VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

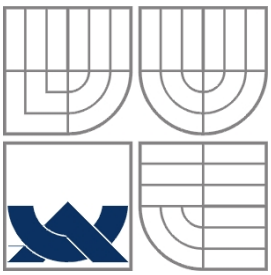
PROGRAMOVÁNÍ JEDNOČIPOVÝCH MIKROKONTROLÉRŮ

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

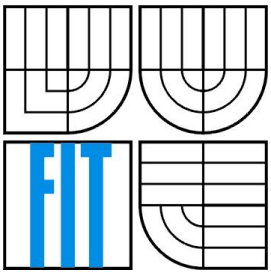
AUTOR PRÁCE
AUTHOR

ROMAN JIRKA

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

PROGRAMOVÁNÍ JEDNOČIPOVÝCH MIKROKONTROLÉRŮ

PROGRAMMING OF SINGLE-CHIP MICROCONTROLLER

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

ROMAN JIRKA

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. Ing. JOSEF SCHWARZ, CSc.

BRNO 2008

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2007/2008

Zadání bakalářské práce

Řešitel: **Jirka Roman**
Obor: Informační technologie
Téma: **Programování jednočipových mikrokontrolérů**
Kategorie: Vestavěné systémy

Pokyny:

1. Porovnejte architektury a vlastnosti vybraných jednočipových mikrokontrolerů.
2. Zpracujte metodiku tvorby aplikačních. programů využívajících vybrané periferní jednotky mikrokontrolérů.
3. Pro mikrokontrolér M68HC908LJ12 vypracujte sadu typických úloh v asembleru a jazyce C, využitelných ve výuce předmětu "Mikroprocesory a vestavěné systémy".
4. Navržené úlohy odlaďte v prostředí Code Warior a doložte vhodnými snímky z vývojového prostředí.

Literatura:

- Rozehnal, Z.: Mikrokontroléry Motorola HC11, Praha 2001, nakladatelství BEN, ISBN 80-86056-77-5.
- Predko, M.: Handbook of Microcontrollers, McGraw-Hill, ISBN 0079137164, 1998.
- Valvano, W.J.: Embedded microcomputer system, Brooks/Cole, USA, 1999, ISBN 0 534-36642-2.

Při obhajobě semestrální části projektu je požadováno:

- Splnění prvních dvou bodů zadání.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Schwarz Josef, doc. Ing., CSc.**, UPSY FIT VUT
Datum zadání: 1. listopadu 2007
Datum odevzdání: 14. května 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

doc. Ing. Zdeněk Kotásek, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Roman Jirka**
Id studenta: 78958
Bytem: Čsl. armády 370/27, 683 01 Rousínov
Narozen: 04. 04. 1986, Brno
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
bakalářská práce

Název VŠKP: Programování jednočipových mikrokontrolérů
Vedoucí/školitel VŠKP: Schwarz Josef, doc. Ing., CSc.
Ústav: Ústav počítačových systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☐ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

Abstrakt

Tato bakalářská práce se zabývá jednočipovými mikrokontroléry. Cílem je seznámit zájemce s nejznámějšími jádry univerzálních mikrokontrolérů, uvést vlastnosti jednotlivých jader, výhody, nevýhody a rozdíly. Součástí práce je sada příkladů pro mikrokontrolér M68HC908LJ12 vypracovaná ve vývojovém prostředí CodeWarrior.

Klíčová slova

Mikrokontrolér, Harvardská architektura, von Neumannova architektura, 8051, AVR, HC11, PIC, ST7

Abstract

This bachelor's work deals with single-chip microcontrollers. The main purpose is to introduce the best known cores of universal microcontrollers to users, to introduce characteristics of individual cores, advantages, disadvantages and differences. The part of this work is a set of examples for microcontroller M68HC908LJ12 worked out in development environment CodeWarrior.

