České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačů

Bakalářská práce

Grafické uživatelské rozhraní pro interaktivní návrhový systém

Karel Kohout

Vedoucí práce: Ing. Petr Fišer, Ph.D.

Studijní program: Elektrotechnika a informatika, strukturovaný, Bakalářský

Obor: Výpočetní technika

22. ledna 2009

Poděkování

Tímto bych rád poděkoval vedoucímu této práce Ing. Petru Fišerovi, Ph.D. za odbornou pomoc, návrhy a připomínky k vytvořené aplikaci a své rodině za morální podporu při programování a sepisování této práce.

Prohlášení

Prohlašuji, že jsem práci vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.
Nemám závažný důvod proti užití tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Mostech u Jablunkova dne 22. ledna 2009 .....

Abstract

The purpose of this work is to create a graphical user interface for an interactive tool for the design of logic circuits, ABC. This interface should simplify work with this tool and make this work more effective. This publication describes the design, implementation and control of application. Furthermore, there is described testing of a program and finally in the end of this work there is discussion of the possible continuation of the development of the application.

Abstrakt

Účelem této práce je vytvořit grafické uživatelské rozhraní pro interaktivní nástroj pro návrh logických obvodů ABC. Toto rozhraní by mělo zjednodušit a zefektivnit práci s nástrojem. V této publikaci je popsán návrh řešení, implementace a ovládání aplikace. Dále je popsáno testování vytvořeného programu a nakonec v závěru je obsažena diskuse nad možným pokračováním ve vývoji aplikace.

Obsah

Seznam obrázků ..... **xiii**

Seznam tabulek ..... **xiv**

1. Úvod **1**

2. Cíle a požadavky aplikace **3**

2.1 Cíle aplikace *3*

2.2 Požadavky aplikace *3*

3. Analýza **5**

3.1 ABC *5*

3.2 WinAPI *6*

4. Implementace **9**

4.1 Úprava programu ABC *9*

4.2 Vytvoření knihovny ABC *10*

4.3 Příklad použití knihovny *10*

4.4 Princip použití knihovny ve WinABC *12*

4.5 Moduly *12*

4.6 Vlákna *13*

4.7 Hlavičkový soubor WinABC.h *14*

4.8 Nastavení *14*

4.9 Zobrazení obvodu ve formátu… *15*

4.10 Doporučená konfigurace počítače *15*

5. Popis a ovládání aplikace **17**

5.1 Hlavní okno aplikace *17*

5.2 Inicializace *19*

5.3 Načtení souboru *19*

5.4 Uložení souboru *19*

5.5 Nastavení *20*

5.6 Nastavení výstupů *21*

5.7 Seznam oblíbených příkazů *23*

5.8 Command Wizard *24*

5.9 Výstupní okno příkazu *25*

6. Testování aplikace **27**

6.1 Testování přesnosti výpočtu *27*

6.2 Testování rychlosti výpočtu *28*

6.2.1 Test I 28

6.2.2 Test II 29

7. Závěr **31**

Literatura **33**

Seznam použitých zkratek **35**

Obsah přiloženého CD **37**

Seznam obrázků

Obrázek 3.1: Pole působnosti ABC 5
Obrázek 4.1: Princip použití knihovny ve WinABC 12
Obrázek 4.2: Znázornění vlákna pro provedení výpočtu 14
Obrázek 5.1: Hlavní okno aplikace 18
Obrázek 5.2: Nastavení aplikace 20
Obrázek 5.3: Nastavení výstupů 22
Obrázek 5.4: Seznam oblíbených příkazů 23
Obrázek 5.5: Přidání/editace položky 23
Obrázek 5.6: Command Wizard 24
Obrázek 5.7: Okno pro zobrazení textového výstupu příkazu 25

Seznam tabulek

Tabulka 4.1: Struktura souboru „settings.dat“ 15
Tabulka 4.2: Minimální a doporučená konfigurace 16
Tabulka 6.1: Výsledky měření testu I pro konfiguraci I 28
Tabulka 6.2: Výsledky měření testu I pro konfiguraci II 29
Tabulka 6.3: Výsledky měření testu II 29

1. Úvod

Účelem této práce je vytvořit grafické uživatelské rozhraní pro interaktivní nástroj pro návrh logických obvodů ABC. V praxi to je tedy vytvoření zcela nové aplikace, která využívá naprogramované funkce původního nástroje. Jelikož nově vytvořená aplikace je určena pro operační systém Windows a využívá jeho základní grafické prvky, dostala název WinABC.

Důvodů, které vedly k vytvoření této aplikace, bylo hned několik. V prvé řadě je to zrychlení a zefektivnění práce s nástrojem ABC, dále zatraktivnění nástroje pro začínající uživatele, anebo zobrazení většího množství informací o zpracovávaném obvodu v jednom okně.

V následující kapitole se nachází podrobný popis cílů a požadavků vytvořené aplikace. Pomocí těchto daných požadavků bude aplikace v samém závěru této práce zhodnocena. Třetí kapitola krátce popíše, co to vůbec interaktivní nástroj ABC je a jaké je jeho použití. V této kapitole je popsán i princip programování ve WinAPI. Čtvrtá kapitola se zabývá konkrétní implementací vyvíjené aplikace a použitými technologiemi v programování. Pátá kapitola popíše již hotový program a seznámí nás se vzhledem a chováním aplikace. V šesté kapitole se dozvíme o průběhu a výsledcích testování. V samém závěru této publikace proběhne zhodnocení práce a nástin případného dalšího vývoje aplikace WinABC.

2. Cíle a požadavky aplikace

2.1 Cíle aplikace

Hlavním cílem aplikace WinABC je přenést konzolovou aplikaci ABC, vyvinutou kalifornskou univerzitou v Berkeley, do grafického prostředí Windows a využít výhody s tímto spojené. Jedná se především o zjednodušení ovládání programu, zrychlení práce s programem, snadná kontrola zpracovávaného obvodu, uživatelsky přátelštější prostředí a další výhody. Pro začínajícího uživatele programu je jistě snazší se naučit ovládat program v grafickém prostředí, než se učit příkazy konzolové aplikace. Aplikace WinABC bude však určena i pro zkušené uživatele konzolové aplikace ABC, neboť možnost zadávání příkazů podobně, jako na příkazové řádce, zde nezmizí a tak přechod z původního rozhraní do rozhraní grafického bude bezproblémový.

2.2 Požadavky aplikace

Požadavky na novou aplikaci využívající grafické prostředí jsou následující. Umožnit zjednodušení a zrychlení práce, nezpomalit příliš výpočet operací, nízké hardwarové nároky, neomezit funkcionalitu původního konzolového programu.

Zjednodušení a zrychlení práce logicky vyplývá z použití grafického rozhraní. Vybírání položek z menu nebo nástrojové lišty je snazší, než vše vypisovat do příkazové řádky. Použití okna s nejpoužívanějšími příkazy šetří i čas, který by byl použit pro ruční napsání příkazu (např. některé příkazy s použitím parametrů cest k souboru mohou dosahovat i více než 100 znaků). Další zrychlení práce je způsobeno i automatickým vypisováním statistických údajů o zpracovávaném obvodu ihned po vykonání příkazu.

Požadavek nezpomalit příliš výpočet operací by měl být dosažen použitím vhodné technologie pro vývoj grafického uživatelského rozhraní a celé aplikace WinABC. Tato zvolená technologie by také neměla způsobit dramatické zvýšení hardwarových nároků oproti původní konzolové verzi programu.

Výpočet každé operace ve WinABC by měl být časově srovnatelný s časem výpočtu v původní konzolové verzi aplikace. Tento požadavek bude ověřen testováním rychlosti výpočtu stejné operace u obou aplikací.

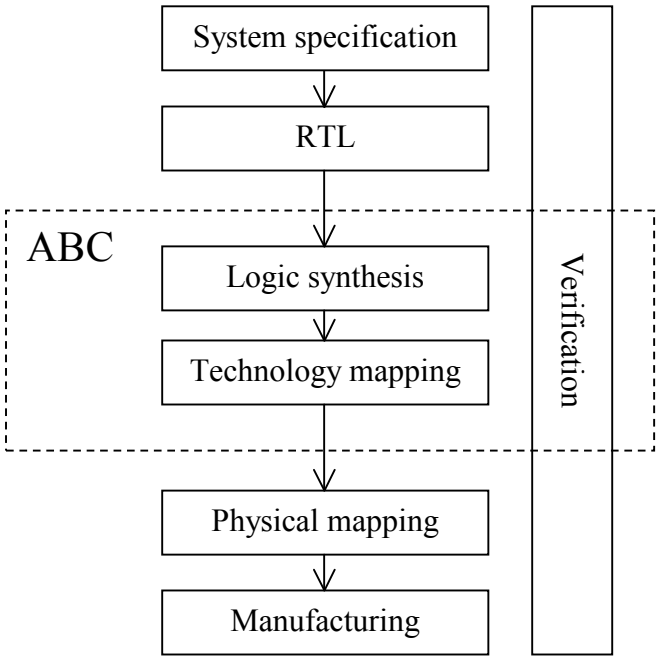
Celková funkcionalita původního konzolového programu bude zachována tak, že v nové aplikaci se ponechá možnost psaní příkazů, jako v příkazové řádce. To znamená, že pokud není potřebná funkce v menu, na panelu nástrojů, v historii nebo v seznamu příkazů, lze ji napsat přímo.

