Vysoké Učení Technické v Brně

Fakulta Informačních Technologií

Modelování a simulace

2015 / 2016

Simulátor číslicových obvodů

Projekt do předmětu Modelování a simulace

Vypracoval:	Martin Pitřík a Jakub Pastuszek
Login:	xpitri00, xpastu00
Kontakt:	xpitri00@stud.fit.vutbr.cz, xpastu00@stud.fit.vutbr.cz
Datum vypracování:	30. listopadu 2015
Datum odevzdání:	6. prosince 2015

Obsah

1 Úvod

Tato práce se zabývá návrhem a implementací simulátoru logických číslicových obvodů sestavených z hradel AND, NAND, OR, NOR a NOT. Simulátor umožňuje simulaci jak kombinačních, tak sekvenčních logických obvodů v synchronních i asynchronních zapojeních.

Výstupem simulátoru je znázornění chování jednotlivých částí obvodu, tedy logických stavů na jednotlivých vstupech, výstupech a vodičích obvodu. Chování je znázorněno prostřednictvím grafické reprezentace časového diagramu za pomocí programu GNU Plot, případně lze využít textový výstup v podobě tabulky přechodů v čase.

Simulátor pro úspěšnou simulaci předpokládá běžné provozní podmínky použitých hradel, mezi které patří rozsah pracovních teplot udaný katalogovým listem a dodržení předepsaných napětí jednotlivých logických úrovní.

1.1 Zdroje faktů

Simulátor je postaven na společné části dvou autorů. Jeden z autorů obstarával návrh popisu simulovaného obvodu a jeho zpracování a převod do interních struktur simulátoru, druhý autor pracoval na provádění samotné simulace. Testování a experimentování je prací obou autorů.

Studium a realizace simulátoru rovněž zahrnovala i návrh popisu zapojení a parametrů simulovaného obvodu. Požadavkem byl jazyk syntakticky srozumitelný a snadno použitelný. V rámci návrhu byla provedena analýza simulačních jazyků používaných prostředím OrCad a jednoduchého simulátoru Atanua a editoru elektronických zapojení Eagle. Výsledný simulační jazyk je inspirován jazykem používaným programem Eagle.

Při koncepci a implementaci simulátoru se stěžejní literaturou stala (1). Pro samotné vyhodnocení správné funkce simulátoru posloužila dále literatura věnovaná číslicové technice a katalogové listy vybraných logických obvodů, konkrétně se jedná o (2), (3), (4), (5), (6), (7), (8).

1.2 Ověření funkčnosti

Správná činnost simulátoru byla ověřena na testovací množině zapojení kombinačních a sekvenčních obvodů, kde simulací získané výsledky byly porovnány s referenčními výsledky. Zdrojem referenčních výsledků se staly katalogové listy, literatura popisující chování daných obvodů a simulační prostředí OrCad.

Detailněji je tato část popsána v kapitole věnující se testování.

2 Fakta

Číslicovým systémem se rozumí logický obvod. Logický obvod je systém, jehož vstupní a výstupní veličiny nabývají pouze dvou hodnot. Pracují tedy s dvouhodnotovým signálem logická nula (L) a logická jednička (H) Logickou jedničkou se rozumí stav „zapnuto“, logickou nulou stav „vypnuto“ (2). Rovněž lze uvažovat neurčitý logický stav (X) (4).

Pro popis logických obvodů se využívá pravdivostních tabulek. Pravdivostní tabulka slouží k určení výstupního stavu podle vstupních proměnných. Pravdivostní tabulkou lze popsat i samotnou činnost jednotlivých hradel, kde udává závislost jednoho vstupního stavu na dvojici vstupních proměnných. Vstupy v tomto případě obvykle označujeme symboly A, B, výstup pak symbolem Y. Chování logických obvodů lze dále popsat prostřednictvím booleovy algebry pomocí tří základních funkcí: logického součtu (OR), logického součinu (AND) a logické negace (NOT) (2) (9).

U hradel definujeme pojem zpoždění průchodu signálu logickým členem. Tato doba označuje čas potřebný k přechodu z jednoho logického stavu na druhý. Během této změny hradlo nereaguje na své vstupy (9).

V rámci logických obvodů rozlišujeme dva základní typy. Jedná se o kombinační a sekvenční obvody. Výstup kombinačních obvodů je určen jen kombinací vstupních veličin. Výstup sekvenčního obvodu je určen nejen kombinací vstupních veličin, ale i hodnotami předcházejícího stavu logického obvodu (2).

2.1 Přehled základních logických funkcí

Logický součin (AND) je logická funkce definována jako $Y = A.B$. Logický součet (OR) je definován jako $Y = A + B$. Funkce logické negace (NOT) se algebraicky vyjádří ve tvaru $Y = \bar{A}$. Jednotlivé výstupy základních logických členů lze negovat, čím dostáváme členy NAND a NOR (2). Pravdivostní tabulky zde zmíněných funkcí jsou uvedeny v Tabulka A-1, Tabulka A-2, Tabulka A-3.

2.2 Klopné obvody

Z jednotlivých hradel lze sestavit základní prvek sekvenčních logických obvodů, kterým je klopný obvod. Klopný obvod je základní paměťový prvek, jehož úkolem je zaznamenat přítomnost přechodné informace i tehdy, když tato informace zmizí (2).

Dle funkce rozlišujeme astabilní, monostabilní a bistabilní klopné obvody. Z hlediska sekvenční logiky jsou podstatné bistabilní klopné obvody, které disponují dvojicí stabilních stavů, kdy jeden přechází skokem v druhý na základě, že je na vstup přiveden budící vstupní signál. Tyto obvody se poté využívají jako registry a statické paměťové obvody (2).

Z hlediska bistabilních klopných obvodů rozlišujeme typy RS, JK, D a T. Nejčastěji je využíván klopný obvod typu D. Dále rozlišujeme bistabilní klopné obvody dle způsobu řízení na asynchronní, hladinové a derivační. Asynchronní klopný obvod reaguje vždy při změně vstupních veličin, hladinový vždy při aktivní úrovní taktovacího signálu a derivační při změně taktovacího signálu (3).

Klopným obvod typu RS má dvojici vstupů RESET a SET a dvojici výstupů Q, !Q. Vstup SET slouží k nastavení klopného obvodu do jedničky, RESET poté k jeho nulování. Spojíme-li tyto vstupy přes logickou negaci, získáme klopný obvod typu D (2). Pravdivostní tabulky zde zmíněných klopných obvodů jsou uvedeny v Tabulka A-4, Tabulka A-5.

3 Koncepce simulátoru

Simulace číslicového obvodu je diskrétní simulací (1).

Logickou úroveň lze chápat, jako proměnnou, která nabývá uzavřenou množinu přípustných hodnot. V rámci simulace si vystačíme s hodnotami reprezentujícími logickou jedničku (H), logickou nulu (L) a neurčitý logický stav (X). Lze rozlišit i další přípustné stavy ve výsledcích simulace, nicméně pro jednoduchost je lze zanedbat a pracovat se sloučením do neurčitého logického stavu.

