

# **Použití nástroje Freescale FreeMASTER pro ladění aplikací s mikropočítačem HCS08**

Eduard Miškařík



Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

akademický rok: 2014/2015

## **ZADÁNÍ BAKALÁŘSKÉ PRÁCE**

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Eduard Miškařík**

Osobní číslo: **A12114**

Studijní program: **B3902 Inženýrská informatika**

Studijní obor: **Bezpečnostní technologie, systémy a management**

Forma studia: **prezenční**

Téma práce: **Použití nástroje Freescale FreeMASTER pro ladění aplikací s mikropočítačem HCS08**

Téma anglicky: **Using the Freescale FreeMASTER Tool for Debugging HCS08 Applications**

Zásady pro vypracování:

1. Prostudujte a popište základní vlastnosti a možnosti nástroje FreeMASTER.
2. Navrhněte vhodné ukázkové aplikace demonstrující možnosti tohoto nástroje.
3. Navržené aplikace implementujte v jazyce C pro mikropočítač HCS08 a ověřte jejich funkčnost.
4. Vytvořte prezentaci popisující ladění a vizualizaci vytvořených aplikací v prostředí FreeMASTER.

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. BARR, Michael a Anthony J MASSA. Programming embedded systems. 2nd ed. Sebastopol: O'Reilly, 2006, xxi, 301 s. ISBN 978-0-596-00983-0.
2. CATSOULIS, John. Designing Embedded Hardware. Sebastopol: O'Reilly Media, 2005. ISBN 978-0-596-00755-3.
3. MANN, Burkhard. C pro mikrokontroléry: ANSI-C, kompilátory C, spojovací programy – linkery, práce s ATMELE AVR a MSC-51, příklady programování v jazyce C, nástroje pro programování, tipy a triky. Praha: BEN, 2003. ISBN 80-730-0077-6.
4. PINKER, Jiří. Mikroprocesory a mikropočítače. Praha: BEN – technická literatura, 2004. ISBN 80-730-0110-1.
5. Freescale. FreeMASTER Run-Time Debugging Tool [online]. 2015 [cit. 2015-01-15]. Dostupné z: [https://www.freescale.com/webapp/sps/site/prod\\_summary.jsp?code=FREEMASTER](https://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=FREEMASTER).

Vedoucí bakalářské práce:

**Ing. Jan Dolinay, Ph.D.**

Ústav automatizace a řídicí techniky

Datum zadání bakalářské práce:

**6. února 2015**

Termín odevzdání bakalářské práce:

**3. června 2015**

Ve Zlíně dne 6. února 2015

doc. Mgr. Milan Adámek, Ph.D.

*děkan*



L.S.

Ing. Jan Valouch, Ph.D.

*ředitel ústavu*



### Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

### Prohlašuji,

- že jsem na diplomové/bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně 3.6.2015

Miškoň

.....  
podpis diplomanta

## **ABSTRAKT**

Bakalářská práce se zabývá problematikou ladění aplikací za pomoci nástroje Freescale FreeMaster. Objasňuje a přibližuje danou problematiku. V teoretické části práce je popsána problematika mikropočítačů. Jsou tam rozebrány jednotlivé části mikropočítače a popsány jejich základní vlastnosti a druhy. V další kapitole teoretické části jsou podrobně popsány jednotlivé funkce a možnosti diagnostického programu FreeMaster. V praktické části práce, jsou přehledně pomocí obrázků vysvětleny funkce FreeMasteru. Funkčnost všech možností programu FreeMaster byla prakticky ověřena za pomoci kitu DEMO9S08QB8. U jednotlivých diagnostických funkcí je i popis jejich praktického využití.

Klíčová slova: Freescale, FreeMaster, CodeWarrior, Mikropočítač, DEMO9S08QB8

## **ABSTRACT**

This Bachelor's thesis deals with the issue of debugging tools with the use Freescale FreeMaster. It clarifies and defines the issue. The theoretical part describes the problems of microcomputers. There are analyzed different parts of microcontroller and described their basic properties and types. In the next theoretical chapter there are describes in detail the various features and possibilities of diagnostic program FreeMaster. In the practical part, there are clearly explained FreeMaster functions through the pictures. The functionality of all FreeMaster options were verified with using kit DEMO9S08QB8. Chapters of individual diagnostic functions also includes the description of their practical use.

Keywords: Freescale, FreeMaster, CodeWarrior, Microcontroller, DEMO9S08QB8

## **PODĚKOVÁNÍ**

Chtěl bych poděkovat Ing. Jan Dolinay, Ph.D. za cenné rady, připomínky a konzultace, kterými přispěl k vypracování této bakalářské práce. Dále bych chtěl poděkovat všem spolužákům, za poskytnuté informace, které mi pomohli při vypracovávání této práce. Taky bych chtěl poděkovat všem svým blízkým za podporu a motivaci nejen při vypracovávání téhle práce, ale i celém studiu

## OBSAH

<b>ÚVOD.....</b>	<b>9</b>
<b>I TEORETICKÁ ČÁST.....</b>	<b>10</b>
<b>1 ZÁKLADNÍ ČÁSTI A FUNKCE POČÍTAČE .....</b>	<b>11</b>
1.1 SBĚRNICE .....	11
1.1.1 Datová sběrnice .....	12
1.1.2 Adresová sběrnice .....	12
1.1.3 Řídící sběrnice .....	12
1.2 PROCESOR .....	12
1.3 PAMĚŤ.....	12
1.3.1 Paměť Programu .....	12
1.3.2 Paměť Dat .....	13
1.4 PERIFERNÍ OBVODY .....	13
1.5 MIKROPOČÍTAČE .....	13
<b>2 JAZYK C.....</b>	<b>16</b>
2.1 PROGRAMOVACÍ JAZYK.....	16
2.2 PROGRAM.....	16
2.3 PŘÍKAZ .....	16
2.4 INSTRUKCE.....	16
2.5 KOMPILÁTOR.....	17
2.6 DEBUGGER .....	17
2.7 STROJOVÝ KÓD.....	17
<b>3 FREEMASTER .....</b>	<b>18</b>
3.1 VLASTNOSTI FREEMASTERU .....	19
3.2 SYSTÉMOVÉ POŽADAVKY .....	19
3.3 INSTALACE .....	20
3.3.1 Instrukce pro stažení a instalaci .....	20
3.4 SPUŠTĚNÍ.....	20
3.5 OKNO APLIKACE.....	21
3.5.1 Panel menu a panel nástrojů.....	22
3.6 FUNKCE FREEMASTERU.....	22
3.6.1 Oscilloscope .....	23
3.6.2 Recorder .....	24
<b>II PRAKTICKÁ ČÁST .....</b>	<b>25</b>
<b>4 KIT DEMO9S08QB8 .....</b>	<b>26</b>
4.1 VLASTNOSTI KITU .....	26
<b>5 CODEWARRIOR .....</b>	<b>27</b>
5.1 CODEWARRIOR - NOVÝ PROJEKT .....	27
<b>6 FREEMASTER - PRAKTICKÉ OVĚŘENÍ FUNKCÍ .....</b>	<b>33</b>
6.1 VYTVOŘENÍ A NASTAVENÍ PROJEKTU.....	33
6.1.1 Synchronizace názvů proměnných.....	35

6.2	NASTAVENÍ PROMĚNNÝCH.....	35
6.2.2	Možnosti Proměnných.....	38
6.3	NASTAVENÍ VIZUÁLNÍ STRÁNKY HTML.....	41
6.4	SLEDOVÁNÍ PROMĚNNÝCH.....	41
6.5	PŘÍKAZY.....	45
6.6	FUNKCE OSCILLOSCOPE .....	48
6.6.1	Vytvoření a Nastavení funkce Oscilloscope .....	49
6.7	FUNKCE RECORDER.....	50
<b>ZÁVĚR .....</b>		<b>52</b>
<b>SEZNAM POUŽITÉ LITERATURY.....</b>		<b>54</b>
<b>SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK .....</b>		<b>56</b>
<b>SEZNAM OBRÁZKŮ .....</b>		<b>57</b>
<b>SEZNAM TABULEK.....</b>		<b>59</b>



## ÚVOD

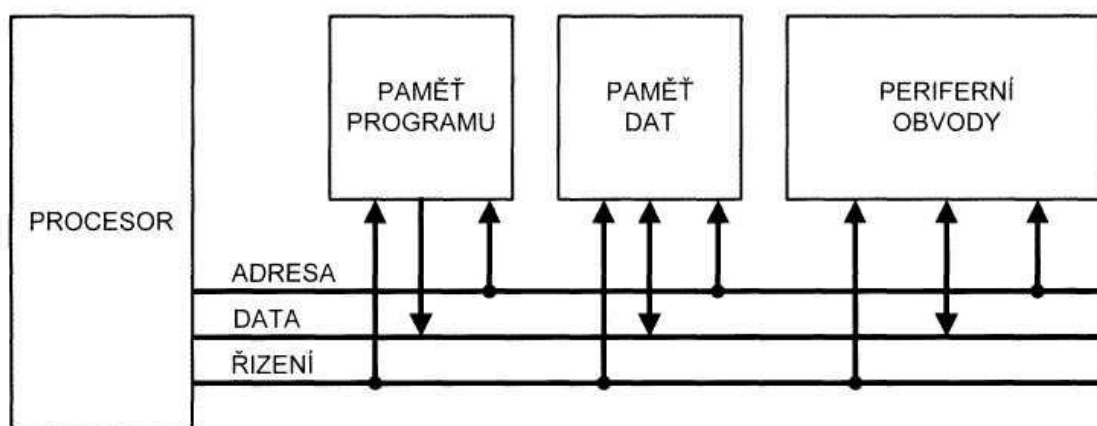
V dnešní době se mikropočítače nacházejí prakticky ve všech elektronických zařízeních různého účelu. Aby takové zařízení mohlo fungovat, je potřeba takové zařízení navrhnout a následně vyrobit. Po jeho mechanickém sestavení je ho ještě potřeba naprogramovat. Pro správnou funkci zařízení je v tomto programu potřeba najít všechny chyby. Chyba může vzniknout při samotném programování, ale taky při nějaké nespecifikované chybě plynoucí z používání tohoto zařízení. Na následujících stranách Bakalářské práce je popsána problematika zabývající se laděním aplikací pomocí nástroje Freescale FreeMaster. Tento program dokáže přistupovat přes specifický protokol pomocí sériové linky do desky kitu respektive do samostatného mikroprocesoru. Z paměti mikroprocesoru potom může zobrazovat pomocí speciálních zobrazovacích funkcí hodnoty, které jsou uloženy v jednotlivých proměnných v reálném čase. Teoretická část rozebírá teorii o mikropočítačích a samostatném programu Freemaster a dále taky bude rozebráno programování v jazyce C. Pomocí programu pro vývoj aplikací CodeWarrioru, bude napsán odladěn a následně nahrán do mikroprocesoru program, který je potřeba pro testovací účely FreeMasteru. V teoretické části o mikropočítačích, která nás lépe uvede do problematiky mikropočítačů, budou popsány základní části počítačů, respektive mikropočítačů tzn. procesor, druhy paměti, periferní obvody a druhy sběrnice. Taky zde budou popsány funkce těchto částí, vysvětlen základní rozdíl mezi počítačem a mikropočítačem a částečně jejich vývoj a vznik. V hlavní části práce budou postupně teoreticky rozebrány funkce a možnosti nastavení aplikace Freemaster. Největší pozornost bude věnována popisu základních zobrazovacích funkcí, tzn. Osciloskop a Recorder, taky jak tyto funkce fungují a jaké je jejich praktické využití. V Praktické části budou podrobně popsány s využitím obrázků jednotlivé kroky. První z kroků je vytvoření projektu v programu Freemaster a nastavení a navázání komunikace s deskou kitu. Dále v praktické části budou ověřeny jednotlivé funkce programu Freemaster, které budou detailně popsány na přiložených obrázcích z programu. Veškeré funkce budou testovány za pomoci kitu DEMO9S08QB8. V praktické části bude taky popsáno prostředí CodeWarrior, které bude použito při psaní a následném odladění a nahrání programu do mikroprocesoru pro testování funkcí FreeMasteru.

## **I. TEORETICKÁ ČÁST**

# 1 ZÁKLADNÍ ČÁSTI A FUNKCE POČÍTAČE

Mikropočítač je malý relativně "levný" počítač, který vychází z všeobecného blokového schématu pro počítače (viz Obrázek 1). Stupeň integrace je na velké úrovni to znamená, že hlavní součásti jsou integrovány v jednom čipu a na desce jsou jen pomocné externí obvody a minimum součástek, které jsou potřeba pro jeho funkci. Na desce se mohou nacházet taky další externí obvody, zprostředkovávající komunikaci a jiné funkce, které jsou potřebné pro danou situaci. Mikropočítač má několik funkčních částí:

- Mikroprocesor
- Paměť programu a dat
- Periferní obvody
- Sběrnice



Obrázek 1 Zjednodušené schéma mikropočítače [1]

## 1.1 Sběrnice

Vzájemné propojení všech částí mikropočítače zajišťuje sběrnice (viz Obrázek 1). Sběrnice je soustava paralelních signálových a datových vodičů, která umožňuje přenos signálů mezi všemi částmi počítače. Pomocí těchto vodičů mezi sebou komunikují a přenášejí data, jednotlivé části počítače. U počítačů respektive mikropočítačů máme tři typy sběrnice a to adresní, datovou a sběrnici řízení. Sběrnice umožňují stavebnicový koncept, tzn., že můžeme rozšiřovat počítač o další jednotky a díly, takzvané moduly, aniž bychom museli zasahovat do vnitřního zapojení počítače. Nevýhodou sběrnice je to, že na ni v jeden okamžik může být připojeno jen jeden zdroj dat. Nelze tak například zapisovat

data do paměti ze dvou zdrojů najednou a podobně. Sběrnici většinou řídí procesor, ale ve výjimečných případech dočasně tuto funkci může přebírat jiná jednotka.[1] [2]

### **1.1.1 Datová sběrnice**

Datová sběrnice slouží k přenášení zpracovaných dat mezi jednotlivými díly. Šířka datové sběrnice je většinou násobkem osmi, tzn. jednoho Bytu.

### **1.1.2 Adresová sběrnice**

Pomocí adresové sběrnice se adresují data do paměti. Šířka adresové sběrnice je u osmi bitových mikropočítačů nejčastěji šestnáct bitů a její určuje maximální počet adres.

### **1.1.3 Řídící sběrnice**

Řídící signály, například pro čtení nebo zápis jsou z procesoru k ostatním dílům přenášeny pomocí řídící sběrnice.

## **1.2 Procesor**

Procesor je hlavní jednotka z částí počítače nebo mikropočítače a řídí činnost celého počítače. Zpracovává data a instrukce, které jsou uloženy v paměti, a zajišťuje jejich správné provádění.

## **1.3 Paměť**

Mikropočítač obsahuje několik typů paměti, které jsou určeny pro ukládání programu nebo jiných dat potřebných pro správnou funkci mikropočítače nebo počítače.

### **1.3.1 Paměť Programu**

V paměti programu jsou uloženy jednotlivé instrukce pro procesor. Postupným vykonáváním těchto instrukcí je realizována požadovaná funkce mikropočítače. V paměti mohou být taky uloženy například různé neměnné konstanty a jiné data potřebné pro funkci programu.

V některých případech, například při vykonávání nějakého stále se opakujícího procesu je program uložen v paměti typu ROM (EPROM, EEPROM, FLASH). U procesů často se měnících je využívána paměť typu RAM, která má ale po ztrátě napájení náhodný obsah. Proto je počítač také ještě vybaven malou programovou pamětí typu ROM. Při

náběhu počítače program začíná zde a načte data z velkokapacitní diskové paměti do rozsáhlejší paměti typu RAM.

### 1.3.2 Paměť Dat

Paměť dat slouží pro dočasné uložení dat získaných například při mezi výpočtech nebo dat získaných ze vstupních obvodů apod.. Paměť dat je většinou realizována pamětí typu RAM. V případě že je i programová paměť stejného typu mohou být realizovány jednou společnou pamětí.

## 1.4 Periferní obvody

Vstupní a výstupní obvody neboli periferní obvody, jsou obvody pro komunikaci počítače s vnějším prostředím. Je to zařízení, které není nezbytně nutné k provozu počítače, avšak rozšiřuje jeho schopnosti.