3. Analýza

3.1 ABC

ABC je softwarový systém pro syntézu a ověřování binárních sekvenčních logických obvodů. Jeho vývoj probíhá na kalifornské univerzitě v Berkeley. Projekt ABC vznikl s posláním vytvořit rychlou a škálovatelnou logickou syntézu. K celému projektu a ke zdrojovým kódům poskytuje univerzita volný přístup. Poslední veřejně přístupná verze programu ABC nabízí optimalizaci, mapování a retime návrhů logických obvodů se 100 000 hradly a 10 000 sekvenčních prvků na optimální zpoždění a minimalizaci prostoru během jediné minuty CPU času. Doba kombinační syntézy a verifikace je typicky rychlejší. Na projekt ABC se poslední dobou začíná upírat pozornost i průmyslového světa, např. firma Xilinx uvažuje o přijetí některých algoritmů použitých v ABC, kvůli jejich o několik řádů rychlejším provedení, než jiné přístupy.

Program ABC nabízí hned několik základních datových struktur pro reprezentaci a manipulaci s kombinačními a sekvenčními logickými obvody (netlist, logické sítě s uzly reprezentovanými jako SOP, nebo BDD, technology-mapped network, síť AIG, …). Formáty souborů, se kterými umí ABC pracovat jsou např. binární BLIF, PLA, formát BENCH, Synopsys equation format, Verilog a jiné.



Obrázek 3.1: Pole působnosti ABC

3.2 WinAPI

Jelikož projekt ABC je psán v jazyce C i nově vytvářená aplikace WinABC bude vyvíjena v tomto jazyce. Pro programování grafického uživatelského rozhraní jsem zvolil možnost přímého programování v „čistém“ WinAPI. Toto přináší několik výhod, ale i pár nevýhod. Nejdříve však popis, co to WinAPI je.

WinAPI (správný název je Windows API) je rozhraní pro programování aplikací určených pro operační systém Windows. Všechny programy v MS Windows musí nezávisle na použitém programovacím jazyce komunikovat prostřednictvím WinAPI, které obsahuje nejen základní funkce, ale i funkce pro vytváření uživatelského rozhraní.

Funkčnost WinAPI lze rozdělit do osmi kategorií.

- **Základní služby**: Poskytuje přístup k nezbytným zdrojům poskytnutých systémem Windows. Zahrnuje souborový systém, periferie, procesy a vlákna, přístup do registrů Windows a ošetření chyb.

- **Pokročilé služby**: Zahrnují API pro práci s registrem Windows, vypnutí/restart systému (nebo jeho zrušení), spuštění/zastavení/vytvoření služeb systému Windows a správu uživatelských účtů.

- **Grafické uživatelské rozhraní (GUI)**: Poskytuje funkce pro výstup grafického obsahu na monitory, tiskárny a jiná výstupní zařízení.

- **Uživatelské Rozhraní**: Poskytuje funkce pro tvorbu a řízení počítačových oken a dalších základních prvků jako jsou tlačítka a posuvníky, zpracovává vstup z klávesnice a myši a jiných funkcí spojených s GUI.

- **Knihovna běžných dialogových oken**: Poskytuje aplikacím standardní dialogová okna pro otevření a ukládání souborů, volbu barvy a fontů, apod. Celkově také patří do kategorie Uživatelské Rozhraní.

- **Knihovna běžných prvků (Common Control Library)**: Poskytuje aplikaci přístup k pokročilejším prvkům operačního systému. Zahrnuje věci jako stavový řádek, zobrazení průběhu výpočtu, toolbary a záložky. Celkově také patří do kategorie Uživatelské Rozhraní.

- **Windows Shell**: Umožňuje aplikacím přístup k funkcím poskytovaných shellem Windows. Celkově také patří do kategorie Uživatelské Rozhraní.

- **Síťové služby**: Poskytuje přístup k různým počítačovým sítím. Zahrnuje také NetBIOS, Winsock, NetDDE, RPC a mnoho dalších funkcí.

Microsoft vyvinul různé nadstavbové knihovny, které umožňují aplikacím pracovat s nízkoúrovňovými WinAPI funkcemi abstraktnějším způsobem. Microsoft Foundation Class Library (MFC) je nadstavba WinAPI pro třídy v C++, která usnadňuje objektově orientované programování při využívání WinAPI. Active Template Library (ATL) je šablonově orientovaná nadstavba pro COM. Windows Template Library (WTL) byla vyvinuta jako rozšíření pro ATL a zamýšlena jako odlehčená alternativa k MFC.

Další nadstavby vyvinula firma Borland: Object Windows Library (OWL) byla vydána jako konkurence k MFC a poskytuje podobnou objektově orientovanou nadstavbu. Nahradila ji Visual Component Library (VCL), která je napsána v Object Pascalu a dostupná jak pro Delphi, tak pro C++Builder.

Většina aplikačních frameworků pro Windows je (alespoň částečně) nadstavbou nad Windows API. Proto také .NET Framework a Java stejně jako ostatní programovací jazyky pod Windows jsou (nebo obsahují) nadstavbové knihovny.

Hlavní výhodou programu napsaném v „čistém“ WinAPI je nízká náročnost na výkon procesoru a velikost použité paměti. Další ocenitelnou výhodou je funkčnost takto vytvořené aplikace i na starších operačních systémech Windows. Na druhou stranu programování ve WinAPI je o něco složitější, než např. při použití nadstavbové knihovny MFC a napsaný kód je hůře čitelný.

4. Implementace

4.1 Úprava programu ABC

Aby bylo možné začít s vlastním programováním nové aplikace využívající grafické rozhraní operačního systému Windows, je nejdříve nutné provést změny v kódu programu původního projektu ABC. Zdrojové kódy ABC lze stáhnout na webových stránkách[1] univerzity v Berkeley. Program je napsán v jazyce C. Spolu se zdrojovými kódy jsou přiloženy i projekty pro Visual Studio 6.0. Jelikož Visual Studio 6.0 je již zastaralé vývojové prostředí staré více než 10 let, pokoušel jsem se tento projekt otevřít v poslední verzi Visual Studio 2008. Bohužel Visual Studio 2008 má se staršími projekty, které jsou psány v jazyce C, problémy, tedy se mi nepodařilo tyto projekty zkompilovat. Padlo tedy rozhodnutí zpracovat projekt ve Visual Studiu 6.0. Stažený projekt ABC nabízí možnost program zkompilovat buď do samostatného spustitelného souboru, nebo po drobných úpravách jako statickou knihovnu, použitelnou v libovolném vlastním projektu. Pomocí této knihovny je možné pracovat s logickém obvodem docela jednoduše. Problémem však zůstává to, že všechny textové výstupy jdou stále na standardní výstup, tedy na konzoli.

Mým cílem bylo pokud možno nezasahovat do původního projektu ABC, aby, pokud vyjde nová verze ABC, stačilo pouze zkompilovat statickou knihovnu a vložit ji do mého projektu WinABC. Tohle se bohužel ukázalo jako nereálné a bylo nutné provést v původním zdrojovém kódu úpravy. Aby byly tyto úpravy pokud možno nejmenší, navrhl jsem, že nejjednodušší řešení textových výstupů bude, když standardní výstup na konzoli změním na výstup do souboru. Po vykonání příkazu se tento soubor programem WinABC jednoduše načte a text dále zpracovává.

Úpravy ve zdrojovém kódu byly následující:

(1) Vytvoření globální proměnné `outStream` typu `FILE*`, která bude obsahovat výstupní proud, tedy soubor, do kterého se má zapisovat.
(2) Všechny příkazy '`printf(`' přepsat na '`fprintf(outStream,`'
(3) Všechny `stdout` a `stderr` přepsat na `outStream`
(4) Do souborů `mem.c`, `satSolver.h`, `vec.h`, `extra.h`, `util_hack.h`, `bar.h`, `cloud.h`, `satStore.h`, `msat.h`, `pr.h`, `espresso.h`, `abc.h` připsat deklaraci '`extern FILE * outStream;`'
(5) Dopsat do souboru `mainFrame.c` funkci `SetOutStream()`, která u zpracovávaného obvodu změní výstupní proudy na `outStream`

Tyto úpravy stačí k tomu, aby všechny výstupy, které směřovaly na konzoli, byly přesměrovány do souboru `outStream`. Proměnná `outStream` se definuje a nastavuje v programu WinABC. Pokud tedy eventuálně vyjde nová verze programu ABC, po

[1] `http://www.eecs.berkeley.edu/~alanmi/abc/`

aplikaci těchto změn bude s velkou pravděpodobností (pokud není nová verze ABC nějak dramaticky jiná) zkompilovaná knihovna stále použitelná v programu WinABC.

Protože původní projekt ABC je napsán v jazyce C, tak i tento nově vytvořený projekt WinABC musí být taktéž v jazyce C. Při pokusu začlenit vytvořenou statickou knihovnu do projektu v jazyce C++ hlásí totiž linker chybu.

4.2 Vytvoření knihovny ABC

Základní stavební prvek nově vytvářené aplikace je statická knihovna `abclib.lib`. Po aplikaci změn uvedených v bodě 4.1, které nám umožní přesměrovat textové výstupy z konzole do souboru, je potřeba knihovnu zkompilovat ve dvou verzích. Pro vývoj je potřeba knihovnu zkompilovat s informacemi pro ladění (verze známá jako Debug) a pro finální verzi je potřeba knihovna ve verzi Release. Obě verze ve VS 6.0 vytvoříme položkou v menu *Build -> Batch Build…*, kde následně zaškrtneme verze *Release* a *Debug* a stiskneme tlačítko *Rebuild All*. Tímto vytvoříme dvě knihovny `abclib_debug.lib` a `abclib_release.lib`, které se použijí v projektu WinABC.