Keywords

Microcontroller, Harvard architecture, von Neumann architecture, 8051, AVR, HC11, PIC, ST7

Citace

Roman Jirka: Programování jednočipových mikrokontrolérů, bakalářská práce, Brno, FIT VUT v Brně, 2008

Programování jednočipových mikrokontrolérů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Josefa Schwarze. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Roman Jirka
12. května 2008

Obsah

1 Úvod

První mikrokontrolér vznikl v roce 1971. Vznikl z toho důvodu, že při návrhu nějakého kombinovaného obvodu stačila jakákoli menší změna a musel se celý návrh dělat znovu. Kdežto při použití mikrokontrolérů stačí pouze změnit program. Mikrokontrolér nebo také jednočipový počítač je většinou monolitický integrovaný obvod obsahující jádro mikroprocesoru společně s nevolatilní pamětí programu (paměť, která udrží data i po odpojení napájení, např. ROM, PROM, FLASH), pamětí dat typu RAM a periferními obvody (logické vstupy/výstupy, časovače, PWM, atd.). Mikrokontrolér může obsáhnout celou aplikaci bez potřeby složitých podpůrných obvodů. Vyznačují se svojí spolehlivostí a kompaktností. Jsou určeny především pro jednoúčelové aplikace jako je měření, řízení, regulace, apod.

Postupem času se začaly mikrokontroléry používat stále častěji a v mnohem komplikovanějších aplikacích. Zvětšovaly se nároky na mikrokontroléry a současné typy přestávaly nárokům vyhovovat. Zejména se vyžadoval větší výkon a větší kapacita paměti dat a programu. Vznikaly tedy stále nové typy mikrokontrolérů a začalo je vyrábět více výrobců. V dnešní době můžeme mikrokontroléry nalézt téměř všude, a proto se stávají nedílnou součástí našeho života.

Pro programátora, který se rozhodl začít programovat jednočipové mikrokontroléry, je těžké zvolit si jednoho zástupce, který by mu nejvíce vyhovoval. Některá jádra se od sebe liší, některá jsou si naopak podobná. Této problematice je věnována druhá kapitola, kde jsou popsána jádra nejznámějších mikrokontrolérů. Ve třetí kapitole jsou popsány metodiky tvorby programů i s uvedenými příklady a čtvrtá kapitola obsahuje souhrnný popis sady příkladů.

2 Přehled jednočipových mikrokontrolérů

V této kapitole se budu věnovat převážně 8-bitovým mikrokontrolérům. Nebudu zde probírat jednotlivé periferie mikrokontrolérů, neboť toto téma je velice rozsáhlé a často velice dobře popsané výrobcem v manuálu k danému mikrokontroléru. Zaměřím se především na rozdíly mezi nejznámějšími univerzálními mikrokontroléry.

Nechci zde rozebírat všechny pojmy související s mikrokontroléry, ale uvedu zde aspoň ty nejzákladnější (viz [9]).

Von Neumannova architektura – pro tuto architekturu je typická společná paměť. Paměť programu a paměť dat sdílejí stejný adresní prostor.

Výhodou této architektury je, že nepotřebujeme rozlišovat instrukce pro přístup k paměti dat a paměti programu, což vede k zjednodušení čipu. Dále je zapotřebí jen jedna datová sběrnice pro přenos obou typů dat. Tato vlastnost je velmi výhodná především u použití externích pamětí, protože se zmenší potřebné množství vstupů/výstupů na mikrokontroléru určených k obsluze paměti, zjednoduší se návrh plošného spoje a jeho velikost, atd.

Nevýhoda je již zmiňovaná sdílená sběrnice. Protože jedna sběrnice přenáší data i instrukce (kód), je komunikace pomalejší než v případě oddělených sběrnic. Tato architektura se nejčastěji využívala v raných dobách mikroprocesorové techniky. I dnes ji však najdeme u mikrokontrolérů s Harvardskou architekturou, u kterých se předpokládá použití externích pamětí pro program a data.

Harvardská architektura – typické je oddělení prostoru paměti dat a paměti programu.