Popis hradla je řešen obecně tak, aby bylo možné dále od něj odvozovat další typy logických hradel. Tento popis pouze definuje dvojici vstupů A, B a výstupu Y, dále pak definuje pojmenování daného prvku a jeho zpoždění. V rámci simulace nepočítáme s více vstupními hradly, což na výslednou simulaci nemá vliv, jelikož lze provést ekvivalentní náhradu dvou vstupním hradlem. Další parametry hradla nemá smysl uvažovat, jelikož nemají vliv na jeho chování.

Popis chování konkrétních hradel je možné provést prostřednictvím booleovy algebry, nicméně v případě, kdy uvažujeme i neurčitý stav (X) je nutné buď nadefinovat operátory vlastní, či na základě pravdivostní tabulky implementovat danou funkčnost pomocí podmíněného větvení.

Dále je nutné zvolit způsob, kterým bude popsáno propojení mezi jednotlivými prvky obvodu. Propojení musí udržovat informace o svém názvu a logickém stavu, ve kterém se nachází, dále je nezbytné uchovávat informace o pinech jednotlivých hradel, které propojuje.

Obvod je také nutné popsat jako celek. Za tímto účelem je potřeba vytvořit seznam všech použitých hradel a seznam všech propojení.

Pro provádění samotné simulace je potřeba uchovávat informace o simulačním čase a o celkové době, po kterou má simulace probíhat. Není nutné uchovávat počáteční čas simulace, jelikož lze implicitně předpokládat, že simulace probíhá od času nula.

Nezbytnou součástí diskrétního simulátoru je plánovač událostí (1). Události lze reprezentovat seřazeným seznamem dle času. Každá položka v seznamu musí obsahovat čas, kdy se má provést, údaj o propojení, jehož stav má v daném čase ovlivnit a logickou úroveň, na kterou má dané propojení nastavit. Rovněž je udržována informace o objektu, který o danou změnu žádal.

Jelikož uvažujeme hradla se zpožděním, je nutné provádět kontrolu, zda dané hradlo může v daném čase plánovat událost, jelikož by vlivem zpoždění nemuselo být schopno danou změnu na svých vstupech zaznamenat. Pokud by toto nebylo ošetřeno, hradlo by teoreticky disponovalo „nekonečnou“ pamětí, tedy by provádělo pouze fázový posun signálu na svém vstupu vůči svému výstupu o hodnotu zpoždění. V reálném čase však dochází k zapomenutí těchto změn a plánovač, jakožto modul s přehledem nad simulačním časem, musí tyto stavy vyřešit. Řešením je prohledání již naplánovaných událostí a kontrola, zda v čase menším, než je zpoždění aktuálně žádajícího hradla, hradlo nenaplánovalo událost. Pokud ano, je aktuálně plánovaná událost zahozena.

V případě simulace je nutné získat i její výsledky. Je tedy nutné znát názvy výstupních souborů, do kterých budou výsledky uloženy, a dále je nezbytné uchovávat změny simulovaného systému v průběhu simulačního času.

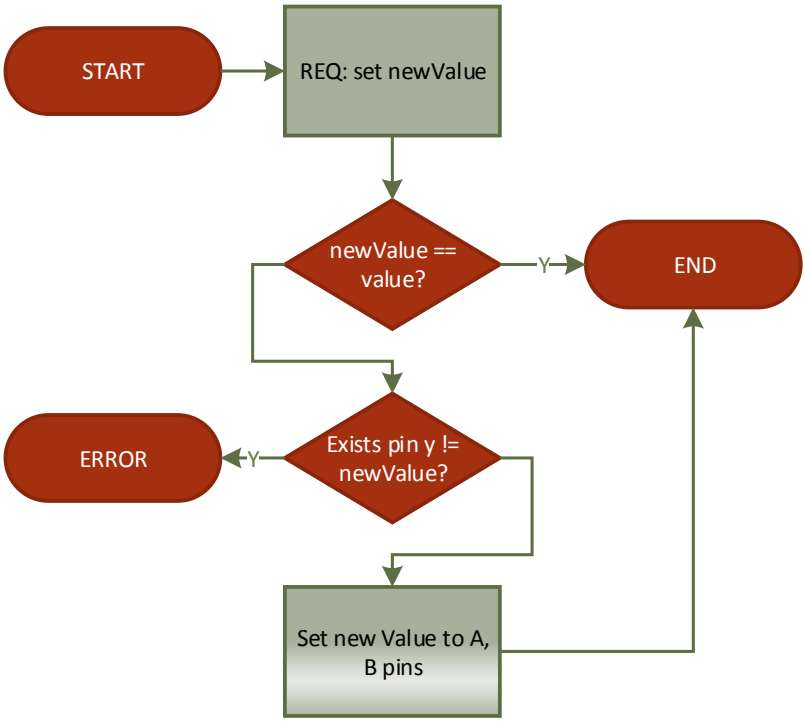
3.1 Popis vybraných algoritmů

Pro ilustraci si uveďme alespoň jeden příklad algoritmizace pravdivostní tabulky (pro hradlo typu AND).

```
if (A or B is L) then Y = L;
else if (A and B is H) then Y = H;
else Y = X
```

Dále je zde popsán algoritmus provádějící nastavení logického stavu na propojení. Pokud se nově nastavovaný stav neliší od aktuálního stavu, není potřeba provádět další činnost. V opačném případě dochází k adresování všech výstupních pinů a kontrole, zda nově nastavovaný stav není v rozporu se stavem, ve kterém je propojení drženo některým z výstupů hradel (neurčitý logický stav X je brán jako nekonfliktní jak k logické jedničce, tak nule). Pokud je vše v pořádku, dochází k adresování připojených vstupů a k jejich nastavení.

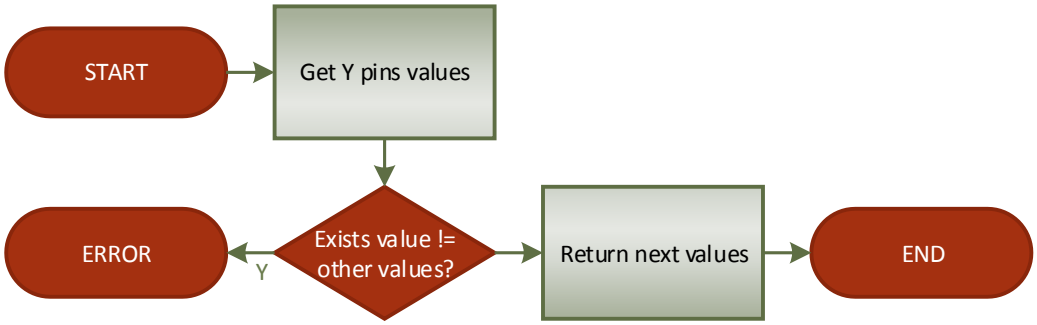
Algoritmus je ve formě UML diagramu znázorněn na Obrázek 3-1.



Obrázek 3-1: Nastavení nové logické úrovně.