Statická knihovna je na rozdíl od dynamické knihovny spojována linkerem v době sestavení programu. Výsledkem je tedy jeden spustitelný soubor (`*.exe`), který v sobě obsahuje používané metody z knihovny. Dynamická knihovna se naopak spojuje se spustitelným souborem až v době spuštění programu.

4.3 Příklad použití knihovny

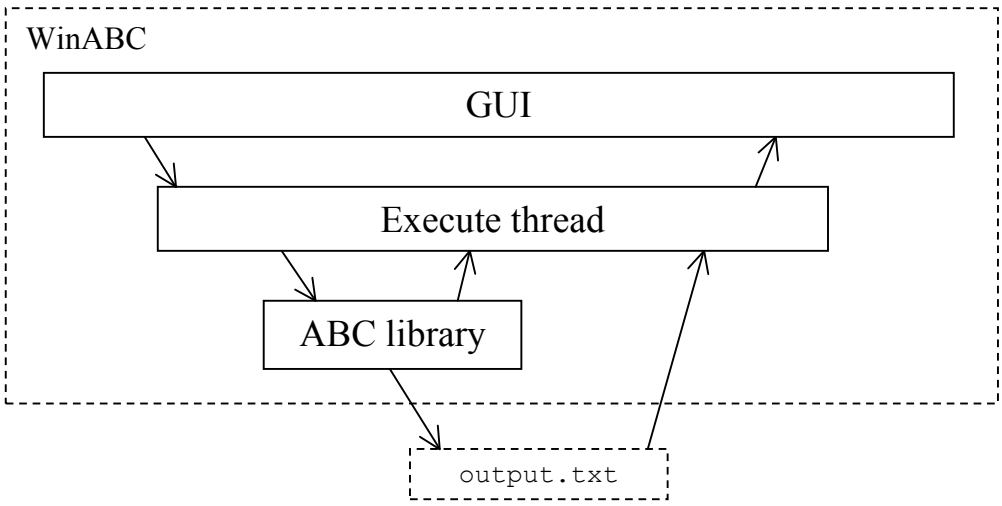
Následující kód obsahuje jednoduchou ukázku využití statické knihovny `abclib_debug.lib`. V ukázce se spustí příkaz *help* a výsledek příkazu se uloží do souboru `output.txt`. V ukázce není pro jednodušší názornost soubor `output.txt` načítán. Všechny použité metody ze statické knihovny mají u sebe komentář, který popisuje jejich základní funkci.

Ze startu programu je nutné zavolat metodu `Abc_Start()`, která alokuje potřebné struktury a proměnné. Dále musíme pomocí funkce `Abc_FrameGetGlobalFrame()` získat ukazatel na strukturu pro práci s logickým obvodem (uvnitř knihovny je to struktura typu `Abc_Frame_t`, navenek ji stačí deklarovat jako `void`). Nyní otevřeme soubor `output.txt` pro zápis a ukazatel na něho uložíme do proměnné `outStream`. Metoda `SetOutStream()` nastaví výstupní proudy na `outStream`, tedy na otevřený soubor `output.txt`. Nyní je již vše připraveno pro spuštění příkazu, které se provede metodou `Cmd_CommandExecute()`. Po vykonání příkazu už jen uzavřeme soubor `output.txt`.

```c
#include <stdio.h>
#pragma comment(lib, "abclib_debug.lib")

FILE * outStream;

/* Deklarace použitých metod ze statické knihovny */
extern void Abc_Start();
extern void Abc_Stop();
extern void * Abc_FrameGetGlobalFrame();
extern int Cmd_CommandExecute(void * pAbc, char * sCommand);
extern void SetOutStream(void * pAbc);

int main(int argc, char * argv[])
{
	void * pAbc;
	char command[1000];

	/* Alokuje potřebné struktury a proměnné */
	Abc_Start();

	/* Získá ukazatel na strukturu pro práci s logickým obvodem */
	pAbc = Abc_FrameGetGlobalFrame();

	outStream = fopen("output.txt", "w");
	if (outStream == NULL)
	{
		fprintf(stderr, "Cannot write to file!\n");
		return 1;
	}

	/* Nastaví stream pro textový výstup */
	SetOutStream(pAbc);

	sprintf(command, "help");

	/* Vykoná příkaz */
	if (Cmd_CommandExecute(pAbc, command))
	{
		fprintf(stderr, "Cannot execute command!\n");
		fclose(outStream);
		return 1;
	}

	fclose(outStream);

	/* Dealokuje použité struktury a proměnné */
	Abc_Stop();

	return 0;
}
```

4.4 Princip použití knihovny ve WinABC

Zadáním příkazu v grafickém rozhraní program vytvoří vlákno `ExecuteThread`, které zadaný příkaz spustí ve statické knihovně ABC. Ta se postará o vykonání příkazu a případný textový výstup zapíše do souboru `output.txt`. Zpět vláknu `ExecuteThread` vrátí pouze informaci o úspěšnosti provedení příkazu. Následně vlákno načte soubor `output.txt` a text v něm obsažený pošle do příslušné komponenty GUI. Toto načítání souboru lze zamezit spuštěním vlákna s příznakem `EX_NOLOAD`.



Obrázek 4.1: Princip použití knihovny ve WinABC

4.5 Moduly

Celý program je rozdělen do několika modulů (souborů se zdrojovým kódem). Následuje seznam modulů a jejich základní stručný popis.

- `DlgAbout.c` Obsluhuje dialog *About*.
- `DlgAdd.c` Obsluhuje dialog pro vložení/změnu příkazu v seznamu oblíbených příkazů.
- `DlgFavourite.c` Obsluhuje dialog seznamu oblíbených příkazů.
- `DlgOutput.c` Obsluhuje výstupní okno *Output*.
- `DlgSettings.c` Obsluhuje dialog *Settings*.
- `DlgWizard.c` Obsluhuje dialog *Command Wizard*.
- `Execute.c` Obsahuje metody pro spouštění příkazů ve statické knihovně ABC.
- `LoadSaveSettings.c` Obsahuje metody pro uložení/načtení nastavení do/ze souboru a pro načtení defaultního nastavení.

- `OpenSave.c` — Obsahuje metody pro vyvolání standardních dialogových oken pro otevření a uložení souboru.
- `OutputsSettings.c` — Obsluhuje dialog *Output settings*.
- `RefreshOutputs.c` — Obsahuje metody pro zobrazení výstupů.
- `WinABC.c` — Hlavní modul aplikace. Obsahuje vstupní metodu, metody pro inicializaci, vytváření GUI prvků hlavního okna a obsluhu hlavního okna.

4.6 Vlákna

Pro správné fungování programu je nutné využít vícevláknovou technologii (multithreading). Pokud bychom tuto technologii nepoužili, aplikace by po dobu výpočtu zcela „zamrzla“, tj. nereagovala by na podněty a nepřekreslovala by se. Toto je jistě velmi nežádoucí stav a u déle trvajících výpočtech by operační systém nabídl ukončení takového neodpovídajícího programu.

Kromě hlavního vlákna se v programu používají v různých okamžicích další tři, která volají funkce ze statické knihovny ABC. Tyto vlákna jsou:

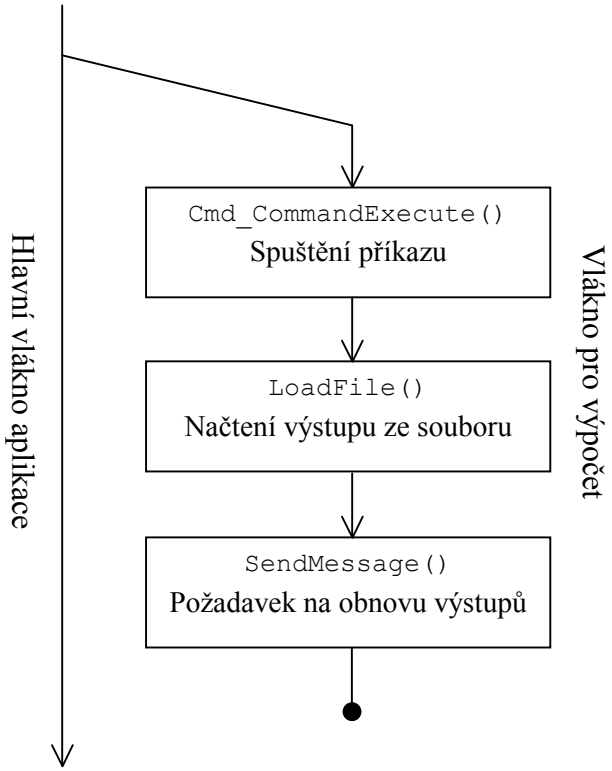
- Vlákno pro inicializaci ABC frameworku.
- Vlákno pro provedení výpočtu
- Vlákno pro obnovu výstupů

Vlákno pro inicializaci ABC frameworku se spouští vždy a pouze při startu aplikace. Jeho úkolem je získat z knihovny ABC strukturu pro práci s logickým obvodem a popřípadě (pokud je to nastaveno v nastavení) vykonat startovní skript `abc.rc`. Tato inicializace sice trvá jen pár desetin sekund, ale na starších počítačích může dosahovat i doby řádově v sekundách a pokud by inicializace nebyla v samostatném vlákně, „zamrznutí“ aplikace by bylo postřehnutelné. Vlastní tělo vlákna je umístěno ve funkci `AbcInit()`.