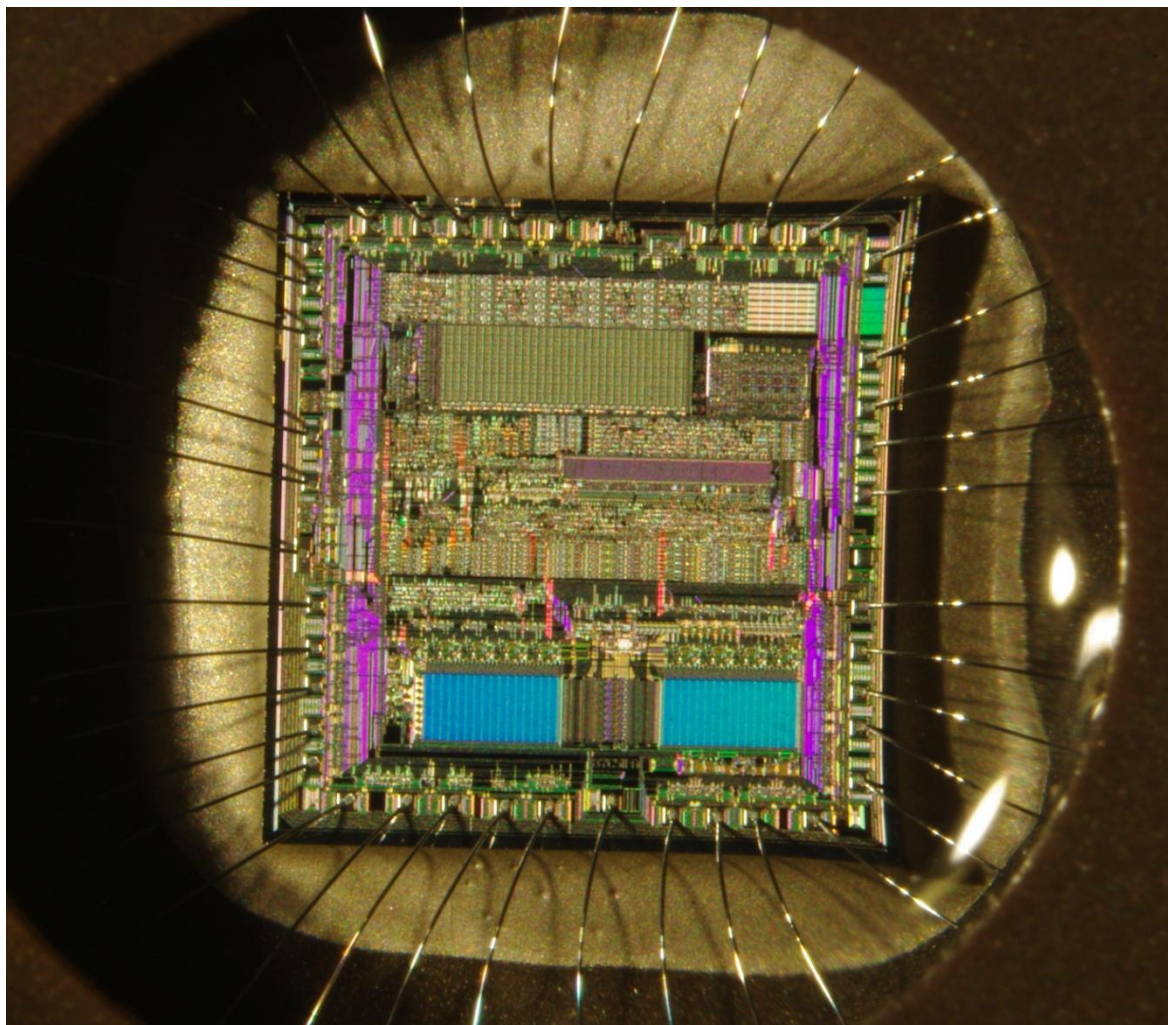
*„Zpravidla obsahují větší počet bran (angl. port), tj. připojovacích míst, rozlišených adresově. Brány mohou být paralelní nebo sériové. Při paralelním přenosu se čte nebo zapisuje najednou celá skupina signálů jako vícebitové slovo (nejčastěji 8bitové). Při sériovém přenosu se data přenášejí postupně bit po bitu. Sériový přenos je mnohem pomalejší než paralelní, ale je úspornější co se týče počtu signálových vodičů.“ [1]*

## 1.5 Mikropočítače

Výše probíraná problematika obecně platí pro všechny počítače. S rychlým rozvojem technologií integrovaných obvodů bylo umožněno zmenšit rozměry jednotlivých částí počítače natolik, že se vejdou do jediného čipu. Tím vzniklo odvětví mikropočítačů. Předpona mikro se vztahuje k fyzickým rozměrům mikropočítačů, nikoliv k omezení jejich funkcí.

*„Vznikla tak řada **jednočipových mikropočítačů**. Zvláště v této skupině počítačů probíhá nejintenzivnější vývoj. Zvětšuje se kapacita paměti programu i paměti dat, rozšiřuje se sestava periferních obvodů. Jednočipový mikropočítač, vhodný pro využití v řízení, je v anglické literatuře označován jako „microcontroller“, česky mikrokontrolér.“ [1]*

Detail mikropočítače a stupeň integrace můžeme vidět na obrázku níže. (viz Obrázek 2 - skutečné rozměry cca 5x5mm)



Obrázek 2 Detail mikropočítače [3]

V dnešní době jsou mikropočítače integrovány snad do všech zařízení nejenom v průmyslové oblasti (například řízení různých výrobních procesů a podobně) nebo automobilový průmysl, ale taky v oblasti člověku mnohem bližší tzv. "chytrých" elektrospotřebičů počínaje drobnou elektronikou (mobilní telefony, fotoaparáty, atd.) přes domácí spotřebiče (lednice, automatické pračky atd.). Mikropočítače v takovýchto aplikacích jsou označovány jako „**embedded computers**“, v doslovném českém překladu tzn. jako zabudované nebo vložené mikropočítače.

*„Na rozdíl od rychlého vývoje sestav periferních obvodů a zvyšování kapacity vnitřní paměti je vývoj nových procesorů podstatně konzervativnější. Souvisí to s nutností kompatibility (tj. slučitelnosti) programů po dlouhou dobu. Finanční prostředky, investované do vývoje programů, jsou obrovské a žádný výrobce integrovaných obvodů tuto skutečnost nemůže ignorovat. Modernizace procesoru se děje nejčastěji přidáváním*



*několika málo instrukcí, zvýšením rychlosti, doplněním o specializované obvody. Kompatibilita programů je tak zaručena směrem nahoru (programy pro starší verzi jsou plně použitelné i pro verzi novou), ale ne nutně směrem dolů." [1]*

## 2 JAZYK C

Pro testování funkcí programu FreeMaster bylo potřeba naprogramovat jednoduchý program v jazyce C. Jazyk C je programovací jazyk, který byl vyvinut pro potřeby operačního systému UNIX počátkem 70. let minulého století Kenem Thompsonem. V současnosti je tento jazyk hodně využíván díky jeho logickému pojmenování příkazů a jeho jednoduchosti. Jazyk C je využíván nejen pro psaní systémového softwaru ale i pro aplikace. Jazyk C je taky využíván díky jeho přenositelnosti na různé architektury. Pro potřeby Bakalářské práce budou v následujících podkapitolách vysvětleny základní pojmy. [4] [5]

### 2.1 Programovací jazyk

Programovací jazyk je forma zápisu jednotlivých příkazů celého programu. Programovací jazyk je způsob komunikace s počítačem, prostřednictvím kterých přikazujeme počítači jakou činnost má vykonat. Jednotlivé programovací jazyky se od sebe navzájem liší klíčovými slovy a dalšími jinými principy. Každý programovací jazyk má svoji syntaxi (způsob zápisu programu) a sémantiku (význam jednotlivých příkazů).

### 2.2 Program

Program je smysluplná posloupnost příkazů vložených do počítače, která má určený význam, sleduje nějaký plán a jejím výsledkem je vykonání úlohy nebo výpočtu. Když vytváříme program, znamená to, že hledáme tuto smysluplnou posloupnost příkazů. Zapsáním posloupnosti příkazů vytvoříme zdrojový kód programu. Po napsání programu se musí přeložit do instrukcí, kterým rozumí procesor (nejčastěji strojový kód). Překlad programu vykonává nástroj, který se nazývá překladač (nebo kompilátor).

### 2.3 Příkaz

Příkaz je podmět pro vykonání určité činnosti, kterou chceme, aby program vykonal. Příkazem může být například součet.

### 2.4 Instrukce

Instrukce je příkaz pro vykonání určité činnosti, kterým rozumí procesor. Sady těchto instrukcí jsou odlišné pro každý typ procesoru.

## 2.5 Kompilátor

Kompilátor neboli překladač je softwarový nástroj pro překlad z vyššího programovacího jazyka, například jazyk C, do nižšího programovacího jazyka nejčastěji strojového kódu. [6]

## 2.6 Debugger

*„Debugger je utilita / softwarový nástroj, který se používá pro hledání chyb při vývoji software ve fázi ladění. Většinou je možné zobrazit zdrojový kód laděného programu, takže je ihned možné vidět místo, kde se objevila programátorská chyba.“ [7]*

## 2.7 Strojový kód

*„Strojový kód je v informatice posloupnost strojových instrukcí prováděných procesorem počítače, která je zapsána pomocí posloupnosti číselných kódů těchto strojových instrukcí.“ [8]*

### 3 FREEMASTER

Na následujících stranách budou rozepsány jednotlivé funkce programu Freemaster od firmy freescale. Freemaster je, dá se říct diagnostický nástroj, pro PC, který dokáže přistupovat do paměti mikropočítače a pracuje v reálném čase. Aplikace byla původně vytvořena vývojáři pro řízení motorů v reálném čase, ale mnoho uživatelů zjistilo, že se dá využít i pro jinou vlastní potřebu. Aplikace FreeMaster může být nainstalována na jakémkoliv systému Microsoft Windows počínaje systémem Windows 98. Stručný přehled podporovaných rodin mikropočítačů viz Tabulka 1, které mají implementovány komunikační protokol Freemaster.

*Tabulka 1 Podporované platformy*

Device / Family	Software Package containing FreeMaster support
56F800/E	Metrowerks IDE / Processor Expert Embedded DSP Software Development Kit (SDK) 56F800_Quick_Start development tool 56F800E_Quick_Start development tool
HC08/HCS08	Stand alone implementation (Freescale Application Note AN2637)
HC12/HCS12	Stand alone implementation derived from HC08 code
MPC500	MPC500_Quick_Start development tool
MPC5500	MPC5500_Quick_Start development tool

Primárním cílem pro rozvoj softwaru FreeMaster bylo vyvinout nástroj pro ladění a demonstraci řídicích algoritmů a aplikací u motorů. Výsledek byl univerzální nástroj, který může být použit pro víceúčelové algoritmy a aplikace například:

- Ladění v reálném čase - FreeMaster umožňuje uživatelům ladění aplikací ve skutečném reálném čase díky své schopnosti sledovat proměnné. Kromě toho, umožňuje ladění na úrovni algoritmu, který pomáhá zkrátit fázi vývoje.
- Diagnostický nástroj - FreeMaster umožňuje, aby byl použit jako diagnostický nástroj pro ladění aplikací na dálku přes síť.
- Ukázky - FreeMaster je vynikající nástroj pro demonstraci algoritmů nebo aplikací s rychle se měnícími výstupy.
- Výuka - FreeMaster může být použit pro vzdělávací účely. Jeho vlastnosti umožňují studentům ovládání a zkoumání programu a učí je, jak program funguje.

### 3.1 Vlastnosti FreeMasteru

- Grafické prostředí
- Snadno pochopitelná navigace v programu
- Jednoduché připojení přes RS232 (USB)
- Přístup v reálném čase k proměnným
- Vizualizace dat v reálném čase na osciloskopu
- Zaznamenání rychle se měnících dat pomocí recorderu
- Vestavěná podpora pro standardní typy proměnných (integer...)
- Interpretace hodnot pomocí vlastní definovaných textových zpráv
- Několik vestavěných transformací pro proměnné typu REAL
- Automatická extrakce proměnných z Metrowerks CodeWarrior výstupních souborů (ELF / DWARF1 / 2, map files...)
- Režim Demo s podporou ochrany heslem
- HTML jazyk na popis nebo navigační stránky
- ActiveX rozhraní umožňuje pomocí VBScript nebo JScript kontrolu nad Embedded Application
- Vzdálený Komunikační server umožňuje připojení k cíli přes síť nebo internet, funkce není ale k dispozici ve volné distribuci
- Vlastní komunikační plug-in moduly jsou k dispozici pro různé typy protokolů (CAN / CCP, JTAG, BDM)

### 3.2 Systémové požadavky

Aplikace FreeMaster může běžet na jakémkoliv PC s operačním systémem Microsoft Windows 98 nebo vyšší.

- Operační systém: Windows 98, Windows NT, až po Windows 7 64bit
- Potřebný software: Internet Explorer 4.5.5 nebo vyšší
- Místo na disku: cca. 108 MB
- Ostatní hardwarové požadavky: Sériová linka RS-232 (USB) pro lokální přístup, přístup k síti pro vzdálený přístup

### 3.3 Instalace

Aplikace FreeMASTER je distribuována buď jako součást většího vývojového nástroje (například Freescale CodeWarrior), nebo jako samostatná instalace.

Samostatná aplikace FreeMaster se skládá ze dvou částí:

- **Aplikace FreeMaster:** je hlavní část obsahující sama o sobě vizualizační nástroj, běžící na PC. Dodává se s vestavěnými ovladači pro komunikaci.
- **Komunikační ovladače FreeMaster:** Tento balíček zahrnuje další ovladače a software např. ovladače pro sériovou linku, komunikační software, a ovladače pro další spojení, jako BDM (Background Debug Mode) nebo CAN a USB.

#### 3.3.1 Instrukce pro stažení a instalaci

Postup pro stažení a instalaci aplikace je následující:

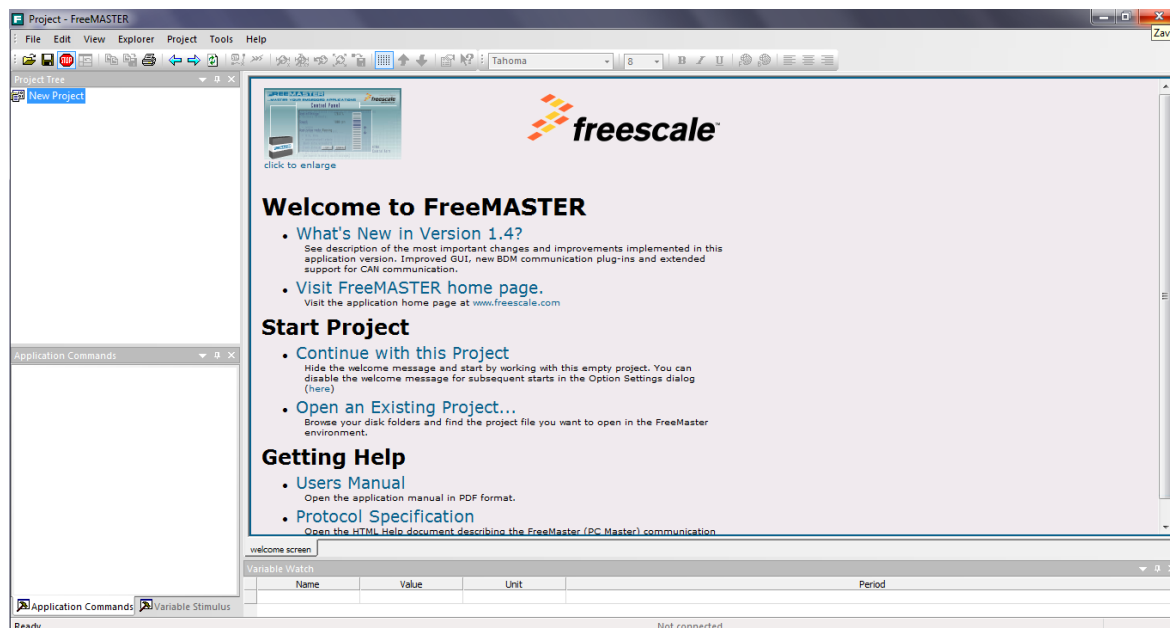
- Na stránkách výrobce ([www.freescale.com](http://www.freescale.com)) vyhledat aplikaci FreeMaster a z tabulky 'Downloads' stáhnout obě části. První část samostatnou aplikaci 'FreeMaster Software (REV 1.4.4)' a druhou část komunikační ovladače 'FreeMaster Communication Driver (REV 1.9)'.
- Spustit instalaci obou částí (nezáleží na pořadí) a nainstalovat je do PC, dle pokynů na obrazovce.

### 3.4 Spuštění

Při spuštění aplikace, se zobrazí na obrazovce hlavní okno. Když není načten žádný projekt, v hlavním panelu okna se zobrazí úvodní uvítací stránka. Počáteční vzhled hlavního okna můžeme vidět na obrázku (viz Obrázek 3).

Úvodní stránka obsahuje odkazy na dokumentaci a na nápovědu. Je tady taky několik dalších odkazů na standardní příkazy menu (například příkaz Otevřít existující projekt a další).





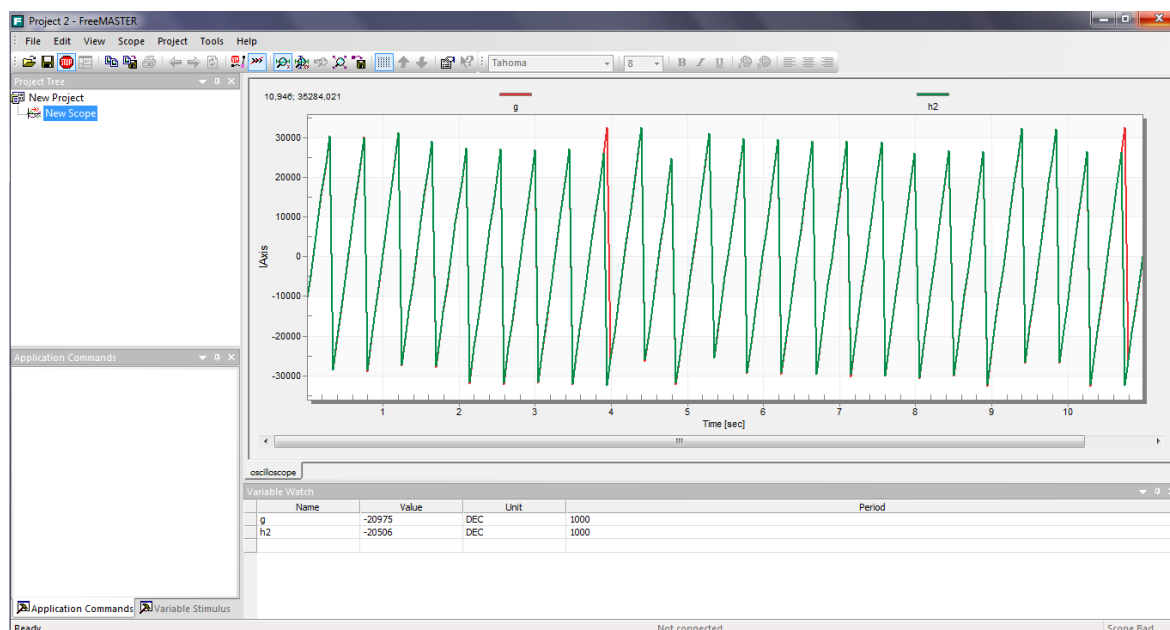
Obrázek 3 Úvodní stránka

### 3.5 Okno aplikace

Okno aplikace se skládá z několika částí, jak je vidět na obrázku (viz Obrázek 4), zobrazení se může trochu lišit v závislosti na nastavení v položce "View" - Zobrazit, podokna mohou být zobrazeny nebo skryty.