Pro zjištění následujících stavů jsou adresovány výstupní piny hradel a zjišťuje se, jaký výstup bude v čase po uplynutí zpoždění. Rovněž se provádí kontrola, zda v totožném čase není požadavek na nastavení konfliktních úrovní.

Algoritmus je znázorněn na Obrázek 3-2.



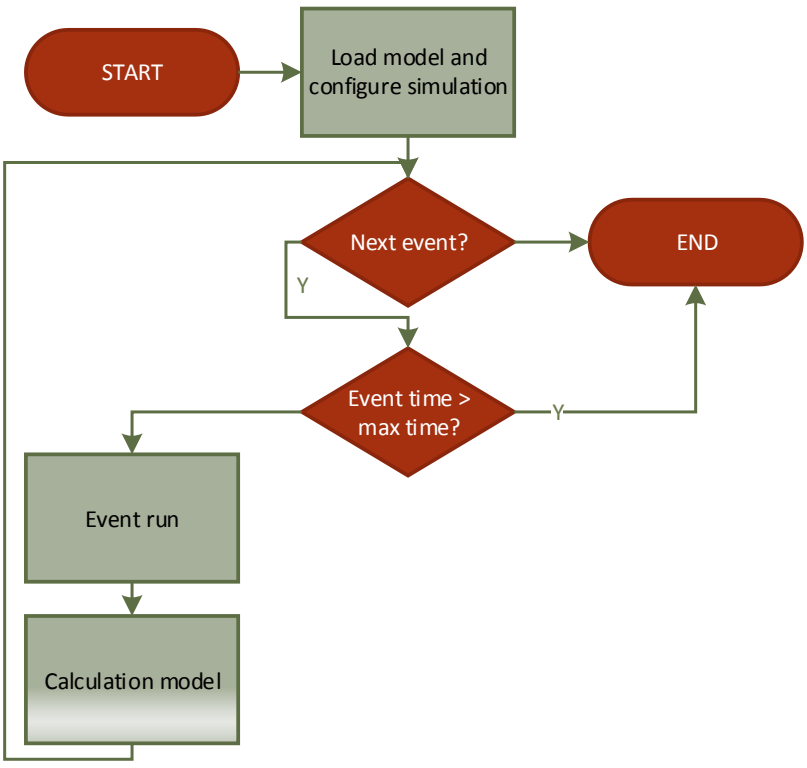
Obrázek 3-2: Získání výstupních hodnot.

Celkově lze činnost simulátoru jako celku popsat následovně. Ve fázi inicializace se načte model a provede se nastavení parametrů simulace a modelu (plánují se události na nulový simulační čas, případně plánování taktovacího signálu přes celý čas simulace).

Poté se přechází k samotné simulaci, kdy je kontrolováno, zda vůbec jsou plánovány nějaké události. Pokud ne, simulace tímto končí. Pokud ano, je provedena kontrola, zda čas naplánované události spadá do intervalu času, po který provádíme simulaci a pokud ne, simulace končí.

V poslední fázi simulátor provede naplánovanou událost (nastavení daného propojení na danou logickou úroveň) a poté provede přepočet, který může vést k naplánování dalších událostí na základě právě provedené události. Takto se pokračuje, pokud existují události, které lze provést a není překročen čas simulace.

Princip činnosti algoritmu je znázorněn na Obrázek 3-3.



Obrázek 3-3: Funkce simulátoru.

4 Implementace a architektura simulátoru

Pro implementaci simulátoru byl zvolen programovací jazyk C++ verze 11, pro následnou grafickou vizualizaci byl použit GNU Plot verze 5.0.

Pro popis simulovaného obvodu byl vytvořen vlastní simulační jazyk. Jazyk myšlenkově vychází z jazyka používaného v programu Eagle a dále pak z programovacího jazyka Assembler. Jazyk umožňuje pomocí nepočetné množiny klíčových slov popsat libovolný logický obvod. Gramatika a sémantika tohoto jazyka je uvedena v příloze C.

Koncept popsaný z kapitoly 3 byl implementován do jednotlivých tříd, struktur a výčtových typů.

Dle popisu obvodu prostřednictvím simulačního jazyka se vytvoří jednotlivé instance komponent simulovaného obvodu. Konkrétně se jedná o instance tříd jednotlivých hradel, které implementují jednotné rozhraní definované bázovou třídou Logics. Dále jsou vytvořeny jednotlivé instance třídy Connection, která popisuje propojení jednotlivých pinů hradel navzájem mezi sebou. Vzniklé objekty hradel jsou následně uloženy do tabulky hradel, objekty popisující propojení jsou vloženy do tabulky všech propojení. Tyto tabulky jsou implementovány pomocí návrhového vzoru jedináček, neb je požadována pouze jedna, sdílená, instance napříč jednotlivými moduly simulátoru.

Dále se provede instance plánovače popsaného třídou Scheduler. Simulace začíná naplánováním událostí v simulačním čase nula (většinou se jedná o veličiny vstupů definované v simulačním popisu). Používá-li simulace i taktovací signál, vytvoří se společně s interním propojením clk instance třídy CLKGen a poté se prostřednictvím plánovače dle periody naplánují změny taktovacího signálu.

Poté se vytváří a inicializuje simulační jádro popsané třídou SimCore, která v metodě run definuje algoritmus popisující provádění diskrétní simulace. Rovněž se vytváří instance třídy PlotCreator, jenž zaznamenává změny v čase v simulovaném obvodu a při skončení simulace generuje soubory s výsledky.

4.1 Uživatelská dokumentace

Simulátor je řešen jako konzolová aplikace, která při svém spuštění očekává jeden parametr, kterým je název souboru popisující simulovaný obvod. V případě nutnosti překladu je k dispozici soubor Makefile.

V případě systému Windows probíhá spuštění následovně:

```
sim.exe netlist.txt
```

V případě systému Linux:

```
./sim netlist.txt
```

Pokud není vyvolána chyba značící selhání při zpracovávání zdrojového popisu, je spuštěna samotná simulace, jejíž postupné výsledky jsou vypisovány na standardní výstup. Seznam chyb je v Tabulka B-1.

Výsledkem simulace je dvojice souborů, pojmenovaných stejným jménem jako zdrojový soubor. První soubor ve formátu *.dat obsahuje textové informace o stavech jednotlivých částí obvodu v závislosti na čase. Druhý soubor ve formátu *.plt obsahuje skript pro použití v nástroji GNU Plot pro grafické vykreslení výsledků simulace.

5 Testování

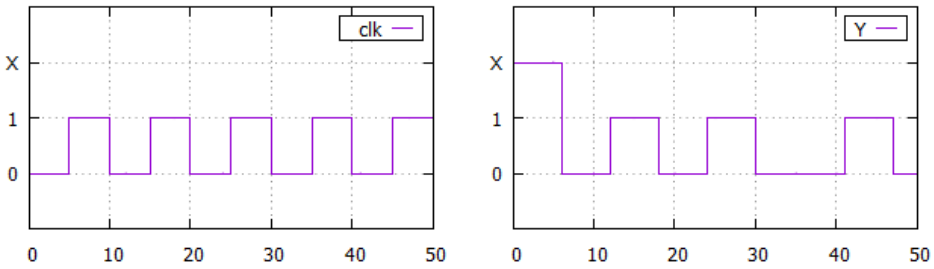
Testovací množina čítala několik zapojení z kombinační a sekvenční logiky. Schémata testovacích zapojení jsou uvedena v příloze D tohoto dokumentu. Simulační popisy jsou součástí souborů k projektu.