Vlákno pro provedení výpočtu se vytvoří vždy, když zadáme nějaký příkaz, ať už ručně do textového pole, nebo nepřímo, např. při otevření souboru. Toto vlákno má na starosti nastavení časovače (timeru), který se používá pro průběžné zobrazení uplynulé doby výpočtu, spuštění zadaného příkazu v knihovně ABC a následné zobrazení přesné doby trvání výpočtu. Dalo by se říci, že toto je ze všech tří vláken vlákno využívané nejdelší dobu. Díky tomuto vláknu aplikace během provádění výpočtu „nezamrzne“ a uživatel tak může díky tomu vidět uplynulý čas výpočtu, nebo např. upravovat seznam oblíbených příkazů. Po provedení výpočtu se z vlákna zašle hlavnímu vláknu požadavek na obnovu výstupů. Vlastní tělo vlákna je umístěno ve funkci `ExecuteThread()`. Vytvoření vlákna provede funkce `StartExecute()`.

Poslední z této trojice, vlákno pro obnovu výstupů, slouží ke spuštění příkazů pro zobrazení výstupů. Vlákno spustí všechny zaškrtnuté příkazy a jejich výstupy zobrazí

v odpovídajícím textovém poli. Vlastní tělo vlákna je umístěno ve funkci `PrintOutputs()`. Vytvoření vlákna provede funkce `StartPrintOutputs()`.



Obrázek 4.2: Znázornění vlákna pro provedení výpočtu

4.7 Hlavičkový soubor WinABC.h

Soubor `WinABC.h` obsahuje definici některých konstant. Jedná se o `MAX_OUTPUTS`, která definuje maximální počet výstupů, `MAX_FAVOURITES`, která definuje maximální počet oblíbených příkazů, `COMMAND_SIZE`, která definuje maximální délku příkazu včetně ukončující nuly, `OUTPUT_FILE_NAME`, která definuje název výstupního souboru, `SETTINGS_FILE_NAME`, která definuje název souboru s nastavením programu a `SETTINGS_FILE_VERSION`, která definuje číslo verze souboru s nastavením.

4.8 Nastavení

Veškeré nastavení, které souvisí s aplikací WinABC se ukládá do souboru `settings.dat`, který je uložen ve stejné složce, jako samotný program. Název souboru je definován v hlavičkovém souboru `WinABC.h` jako `SETTINGS_FILE_NAME` a může být tak

před kompilací programu snadno změněn. Tento soubor je načítán při každém spuštění aplikace. Pokud soubor `settings.dat` neexistuje, program jej vytvoří s defaultním nastavením. Nastavení se zapisuje do souboru při změně nastavení a při ukončení programu.

Počet	Popis	Datový typ	Velikost [B]
1 ×	Verze souboru	BYTE	1
1 ×	Velikost a pozice okna	WINDOWPLACEMENT	44
1 ×	Popisky v nástrojové liště	BOOL	4
1 ×	Čas ve stavovém řádku	BOOL	4
1 ×	Načítat `abc.rc`	BOOL	4
1 ×	Potvrzovat ukončení aplikace	BOOL	4
1 ×	Písmo	LOGFONT	60
1 ×	Zalamovat řádky	BOOL	4
1 ×	Počet výstupů = *m*	int	4
m ×	Délka textu příkazu = *d*	WORD	2
	Příkaz výstupu	char[*d*]	*d*
	Zaškrtnuto	BOOL	4
1 ×	Počet oblíbených příkazů = *n*	int	4
n ×	Délka textu příkazu = *d*	WORD	2
	Oblíbený příkaz	char[*d*]	*d*

Tabulka 4.1: Struktura souboru „settings.dat“

4.9 Zobrazení obvodu ve formátu…

Uživatel má možnost nechat si zobrazit rozpracovaný logický obvod v různých formátech, aniž by musel tento obvod ukládat do souboru příslušného formátu. Tato nabídka zobrazení je v hlavním menu pod položkou *Show -> Show net in format*. Na výběr je zde 11 různých formátů. Po výběru libovolného formátu se vytvoří v adresáři se spuštěným programem soubor `temp.tmp`, do kterého se zapíše textový řetězec „*This format is not available now. See instructions in output window!”*. Pokud se následné uložení do vybraného formátu nezdaří, bude tato uložená zpráva uživatele informovat o nastalém problému. Po uložení je soubor `temp.tmp` otevřen pomocí Poznámkového bloku operačního systému Windows. Zde je však na místě upozornit, že při práci s rozsáhlými logickými obvody může soubor `temp.tmp` dosáhnout velikosti několik desítek, či stovek MB. Načítání takového souboru Poznámkovým blokem pak může být velmi pomalé!

4.10 Doporučená konfigurace počítače

Minimální konfigurace počítače pro spuštění WinABC je na dnešní dobu velmi nízká. Grafické uživatelské rozhraní funguje svižně i na počítačích starých 10 let.

Aplikace WinABC byla testována v operačních systémech Windows 98 a Windows XP. Protože z WinAPI byly použity jen základní funkce, je předpoklad, že i ve verzi Windows 95 by neměly být žádné problémy.

Nejnáročnější část programu na výkon jsou výpočty se zpracovávaným obvodem. Z toho vyplývá požadavek na rychlost procesoru a velikost operační paměti. Jako vhodnou konfiguraci PC lze doporučit dnešní běžný standard, což je procesor 2 GHz, 1 GB RAM a Windows XP.

	Minimální	**Doporučená**
Operační systém	Windows 95	Windows XP
Procesor	75 MHz	2 GHz
Operační paměť	16 MB	1 GB

Tabulka 4.2: Minimální a doporučená konfigurace

5. Popis a ovládání aplikace

Tato kapitola popisuje vzhled aplikace a její ovládání. Seznámí uživatele se způsobem práce v programu, popisuje posloupnost jednotlivých kroků a rozebere jednotlivě všechna dialogová okna, která se mohou při práci objevit.

Celé grafické uživatelské rozhraní je však navrženo tak, aby bylo intuitivní a pro uživatele jiných programů operačního systému Windows nebyl problém si ovládání této aplikace osvojit.

5.1 Hlavní okno aplikace

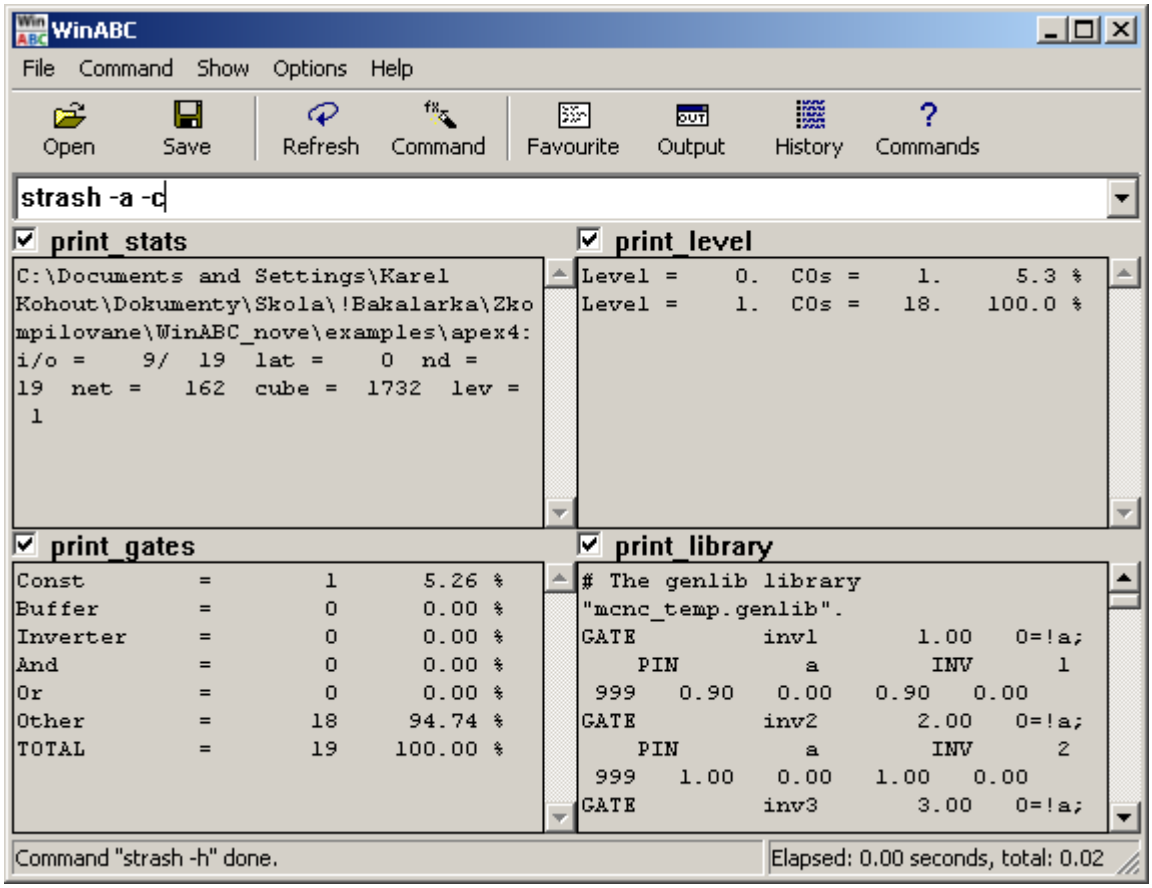
Po spuštění aplikace WinABC se zobrazí hlavní okno, které se skládá z pěti základních částí.