Základní části okna:

- "Menu and Toolbar" - Lišta menu a panel nástrojů
- "Project Tree" - Strom projektů
- "Detail View Area" - Hlavní podokno
- "Application Commands Pane" - Panel příkazů
- "Variable Watch Grid" - Tabulka proměnných



Obrázek 4 Okno aplikace

**"Menu and Toolbar"** - Lišta menu a panel nástrojů jedná se o hlavní ovládací prvky aplikace, přes které se aplikace ovládá. Detailně jsou popsány v jedné z následujících kapitol.

**"Project Three"** - **Strom projektů** obsahuje logickou strukturu projektu, a tvoří základní ovládací prvek projektu. Uživatelé mohou přidávat a definovat sub-bloky projektů, tzn. osciloskop, rekordér a jiné funkce.

**"Detail View Area"** - **Hlavní podokno** je největší část okna aplikace. V hlavním podokně aplikace se zobrazují informace v závislosti na vybrané položce ve stromu projektů. V okně se mohou zobrazovat detaily například graf osciloskopu nebo rekordéru v závislosti na nastavení jejich vlastností. V hlavním okně může být taky zobrazena grafická stránka ve formátu HTML.

### 3.5.1 Panel menu a panel nástrojů

Na panelu menu a panelu nástrojů se nachází veškeré hlavní ovládací prvky programu. Přes panel menu a panel nástrojů je možné přistupovat například do nastavení programu a jiné.

## 3.6 Funkce Freemasteru

Freemaster komunikuje s mikropočítačem pomocí sériové linky. Umí číst a zapisovat do interních proměnných uložených v paměti mikropočítače. Aby mohl program

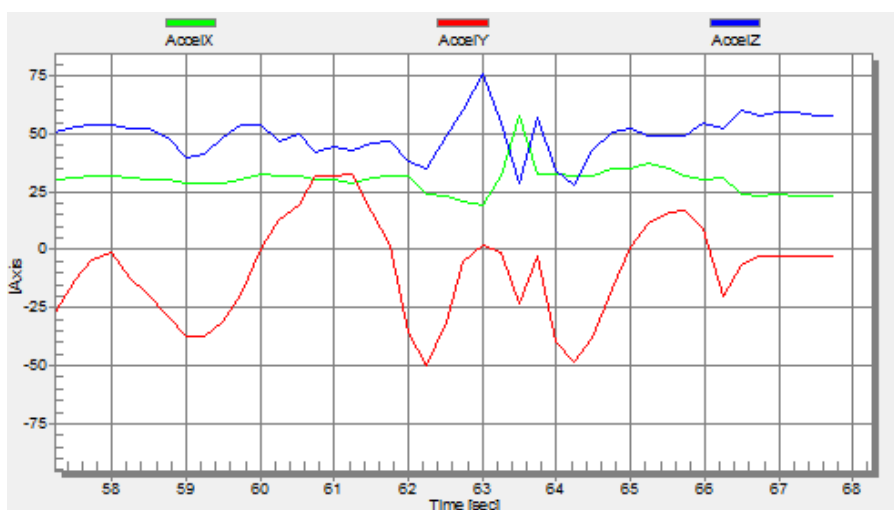
pracovat s proměnnými, nejprve je třeba ho odkázat na soubor typu DWARF/ELF z programu CodeWarrior ze kterého si automaticky extrahuje a synchronizuje názvy proměnné. Po načtení proměnných do programu FreeMaster již s nimi můžeme pracovat. To znamená, že můžeme přidávat a zobrazovat průběhy proměnných v jednotlivých funkcích Freemasteru například osciloskopu. FreeMaster poskytuje vizualizační funkce pro zobrazení informací z proměnných, které jsou popsány v následujících kapitolách.

Informace z paměti může být zobrazena v jednom nepřetržitém bloku, nebo rozdělena do několika bloků, podle toho, která možnost lépe odráží parametry funkce. Každý vstupní blok informací může být podrobně prozkoumán/zobrazen a můžeme pozorovat, jak ovlivňuje výstupní parametry. Každý blok má kartu popis pro vysvětlení detailů.[9]

### 3.6.1 Oscilloscope

FreeMaster Osciloskop - umožňuje zobrazování / vizualizaci informací (viz

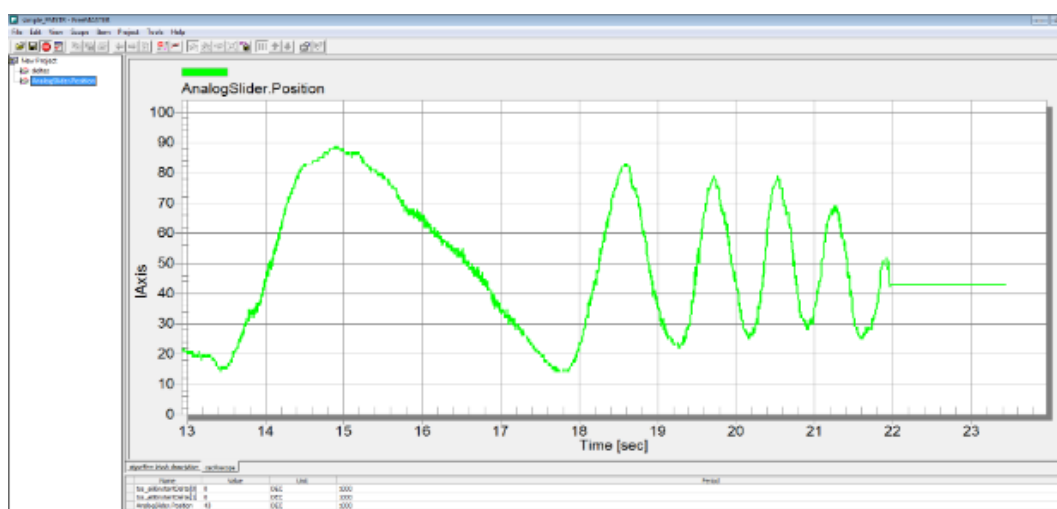
Obrázek 5) v reálném čase z vybraných proměnných stejným způsobem jako klasický osciloskopem s CRT obrazovkou. Hodnoty z proměnných jsou načítány z desky kitu v reálném čase přes sériovou komunikační linku. V tomto případě, je ale zobrazovací rychlost omezena tím, že rychlost sériové komunikace není dostatečná. Čím více je zobrazovaných proměnných najednou tím tato schopnost ještě více klesá. Maximální počet zobrazených proměnných najednou je osm.



Obrázek 5 Osciloskop [10]

### 3.6.2 Recorder

Recorder - umožňuje zobrazování / vizualizaci informací z proměnných, které se mění větší rychlostí než je vzorkovací rychlost osciloskopu. Zatímco funkce Osciloskop pravidelně čte hodnoty proměnných a zobrazuje je v reálném čase. Recorder je spuštěn na cílové desce a proměnné hodnoty se ukládají do vyrovnávací paměti na desce, potom se údaje a vzorky stáhnou z desky do FreeMasteru. Tento mechanismus umožňuje mnohem větší vzorkování a umožňuje vykreslení velmi rychlých akcí (viz Obrázek 6).

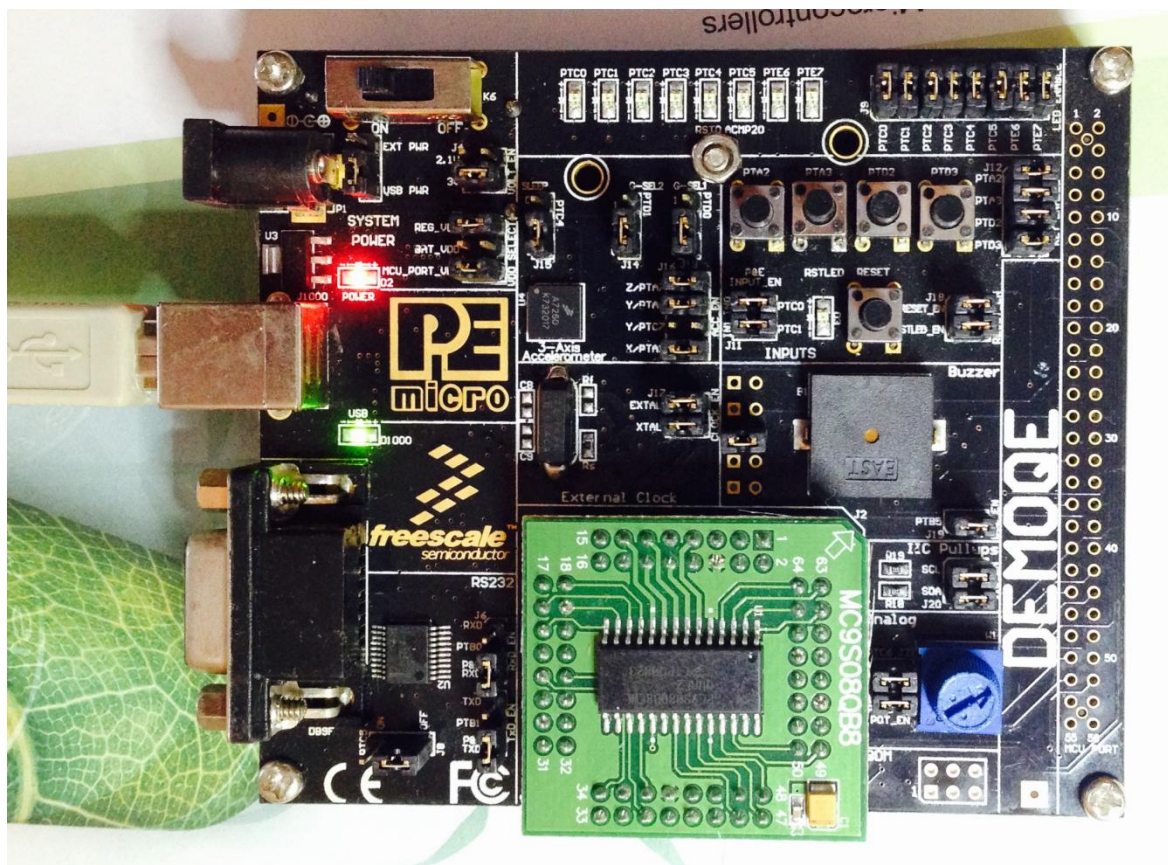


Obrázek 6 Recorder [10]

## **II. PRAKTICKÁ ČÁST**

## 4 KIT DEMO9S08QB8

Praktická část Bakalářské práce byla testována na kitu DEMO9S08QB8 firmy Freescale, na kterém se nachází mikroprocesor HCS-08 (viz Obrázek 7).



Obrázek 7 Deska kitu

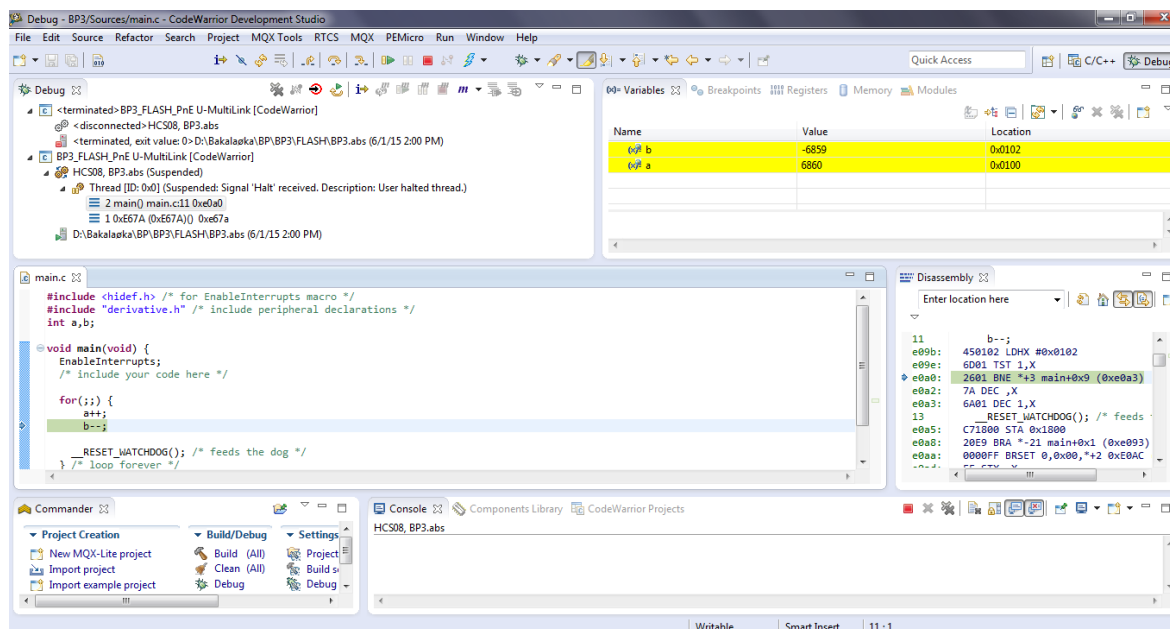
### 4.1 Vlastnosti kitu

Na desce kitu se dole uprostřed (na zeleném plošném spoji) nachází 8bitový mikroprocesor MC9S08QB8, který obsahuje 512 B datové paměti typu RAM a 8 kB programové paměti typu Flash. Maximální frekvence procesoru je 20MHz. Vlevo dole se nachází USB a RS232 konektory pro komunikaci s PC. V pravé horní části desky kitu se nachází ostatní periferní obvody, které ale nebyly v bakalářské práci využity. Napájení desky kitu je zprostředkováno pomocí USB kabelu, ale může být využito i externího napájení pomocí adapteru nebo bateriového napájení.



## 5 CODEWARRIOR

Pro testování funkcí programu FreeMaster bylo potřeba nahrát do mikroprocesoru program. Program byl napsán v jazyce C v prostředí CodeWarrior. Vzhled programu Codewarrior můžeme vidět na následujícím obrázku (viz Obrázek 8).

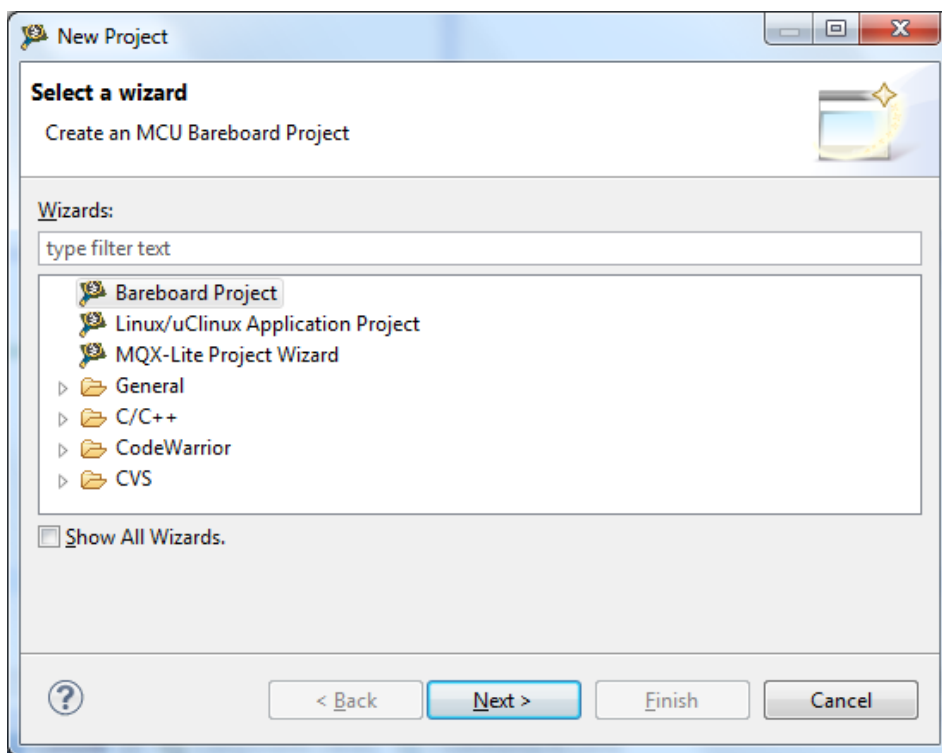


Obrázek 8 Ukázka prostředí CodeWarrior

### 5.1 Codewarrior - Nový projekt

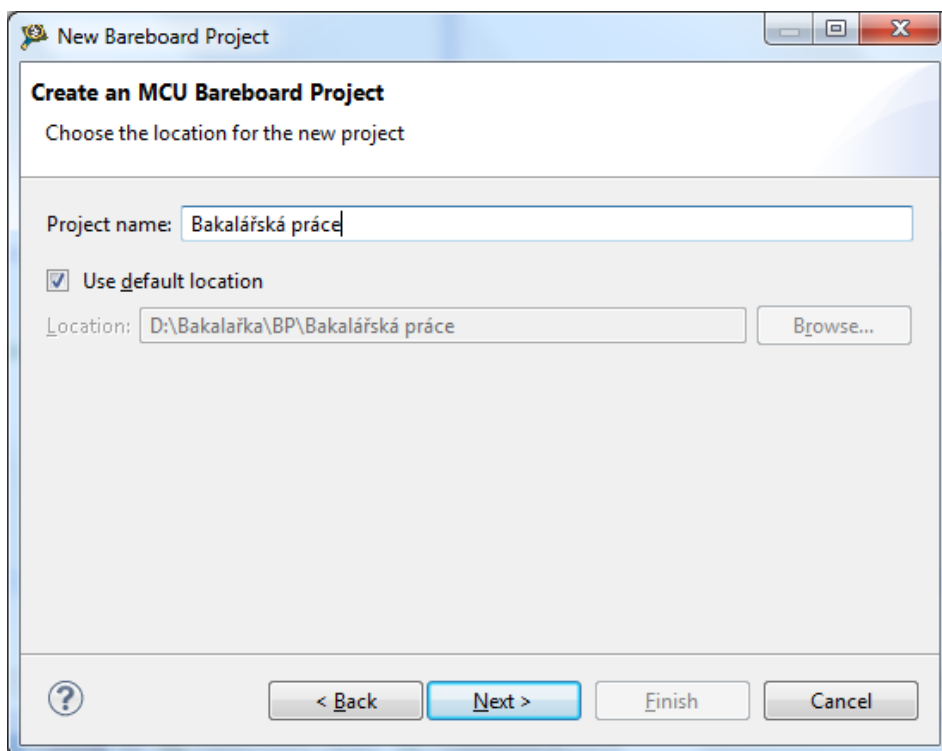
Tato kapitola obsahuje návod pro vytvoření nového projektu v programu Codewarrior. Taky zde bude popsáno nastavení všech potřebných funkcí a následného naprogramování a nahrání programu do mikroprocesoru.