5.1 Testování zpoždění

Správná implementace zpoždění je stěžejním bodem pro činnost simulátoru a hraje významnou roli v případech, kdy se při experimentování pohybujeme za hranicemi časových parametrů hradel.

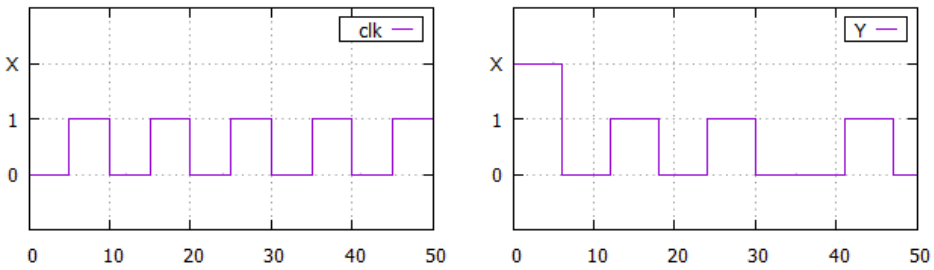
Pro vstup jsme předpokládali obvod složený s hradla AND se spojenými vstupy přivedenými na zdroj taktovacího signálu. Hradlo se zpožděním 6 ns. Perioda hodinového signálu je 10.

Referenční výstup:



Obrázek 5-1: Referenční výstup.

Výstup po provedení simulace:



Obrázek 5-2: Výstup simulace.

Výsledky provedeného testování: Jak je patrno na Obrázek 5-1, Obrázek 5-2, je přepokládaný výstup shodný s výstupem provedené simulace. V čase nula hradlo zpracovává informaci ze svých vstupů a po uplynutí zpoždění provede změnu stavu svého výstupu (Y), jelikož nevíme, jaká hodnota byla na daném výstupu zezačátku, musíme jí označit jako X. Hradlo společně s nastavením svého výstupu je schopno zpracovat nový vstup a po uplynutí zpoždění se tento vstup opět projeví na výstupu. Jelikož je však změna vstupní veličiny rychlejší než je zpoždění hradla, dojde v čase 30 k zapomenutí vstupu a hradlo svůj výstup nezmění.

5.2 Testování detekce kolize na propojení

Pro vstup jsme předpokládali obvod složený z dvojice hradel AND, jejichž výstup byl navzájem propojen. V případě prvního testování měly hradla nadefinované stejné zpoždění, v případě druhého testu bylo nastaveno zpoždění rozdílné.

Referenční výstup: zahlášení chyby v obou dvou případech.

Výstup po provedení simulace: zahlášení chyby v obou dvou případech.

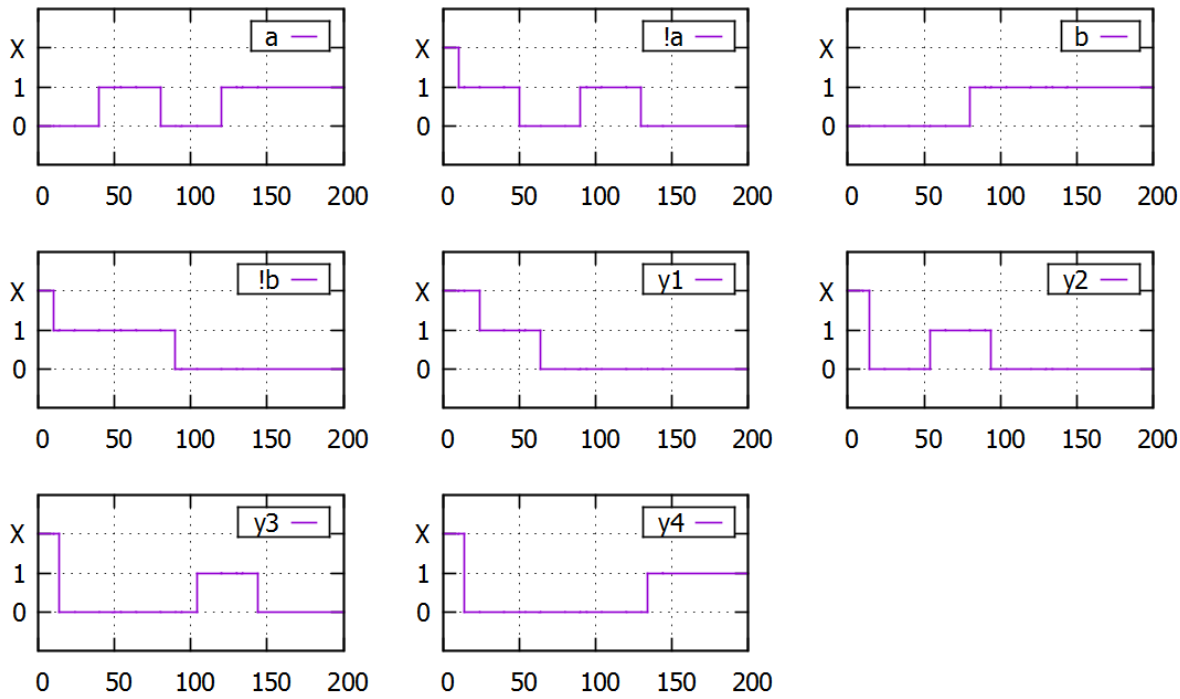
Výsledky provedeného testování: Výstup simulátoru odpovídá očekávání. V prvním případě ve stejný čas hradla dávají na svůj společný výstup rozdílné logické hodnoty, což vede k nejednoznačnému výstupu (zkratu). V druhém případě hradlo s menší hodnotou zpoždění úspěšně nastaví propojení, avšak druhé hradlo záhy žádá o provedení změny stavu na daném společném výstupu. Toto opět vede k nejednoznačnému výstupu (zkratu).

5.3 Dekodér binárního kódu 1 ze 4

Pro vstup jsme předpokládali obvod znázorněný na Obrázek D-1. Na vstupech obvodu byly měněny logické úrovně tak, aby obvod postupně sepnul výstupy y_1, y_2, y_3, y_4.

Referenční výstup: postupná změna výstupů y_1, y_2, y_3, y_4 dle zadaných vstupních kombinací. Změna na výstupech se musí projevit s maximálním zpožděním, které je dáno zpožděním jednotlivých členů v řetězci.

Výstup po provedení simulace:



Obrázek 5-3: výstup simulace dekodéru 1 ze 4.