- Hlavní nabídka (menu)
- Nástrojová lišta (toolbar)
- Textové pole pro zadávání příkazů (combo box)
- Pole pro zobrazování informací o obvodu (pole výstupů)
- Stavový řádek (statusbar)

Pomocí hlavní nabídky (menu) se uživatel dostane ke všem příkazům a nastavení, které program nabízí. Jedná se zejména o možnost načtení a uložení logického obvodu ze respektive do souboru, možnost ukončení aplikace, zobrazení nemodálních oken *Favourite commands* a *Output*, zobrazení dialogu nastavení a příkazy pro práci s obvodem.

Nástrojová lišta slouží k rychlému vyvolání nejčastějších a nejpoužívanějších příkazů, jako je otevření a uložení souboru, znovuzobrazení (obnova) výstupů a jiné.

Textové pole pro zadávání příkazů je ve skutečnosti tvořeno combo boxem (kombinace jednoduchého edit boxu a list boxu), tudíž na pravé straně je umístěno tlačítko, které zobrazí seznam již zadaných příkazů v pořadí od nejnovějšího po nejstarší. Po napsání a potvrzení příkazu se stane toto textové pole neaktivní a nelze měnit jeho obsah. Neaktivní zůstane po celou dobu vykonávání příkazu. Po skončení příkazu se textové pole změní opět na aktivní. Pokud byl příkaz vykonán úspěšně, vymaže se obsah textového pole a tím je i připraveno pro zadání dalšího příkazu. Byl-li však příkaz neúspěšný (např. neznámý příkaz), původní obsah textového pole v něm zůstane nadále. Toto je například užitečné, když uživatel v příkazu udělá překlep a program vyhodnotí příkaz jako neznámý. Nemusí tedy opět vypisovat celý příkaz znova, nebo jej hledat v historii příkazů.



Obrázek 5.1: Hlavní okno aplikace

Pole pro zobrazování informací o obvodu je největší a nejdůležitější část hlavního okna. Skládá se z proměnného počtu jednotlivých výstupů. Tento počet výstupů lze nastavit v nastavení programu a je omezen minimálním počtem 1 a maximálním 20. Maximum lze snadno nastavit při kompilaci programu v hlavičkovém souboru `WinABC.h`. 20 je však plně dostačující počet. Jednotlivý výstup je tvořen check boxem s popisem výstupu (příkazem, který jej vykoná) a víceřádkovým text boxem. Pokud check box není zaškrtnut, nebude při následujícím zobrazování výstupů tento výstup proveden a přidružený text box se stane neaktivní (zašedlý text). Neaktivní výstup (jeho text je zašedlý) tedy vyjadřuje, že informace v něm nemusí být aktuální, protože při posledním zobrazení výstupů se tento výstup neobnovil. Toto vypínání obnovy výstupů má smysl z toho důvodu, protože některé algoritmy pro výpis informací o logickém obvodu můžou v rozsáhlých obvodech trvat i několik minut. Obnova (refresh) výstupů se provádí po každé úspěšně vykonaném příkazu, je tedy zřejmé, že čekat po každém příkazu několik minut na zobrazení výstupů by bylo velmi nežádoucí. Proto má uživatel možnost takovéto náročnější výstupy vypnout a znovu je zapnout až na požádání. Obnova výstupů se provádí v pořadí zleva doprava a shora dolů. V nastavení lze nastavit, zda chceme mít u výstupů horizontální posuvník nebo se má text automaticky zalamovat. Například pro zobrazení Karnaughovy mapy je vhodnější použit posuvník, protože automatické

zalamování textu může úplně rozhodit zobrazení mapy a ta tak bude těžko čitelná. Naopak pro jiné výstupy může být vhodnější zalamování textu, neboť tak uživatel uvidí bez posouvání více informací.

Poslední částí hlavního okna je stavový řádek. Ten podle nastavení je rozdělen buď na jednu, nebo dvě části. Levá část zobrazuje informace o právě probíhajícím výpočtu, nebo o připravenosti na další příkaz. Pokud je v nastavení povoleno zobrazování času výpočtu, obsahuje stavový řádek i informace o uplynulé době výpočtu a celkovém výpočetním času. Při vykonávání příkazu je zobrazován orientační čas v celých sekundách a po skončení je zobrazen uplynulý čas v sekundách s přesností na dvě desetinná místa.

5.2 Inicializace

Po spuštění aplikace a zobrazení hlavního okna se provede inicializace programu. Během této doby je ve stavovém řádku zobrazen text *Initializing ABC framework…* a není možné zadávat příkazy. Tato inicializace trvá na dnešních počítačích jen pár desetin sekund a tedy uživatele nijak časově neomezuje. Jen na starších počítačích (procesor PII 300 MHz a níže) může inicializace trvat řádově v sekundách. Po inicializaci se načte soubor `abc.rc`, který obsahuje některá nastavení pro ABC a aliasy (nahrazení dlouhého příkazu kratším názvem a nahrazení sekvence několika příkazů příkazem jediným). Načítání tohoto souboru při spuštění lze vypnout v nastavení. Dále je provedeno obnovení výstupů, do kterých se ve většině případů vypíše *Empty network*. Nyní je již aplikace připravena k použití.

5.3 Načtení souboru

Po výběru položky z menu *File -> Open…*, nebo tlačítka *Open* na nástrojové liště, se otevře standardní dialogové okno pro otevření souboru. Uživatel má možnost filtrovat zobrazené soubory podle typu. Nabízené možnosti jsou buď zobrazit všechny typy souborů podporované aplikací *ABC input files (*.aig; *.baf; *.bench; *.blif; *.cnf; *.dot; *.eqn; *.gml; *.pla; *.v)*, nebo vyfiltrovat soubory podle jednotlivých typů. Ve filtru lze vybrat i možnost *All files*, ale je nutné upozornit, že program rozpoznává formát souboru podle přípony, takže pokud má soubor jinou příponu, než standardní, nepodaří se ho otevřít.

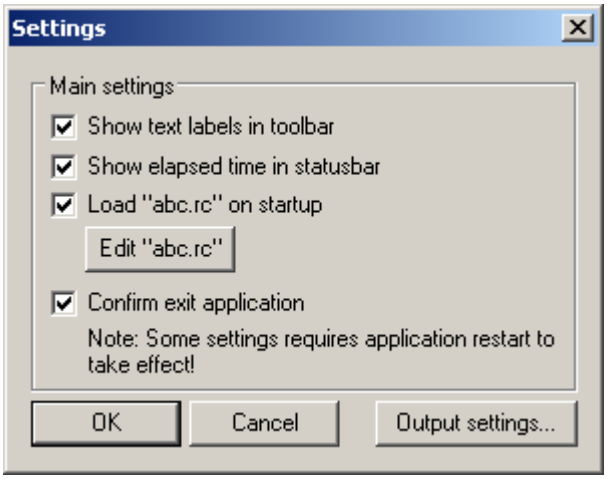
5.4 Uložení souboru

Výběrem položky *File -> Save…*, nebo tlačítka *Save* na nástrojové liště, se otevře standardní dialogové okno pro uložení souboru. V tomto dialogovém okně má uživatel možnost si vybrat formát souboru, ve kterém bude zpracovávaný obvod uložen. Na výběr

je deset různých formátů *(*.aig; *.baf; *.bench; *.blif; *.cnf; *.dot; *.eqn; *.gml; *.pla; *.v)*. Pokud obvod nelze uložit do vybraného formátu, program ukládání přeruší a do okna *Output* vypíše důvod, proč soubor nelze uložit.

5.5 Nastavení

K základnímu nastavení aplikace se uživatel dostane pomocí nabídky hlavní nabídky menu. Nastavení se nachází pod položkou *Options -> Settings*…. Po výběru této položky se otevře dialogové okno *Settings*.



Obrázek 5.2: Nastavení aplikace

Zaškrtávací tlačítko (check box) *Show text labels in toolbar* slouží k zobrazení, respektive skrytí textových popisků tlačítek na panelu nástrojů. Pro začínající uživatele aplikace je jistě vhodnější, když jsou tlačítka i s textovým popisem a tak rozpoznat funkci tlačítka je jednodušší. Na druhou stranu, pokud si uživatel časem zapamatuje jednotlivé ikony tlačítek, může tímto nastavením vypnout textové popisky, takže tlačítka budou menší, zaberou méně místa a pro důležitější pole výstupů zbude větší prostor v hlavním okně. Jelikož panel nástrojů se vytváří při startu aplikace, tak změna tohoto nastavení se projeví až opětovným spuštěním programu.

Položka *Show elapsed time in status bar* slouží k zobrazení a skrytí uplynulého času výpočtu ve stavovém řádku. Pokud je položka zaškrtnuta, rozdělí se stavový řádek na dvě části, kde jeho pravá část slouží právě pro toto zobrazení času. Během výpočtu se zobrazuje uplynulý čas tohoto výpočtu a po dokončení se zobrazí navíc i celkový uplynulý čas všech výpočtů od spuštění programu. Pokud zrovna program provádí výpočet se zpracovávaným obvodem, není možné toto nastavení měnit a check box *Show elapsed time in status bar* se stane neaktivní. I když je zobrazení času vypnuto, program stále počítá celkový uplynulý čas výpočtů, jen tato informace není zobrazena.