Po spuštění programu Codewarrior je v první řadě potřeba vytvořit nový projekt. Nový projekt se vytváří přes nabídku File->New->Project a nebo přes klávesovou zkratku Alt + Shift + N. Po vykonání toho kroku se nám zobrazí následující okno (viz Obrázek 9). V tomto okně vybereme Bareboard Project a stiskneme tlačítko Další (Next).



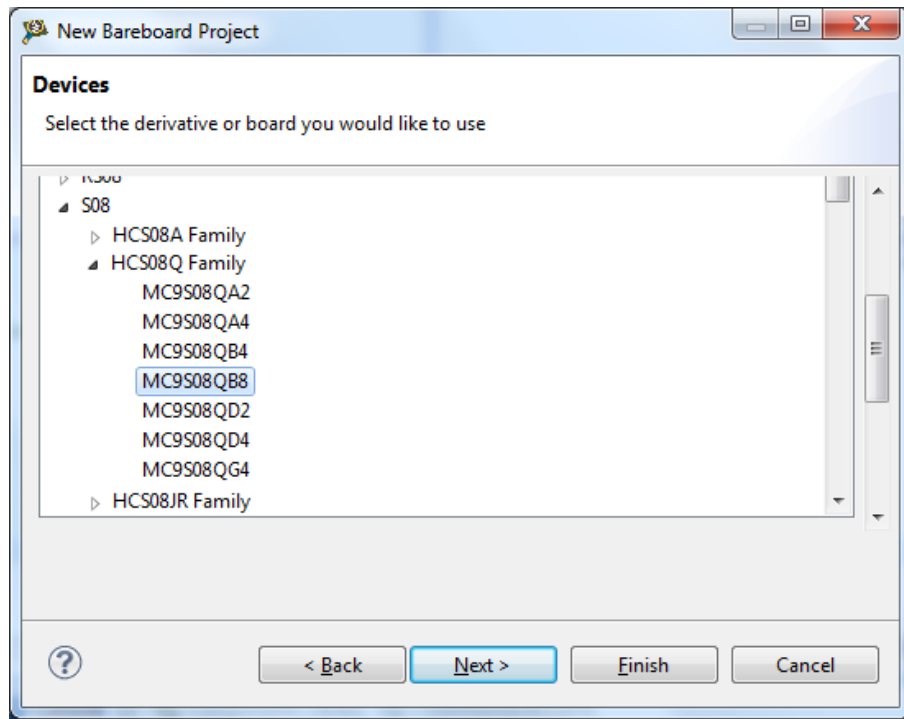
*Obrázek 9 Codewarrior - New Project*

V následujícím okně (viz Obrázek 10) do pole Project name napíšeme jméno projektu a případně zvolíme cestu kam projekt uložit. Nyní můžeme stisknout tlačítko Další (Next) a tím přejít k dalšímu kroku.



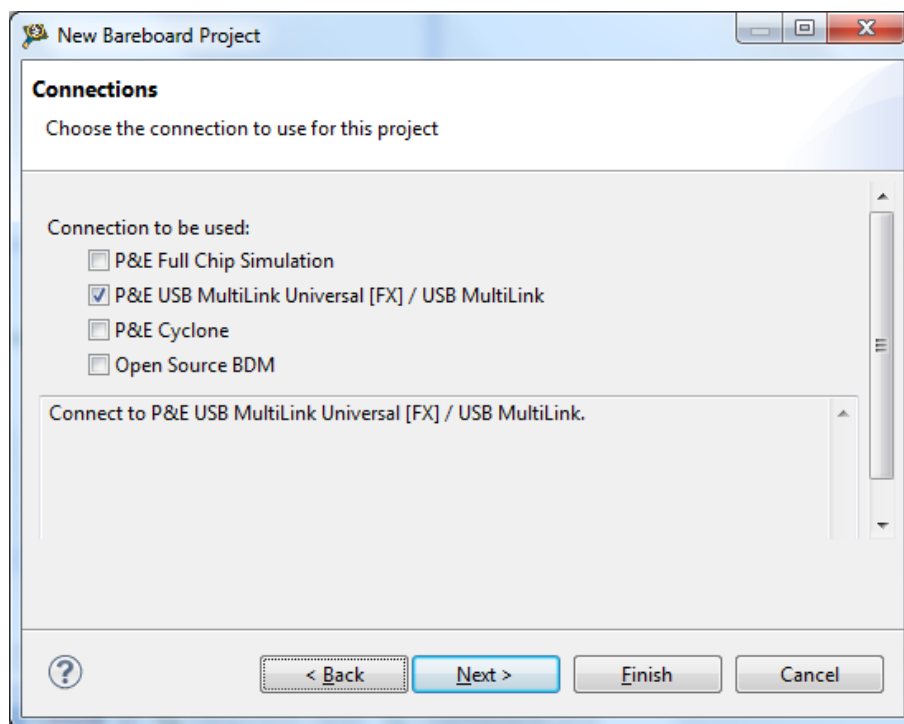
*Obrázek 10 Codewarrior - New Bareboard Project*

V tomto kroku (viz Obrázek 11) je potřeba vybrat možnost, která odpovídá našemu zařízení. To znamená MC9S08QB8. Potom můžeme přejít přes tlačítko Další (Next) k poslednímu kroku.



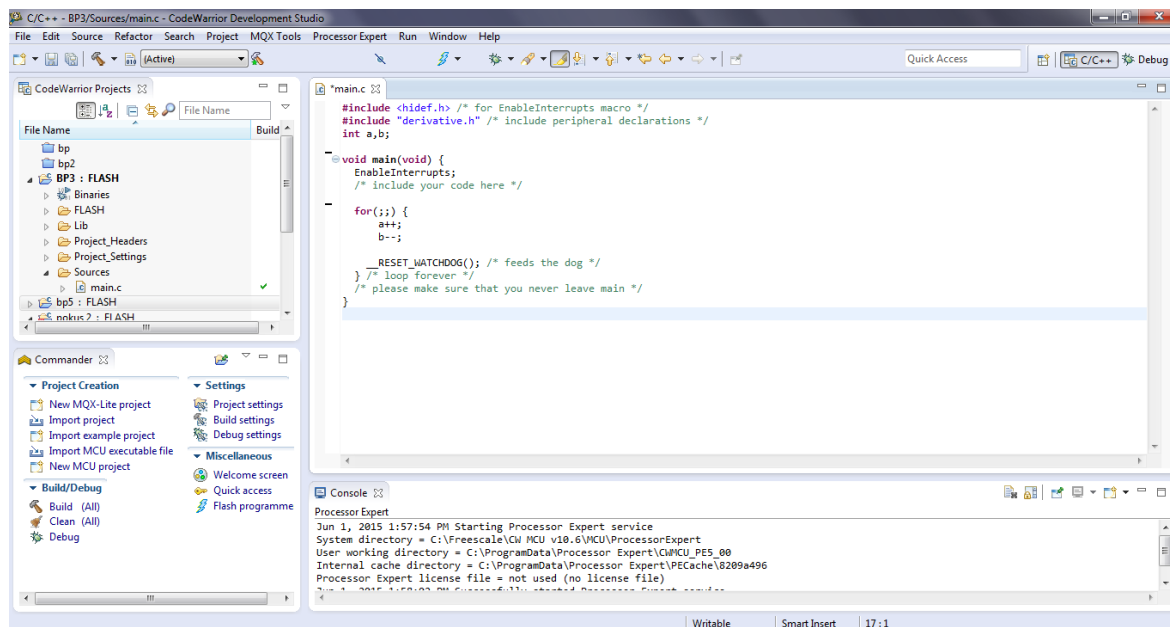
Obrázek 11 Codewarrior - Devices

Poslední krok spočívá v tom, že vybereme možnost komunikace (viz Obrázek 12).



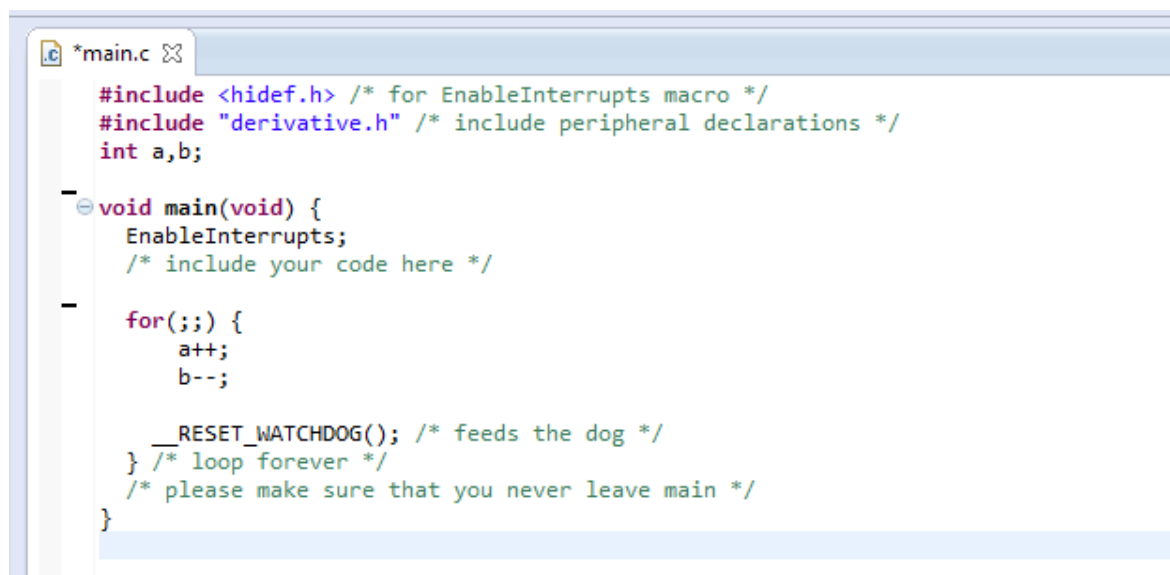
Obrázek 12 Codewarrior - Connections

Po výběru správné komunikace můžeme přejít k dokončení vytváření projektu přes tlačítko Finish. Po dokončení projektu se nám otevře soubor s názvem main. c (viz Obrázek 13).



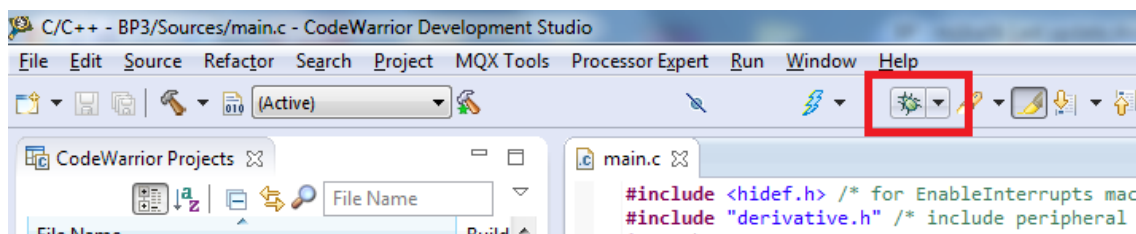
Obrázek 13 Codewarrior - main.c

Do tohoto souboru napíšeme program pro mikroprocesor, pomocí kterého programu budeme následně testovat funkce programu FreeMaster. Pro testovací funkci nám postačí následující program (viz Obrázek 14). Funkce programu spočívá v tom, že neustále přičítá nebo odečítá hodnotu jedna k hodnotě uložené v proměnné.



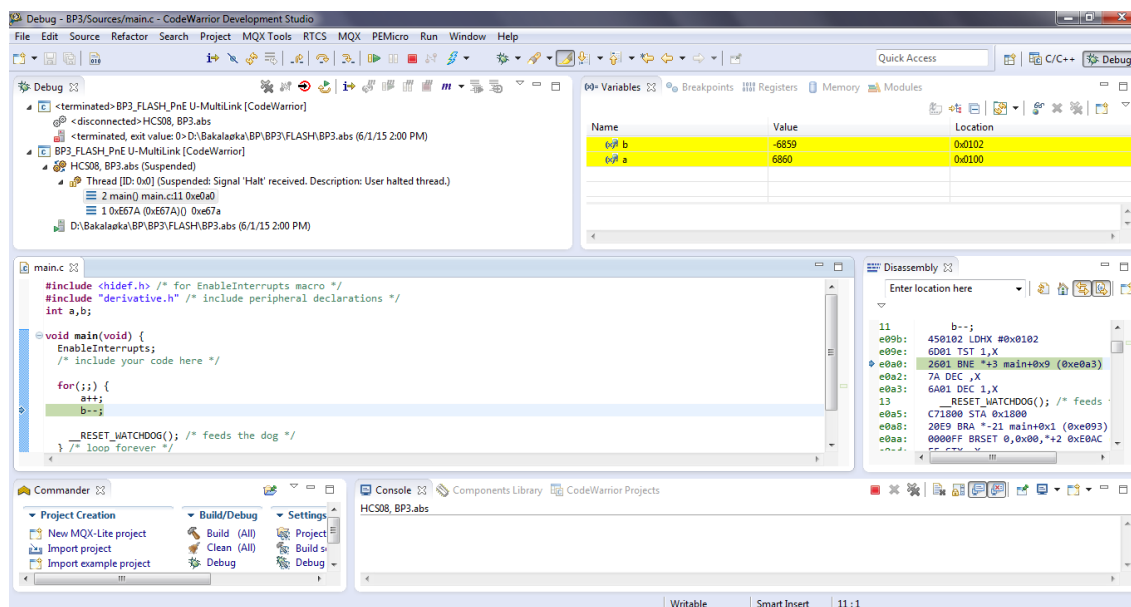
Obrázek 14 Codewarrior - Program pro mikroprocesor

Po napsání programu můžeme nechat program přeložit, zkontrolovat v něm chyby a následně nahrát do mikroprocesoru. Toto provedeme pomocí tlačítka Debug, které se nachází na hlavním panelu (viz Obrázek 15). Pro odeslání programu do mikroprocesoru musíme mít samozřejmě desku kitu propojenou s PC a oživenou.



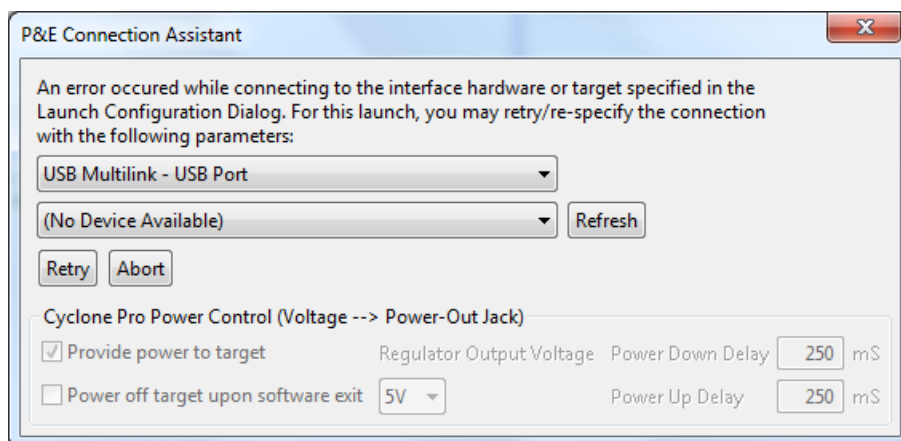
Obrázek 15 Codewarrior - Debug

Po stisknutí tohoto tlačítka musíme vyčkat než se provede celý proces. Po dokončení tohoto procesu se nám zobrazí následující okno (viz Obrázek 16). Pomocí příslušných tlačítek můžeme spustit, zastavit nebo krokovat běh programu. Při překladu programu Codewarrior vytvoří i soubor s příponou abs, který budeme později potřebovat pro synchronizaci názvů proměnných v programu Freemaster.



Obrázek 16 Codewarrior

Kdyby se vyskytla nějaká chyba program Codewarrior nám to ohlásí. Například může nastat chyba v komunikaci (viz Obrázek 17).



*Obrázek 17 Codewarrior - Chyba v komunikaci*

Po úspěšném provedení všech kroků a nahrání programu do mikroprocesoru můžeme přejít k testování funkcí programu FreeMaster.

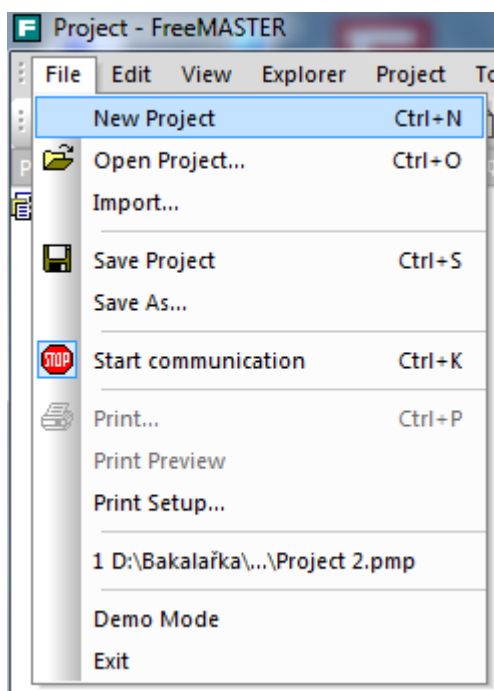


## 6 FREEMASTER - PRAKTICKÉ OVĚŘENÍ FUNKCÍ

Na následujících stranách bude popsáno praktické ověření jednotlivých funkcí programu FreeMaster za použití kitu DEMO9S08QB8, který je popsán v předchozí kapitole (viz kapitola 4.Kit DEMO9s08qb8). Před zahájením testování je nutné mít naprogramovaný program nahraný v mikroprocesoru. Pro testování je nezbytné vytvořit nový projekt v programu FreeMaster a následně nastavit jeho specifikace a komunikaci. Toto nastavení je popsáno v následující kapitole.