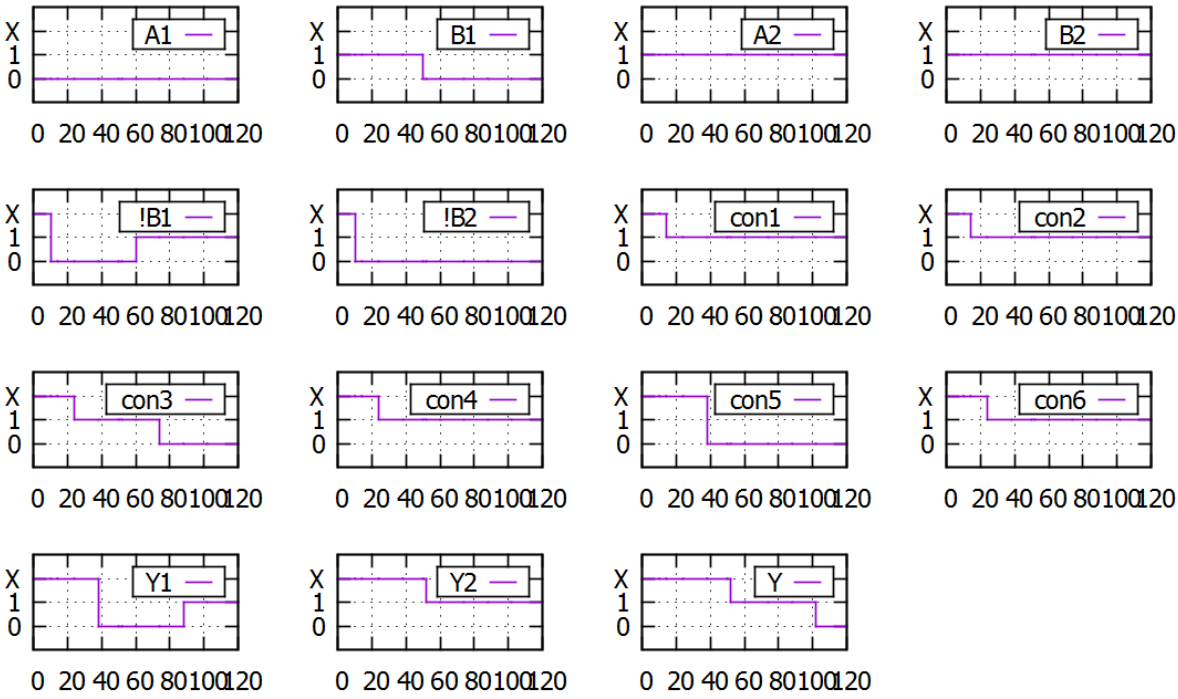
Výsledky provedeného testování: Výstup ze simulátoru na Obrázek 5-3 odpovídá referenčnímu výstupu, kdy postupně dekodér správně spíná své výstupy v závislosti na svých vstupech. Hodnota na výstupu se také projevuje v očekávaném čase.

5.4 Dvoubitový komparátor

Pro testování jsme předpokládali obvod znázorněný na Obrázek D-2. Na vstupech obvodu byla měněna logická hodnota a byl sledován výstup komparátoru.

Referenční výstup: V případě rovnosti čísel na vstupech komparátoru je očekáván přechod výstupu na úroveň logické nuly. V opačném případě je na výstupu komparátoru očekávaná logická jednička. Změna na výstupu musí být provedena opět do času od změny vstupu, který je dán zpožděním jednotlivých hradel v pracovním řetězci.

Výstup po provedení simulace:



Obrázek 5-4: Výstup ze simulace dvoubitového komparátoru.

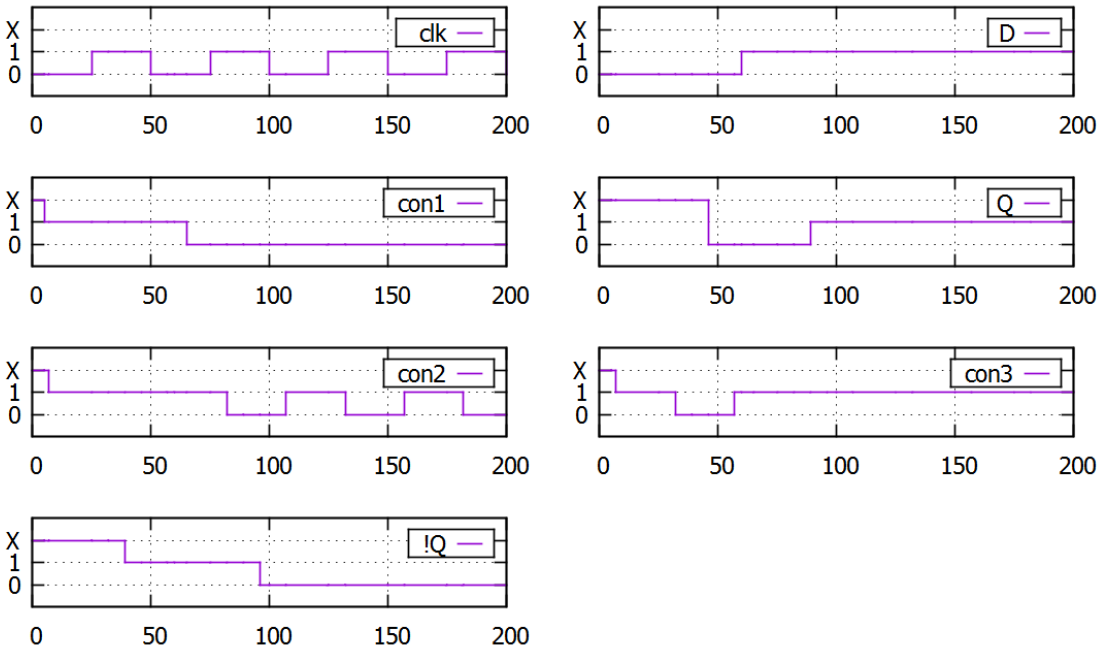
Výsledky provedeného testování: Jak je patrno z Obrázek 5-4, obvod v případě nerovnosti čísel na svých vstupech přechází výstupem do logické jedničky a v případě, že jsou čísla na vstupech rovna, výstup komparátoru přechází do logické nuly. Změna se odehrává rovněž v předpokládaném čase.

5.5 Synchronní hladinový klopný obvod typu D

Pro testování D klopného obvodu bylo předpokládáno zapojení znázorněné na Obrázek D-3. Na vstupu klopného obvodu byla měněna vstupní hodnota a byly sledovány výstupy klopného obvodu.

Referenční výstup: Klopný obvod tohoto typu může reagovat na změnu pouze v případě, že je taktovací signál v aktivní úrovni (v tomto zapojení logická jednička). V tomto případě se změna vstupu přenese i na výstup klopného obvodu, pokud není taktovací signál aktivní, ne-provede se nic.

Výstup po provedení simulace:



Obrázek 5-5: Výsledky simulace hladinového D klopného obvodu.