Další volba v nastavení je položka *Load "abc.rc" on startup*. Pokud je tato volba zaškrtnuta, program při inicializaci načte a spustí skript uložený v souboru `abc.rc`, který se nachází ve stejném adresáři, jako spuštěná aplikace. Do tohoto skriptu lze uložit jakýkoliv příkaz, který lze zadat do pole zadávání příkazů (např. tak lze nastavit automatické načtení uloženého obvodu při startu), ale hlavní účel skriptu `abc.rc` je spíše pro nastavení ABC frameworku a nastavení aliasů příkazů. Pokud je toto načítání nastaveno, ale soubor `abc.rc` neexistuje, je načtení skriptu přeskočeno a program pokračuje dál.

Tlačítko *Edit "abc.rc"* má přímou souvislost s posledně uvedeným check boxem. Toto tlačítko slouží k editaci už zmíněného skriptu `abc.rc`. Program otevře tento skript v Poznámkovém bloku (`notepad.exe`) operačního systému Windows. Uživatel si tak může jednoduše skript prohlédnout, nebo ho i editovat. Pokud soubor neexistuje, Poznámkový blok na to uživatele upozorní a nabídne mu možnost tento soubor automaticky vytvořit.

Zaškrtávací tlačítko *Confirm exit application* slouží k vypnutí potvrzovacího dialogu při ukončení aplikace. Možnost zapnout potvrzování ukončení je zde z toho důvodu, protože omylem ukončený program má za následek ztrátu rozpracovaného obvodu.

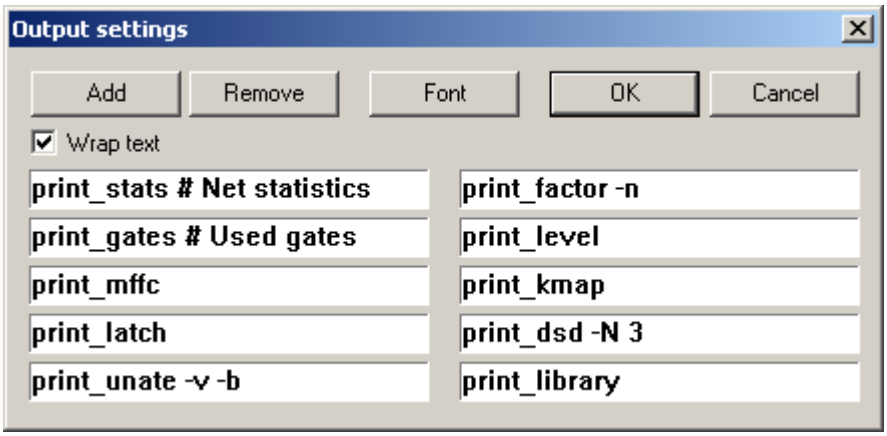
Pomocí tlačítka *Output settings…* se uživatel dostane k nastavení výstupů informací o zpracovávaném obvodu. Toto dialogové okno je popsáno dále v textu.

Tlačítka *OK* a *Cancel* slouží standardně k zavření dialogového okna *Settings* buď s uložením změn, respektive bez uložení. Tento výběr však neplatí pro editaci souboru `abc.rc` (tam záleží na uložení souboru Poznámkovým blokem) a nastavení výstupů pomocí tlačítka *Output settings…*. Tam se jedná o samostatné dialogové okno.

5.6 Nastavení výstupů

Pro nastavení výstupů informací o zpracovávaném obvodu, které jsou zobrazeny v hlavním okně aplikace a zabírají jeho největší část, slouží dialogové okno *Output settings*. K tomuto nastavení se lze dostat buď pomocí hlavní nabídky pod položkou *Options -> Output settings…*, nebo z dialogového okna *Settings* pomocí tlačítka *Output settings…*. Příkazů pro výpis informací, které aplikace nabízí, je kolem dvaceti, proto počítat a zobrazovat všechny tyto informace je časově a prostorově náročné, ale i zbytečné. Proto si uživatel vybere, jaké chce zobrazovat informace, podle svého uvážení. Horní část dialogového okna obsahuje ovládací tlačítka a spodní část text boxy, které odpovídají jednotlivým výstupům v hlavním okně. Počet těchto text boxů je proměnný, minimálně je však vždy jeden a maximálně dvacet. Do těchto text boxů se vypisují příkazy, které slouží pro výpis informací. Tyto příkazy lze nalézt v seznamu příkazů v kategorii *Printing commands*. Lze zde však vyplnit i příkazy z jiných kategorií. To může být výhodné, když chceme po každém námi ručně napsaném příkazu spouštět

automaticky jiné příkazy. Například pokud napíšeme do jednoho výstupu příkaz *write_blif network.blif*, program uloží po každém vykonaném příkazu zpracovávaný obvod do souboru `network.blif`. Protože program ignoruje v příkazu znaky, které následují po # (mřížka), lze toho využít k psaní popisků a komentářů.



Obrázek 5.3: Nastavení výstupů

Tlačítko *Add* přidá další výstup (text box). Přidá jej za výstup, ve kterém se nachází kurzor. Pokud je dosažen maximální limit počtu výstupů, tlačítko *Add* se stane neaktivní.

Tlačítko *Remove* slouží k odebrání (smazání) výstupu, ve kterém se zrovna nachází kurzor. Pokud je dosažen minimální limit počtu výstupů, tlačítko *Remove* se stane neaktivní.

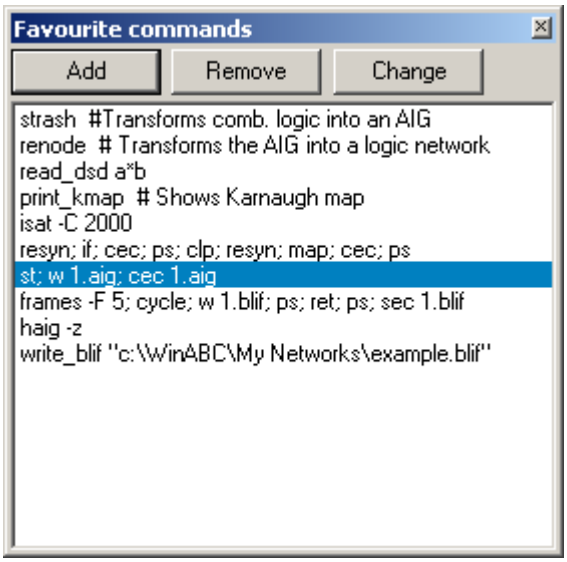
Tlačítko *Font* slouží k otevření dialogového okna výběru typu a velikosti písma. Toto dialogové okno je ze standardních systémových dialogů operačního systému Windows. Písmo vybrané tímto dialogovým oknem se bude zobrazovat ve výstupech. Doporučuje se tedy vybrat písmo, které je neproporcionální a má tedy stejnou šířku všech znaků. Vhodná písma jsou např. *Courier*, *Courier New*, *Fixedsys* a *Terminal*. Pokud bychom vybrali proporcionální písmo, zobrazení např. Karnaughovy mapy by bylo nesprávné a nečitelné.

Tlačítka *OK* a *Cancel* slouží standardně k zavření dialogového okna *Output Settings* buď s uložením změn, respektive bez uložení.

Zaškrtávací tlačítko *Wrap text* slouží s zalomení textu ve výstupních text boxech. Pokud je volba zaškrtnuta, text box má pouze vertikální posuvník a text je automaticky zalamován. Pokud není volba zaškrtnuta, má text box navíc i horizontální posuvník a text je zobrazován tak, jak jej program generuje. Například pro zobrazení Karnaughovy mapy je vhodnější použit posuvník, protože automatické zalamování textu může úplně rozhodit zobrazení mapy a ta tak bude těžko čitelná. Naopak pro jiné výstupy může být vhodnější zalamování textu, neboť tak uživatel uvidí bez posouvání více informací.

5.7 Seznam oblíbených příkazů

Aplikace umožňuje uživateli vytvořit si seznam oblíbených (častých) příkazů. Protože některé příkazy mohou i s parametry být velmi dlouhé a v jednom příkazu může být obsaženo i příkazů několik, může tento seznam oblíbených příkazů uživateli velmi urychlit práci. Tento seznam příkazů se ukládá spolu s jinými nastaveními aplikace do souboru `settings.dat` a je tedy načten i při dalším spuštění aplikace. Tento seznam příkazů se zobrazí buď pomocí hlavní nabídky pod položkou *Show -> Favourite commands*, nebo pomocí přepínacího tlačítka *Favourite* na nástrojové liště hlavního okna. Po zapnutí se zobrazí okno *Favourite commands*, které je nemodální a umožňuje tedy dále pokračovat v práci s aplikací, i když je okno zobrazeno. Skrytí okna se provede buď kliknutím na křížek v pravém horním rohu okna, nebo pomocí přepínacího tlačítka *Favourite* na nástrojové liště hlavního okna.



Obrázek 5.4: Seznam oblíbených příkazů

Tlačítko *Add* slouží k přidání dalšího příkazu do seznamu. Po stisknutí se zobrazí dialogové okno, ve kterém se do text boxu vepíše požadovaný příkaz. Po stisknutí tlačítka *OK*, nebo stisknutí klávesy *<Enter>* se vytvoří nová položka v seznamu. Pokud je dosažen maximální limit počtu možných položek v seznamu (limit je nastaven na 1000 položek), tlačítko *Add* se stane neaktivní. Tak jako v případě nastavení výstupů, i zde můžeme využít psaní popisků a komentářů k příkazu pomocí znaku # (mřížka).