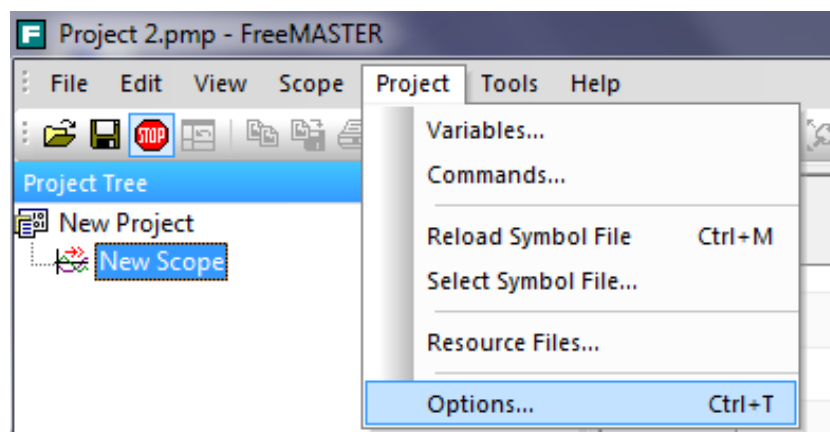
### 6.1 Vytvoření a Nastavení projektu

Pro testování níže uvedených funkcí programu FreeMaster bylo zapotřebí vytvořit nový projekt. Ten lze vytvořit následujícím způsobem (File->New Project) nebo přes klávesovou zkratku Ctrl + N (viz Obrázek 18).



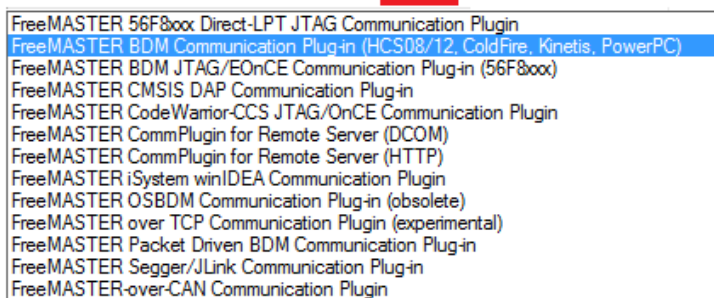
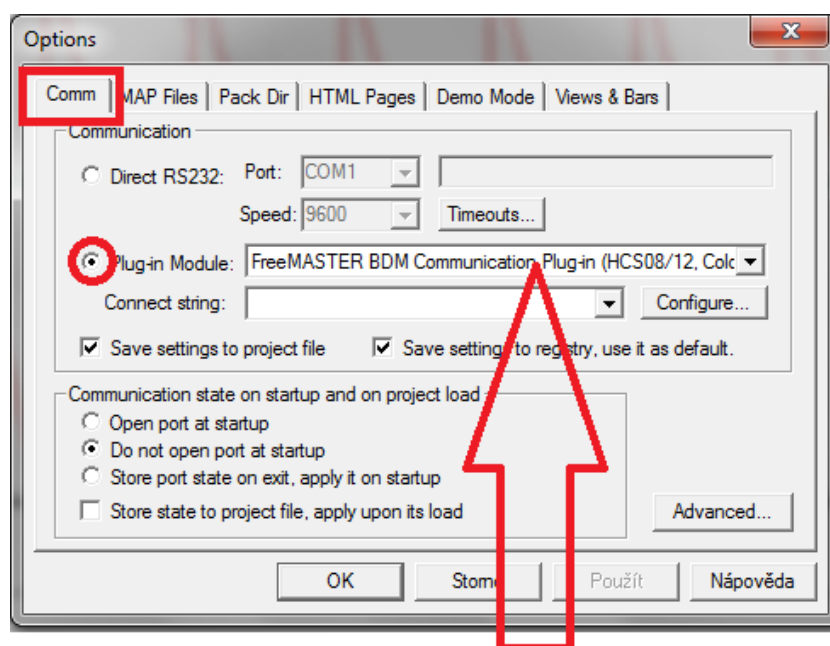
Obrázek 18 FreeMaster - New Project

Dalším krokem, po vytvoření nového projektu, je správným způsobem nakonfigurovat komunikaci programu FreeMaster s deskou kitu. Do nastavení projektu se přistupuje přes položku (Project->Options) nebo zkratkou Ctrl + T (viz Obrázek 19).



Obrázek 19 FreeMaster - Options

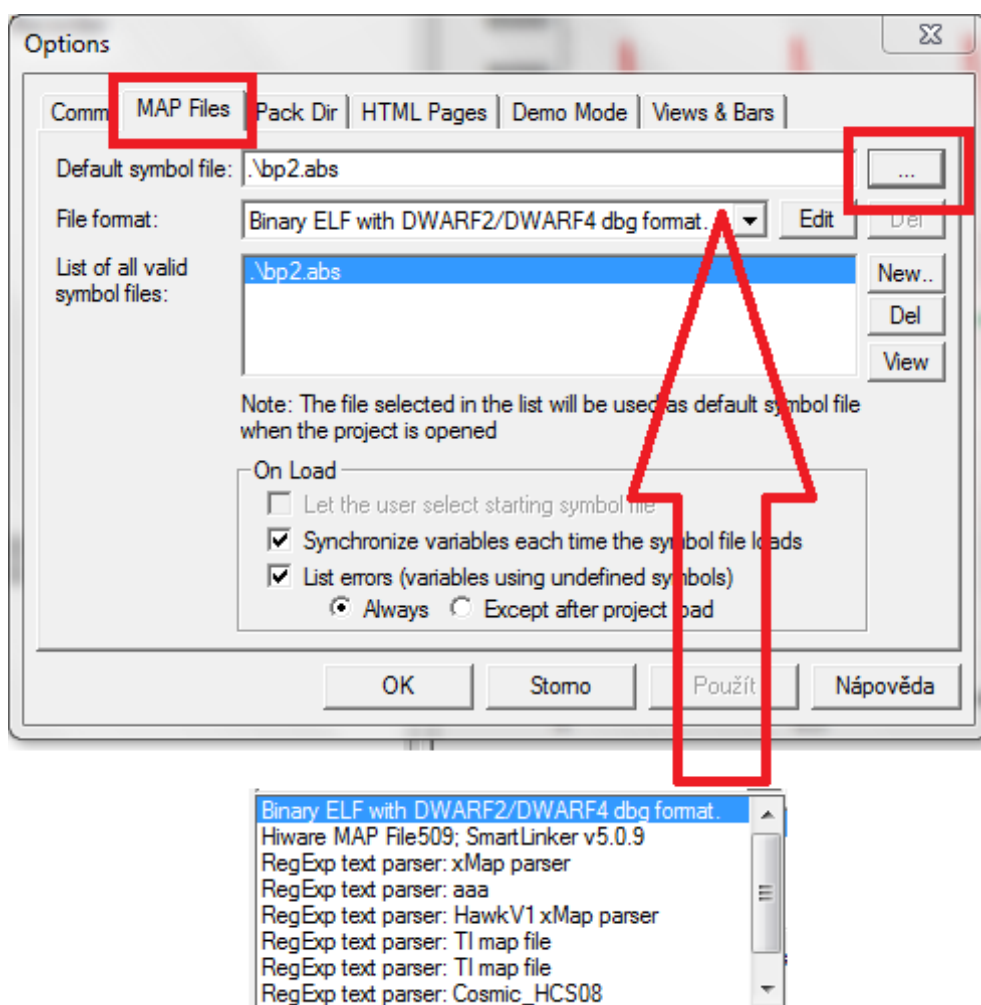
V Nastavení je zapotřebí vybrat záložku Comm tzn. Communication (Komunikace). Na této záložce se konfiguruje komunikace například přes RS232 nebo jiné. Pro kit DEMO9S08QB8 je zapotřebí zatrhnout Plug-in Module a z rolovací lišty vybrat položku Free MASTER BDM Communication Plug-in (HCS08/12,ColdFire, Kinetis, PowerPC), která je zapotřebí pro správnou komunikaci s deskou kitu. (viz Obrázek 20).



Obrázek 20 FreeMaster - Communication

### 6.1.1 Synchronizace názvů proměnných

V položce Options ještě zrovna lze provést nastavení v záložce MAP Files. Zprv je zde potřeba nastavit cestu k souboru s příponou abs a následně vybrat typ formátu souboru (Binary ELF with DWARF2/DWARF4 dbg format),(viz Obrázek 21). Tento soubor slouží k synchronizaci Názvů proměnných s nástrojem Codewarrior, aby se nemuseli ručně hledat jednotlivé adresy proměnných v paměti. Tento soubor získáme po přeložení programu pro mikroprocesor v programu Codewarrior. Po dokončení všech nastavení můžeme tyto změny uložit stisknutím tlačítka OK.



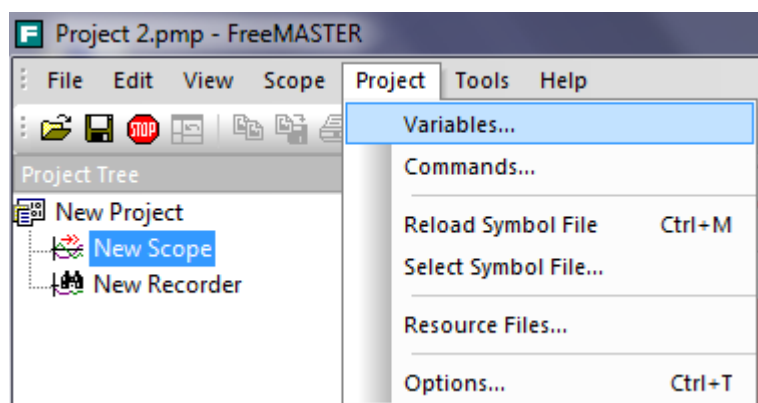
Obrázek 21 FreeMaster - MAP Files

## 6.2 Nastavení Proměnných

FreeMASTER komunikuje s deskou kitu mikropočítače přes specifický komunikační protokol. Tento protokol podporuje odesílání příkazů z aplikace v PC do

desky kitu mikropočítače a čtení nebo zápis do jeho proměnných. Všechny příkazy a proměnné použité v rámci projektu FreeMASTER musí být uvedeny v programu pro mikropočítač.

Abychom mohli snadno pracovat s proměnnými z programu CodeWarrior, je zapotřebí provést krok z kapitoly 6.1.1 Synchronizace názvů proměnných. Po jeho úspěšném dokončení můžeme přejít k nastavení jednotlivých proměnných. Do seznamu proměnných (Variables List) se přistupuje z menu přes položku Projekt (Project->Variables List), (viz Obrázek 22).

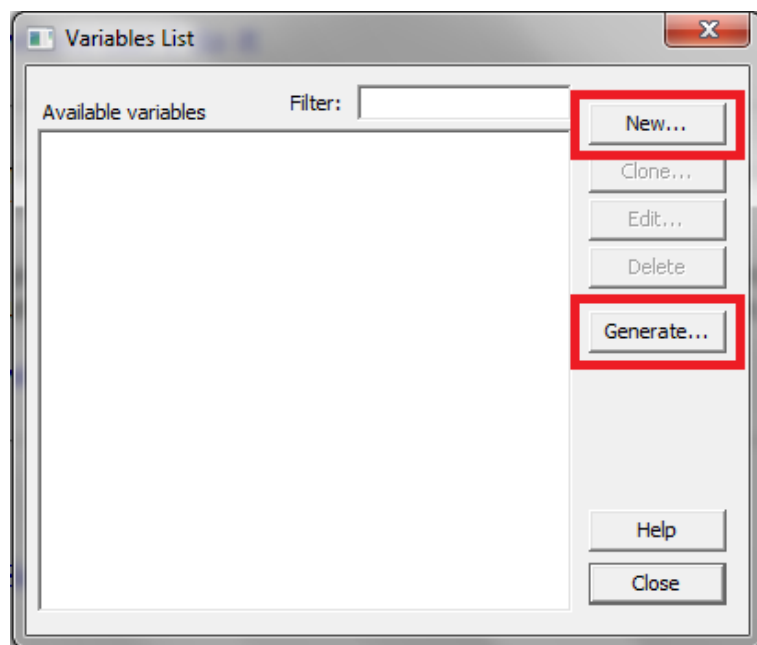


Obrázek 22 FreeMaster - Variables

Po zobrazení Seznamu proměnných (Variables List) můžeme proměnné vytvářet několika způsoby. Chcete-li definovat novou proměnnou, použijte tlačítko Nová Proměnná (New Variable), (viz Obrázek 23). Tím se otevře dialogové okno, kde si můžete nastavit vlastnosti proměnné popsané v kapitole 6.2.2 Možnosti Proměnných. Nebo můžeme proměnné hromadně vygenerovat. Tato možnost je popsána v následující kapitole.

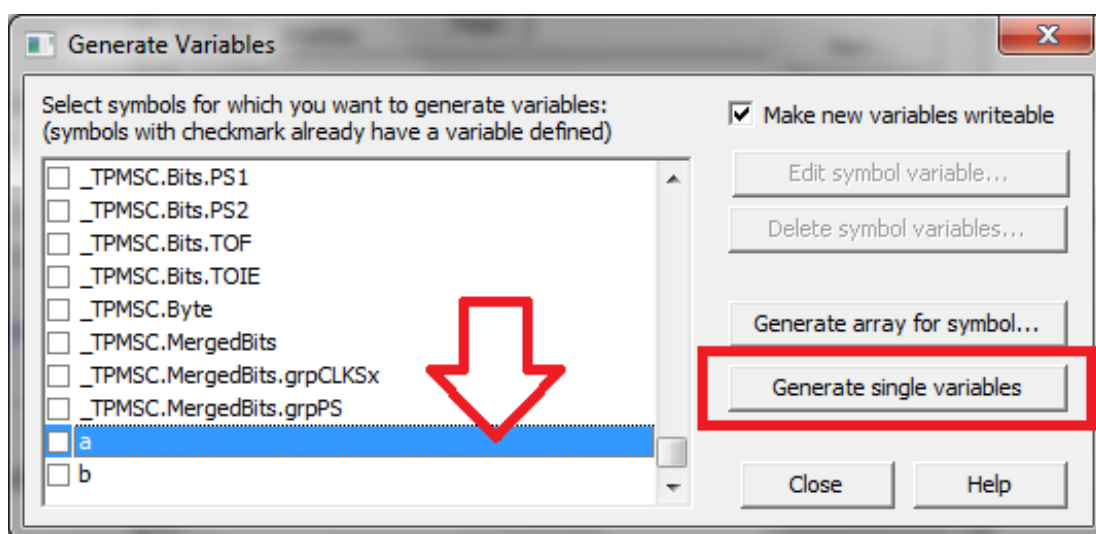
### 6.2.1 Hromadné generování proměnných

Pro hromadné generování proměnných ze souboru s příponou .abs je zapotřebí kliknout na položku Generovat Proměnné (Generate Variables), (viz Obrázek 23), která způsobí to, že se otevře nabídka, odkud bude možno hromadně vygenerovat nové proměnné.



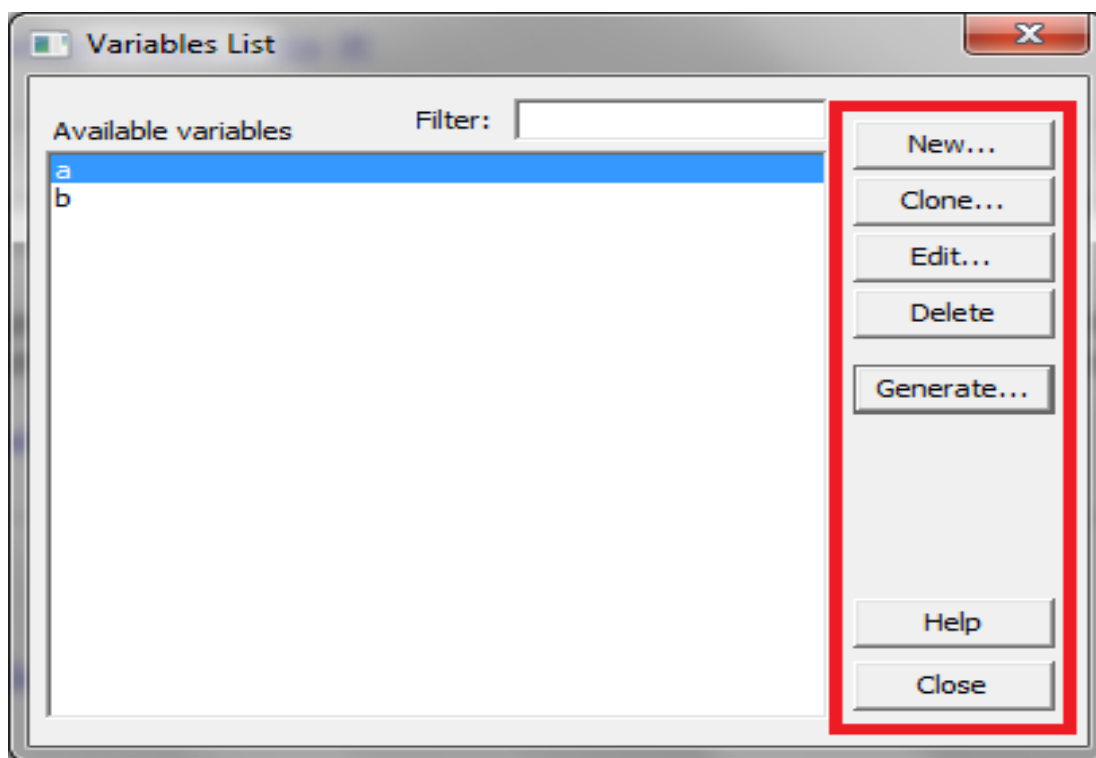
Obrázek 23 FreeMaster - Variables List

Po otevření nabídky Generovat Proměnné (Generate Variables), vyhledáme požadované proměnné, označíme je a kliknutím na položku Generovat samostatnou proměnnou (Generate Single Variables), (viz Obrázek 24) tímto přidáme proměnnou do Seznamu Proměnných (Variables List). Po úspěšném zvládnutí tohoto kroku se proměnné objeví v seznamu proměnných (Variables List). Tento postup opakujeme pro všechny požadované proměnné. Po úspěšném přidání všech požadovaných proměnných můžeme tuto nabídku zavřít.