Výsledky provedeného testování: Jak je patrno na Obrázek 5-5, tak v čase 25 dochází k aktivaci taktovacího signálu a vstup D je v logické nule. Po uplynutí zpoždění klopného obvodu dochází k nastavení jeho výstupu do úrovně logické nuly, negovaný výstup přechází do logické jedničky. Tento stav je poté klopným obvodem držen. V čase 75 dochází k další aktivaci taktovacího signálu, kdy je již vstup D nastaven na úroveň logické jedničky. Tímto dochází k nastavení klopného obvodu, který po čase zpoždění přechází do logické jedničky a jeho negovaný výstup přechází do úrovně logické nuly.

6 Průběh a výsledky simulačních experimentů

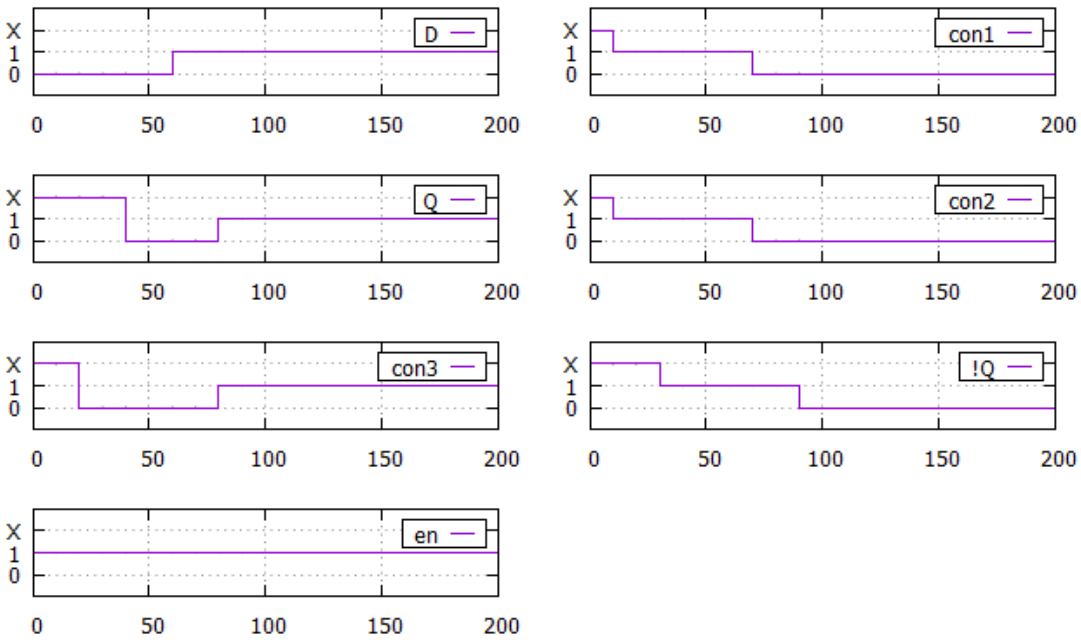
Jako simulační experiment jsme zvolili prozkoumání vnitřní struktury integrovaného obvodu 74HC374 (11). Jedná se o integrovaný obvod obsahující soustavu hladinových D klopných obvodů. Při experimentu zanedbáváme vliv výstupních třístavových zesilovačů.

Zadání experimentu: Realizovat interní zapojení klopného obvodu a experimentální zjištění možných zpoždění na jeho jednotlivých logických členech, aby byly splněny hodnoty z katalogového listu (zpoždění 15 ns). Při experimentování se pokusíme použít jen takové parametry, které lze u dostupných logických členů nalézt (pokud budou k dispozici).

Očekávané výstupy: Očekávaným výstupem je korektní chování hladinového D klopného obvodu s časovými parametry stejnými či lepšími, než je uvedeno v katalogovém listě.

Vstupy po první experiment: Zvažme konstrukci z integrovaných obvodů 74LS00 (NAND s typickým zpožděním 10 ns) a obvodu 74LS04 (NOT s typickým zpožděním 10 ns). Vstup klopného obvodu poprvé nastavme na logickou nulu, poté v čase 60 ns na logickou jedničku. Na povolovací vstup je trvale přivedena logická jednička.

Výstup po provedení simulace:

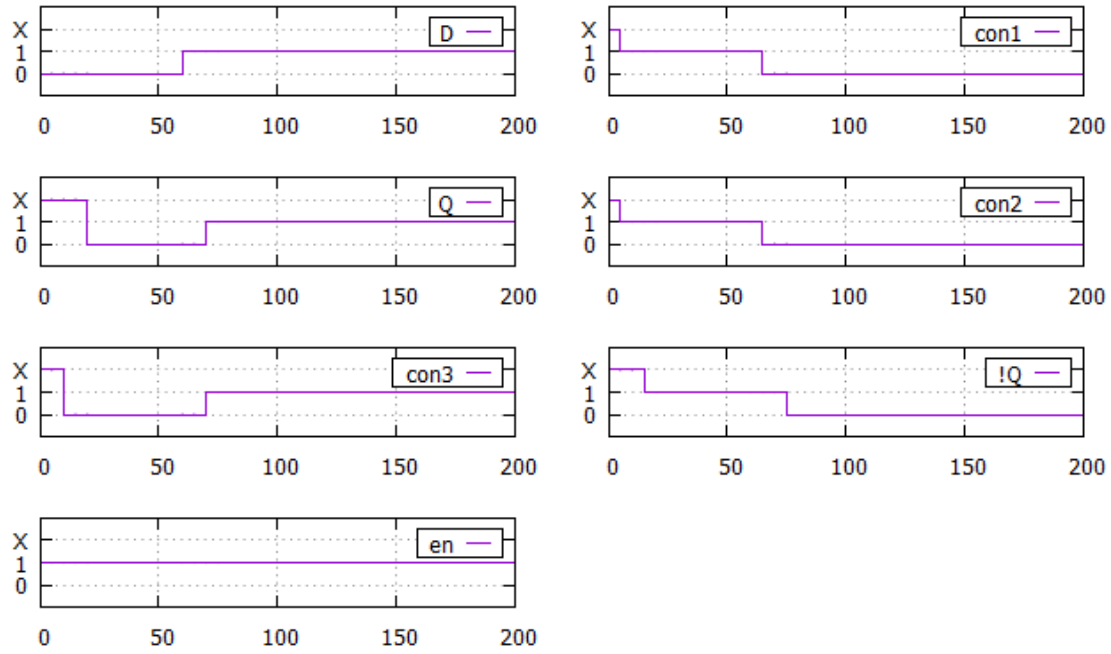


Obrázek 6-1: Výsledek prvního experimentu.

Závěr prvního experimentu: Z výsledků experimentu na Obrázek 6-1 je patrno, že klopný obvod plní svou funkci. Nyní ověříme, zda použité hradla garantují výstup v požadovaném čase. Pro nastavení klopného obvodu do stavu logické nuly je potřeba 40 ns. Již zde je zřejmé, že očekávaných časů nebylo dosaženo.

Vstupy pro druhý experiment: Nyní zkusíme použít rychlejší řadu obvodů. Jako hradla NAND použijme obvod SN74S00 s typickým zpožděním 5 ns. Jako hradlo NOT použijme SN74S04 se stejnou hodnotou zpoždění. Ostatní parametry jsou stejné, jako v případě prvního experimentu.