Obrázek 5.5: Přidání/editace položky

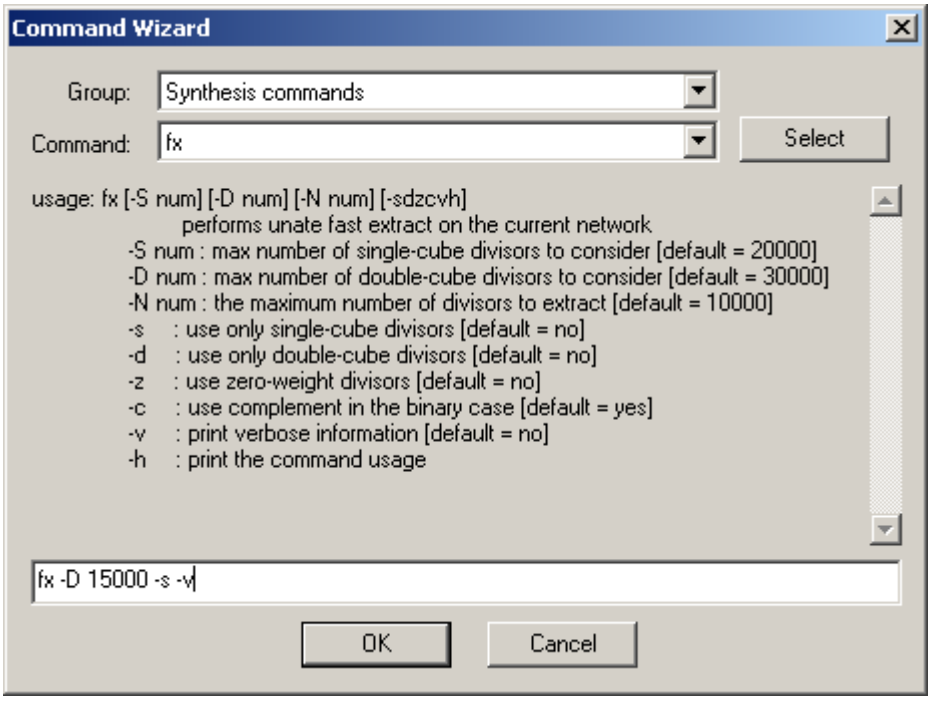
Tlačítko *Remove* odstraní právě vybranou položku ze seznamu.

Tlačítko *Change* slouží k editaci právě vybrané položky seznamu. Po stisknutí se zobrazí stejné dialogové okno, jako v případě přidávání nové položky a do text boxu je načten editovaný příkaz. Po stisknutí tlačítka *OK*, nebo stisknutí klávesy <*Enter*> se změní obsah položky v seznamu.

Posledním objektem na dialogovém okně *Favourite commands* je seznam příkazů. Kliknutím na příkaz v tomto seznamu se příkaz přenese do textového pole pro zadávání příkazů na hlavním okně, kde můžeme dodatečně příkaz upravit a poté spustit klávesou <*Enter*>. Pokud na položku v seznamu poklepáme (dvojklik myší), příkaz se spustí okamžitě. Během provádění výpočtu samozřejmě nelze spustit další příkaz. Pokud se o to pokusíme, program nás na to upozorní. Seznam však můžeme měnit během této doby bez omezení (přidání, odebrání, změna položky).

5.8 Command Wizard

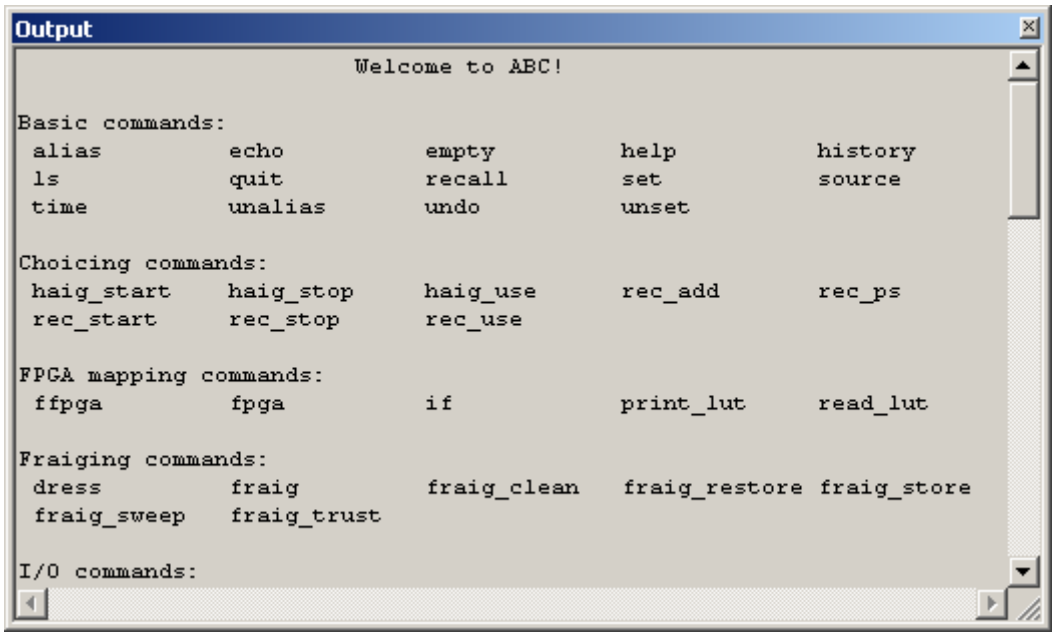
Po výběru položky *Command -> Command Wizard...* z hlavního menu, nebo stisknutím tlačítka *Command* na panelu nástrojů se zobrazí okno, ve kterém si uživatel může prohlížet nápovědu pro všechny příkazy a selektovat příkazy podle kategorií. Ve spodním textovém poli může k příkazu podle nápovědy dopsat požadované parametry a příkaz spustit.



Obrázek 5.6: Command Wizard

5.9 Výstupní okno příkazu

I některé příkazy, které nejsou určeny přednostně pro vypisování informací o obvodu, mohou vyprodukovat nějaký textový výstup. V tomto případě je tedy nutné mít ještě jedno okno, které bude sloužit právě pro zobrazení výstupu zadaného příkazu. Toto okno je v aplikaci nazváno *Output* a je nemodální, podobně jako okno se seznamem oblíbených příkazů. Okno *Output* bývá standardně skryto a zobrazí se automaticky, pokud příkaz vygeneruje nějaký textový výstup. Zobrazit ho lze i ručně pomocí hlavní nabídky pod položkou *Show -> Show output window*, nebo pomocí přepínacího tlačítka *Output* na nástrojové liště hlavního okna. Toto okno se využívá i k zobrazení historie příkazů pomocí tlačítka *History* a k zobrazení seznamu všech příkazů pomocí tlačítka *Commands*.



Obrázek 5.7: Okno pro zobrazení textového výstupu příkazu

6. Testování aplikace

V této kapitole je popsáno testování vytvořené aplikace. Aplikace je porovnávána s původní konzolovou verzí ABC. Základní sledované parametry testování jsou rychlost výpočtu a přesnost výpočtu (zda-li je výsledný logický obvod v obou aplikacích totožný). Testy jsou prováděny několikanásobně a na více počítačích různých konfigurací. Základní požadavky, které zhodnotí testování jako úspěšné, jsou tyto: Programy musí pracovat s obvodem stejně a rychlost zpracování nové aplikace WinABC nesmí být výrazně pomalejší.

K testování je použit skript `regtest.script`, který pracuje s obvody uloženými v adresáři `examples/`. Skript obsahuje příkazy z různých kategorií. Skript se v programu spustí příkazem `source regtest.script`. Obsah souboru `regtest.script` je následující:

```
r examples/apex4.pla;     resyn; if; cec; ps; clp; resyn; map; cec; ps
r examples/C2670.blif;    st; w 1.aig; cec 1.aig
r examples/C2670.blif;    st; short_names; w 1.bench; cec 1.bench
r examples/C2670.blif;    st; short_names; ren -s; w 1.eqn; cec 1.eqn
r examples/C2670.blif;    resyn2; if -K 8; cec; ps; u; map; cec; ps
r examples/frg2.blif;     dsd; muxes; cec; ps; clp; share; resyn; map; cec; ps
r examples/frg2.blif;     bdd; muxes; cec; ps; clp; st; ren -b; muxes; cec; ps
r examples/i10.blif;      resyn2; fpga; cec; ps;  u; map; cec; ps
r examples/i10.blif;      choice; fpga; cec; ps;  u; map; cec; ps
r examples/pj1.blif;      st; if; cec; ps;   u; map; cec; ps
r examples/s38417.blif;   comb; w 1.blif; resyn; if; cec 1.blif; ps
r examples/s38417.blif;   resyn; if; cec; ps;   u; map; cec; ps
r examples/s38584.bench; resyn; ren -s; fx; if; cec; ps;   u; map; cec; ps
r examples/s444.blif;     b; esd -v; print_exdc; dsd; cec; ps
r examples/s444.blif;     double; frames -F 5; w 1.blif; ffpga -K 8; cec 1.blif
r examples/s5378.blif;    frames -F 5; cycle; w 1.blif; ps; ret; ps; sec 1.blif
r examples/s6669.blif;    cycle; w 1.blif; ps; ret -M 3; resyn; ps; sec 1.blif
time
```

6.1 Testování přesnosti výpočtu

Po vykonání výše uvedeného skriptu se v adresáři se spuštěným programem (buď ABC nebo WinABC) vytvoří 6 souborů s uloženými zpracovanými obvody (`1.blif`, `1.bench`, `1.eqn`, `1.aig`, `mcnc_temp.genlib_temp` a `mcnc_temp.super`). Porovnáním těchto vytvořených souborů z obou aplikací (ABC a WinABC) můžeme zjistit, zda jsou spolu odpovídající soubory shodné a tedy obě aplikace zpracovaly logický obvod stejně.