Za hlavní výhodu lze považovat možnost jiné šířky programové a datové sběrnice. Toho se nejvíce využívá u 8-bitových mikrokontrolérů, kde 8-bitová šířka instrukčního slova nestačí pro pokrytí všech instrukcí. Této možnosti se široce využívá. Můžeme tedy najít mikrokontroléry s šířkou programové sběrnice 12, 14 i 16 bitů. Další výhodou je větší rychlost vykonávání instrukcí (při stejné technologii), protože instrukci i potřebná data lze často číst ve stejný okamžik.

Nevýhodou je větší technologická náročnost čipu z důvodu existence 2 sběrnic a nutnost speciálních instrukcí pro čtení alespoň části obsahu programové paměti. Čtení obsahu programové paměti je sice proti čisté definici harvardské architektury, ale ve většině programů je nutné mít nějaké konstanty nebo tabulky. Tyto konstanty a tabulky je výhodnější dát do nepoužité části programové

paměti, než předem pevně vyhradit nějakou paměť pro úschovnu konstant (které by se nejspíše na začátku programu kopírovaly právě z paměti programu).

Různé šíře programové a datové sběrnice využívá nejvíce harvardská architektura VLIW (zkratka z anglických slov Very Long Instruction Word, česky velmi dlouhé instrukční slovo). V podstatě se jedná o myšlenku co nejvíce využít systémových zdrojů vlastního mikrokontroléru tím, že je programátorovi dána možnost ovlivnit využití jednotlivých částí mikrokontroléru. Těmito částmi mohou být aritmetická jednotka pro pevnou desetinou čárku, logická jednotka, aritmetická jednotka pro pohyblivou čárku, adresovací jednotka atd. Mnohdy mají mikrokontroléry výše uvedených jednotek více. Díky možnosti silně ovlivnit zpracování dat jednotlivými jednotkami mikrokontroléru je dána programátorovi možnost zvětšit výkon mikrokontroléru tím, že může zpracovávat více dat v jednom okamžiku.

Instrukční soubor CISC – z anglického Complex Instruction Set Controller, česká varianta překladu je procesor s úplným instrukčním souborem. Jde o procesor, u kterého najdeme široké spektrum instrukcí, často velmi komplikovaných. Cílem takovéto sady instrukcí je co nejvíce potřeb programátora vyřešit pomocí jediné instrukce. Na jednu stranu to znamená úsporu místa v programové paměti, na druhou stranu to znamená složitější dekodér instrukcí v mikrokontroléru a tím pádem pomalejší zpracování instrukcí. Tento instrukční soubor je dobře uplatnitelný v aplikacích, které jsou náročné na výpočty a požadují speciální instrukce.

Daný instrukční soubor se nejčastěji používal v počátcích mikroprocesorové techniky. V té době byly přístupová doba a velikost paměti limitujícími faktory při zvětšování výkonnosti mikrokontrolérů. Bylo snazší vyrobit relativně malý, komplikovanější instrukční dekodér, než větší a rychlejší paměť. S postupem času, jak se paměti staly rychlejšími a kapacita také přestala být problém, začal instrukční soubor CISC stále častěji vytlačovat instrukční soubor RISC.

Instrukční soubor RISC – z anglického Reduced Instruction Set Controller, česky procesor s redukovaným (omezeným) instrukčním souborem. Základní myšlenkou této architektury je, že je snazší vykonat více jednodušších instrukcí než jednu složitou. Omezení instrukčního souboru způsobilo zjednodušení dekodéru instrukcí, takže jednotlivé instrukce mohou být vykonávány mnohem rychleji, často během jediného systémového taktu. To samozřejmě vede k zjednodušení výpočetního výkonu mikrokontroléru, neboť komplikované instrukce v architektuře CISC jsou v programech využívány jen v omezené míře a jednoduché instrukce (také režie mezi komplikovanými výpočty), používané častěji, mohou být prováděny rychleji.