Výstup po provedení simulace:

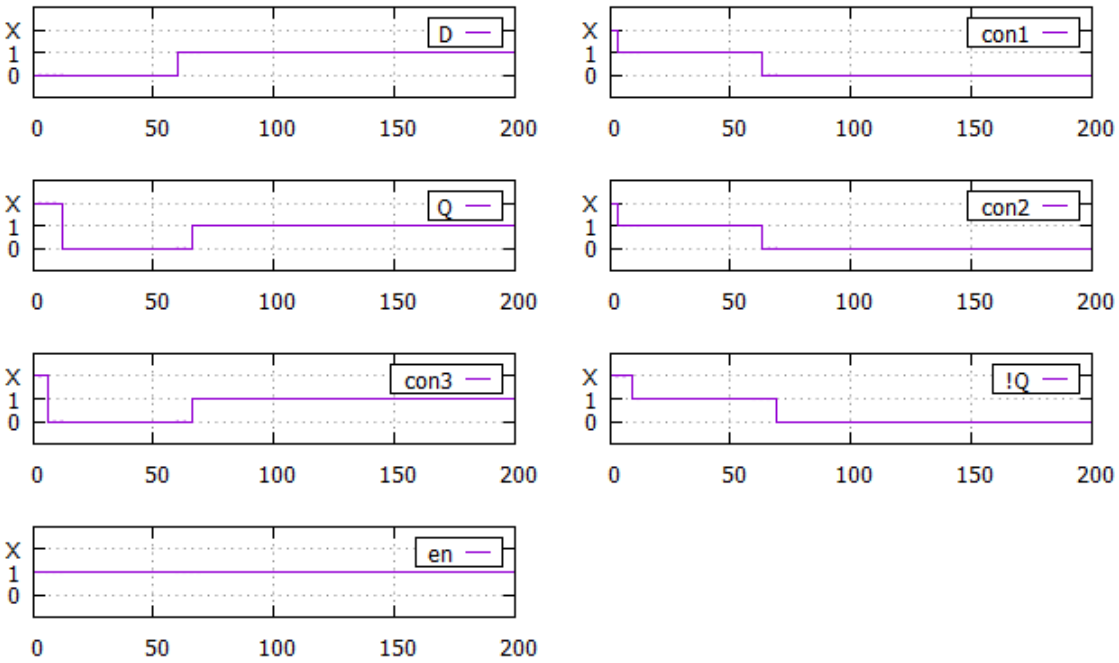


Obrázek 6-2: Výsledek druhého experimentu.

Závěr druhého experimentu: Z výsledků simulace na Obrázek 6-2 je patrno, že klopný obvod plní svou funkci. Nastavení do stavu logické nuly trvá 20 ns, do logické jedničky 10 ns.

Vstup pro třetí experiment: Pro další experiment uvažujme zpoždění 3 ns, které lze při snížení kapacity $C_L = 15\,pF$ od těchto obvodů očekávat. Ostatní parametry experimentu jsou stejné.

Výstup po provedení simulace:



Obrázek 6-3: Výsledek třetího experimentu.

Závěr třetího experimentu: Jak je patrno na Obrázek 6-3, klopný obvod přechází do stavu logické nuly za čas 12 ns. Ze stavu logické nuly do stavu logické jedničky přejde za čas 6 ns.

Závěr: Lze tedy usoudit, že struktura námi zkoumaného integrovaného obvodu obsahuje logické členy se zpožděním 3 ns. Podle katalogového listu obvodů řady S lze takto nízkých zpoždění dosáhnout při snížení kapacity $C_L = 15\,pF$. Lze tedy usoudit, že námi zkoumaný obvod bude obsahovat zapojení s těmito parametry.

7 Závěr

V rámci projektu vznikl simulační nástroj umožňující simulovat chování asynchronních a synchronních kombinačních a sekvenčních logických obvodů složených z hradel typu AND, OR, NOT, NAND a NOR s možností definovat zpoždění na jednotlivých hradlech. Výsledky simulace lze dále zpracovávat v textové podobě programy třetích stran, případně je možné využít generovaného skriptu pro vizualizaci prostřednictvím programu GNU Plot.

Validita výsledků simulátoru byla ověřena na testovací množině skládající se z logických obvodů různých typů. V rámci validace byla prokázána ekvivalence mezi referenčními očekávanými výstupy a výstupy z prováděných simulaci.

Rovněž jsme v závěru experimentálně zjistili parametry jednotlivých členů ve vnitřním zapojení integrovaného obvodu 74HC374.

8 Citovaná literatura

1. **Peringer, Petr.** Modelování a simulace. 2012.

2. **Antošová, Marcela a Vratislav, Davídek.** *Číslicová technika.* České Budějovice : Kopp, 2009. 978-80-7232-394-4.

3. **Juránek, Leoš.** Klopné obvody.

4. **Kesl, Jan.** *Elektronika 3 číslicová technika.* Praha : BEN, 2008. 978-80-7300-182-7.

5. **Texas Instruments.** SN7400 (NAND).

6. —. SN7402 (NOR).

7. —. SN7408 (AND).

8. —. SN7432 (OR).

9. **Wikipedie.** Booleova algebra. *Wikipedie.* [Online] [Citace: 22. 11 2012.] https://cs.wikipedia.org/wiki/Booleova_algebra.

10. **Škuta, Jaromír.** Technické řešení logických obvodů. [Online] [Citace: 22. 11 2015.] http://352lab.vsb.cz/ServerFinalVer/UcebniceLPaS/text/Techreal/Techreal.htm.

11. **Philips.** 74HC/HCT374 (D).

A) Pravdivostní tabulky

Tabulka A-1: Pravdivostní tabulka AND.

A	B	Y
1	1	1
1	0	0
0	1	0
0	0	0
0 / X	X / 0	0
1 / X	X / 1	X
X	X	X

Tabulka A-2: Pravdivostní tabulka OR.

A	B	Y
1	1	1
1	0	1
0	1	1
0	0	0
0 / X	X / 0	X
1 / X	X / 1	1
X	X	X

Tabulka A-3: Pravdivostní tabulka NOT.

A	Y
1	0
0	1
X	X

Tabulka A-4: Pravdivostní tabulka RS klopného obvodu.

S	R	Qt+1	!Qt+1
1	1	(1)	(1)
1	0	1	0
0	1	0	1
0	0	Qt	!Qt

Tabulka A-5: Pravdivostní tabulka D klopného obvodu s povolovacím vstupem.

T	D	Qt+1
0	0 / 1	Qt
1	0	0
1	1	1

B) Seznam a popis chyb

Tabulka B-1: Seznam a popis chyb.

