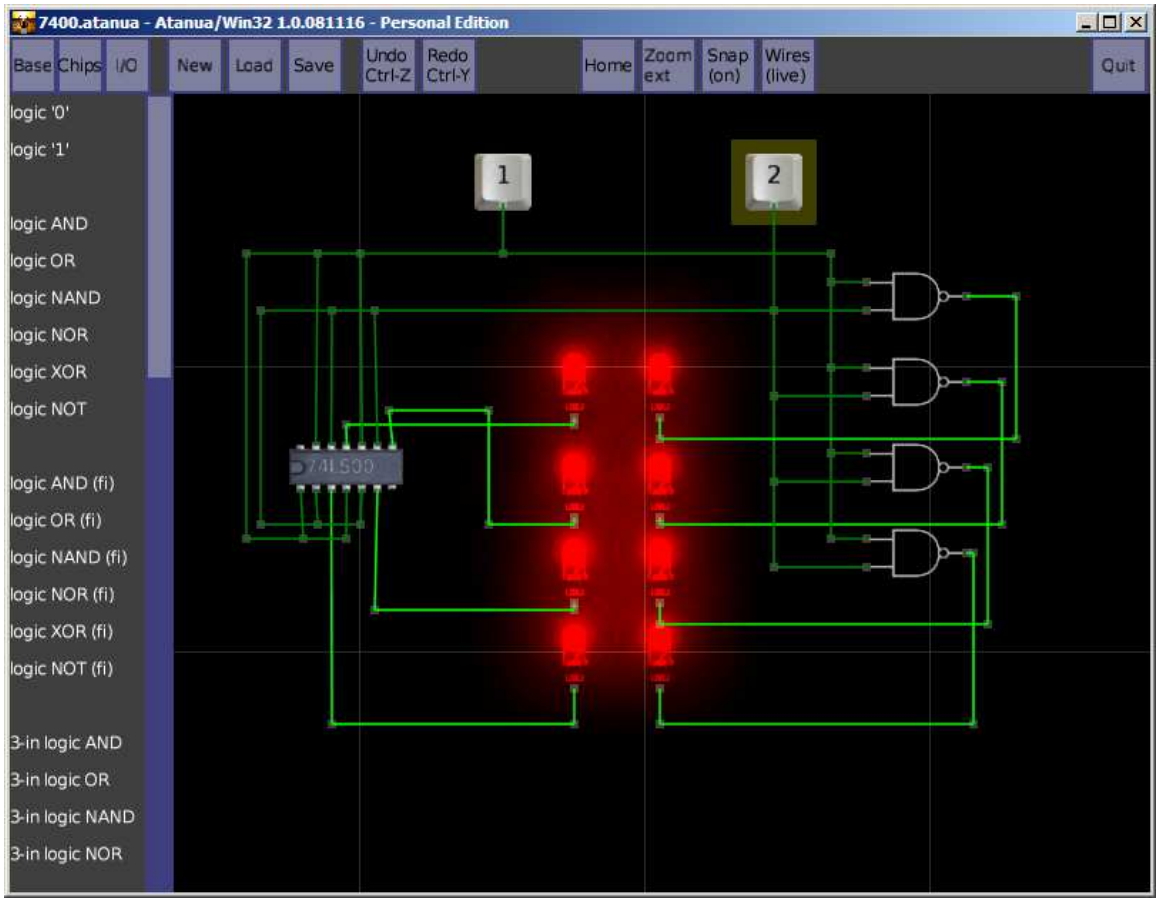
Obrázok č.2: LOGiX

2.2.3 ATANUA

Atanua 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.