Dlouhou dobu jsem přemýšlel jestli mikrokontroléry členit podle výrobce nebo podle jádra. Nakonec jsem se rozhodl pro členění podle jádra (architektury). Výrobců mikrokontrolérů je nepřeberné množství, ale nejčastěji se jedná o několik nejznámějších jader, ke kterým jsou přibaleny určité periferní obvody. Pokud jsem narazil na jádra, která jsou zpětně kompatibilní se staršími jádry, rozvedl jsem to nejnovější a o starších typech jádra jsem se alespoň zmínil.

2.1 Přehled 8-bitových mikrokontrolérů

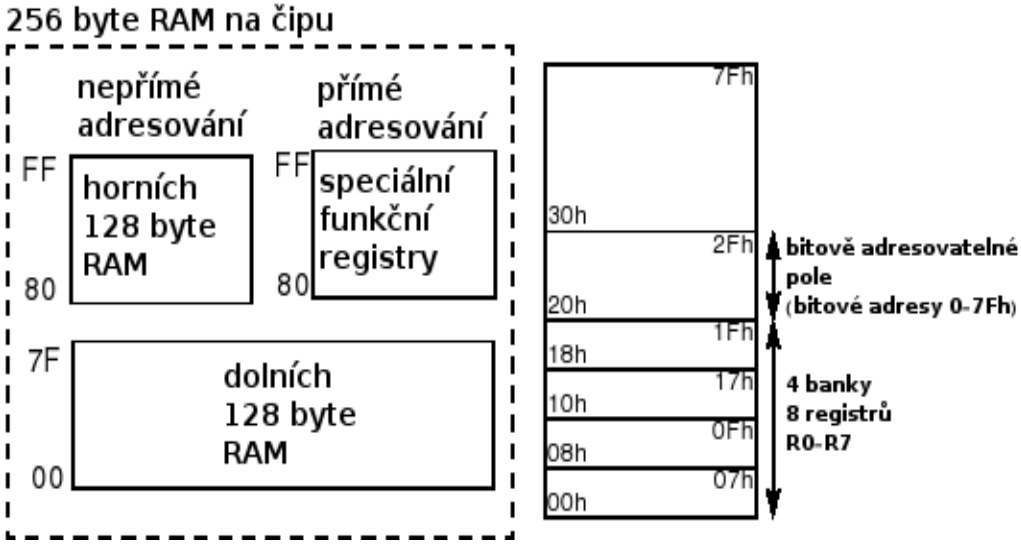
2.1.1 8051

První mikrokontrolér s touto architekturou byl představen v 70. letech firmou Intel. V 80. letech se jednalo o nejoblíbenější mikrokontroléry. Intel oficiálně pojmenoval tuto řadu mikrokontrolérů jako **MCS 51** (viz [14]), ale v povědomí mnoha uživatelů stále přetrvává název 8051. Mikrokontroléry 8051 nebo jejich klony vyrábějí firmy Intel, Atmel, Texas Instruments, STMicroelectronics, Silicon Laboratories, Cypress Semiconductor, Philips Semiconductor, Dallas Semiconductor, Analog Devices, Infineon, Winbond a další.

Mikrokontroléry 8051 jsou založeny na harvardské architektuře a mají instrukční soubor CISC. Pro vykonání základní instrukce je zapotřebí 12 hodinových taktů. U novějších klonů existuje mód x2, kdy je základní instrukce vykonána za 6 hodinových taktů. Šíře instrukčního slova je 8-bitová a šíře adresové sběrnice je 16-bitová. Po připojení napájení musí být reset prováděn externími součástkami.