Obrázek 24 FreeMaster - Generate Variables

V seznamu proměnných, jak je vidět na obrázku (viz Obrázek 25), se nachází tlačítka Nová Proměnná (New Variable), Klonovat Proměnnou (Clone Variable), Upravit Proměnnou (Edit Variable), Smazat (Delete) a další, přes které můžeme taky vybrané proměnné upravovat dle požadavků.

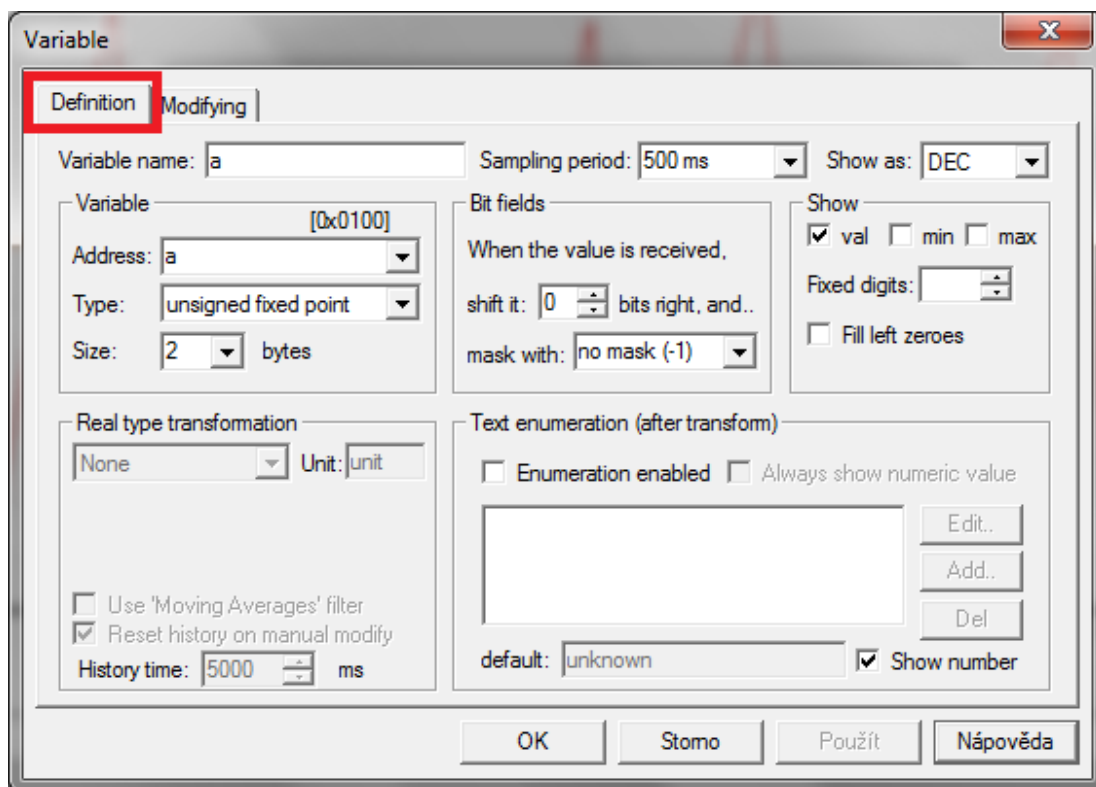


Obrázek 25 FreeMaster - Edit Variables

Pokud potřebujete vytvořit kopii existující proměnné, vyberte původní proměnnou a stiskněte tlačítko Klonovat (Clone). Tlačítko Upravit Proměnnou (Edit Variable) otevře stejné okno, jako při novém vytváření proměnné, kde je možno nastavit vlastnosti proměnné. Po stisknutí tlačítka Odstranit (Delete), je uživatel vyzván k potvrzení vymazání a vybraná proměnná bude vymazána.

### 6.2.2 Možnosti Proměnných

Pomocí tlačítka Upravit proměnnou (Edit Variable) lze definovat vlastnosti vybrané proměnné. V zobrazeném okně se nachází dvě záložky, v kterých lze definovat příslušné vlastnosti proměnné (viz Obrázek 26).



Obrázek 26 FreeMaster - Variable Definition

Na záložce Definovat (Definition) lze nastavit obecné vlastnosti proměnné:

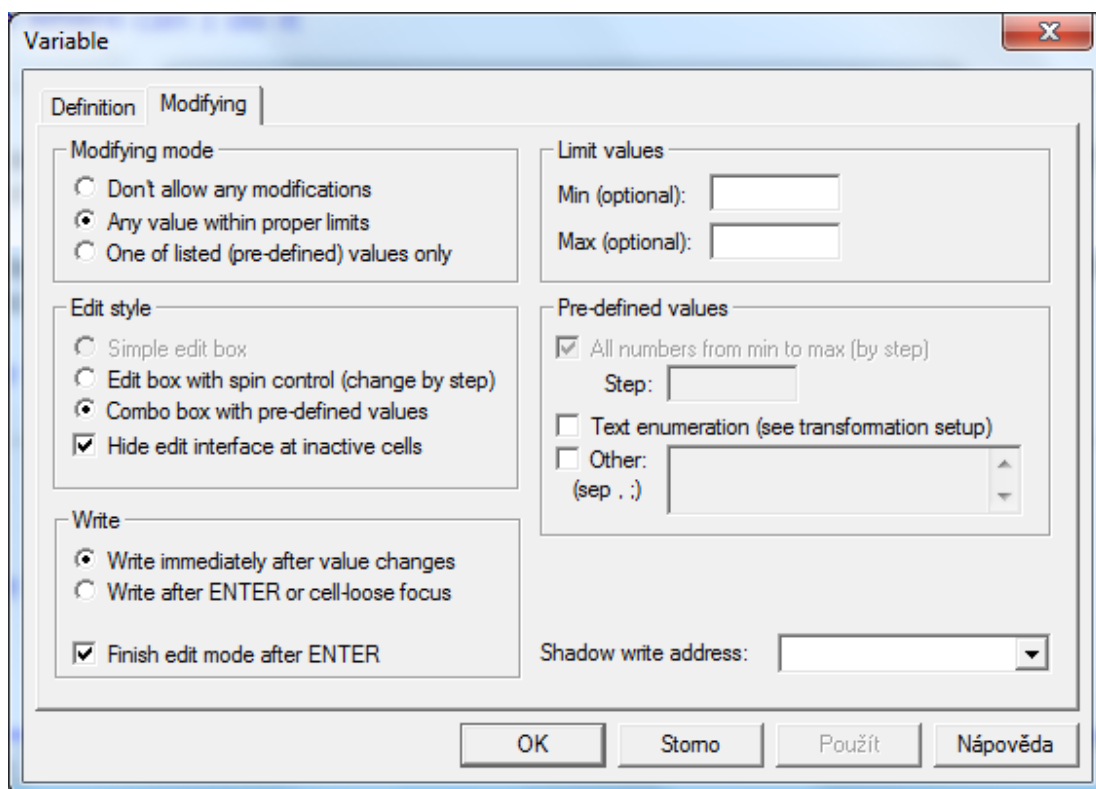
- Variable name = Název proměnné jako variabilní identifikátor v projektu
- Sampling period = Je frekvence načítání dat z proměnné
- Show as = Je formát, ve kterém je hodnota proměnné zobrazována. (DEC, HEX, BIN, ASCII nebo REAL).
- Variable = Zde se nastavují informace o proměnné:
- Address = Fyzická adresa proměnné v paměti. Přestože lze zadat přímou hexadecimální hodnotu, je možno vybrat přímo název proměnné ze seznamu proměnných, pokud byly názvy synchronizovány z externího souboru. (viz kapitola 6.1.1 Synchronizace názvů proměnných).
- type = Typ proměnné, jak je definována v cílové aplikaci
- Size = Velikost proměnné, jak je definována v cílové aplikaci
- Bit fields = parametry pro extrakci jednoho bitu nebo více bitů z dané proměnné
- Show = Parametry v této oblasti nastavení určují, jak bude proměnná zobrazena
- Val, min, max = Zaškrtněte, pokud chcete zjišťovat aktuální hodnotu proměnné nebo jen maximální a minimální hodnotu proměnné. Maximální hodnoty lze

obnovit klepnutím pravým tlačítkem myši na proměnnou v okně Variable watch a výběrem příkazu Reset Min / Max.

- Real type transformation = Když je nastaven formát na REAL, lze definovat další číselné zpracování, která se vztahuje na hodnoty proměnné.
- Text enumeration = Umožňuje popsat význam některých proměnných a přiřadit textový popis ke každé z nich, který je pak zobrazen u proměnné ve Variable watch společně nebo namísto číselné hodnoty. Pomocí tlačítek Edit, Add a Del lze upravovat, přidávat a mazat tyto popisky.
- Default = výchozí textový popis, který se zobrazí, když není odpovídající text nalezen v tabulce.

Na záložce Modifying lze nastavit tyto vlastnosti proměnné (viz Obrázek 27):

- Don't allow any modifications = Proměnná je pouze pro čtení, Všechna ostatní nastavení na této stránce jsou zakázány
- Any value within proper limits = Můžete zadat min a / nebo Max hodnotu a jiné.

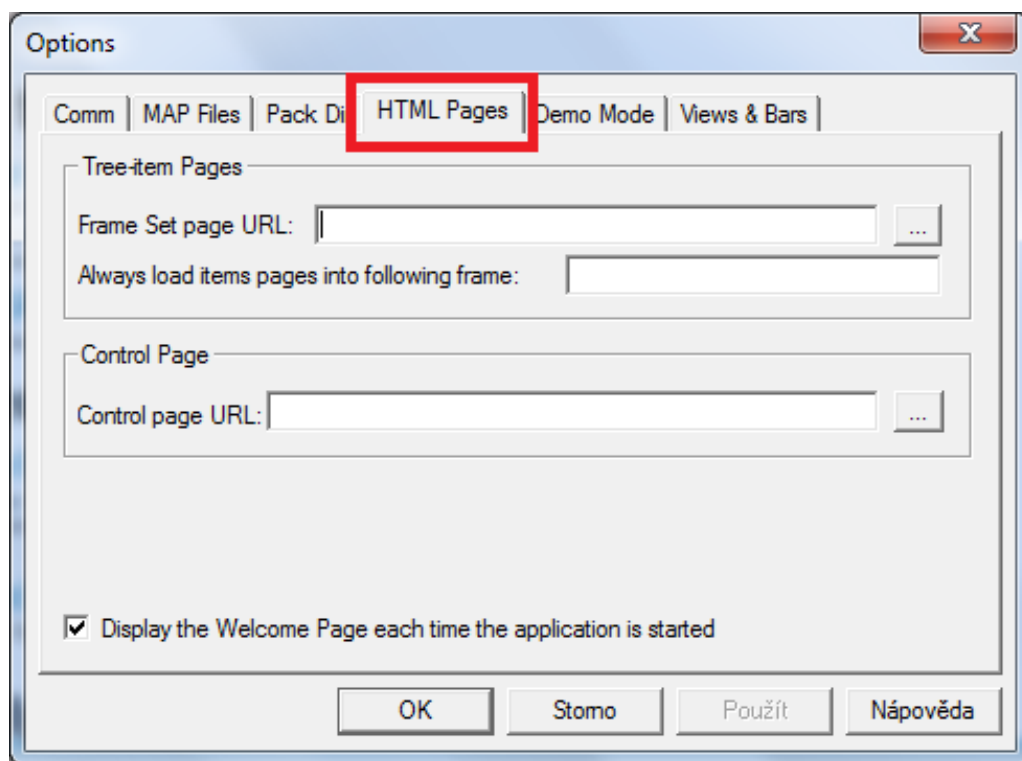


Obrázek 27 FreeMaster - Modifying



### 6.3 Nastavení vizuální stránky HTML

V programu FreeMaster je možné využívat HTML stránky, pro zobrazení popisu a ovládacích prvků, vztahující se k vybrané položce ve stromu projektu. Cesta nebo URL stránky jsou uvedeny ve vlastnostech pro každou položku stromu. Na záložce HTML Stránky (HTML Pages) v nastavení (Options), je možno nastavit cestu k HTML souboru, který se bude zobrazovat v hlavním okně aplikace (viz Obrázek 28).

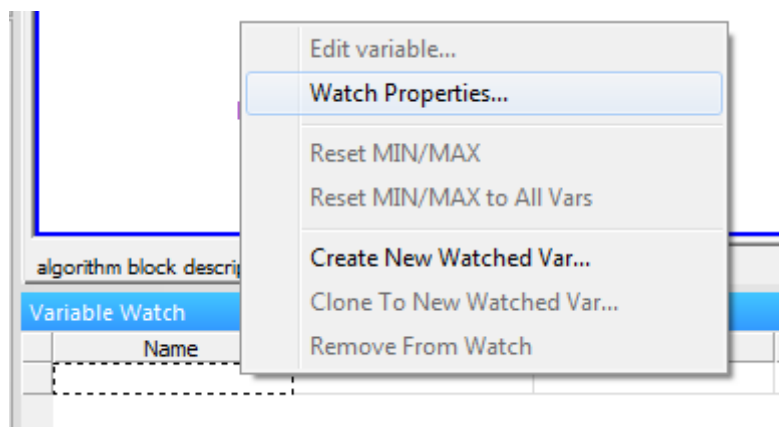


Obrázek 28 FreeMaster - HTML Pages

Vizuální HTML stránky mohou být propojeny přímo s deskou kitu pomocí Visual Basic Skriptů. To znamená, že mohou přistupovat k proměnným nebo příkazům.

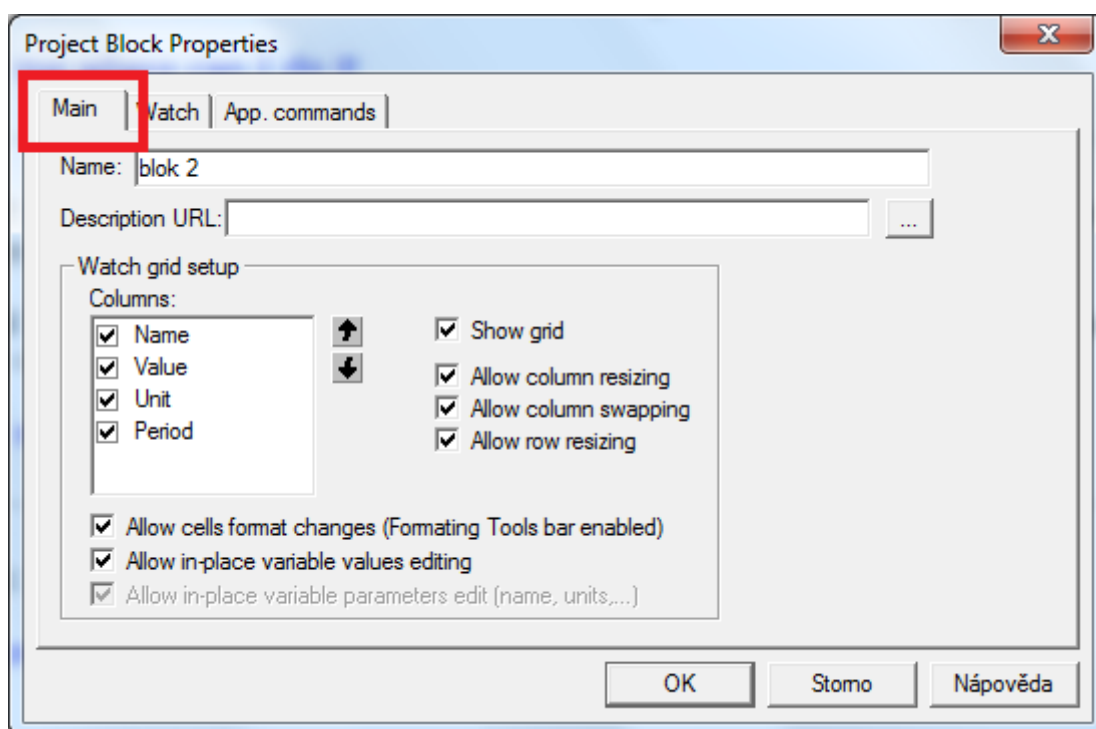
### 6.4 Sledování Proměnných

Aby bylo možno sledovat v programu hodnoty uložené v proměnných v první řadě je třeba tyto proměnné přidat na seznam Sledovaných proměnných (Variable Watch). Kliknutím pravým tlačítkem myši v oblasti tohoto okna se nám zobrazí nabídka (viz Obrázek 29).



Obrázek 29 FreeMaster Menu Variable Watch

Při výběru položky Vytvořit novou sledovanou proměnnou (Create New Watched Variable) se zobrazí stejná nabídka jako při vytváření nové proměnné (viz 6.2.2 Možnosti Proměnných). V případě že jsou proměnné již definovány, můžeme využít nabídky Vlastnosti sledování (Watch Properties). Po zvolení této možnosti se zobrazí okno Vlastnosti Bloku Projektu (Project Block Properties), (viz Obrázek 30).