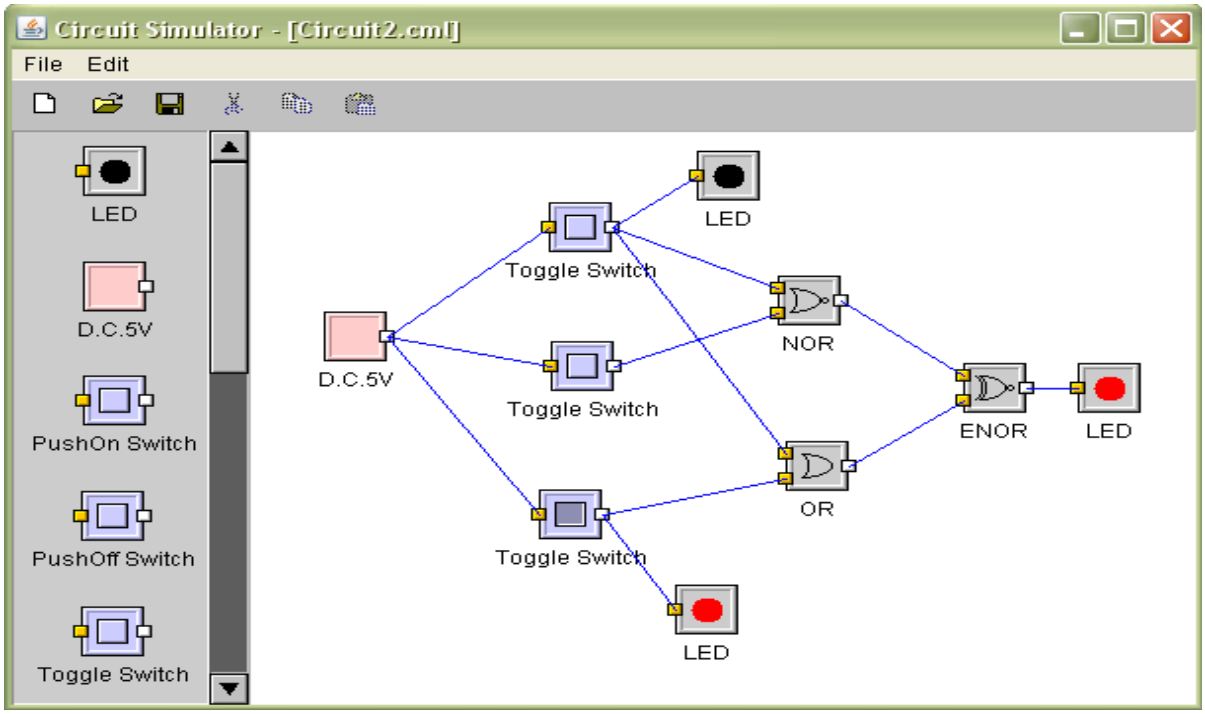
Zaujímavá je aj vlastnosť nazvaná Anti-cheating tool, ktorá je určená pre učiteľov na skontrolovanie domácej úlohy študentov. Atanua je k dispozícií pre platformy Windows, MAC OS a Linux. Pre nekomerčné využitie je program zadarmo.



Obrázok č.3: Atanua

2.2.4 SIMCIR

Simcir 1.2.1 umožňuje používateľovi zapájať ponúknuté súčiastky a na ich vstup vysielať 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 ľubovoľné 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 ľubovoľné zapájať. Zmeny v zapojení alebo v signáloch sa prejavia hneď ako sa uskutočnia.



Obrázok č.4: Simcir 1.2.1

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

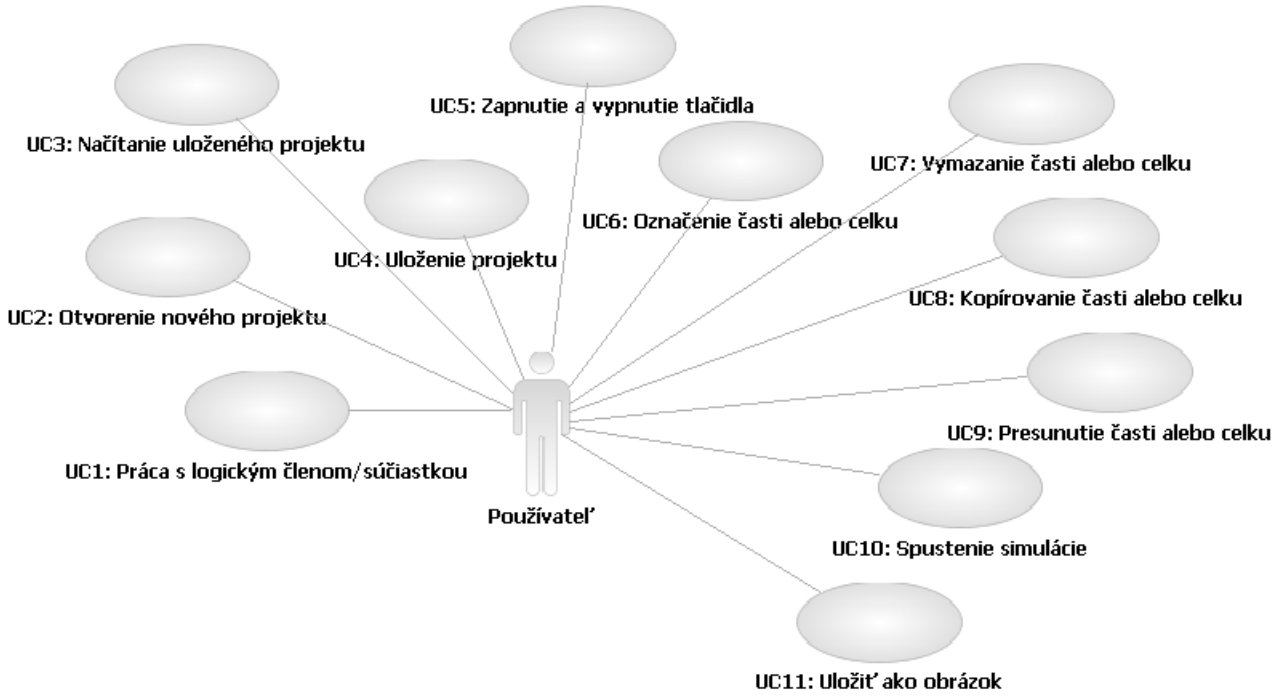
Navrhovaný programový systém by mal spĺňať zadanie v plnom rozsahu. Výsledný produkt má mať formu programu, ktorý je ľahko prevádzkovateľný na školských osobných počítačoch. Keďže tento programový systém bude používaný v pedagogickom procese, finálny produkt musí byť prívetivý pre používateľa, jednoducho 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. Taktiež bude možné vybrať časť alebo celý model a vymazať označenú časť. Logický obvod bude reprezentovaný graficky, ovládať sa bude myšou. Bude podporovaná veľká škála logických členov s rôznym počtom vstupov, ktoré budú uložené v knižnici logických členov. S tejto knižnice bude možné pomocou myši vybrať logický člen a uložiť ho na pracovnú plochu. 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 nula a jednotku. Výsledný produkt sa bude skladať z dvoch častí, editačnej a simulačnej. V editačnej časti bude možné navrhnúť 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.