Všechny výpočetní výkony se provádí přes střadač, někdy taky nazývaný jako akumulátor. Pro násobení a dělení je zde pomocný registr B. Dále jsou k dispozici 4 tzv. banky po 8 pomocných registrech R0-R7. Banky se nacházejí v dolní části paměti dat. Banka 0 začíná na adrese 0, banka 1 na adrese 8, atd. Uspořádání vnitřní paměti můžete vidět na obrázku 1. Registry R0 a R1 slouží i jako ukazatele pro nepřímé adresování prvních 256 bytů paměti dat. Po resetu je aktivní banka 0. Aktivní banku můžeme změnit za chodu programu nastavením bitů RS0 a RS1 v registru PSW. Za bankami, na adrese 20h-2Fh se nachází bitově adresovatelné pole (v jazyce C se dá srovnat polem typu boolean). Bitově adresovatelné pole je velmi výhodné, pokud potřebujete nastavovat příznak, podle kterého se dále bude program členit. Nemusíme tak zabírat zbytečně pamět a můžeme použít jen 1 bit, který je k tomu zapotřebí. Interní paměť se dělí na horní a dolní. Dolní paměť je v rozmezí 0h-7Fh a obsahuje zmíněné banky a bitově adresovatelné pole. Horní paměť je potom v rozmezí 7Fh-FFh. Ostatní paměť už je brána jako externí a přistupuje se k ní přes speciální instrukce. Dolní paměť je adresovatelná přímo i nepřímo. Horní paměť je adresovatelná pouze nepřímo. Je to z toho důvodu, že do stejné části paměti jsou mapované speciální funkční registry (porty, stavová slova,

atd.), které jsou adresované přímo. Externí paměť se adresuje od nuly. I když je na čipu více paměti než zmíněných 256 bytů, přistupuje se k ní jako k externí paměti. Když je připojena externí paměť, mikrokontrolér podle vnitřní elektroniky sám rozpozná jestli daná oblast je ještě interní nebo už externí. Nevýhoda je, že z externí paměti bude nevyužita část paměti, která je adresována jako interní.



Obrázek 1: Interní paměť programu mikrokontroléru AT90C5122

Zásobník je realizován programově. Ukazatel zásobníku je 8-bitový, inkrementační, a ukazuje do paměti RAM. Protože je zásobník inkrementační, je dobré jej umístit za bitově adresovatelné pole a námi definované proměnné. Zásobník může být použit kdekoliv v interní paměti (256 bytů), ale nemůže být použit v externí paměti. Dále máme pro nepřímé adresování celkem 4 ukazatele. Jedná se o registry R0 a R1, akumulátor a ukazatel DPTR. Akumulátor, společně s registry R0 a R1 mohou adresovat pouze prvních 256 bytů paměti dat (interní i externí). Pokud chceme adresovat více paměti, musíme použít ukazatel DPTR, který je jediný 16-bitový a dělí se na DPH a DPL. Pro práci s většími datovými typy obsahuje instrukční soubor sčítací instrukce pracující s předchozím příznakem přenosu (anglicky carry). Odečítací instrukce jsou k dispozici jen pracující s předchozím příznakem přenosu, takže před započetím odečítání je nutno vynulovat příznak přenosu.

Chtěl jsem zde uvést členění rodiny 8051, ale každý výrobce značí svoje výrobky po svém. Výrobci ovšem na skutečnost, že se jedná o jádro 8051 nebo jeho klon, často upozorňují, protože je řada 8051 dobře známá a většina uživatelů je s nimi obeznámena.

Mikrokontroléry 8051 obsahují 16-bitové časovače/čítače. Při práci s nimi musí být časovače/čítače vypnuty, jinak může dojít k nesprávně vyhodnoceným výsledkům. Není zde způsob jak s nimi pracovat za chodu, aniž by nenastalo riziko nekorektně přečtených dat.

K čipu musí být připojen oscilátor nebo krystal. Samotný mikrokontrolér neobsahuje integrovaný oscilátor. Novější typy mikrokontrolérů mají fázový závěs, takže může být použit nízkofrekvenční krystal.