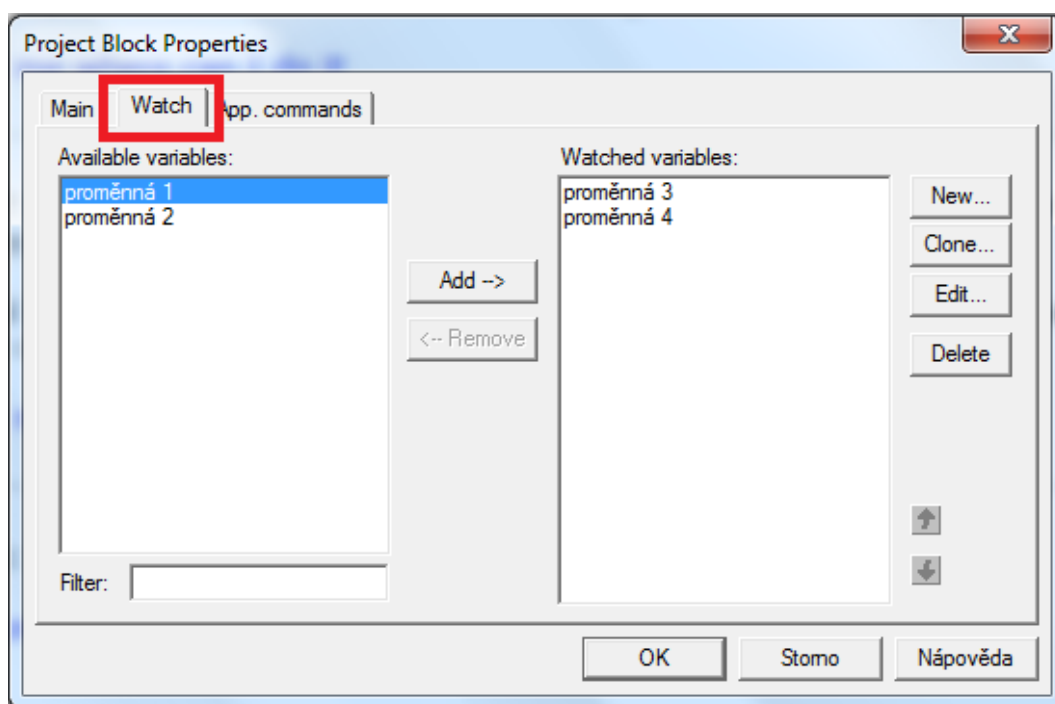
Obrázek 30 FreeMaster - Project Block Properties - Main

Na Hlavní (Main) záložce lze konfigurovat následující vlastnosti:

- Name = Název bloku projektu, který bude zobrazen ve stromové struktuře projektu

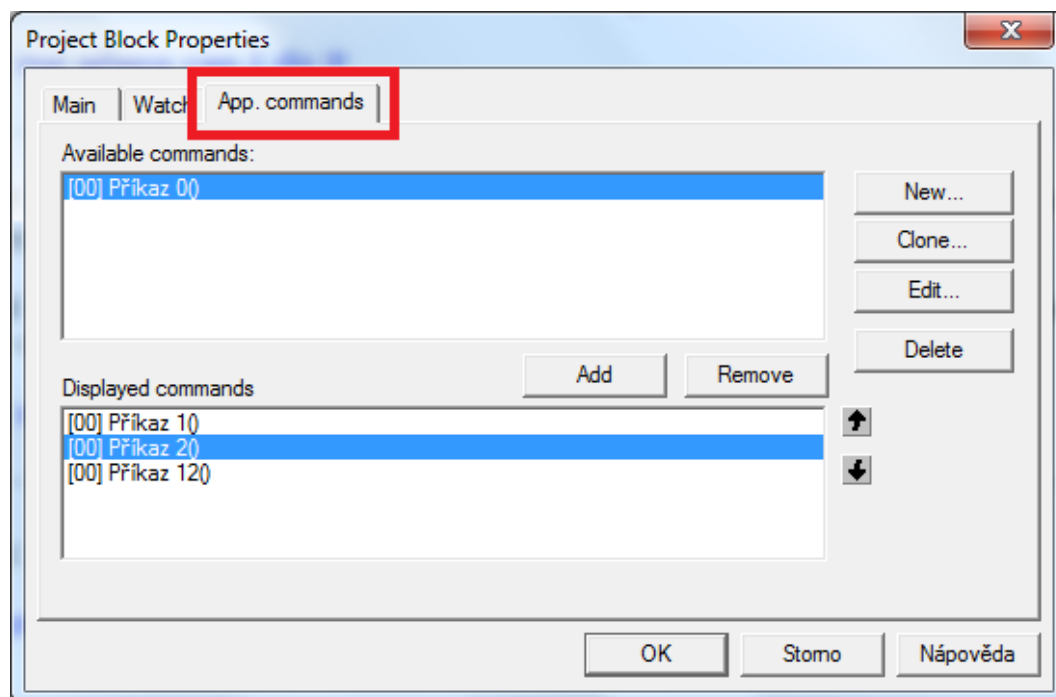
- Description URL = Zde je možné zadat URL nebo cestu k HTM nebo HTML souboru, který se zobrazí v hlavním okně.
- Watch-grid setup = Zde je možno zvolit které sloupce se zobrazí v tabulce Sledovaných proměnných (Variable Watch) (jméno, hodnota, jednotka, období); Lze určit pořadí sloupců pomocí tlačítek se šipkami nahoru a dolů;

V další záložce (Watch) je možno vybrat, které proměnné chceme sledovat v rámci tohoto bloku projektu (viz Obrázek 31). Pomocí tlačítek Přidat (Add) a Odebrat (Remove) lze přidávat a odebírat proměnné ze sledovaného seznamu. Na bočním panelu jsou tlačítka Nová (New), Klonovat (Clone), Upravit (Edit) nebo Smazat (Delete) pomocí kterých lze pracovat s proměnnými dle požadavků.



Obrázek 31 FreeMaster - Project Block Properties - Watch

Na poslední záložce (viz Obrázek 32) v nastavení je možno nastavit "příkazy" v rámci bloku projektu.



Obrázek 32 FreeMaster - Project Block Properties - App. Commands

FreeMASTER komunikuje s deskou kitu a umožňuje čtení a zápis hodnot do proměnných anebo taky umožňuje zaslat tzv. "Příkazy" (App. Commands). (viz Kapitola 6.5 Příkazy)

Pomocí tlačítek Přidat (Add) a Odebrat (Remove) lze přidávat a odebírat příkazy ze seznamu pro rychlý přístup, které budou k dispozici v podokně Rychlý přístup v rámci tohoto bloku projektu. Na bočním panelu jsou tlačítka Nový (New), Klonovat (Clone), Upravit (Edit) nebo Smazat (Delete) pomocí kterých lze pracovat s příkazy dle požadavků. Po nastavení všech požadavků můžeme nastavení zavřít tlačítkem OK.

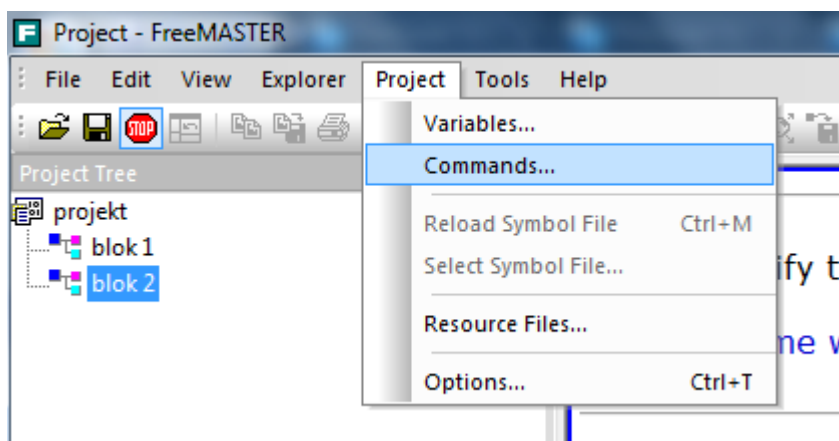
Úspěšné přidání sledovaných proměnných do okna Watch Variable můžeme vidět na následujícím obrázku (viz Obrázek 33). Ve sloupci Value můžeme sledovat aktuální hodnotu uloženou v proměnné.

Variable Watch				
	Name	Value	Unit	Period
	a	31091	DEC	1000
	b	-31091	DEC	1000

Obrázek 33 FreeMaster - Variable Watch

## 6.5 Příkazy

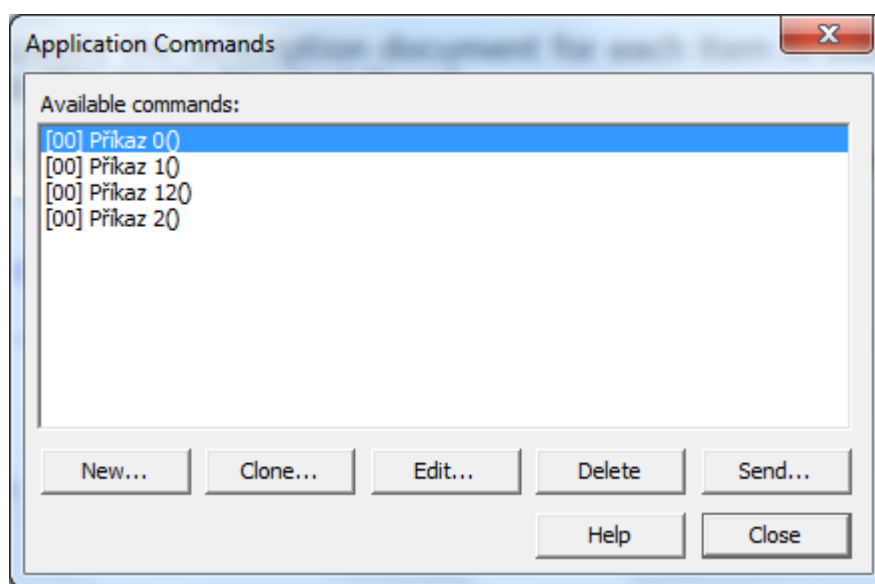
Seznam příkazů definovaných v projektu lze otevřít přes volbu Projekt (Project) -> Příkazy (Commands), (viz Obrázek 34).



Obrázek 34 FreeMaster - Commands

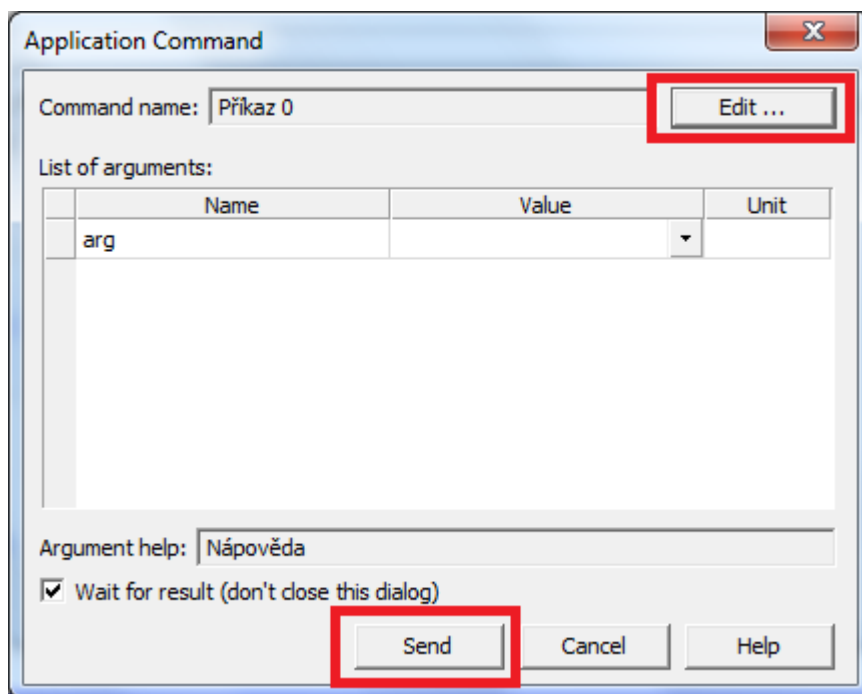
Pomocí tlačítek, (viz Obrázek 35) které jsou na spodním panelu, lze pracovat s příkazy dle požadavků.

- New = Vytvoří nový příkaz aplikace
- Edit = Upraví vlastnosti vybraného příkazu aplikace
- Clone = Vytvoří kopii vybraného příkazu
- Delete = Odstraní vybraný příkaz
- Send = Otevře rozhraní, které umožňuje, odesílat příkazy do desky kitu.



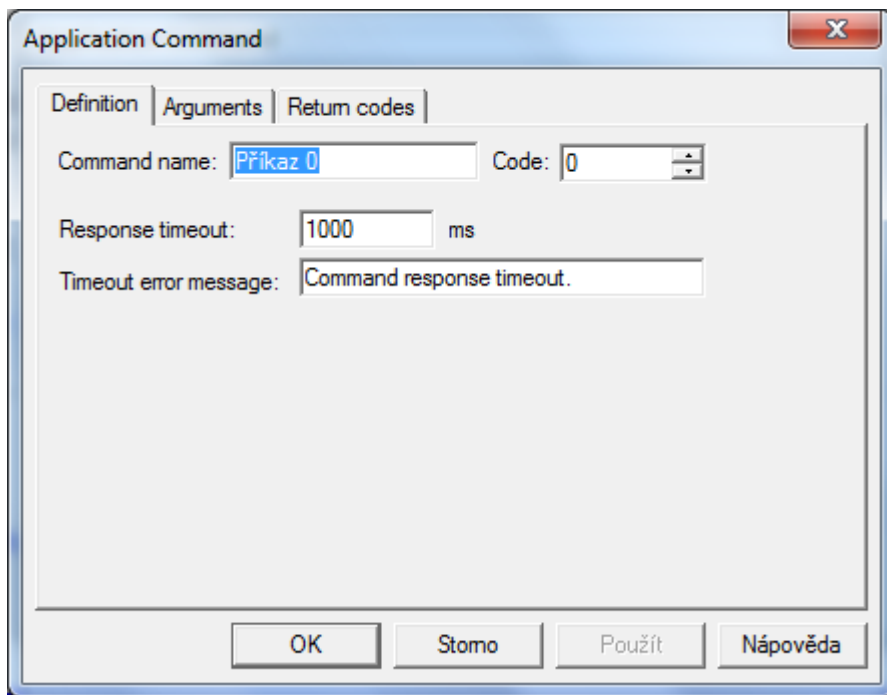
Obrázek 35 FreeMaster - App. Commands

Při využití tlačítka Odeslat (Send) se otevře okno ze kterého je možno posílat příkazy(hodnoty) do desky kitu. U každého argumentu, můžete definovat nápovědu, která se zobrazí v tomto dialogu při zadávání hodnoty (viz Obrázek 36).



Obrázek 36 FreeMaster - App. Commands - Send

Před odesláním příkazu, pomocí tlačítka Odeslat (Send), je možnost příkaz ještě upravit pomocí tlačítka Edit. Toto tlačítko má stejnou funkci jako tlačítko v okně Příkazy (Commands) na předešlém obrázku (viz Obrázek 35). Při využití tohoto tlačítka se otevře následující nastavení příkazu (viz Obrázek 37). Na záložce Definice (Definition) je možné konfigurovat jméno příkazu, které se bude zobrazovat v Projektu a specifický jedno bajtový kód příkazu, který identifikuje příkaz v programu cílové desky. Response timeout je maximální doba po kterou bude FreeMaster čekat na odpověď z desky kitu pokud je tato možnost aktivována. Po uplynutí této doby se v okně upozornění zobrazí chybová zpráva zadaná v políčku Timeout error message.

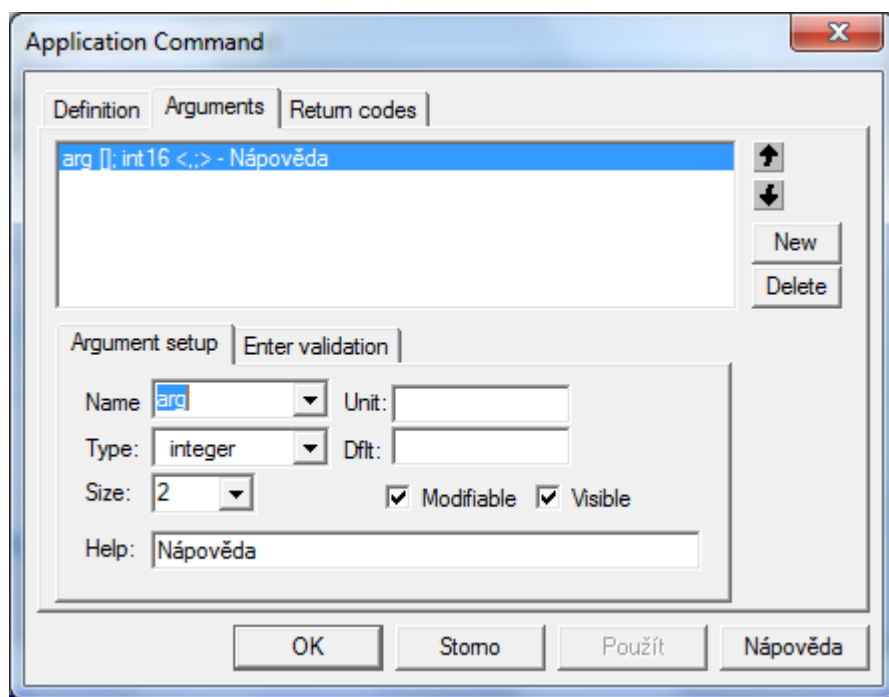


Obrázek 37 FreeMaster - App. Commands - Definition

Na záložce Argumenty (viz Obrázek 38), se dají definovat příkazy argumentů. Příkazy mohou do cílové desky předávat hodnoty do proměnných spolu se specifickým kódem. Pro vytvoření nového argumentu lze využít tlačítko Nový (New), pro smazání argumentu vybraného ze seznamu tlačítko Smazat (Delete) a pro změnu pořadí argumentu a šipky nahoru a dolů. V podzáložce Nastavení Argumentu (Argument setup) je možno nastavit tyto hodnoty:

- Name = Název argumentu, který se objeví v seznamu argumentů a v seznamu příkazů, když je uživatel vyzván k zadání argumentu hodnoty
- Type = Specifikuje numerickou hodnotu argumentu
- Size = Specifikuje velikost hodnoty argumentu v bajtech
- Unit = Může být zadán libovolný text, který bude zobrazen jako jednotky argumentu. Tento text není odesílán do cílové aplikace.
- Dflt = výchozí hodnota argumentu. Tato hodnota bude zobrazena v seznamu argumentů. Pokud není zadána žádná hodnota, musí uživatel zadat hodnotu, při každém odesílání příkazu.
- Modifiable = Pokud je toto políčko zaškrtnuto, uživatel nemůže změnit výchozí hodnotu argumentu v seznamu argumentů.