3.2 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 5. Bude poskytovaná nasledovná funkcionalita systému:

- práca s logickým členom alebo súčiastkou
  - pridanie
  - presunutie
  - prepojenie
  - vymazanie
- otvorenie nového projektu
- načítanie uloženého projektu
- uloženie projektu
- zapnutie a vypnutie tlačidla
- označenie časti alebo celku
- vymazanie časti alebo celku
- kopírovanie časti alebo celku
- presunutie časti alebo celku
- spustenie simulácie
- uložiť ako obrázok



Obrázok č.5: Prípady použitia.

Grafické používateľské rozhranie sa bude skladať z nasledovných častí:

- pracovná plocha
- panel podporných funkcií
- knižnica logických členov

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. 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

Systém bude zložený z viacero podsystémov, ktoré sú medzi sebou navzájom nezávislé. Toto zaručí, že jednotlivé podsystémy, 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 6.

Obrázok č.6: Moduly systému.

4.1.1 Modul logických súčiastok

Tento modul bude obsahovať abstraktnú reprezentáciu jednotlivých súčiastok logických obvodov. Výsledný produkt bude podporovať použitie 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.

- Žiarovka – indikácia 1 alebo 0. Bude sa používať na indikovanie vstupu a aj výstupu.
- Spoj – prechod medzi jednotlivými súčiastkami.
- a iné

Definície logických súčiastok respektíve členov budú udávať ich funkcionalitu. Všetky logické členy budú dediť od triedy *Gate*, ktorá bude vyzerať 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 stvrtom vstupe

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

        public int input1ID = 0; // id prveho vstupu
        public int input2ID = 0; // id druheho 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 logiského obvodu vždy, keď sa udeje nejaká zmena. Príkladom zmeny môže byť odstránenie členu, prepojenie členov, zmena vstupnej premennej obvodu. Metóda *processInputs* pre trojvstupový člen AND vyzerá nasledovne:

```
public void processInputs()
{
        if (input1 && input3 && 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.

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.

Hlavnou črtou používateľského rozhrania bude funkcia nazývaná Drag & Drop, ktorá umožní jednoduché a intuitívne pridávanie logických členov na pracovnú plochu. 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 členu, 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 funkcionalitu programu ktorá priamo nesúvisí so zadaním, ako možnosť uloženia a načítania namodelovaného logického obvodu, ukladanie a načítavanie zmien v modeli a podobne. Tieto funkcie bude možné ovládať pomocou riadiacej lišty, ktorá bude v hornej časti grafického používateľského rozhrania.

5 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].

[2] Frištacký N., Kolesár M., Kolenička J., Hlavatý J: *Logické systémy*, 1986. Alfa – Vydavateľstvo technickej a ekonomickej literatúry Bratislava. [cit. 2009-10-24].

[3] Pivarček Ján: Virtuálny verifikačný panel s členmi XOR a OR, Bakalárska práca, FIIT STU 2009

[4] Sivák Marek: Hradlá a hradlové štruktúry, podpora výuky – aplikácia pod OS Windows, Bakalárska práca, FIIT STU 2009

[5] Palaj Tomáš: Virtuálny verifikačný panel s členmi AND-NOR a NAND, Bakalárska práca, FIIT STU 2009

[6] Pilarčík Viliam: Virtuálny verifikačný panel s členmi NOR, Bakalárska práca, FIIT STU 2009

[7] 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.

[8] 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

[9] Komppa: Atanua Logic Simulator. [Online]. [cit. 2009-11-07]. Dostupný z WWW: http://sol.gfxile.net/atanua/

[10] 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.

[11] Tetzl, Andreas: Logic Simulator. [Online]. 22. novembra 2008. [cit. 2009-11-07]. Dostupný z WWW: http://www.tetzl.de/java_logic_simulator.html.