Z různých měření na různých konfiguracích počítače vyšlo najevo, že soubory `mcnc_temp.genlib_temp` jsou naprosto shodné. U ostatních souborů je jejich obsahem i datum a čas uložení. Tyto data v souboru samozřejmě z pochopitelných důvodů nebudeme brát v potaz, neboť aplikace byly spuštěny v různou dobu. Zbytek souboru je již však shodný. Proto můžeme konstatovat, že obě aplikace zpracovaly obvody stejně a je tedy splněn první požadavek úspěšnosti testů.

6.2 Testování rychlosti výpočtu

Pro testování rychlosti výpočtu byly použity testy dva. V prvním testu byl použit stejný skript jako v bodě 6.1. Pro obě aplikace (ABC a WinABC) byly nastaveny stejné podmínky, tj. v aplikaci WinABC bylo vypnuto zobrazování předdefinovaných výstupů, aby obě aplikace vykonaly stejnou sekvenci příkazů. Testování proběhlo na dvou konfiguracích PC, v počtu 5 měření na konfiguraci. Z těchto pěti měření byl vypočítána průměrná doba výpočtu pro každou aplikaci.

Druhý test rychlosti výpočtu byl zvolen tak, aby výpočet negeneroval žádný textový výstup, neboť ten se u obou aplikací liší. Algoritmus výpočtu v tomto testu by měl v obou aplikacích provádět stejné kroky. Pro test byl použit soubor `pj1.blif` z adresáře `examples/`. Na tomto souboru byl proveden příkaz *strash* a následně *renode*. A právě doba výpočtu příkazu *renode* je použita pro porovnání obou aplikací. Dobu trvání výpočtu lze zjistit pomocí příkazu *time*. Testování proběhlo v počtu 5 měření. Z těchto pěti měření byl vypočítána průměrná doba výpočtu pro každou aplikaci.

6.2.1 Test I

Konfigurace I

Procesor: AMD Athlon 64 X2 3800+ / 2 GHz
Operační paměť: 2 GB DDR2-SDRAM / 667 MHz
Operační systém: Windows XP SP3

Měření	ABC	WinABC
1.	*33,52 s*	*30,70 s*
2.	*33,42 s*	*30,75 s*
3.	*33,61 s*	*30,72 s*
4.	*33,49 s*	*30,92 s*
5.	*33,39 s*	*30,58 s*
Průměr	***33,49 s***	***30,73 s***

Tabulka 6.1: Výsledky měření testu I pro konfiguraci I

Konfigurace II

Procesor: Intel PII Celeron / 500 MHz
Operační paměť: 320 MB RAM
Operační systém: Windows 98 SE

Měření	ABC	WinABC
1.	667,42	658,46
2.	670,66	661,11
3.	668,96	665,78
4.	669,24	654,59
5.	665,10	656,93
Průměr	**668,28**	**659,37**

Tabulka 6.2: Výsledky měření testu I pro konfiguraci II

Z měření vyplynulo, že nově vytvořená aplikace WinABC je překvapivě rychlejší, než její konzolový předchůdce ABC. Testovací skript vykonala v konfiguraci I o *8,22 %* rychleji a v konfiguraci II o *1,33 %* rychleji. Toto zrychlení by se dalo vysvětlit tím, že aplikace WinABC zobrazí výsledek skriptu až po jeho skončení, zatímco aplikace ABC vypisuje výsledky průběžně. Proto pro objektivnější výsledek testu rychlosti výpočtu je určen Test II.

6.2.2 Test II

Konfigurace

Procesor: AMD Athlon 64 X2 3800+ / 2 GHz
Operační paměť: 2 GB DDR2-SDRAM / 667 MHz
Operační systém: Windows XP SP3

Měření	ABC	WinABC
1.	*6,94 s*	*6,83 s*
2.	*6,92 s*	*6,84 s*
3.	*6,91 s*	*6,81 s*
4.	*6,94 s*	*6,86 s*
5.	*6,91 s*	*6,88 s*
Průměr	***6,92 s***	***6,84 s***

Tabulka 6.3: Výsledky měření testu II

Výsledek tohoto testu ukázal, že aplikace WinABC je při provádění příkazu *renode* o *1,16 %* rychlejší, než její předchůdce ABC. Druhá podmínka úspěšnosti testů byla tedy splněna.

7. Závěr

Úkolem této bakalářské práce bylo vytvořit grafické uživatelské rozhraní pro interaktivní nástroj pro návrh logických obvodů ABC. Tento úkol se podařilo splnit podle podmínek obsažených v zadání. První požadavek na toto grafické uživatelské rozhraní byl, aby rozhraní průběžně zobrazovalo informace o zpracovávaném obvodu. Možnost zvolit si, jaké informace o obvodu se budou zobrazovat, je ponechána na uživateli. Ten má tak možnost nastavit si aplikaci podle svého. Dalším požadavkem bylo urychlit práci uživatele tím, že grafické uživatelské rozhraní bude obsahovat nástroje pro rychlé zadávání často používaných příkazů. I tento požadavek byl splněn tím, že byl vytvořen seznam oblíbených příkazů, který zcela jistě ušetří čas, potřebný pro napsání takového příkazu. Dále bylo nutné příliš nezpomalit provádění výpočtů. I tento požadavek byl zcela splněn, jak již bylo dokázáno v testování. Výsledky testování výkonu aplikace ukázaly, že aplikace přenesená do grafického rozhraní nijak zásadně nezpomaluje prováděné výpočty. S tímto rychlým zpracováním příkazů souvisí i nízké hardwarové nároky pro spuštění aplikace. Obojí bylo dosaženo napsáním aplikace v jazyce C s použitím technologie programování WinAPI.

Možností, jak rozšířit aplikaci WinABC, se nabízí hned několik. Aplikace by mohla obsahovat lepší podporu skriptů. Nyní lze skript spustit pouze příkazem `source`. Nevýhoda tohoto je, že výsledek (textový výstup skriptu) vidíme až po úplném vykonání celého skriptu. Vhodnější by bylo, aby se tento skript otevřel v aplikaci WinABC a ta příkazy v něm uvedené vykonávala sama. Výstupy by tak byly vidět průběžně a byla by zde i možnost skript pozastavit. Další vylepšení aplikace by mohla přinést práce s více obvody najednou, nebo třeba podpora tisku informací o obvodu na tiskárně. Takovýchto vylepšení by se jistě našlo ještě více.

Literatura

[1] R. Chalupa. *1001 tipů a triků pro Visual C++*. Computer Press s.r.o., Brno, 1. vydání, 2003.

[2] P. Herout. *Učebnice jazyka C*, KOPP, České Budějovice, dotisk 3. vydání, 2000

[3] P. Herout. *Učebnice jazyka C – 2. díl*, KOPP, České Budějovice, dotisk 1. vydání, 1998

[4] Builder – Učíme se WinAPI
`http://www.builder.cz/art/cpp/winapi1.html`

[5] ABC: A System for Sequential Synthesis and Verification
`http://www.eecs.berkeley.edu/~alanmi/abc/`

[6] MSDN (Microsoft Developer Network) Library
`http://msdn.microsoft.com/en-us/library/default.aspx`

[7] Wikipedie – Windows API
`http://cs.wikipedia.org/wiki/Windows_API`

[8] SCDsource: Berkeley ABC project reshapes logic synthesis
`http://www.scdsource.com/academic.php?id=10`

Seznam použitých zkratek

ABC	Jméno konzolového programu pro návrh logických obvodů vyvinutého kalifornskou univerzitou v Berkeley
AIG	And-Inverter-Graph, obvod, který se skládá z dvouvstupových AND hradel a invertorů
API	Application Programming Interface (Rozhraní pro programování aplikací)
ATL	Active Template Library, soubor C++ tříd zjednodušující programování aplikací
BDD	Binary Decision Diagram, a canonical graph-based representation of Boolean functions
BLIF	Berkeley Logic Interchange Format, formát pro reprezentaci logických obvodů
COM	Component Object Model
GUI	Graphical User Interface (Grafické uživatelské rozhraní)
LUT	Look-up table
MFC	Microsoft Foundation Class Library
RTL	Register transfer level, způsob popisu funkce synchronního digitálního obvodu
SOP	Sum-of-Products, a non-canonical representation of Boolean functions
VS	Visual Studio, softwarový nástroj pro tvorbu aplikací od firmy Microsoft
WinABC	Jméno Win32 aplikace rozšiřující program ABC o GUI, aplikace vytvářené v této práci
WinAPI	Windows Application Programming Interface (Rozhraní pro programování aplikací operačního systému Windows)

Obsah přiloženého CD

Adresářová struktura

```
├─ [ABC]                - Původní konzolová aplikace ABC
├─ [abc70930_upraveno]  - Projekt a kódy ABC upravené pro WinABC
├─ [examples]           - Adresář se vzorovými obvody
├─ [text]               - Adresář s bakalářskou prací
├─ [WinABC]             - Zkompilovaná aplikace WinABC
└─ [WinABC_project]     - Projekt VS 6.0 a zdrojové kódy WinABC
```

Pro spuštění aplikace WinABC je nutné mít spustitelný soubor `winabc.exe` v adresáři, do kterého lze zapisovat. Překopírujte proto adresář s programem z přiloženého CD na lokální disk PC.