- Visible = Pokud toto políčko není zaškrtnuto, argument nebude zobrazen v seznamu argumentů a jeho výchozí hodnota bude vždy odeslána do cílové aplikace.
- Help = Jakékoli textové informace, které se zobrazí v dialogovém okně příkazu, když bude uživatel vyzván pro zadání hodnoty argumentu.

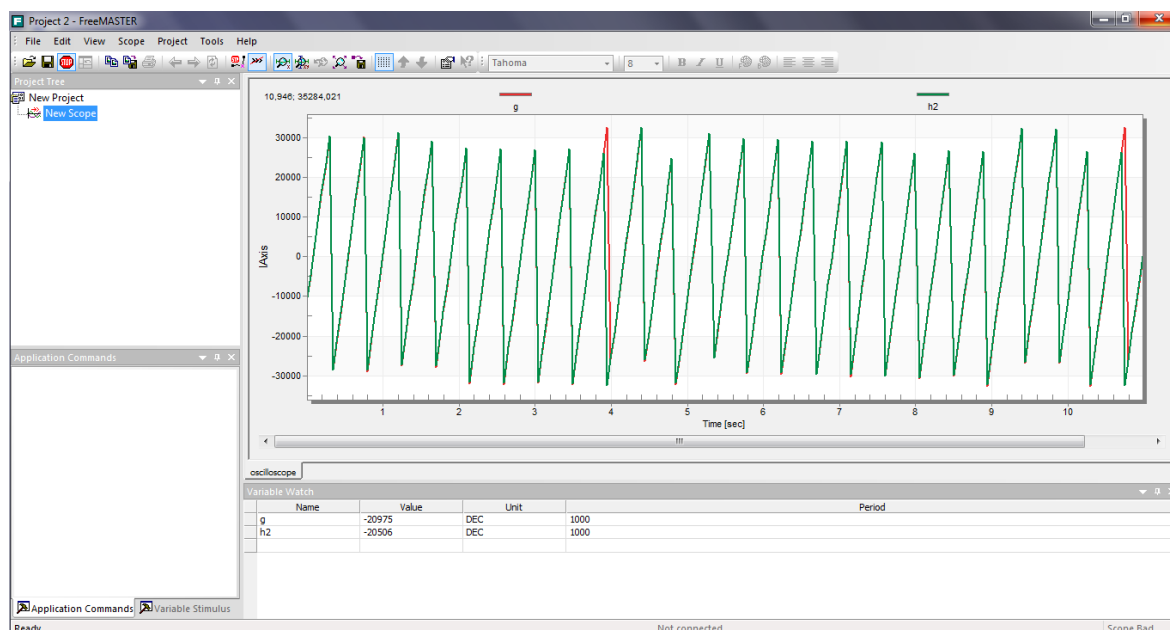


Obrázek 38 FreeMaster - App. Commands - Arguments

## 6.6 Funkce Oscilloscope

Bylo prakticky ověřeno, že tato funkce je vhodná pro sledování takových funkcí v reálném čase, které nemají velkou změnu průběhu nebo frekvence v čase. Při sledování funkcí, které mají velkou změnu průběhu nebo frekvence je vzorkování nedostatečné, protože je omezeno rychlostí přenášáním dat po sériové sběrnici USB. Při sledování většího počtu funkcí nebo proměnných najednou je tato frekvence vzorkování ještě více omezena. Maximální počet sledovaných funkcí je osm. Detail této funkce, najdeme na následujícím obrázku (viz Obrázek 39).

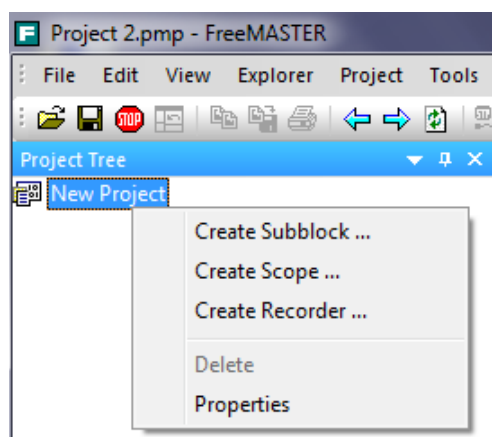




Obrázek 39 Detail Funkce Oscilloscope

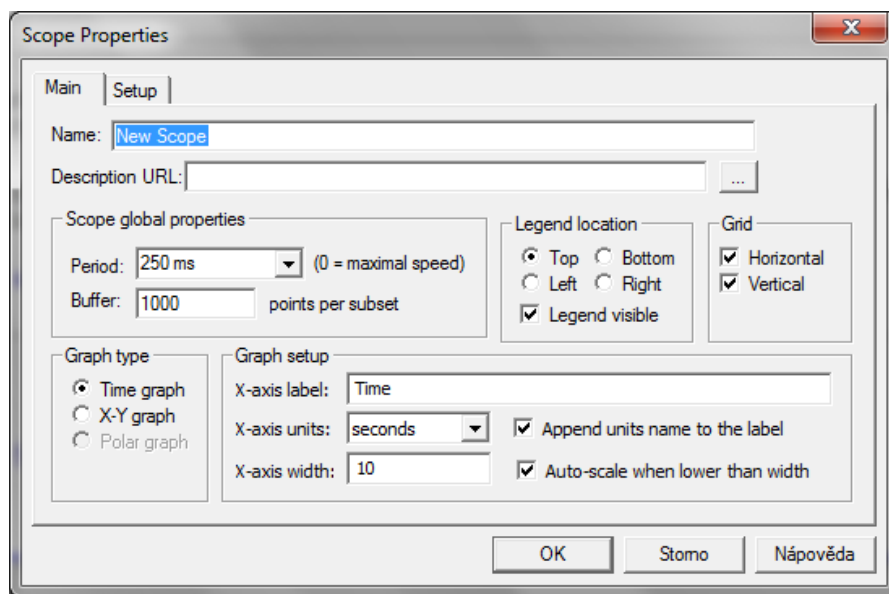
### 6.6.1 Vytvoření a Nastavení funkce Oscilloscope

Postup pro vytvoření zobrazovací funkce Osciloskop je následující. Nejprve je potřeba kliknout pravým tlačítkem na jméno projektu ve stromu projekt (Project Tree). Potom se nám zobrazí nabídka zobrazena na následujícím obrázku (viz Obrázek 40)



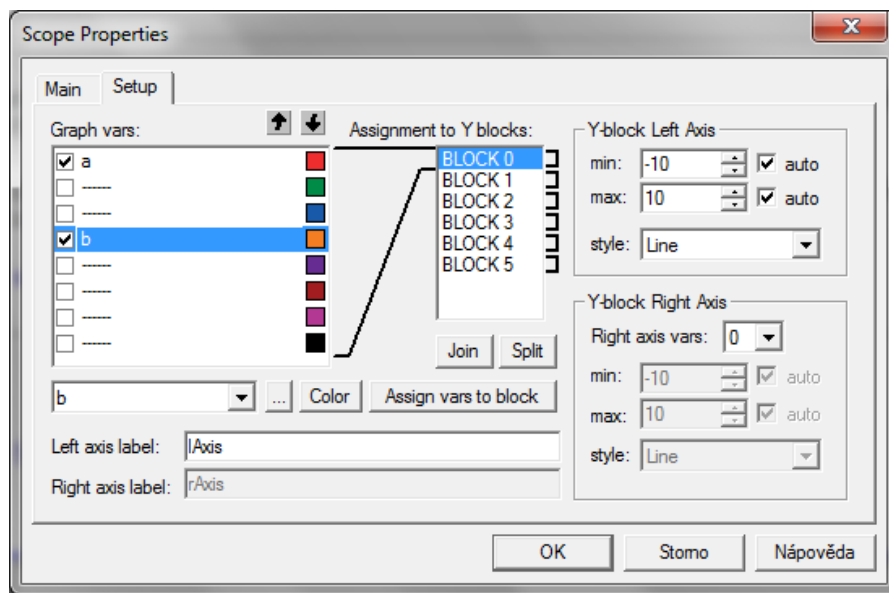
Obrázek 40 FreeMaster - Scope

Vybereme možnost Create Scope... pro vytvoření zobrazovací funkce osciloskop. Po výběru předešlé možnosti se nám zobrazí následující okno (viz Obrázek 41). V záložce Main můžeme nastavovat základní vlastnosti jako je jméno, jaký typ grafu, rychlost obnovování a jiné.



Obrázek 41 FreeMaster - Scope Properties - Main

V záložce Setup (viz Obrázek 42) můžeme nastavit přiřazení jednotlivých proměnné k barvám a taky další pomocné nastavení. Po nastavení všech požadavků můžeme, stisknou tlačítko OK, které uloží všechny nastavené hodnoty a vytvoří požadovanou zobrazovací funkci (viz Obrázek 39).



Obrázek 42 FreeMaster - Scope Properties - Setup

## 6.7 Funkce Recorder

Tato funkce je vhodná pro sledování rychlejších funkcí. Tato funkce nebyla prakticky ověřena. U této funkce je zapotřebí nahrát do programu mikroprocesoru část kódu, která zprostředkuje tuto funkci. Potom při navázání komunikace a spuštění funkce

recorder v programu Freemaster je v mikroprocesoru spuštěno na určitý krátký časový úsek zaznamenávání proměnných. Následně je tento balíček dat poslán do programu Freemaster kde je vyhodnocen. Nahrávání části kódu do programu bylo prováděno dle manuálu výrobce, tento krok byl z nezjištěných důvodů neúspěšný. Pravděpodobně díky nekompatibilitě ovladačů daného kitu nebo díky špatně zvolené komunikaci s PC. K této funkci je pravděpodobně zapotřebí komunikace přes sériovou linku RS-232.

## ZÁVĚR

V teoretické části bakalářské práce je shrnuta problematika zabývající se základními částmi počítačů respektive mikropočítačů. Kde jsem stručně popsal základní typy a druhy částí mikropočítače. V další kapitole je stručně popsán jazyk C a jsou shrnuty základní pojmy užívané v oblasti programování nejen mikropočítačů.

Hlavní obsah teoretické části bakalářské práce tvoří popis programu FreeMaster od firmy Freescale. V prvních z podkapitol jsou popsány základní vlastnosti programu FreeMaster a taky teoretické využití tohoto programu. Dále je zde popsáno stažení a instalace programu a všech jeho částí potřebných ke správné funkčnosti programu. Je zde taky popsán princip funkčnosti Freemasteru.

V první z kapitol praktické části je popsána deska kitu DEMO9S08QB8, která byla použita pro praktické ověření funkcí programu Freemaster. Na této desce se nachází rozsáhlé spektrum periferních obvodů pro různé funkce. Tato deska kitu má výhodu, že je napájena USB kabelem a zjednodušuje její použití. Další kapitola se zabývá programem Codewarrior, který byl využit v bakalářské práci pro potřeby naprogramování mikroprocesoru. Program Codewarrior má poměrně rozsáhlé funkce a možnosti. My jsme ho využili pro napsání programu pro mikroprocesor a následné odladění a nahrání do mikroprocesoru. Všechny kroky, jsou zaznamenány pomocí obrázků s komentářem, včetně programu pro mikroprocesor.

Po naprogramování mikroprocesoru bylo přistoupeno k testování funkcí a možností programu Freemaster, které bylo hlavním cílem bakalářské práce. V příslušné části bakalářské práce je popsáno toto testování. Jsou zde taky popsány možnosti jednotlivých nastavení, které bylo nutno provést pro správnou funkci programu Freemaster. Například nastavení a navázání komunikace s deskou kitu a následné testování jednotlivých funkcí, které spočívá v přístupu Freemasteru do paměti mikroprocesoru. Freemaster může z této paměti získávat data z jednotlivých proměnných v reálném čase. Tyto data může Freemaster následně vyhodnocovat a zobrazovat pomocí speciálních zobrazovacích funkcí. První ze zobrazovacích testovaných funkcí byla funkce Oscilloscope, která pracuje na podobném principu jako běžný CRT osciloskop. Je vhodná pro zobrazování pomalejších průběhů funkcí, protože rychlost vzorkování je omezena rychlostí sběrnice USB. Tato funkce dokáže zobrazovat data až z osmi proměnných najednou, ale čím více proměnných je zobrazováno současně tím více klesá rychlost vzorkování. Hlavním přínosem této

funkce je diagnostika mikroprocesorů, která může být využita studenty ve výuce, při vývoji různých aplikací pro mikroprocesory. Další prověřovanou zobrazovací funkcí byla funkce recorder, která by měla být vhodná pro sledování rychlejších průběhů funkcí. Funkci recorder programu FreeMaster se mi bohužel nepodařilo prakticky ověřit z nezjištěného důvodu.

## SEZNAM POUŽITÉ LITERATURY

- [1] PINKER, Jiří. Mikroprocesory a mikropočítače. Praha: BEN - technická literatura, 2004. ISBN 80-730-0110-1.
- [2] Sběrnice. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2015-06-02]. Dostupné z: <http://cs.wikipedia.org/wiki/Sběrnice>
- [3] *Extract Microcontroller Embedded Firmware* [online]. In: . [cit. 2015-06-02]. Dostupné z: <http://www.extract-ic.com/extract-microcontroller-embedded-firmware/>
- [4] MANN, Burkhard. C pro mikrokontroléry: ANSI-C, kompilátory C, spojovací programy - linkery, práce s ATMELE AVR a MSC-51, příklady programování v jazyce C, nástroje pro programování, tipy a triky. Praha: BEN, 2003. ISBN 80-730-0077-6
- [5] Programování v jazyce C [online]. 2014 [cit. 2015-06-02]. Dostupné z: <http://www.itnetwork.cz/programovani-v-jazyce-c-zaklady>
- [6] MARTINEK, David. *Překlad programu* [online]. 2014 [cit. 2015-06-02]. Dostupné z: <http://www.fit.vutbr.cz/~martinek/clang/gcc.html>
- [7] Debugger. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2015-06-02]. Dostupné z: <http://cs.wikipedia.org/wiki/Debugger>
- [8] Strojový kód. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2015-06-02]. Dostupné z: [http://cs.wikipedia.org/wiki/Strojový\\_kód](http://cs.wikipedia.org/wiki/Strojový_kód)
- [9] *Freescale. FreeMASTER Run-Time Debugging Tool* [online]. 2015 [cit. 2015-01-15]. Dostupné z: [https://www.freescale.com/webapp/sps/site/prod\\_summary.jsp?code=FREEMASTER](https://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=FREEMASTER)
- [10] *Tutorial: FreeMASTER Visualization and Run-Time Debugging* [online]. [cit. 2015-06-02]. Dostupné z: <http://mcuoneclipse.com/2013/08/24/tutorial-freemaster-visualization-and-run-time-debugging/>
- [11] BARR, Michael a Anthony J MASSA. Programming embedded systems. 2nd ed. Sebastopol: O'Reilly, 2006, xxi, 301 s. ISBN 978-0-596-00983-0.

- [12] CATSOULIS, John. Designing Embedded Hardware. Sebastopol: O'Reilly Media, 2005. ISBN 978-0-596-00755-3.

**SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK**

RS232 Typ sériové linky

USB Univerzální sériová linka

BDM (Background Debug Mode) speciální sběrnice pro programování mikroprocesorů

CRT Typ obrazovky



**SEZNAM OBRÁZKŮ**

Obrázek 1 Zjednodušené schéma mikropočítače [1] .....	11
Obrázek 2 Detail mikropočítače [3].....	14
Obrázek 3 Úvodní stránka .....	21
Obrázek 4 Okno aplikace.....	22
Obrázek 5 Osciloskop [10] .....	23
Obrázek 6 Recorder [10].....	24
Obrázek 7 Deska kitu.....	26
Obrázek 8 Ukázka prostředí CodeWarrior .....	27
Obrázek 9 Codewarrior - New Project .....	28
Obrázek 10 Codewarrior - New Bareboard Project.....	28
Obrázek 11 Codewarrior - Devices .....	29
Obrázek 12 Codewarrior - Connections .....	29
Obrázek 13 Codewarrior - main.c.....	30
Obrázek 14 Codewarrior - Program pro mikroprocesor .....	30
Obrázek 15 Codewarrior - Debug.....	31
Obrázek 16 Codewarrior.....	31
Obrázek 17 Codewarrior - Chyba v komunikaci .....	32
Obrázek 18 FreeMaster - New Project .....	33
Obrázek 19 FreeMaster - Options.....	34
Obrázek 20 FreeMaster - Communication.....	34
Obrázek 21 FreeMaster - MAP Files .....	35
Obrázek 22 FreeMaster - Variables .....	36
Obrázek 23 FreeMaster - Variables List.....	37
Obrázek 24 FreeMaster - Generate Variables.....	37
Obrázek 25 FreeMaster - Edit Variables .....	38
Obrázek 26 FreeMaster - Variable Definition .....	39
Obrázek 27 FreeMaster - Modifying .....	40
Obrázek 28 FreeMaster - HTML Pages.....	41
Obrázek 29 FreeMaster Menu Variable Watch .....	42
Obrázek 30 FreeMaster - Project Block Properties - Main .....	42
Obrázek 31 FreeMaster - Project Block Properties - Watch.....	43
Obrázek 32 FreeMaster - Project Block Properties - App. Commands.....	44

Obrázek 33 FreeMaster - Variable Watch .....	44
Obrázek 34 FreeMaster - Commands .....	45
Obrázek 35 FreeMaster - App. Commands .....	45
Obrázek 36 FreeMaster - App. Commands - Send .....	46
Obrázek 37 FreeMaster - App. Commands - Definition .....	47
Obrázek 38 FreeMaster - App. Commands - Arguments .....	48
Obrázek 39 Detail Funkce Oscilloscope .....	49
Obrázek 40 FreeMaster - Scope.....	49
Obrázek 41 FreeMaster - Scope Properties - Main.....	50
Obrázek 42 FreeMaster - Scope Properties - Setup .....	50

## SEZNAM TABULEK

Tabulka 1 Podporované platformy .....	18
---------------------------------------	----

## SEZNAM PŘÍLOH

Příloha P I: disk CD s bakalářskou prací