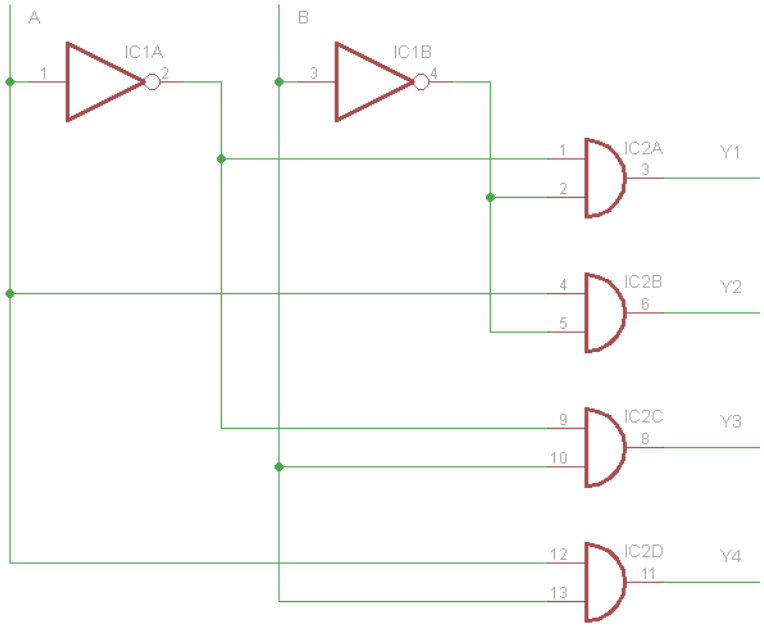
Chybová zpráva	Popis chyby
file unreachable	Nelze zpracovat soubor, soubor neexistuje nebo k němu proces nemá přístup.
wrong keyword <token> in this content	Chybné klíčové slovo v daném kontextu.
keyword <token> not found	Klíčové slovo nebylo nalezeno.
time redefinition	Předefinování času simulace.
time value has to be a number	Hodnota času musí být číslo.
time too short	Hodnota času je příliš malá. Minimální hodnota je dva.
clk redefinition	Předefinování CLK.
clk value has to be a number	Hodnota CLK musí být číslo.
connection redefinition - <con>	Předefinování propojení.
gate redefinition - <gate>	Předefinování hradla.
gates name cannot contain dot(s) - <gate>	Název hradlo nesmí obsahovat tečku.
delta not found	Zpoždění na hradle nebylo definováno
delta has to be a number	Zpoždění musí být číslo.
semicolon not found	Očekáván znak středníku.
unspecified	Blíže nespecifikovaná chyba při zpracovávání netlistu.
SET - connection <con> not defined	Spojení nebylo definováno
SET - time not found	Čas nastavení spojení musí být definovaný.

scheduled time has to be a number - <con>	Naplánovaný čas musí být číslo.
SET - level not found	Při plánování musí být zadána požadovaná hladina.
scheduled level has to be high(H|1) or low(L|0) or X	Naplánovaná změna sběrnice musí být do stavu log. 1 nebo 0, popř. nedefinovaný stav X.
ADD - connection <con> not defined	Použité spojení nebylo definováno.
ADD – after <con> comma not found	Po vybrání spojení musí následovat čárka.
ADD - gate <gate> not defined	Použité hradlo nebylo definováno.
ADD - missing gate port - <gate>	Nebyl použit žádný port hradla.
ADD - wrong gate port - <port>	Byl použit nedefinovaný port hradla.
ADD - comma or semicolon not found	Očekáván znak čárky nebo středníku.
comma not found	Očekáván znak čárky.
not defined level	Simulační chyba, došlo k logickému konfliktu.
sim error	Chyba parametrů simulace.

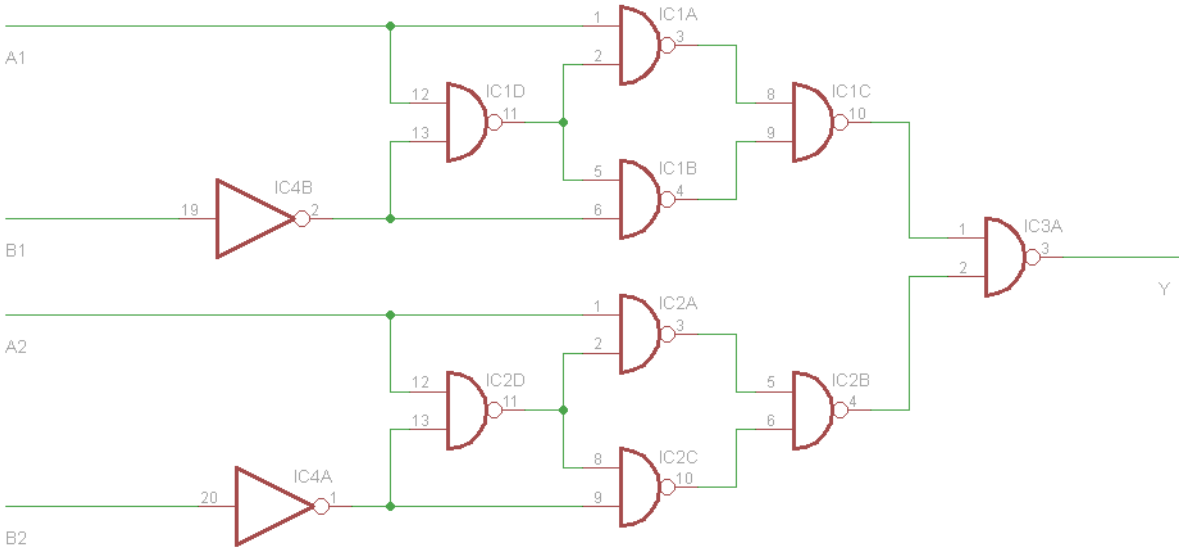
C) Gramatika simulačního jazyka

```
NETLIST > DEF MAIN eof

DEF > eps
DEF > time {val} ; DEF
DEF > clk {val} ; DEF
DEF > gate {id_gate} , {val} ; DEF
DEF > con {id_con} ; DEF

MAIN > begin BODY end

BODY > eps
BODY > set {id_con} , {val} , {level} ; BODY
BODY > add {id_con} , GATEID ADD BODY

GATEID > {id_gate} . {port}

ADD > ;
ADD > , GATEID ADD
```

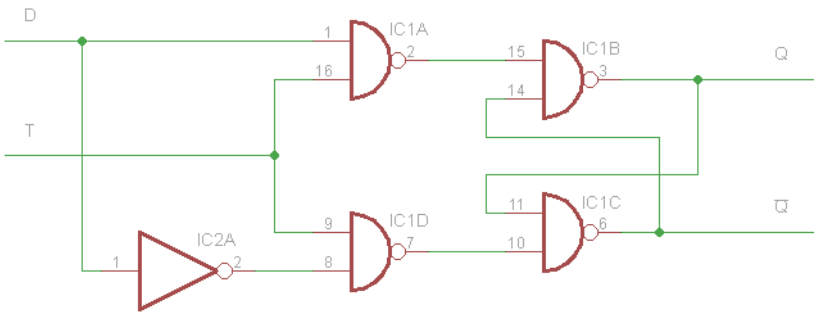
D) Schémata zapojení testovacích obvodů



Obrázek D-1: Zapojení dekodéru 1 ze 4.



Obrázek D-2: Zapojení dvoubitového komparátoru.



Obrázek D-3: Zapojení D hladinového klopného obvodu.