Porty jsou součástí speciálních funkčních registrů, které jsou mapovány do horní paměti dat s přímým adresováním. K ovládání portů se tedy používají stejné instrukce jako pro práci s pamětí dat. Porty jsou tvořeny záchytným obvodem a tranzistory. Zapojení pro vstupní a výstupní používání portu se nemění. Elektronika portu při čtení dat vyžaduje, aby byl port nastaven na jedničky (výstup portu musí být nastaven na vysokou úroveň napětí), jinak se z portu přečtou nekorektní data. Při použití portu pro alternativní funkce (sériový kanál, atd.) musí být port nastaven také na jedničky.

Řadič přerušení je plně vektorový systém, kdy každá periferie, využívající přerušení, má vlastní vektor přerušení. Obslužný program pro obsluhu přerušení se skládá ze samostatných programů, které obsluhují stav dané periferie, který přerušení vyvolal. Vektorový systém vyžaduje složitější návrh mikrokontroléru. Jeho výhodou je rychlejší reakce na vzniklé přerušení a jednodušší obslužné programy. Jestliže při obsluze přerušení vznikne požadavek na přerušení, které má vyšší prioritu než právě prováděné přerušení, bude nejdříve obslouženo přerušení s vyšší prioritou. Po obsloužení přerušení s vyšší prioritou se běh programu vrátí k obsluze předchozího přerušení. Kdyby nebylo možné přerušit aktuální obsluhu přerušení za účelem obsloužení přerušení s vyšší prioritou, mohlo by se stát, že by mikrokontrolér ignoroval signál resetu, protože právě probíhá jiné přerušení. Na požadavek resetu mikrokontrolér musí reagovat vždy. Reset má tedy nejvyšší prioritu a jde o nemaskovatelné přerušní.

Jak je vidět, některé části mikrokontroléru jsou již pro požadavky dnešního trhu nedostačující nebo nevhodné. Architektura 8051 má stále velkou zákaznickou podporu a tak prošel mikrokontrolér několika změnami. Byli mu přidány nové periferie, některé věci byli modernizovány (porty, reset, zdroje hodinového taktu, atd.) a objevili se překladače vyšších programovacích jazyků jako je BASIC, Pascal, PL/M a C. Když chce nějaká firma začít vyrábět mikrokontroléry, má na výběr ze dvou možností. Buď investuje do vývoje mikrokontroléru a osvěty mezi uživateli, nebo si koupí již existující jádro, které je uživatelům dobře známo a zbylé finanční prostředky použije pro specializaci jádra pro předem plánovaný účel. Proto existuje spousta mikrokontrolérů založených na jádře 8051 (tzv. klonů). Například firma Triscend využila jádra 8051, doplnila ho o signálovou část a vznikla nová řada mikrokontrolérů s označením TE5.

Informace jsem čerpal z [3][7][10][14].

2.1.2 AVR

Rodina mikrokontrolérů 8051 má svá omezení a přestala stačit stále rostoucím požadavkům na výpočetní výkon. Bylo zapotřebí navrhnout novou strukturu vyhovující vyšším požadavkům na

výkon a dobře použitelnou s vyššími programovacími jazyky, zejména široce používaným jazykem C. Vznikla tedy rodina mikrokontrolérů s harvardskou architekturou a instrukčním souborem RISC, nazývaná AVR. Mikrokontroléry AVR jsou vyráběny korporací Atmel.

Přes zdánlivou podobnost s jádrem mikrokontroléru 8051 jsou zde podstatné odchylky. Především je to šíře instrukčního slova, která je 16-bitová. To zvýšilo požadavky na velikost paměti, ale zároveň umožnilo zrychlit načítaní instrukcí, které jsou nejčastěji o velikosti jednoho slova, takže je mikrokontrolér dokáže přečíst během jednoho hodinového taktu. Po připojení na pájení je reset prováděn interními obvody.

Dalším rozdílem je propojení centrální výpočetní jednotky (ALU) s polem 32 pracovních registrů, kde 16 dolních registrů má omezenou funkčnost. Přes tyto registry se provádějí všechny výpočty. Jsou tedy ulehčeny výpočty potřebující více mezivýsledků a odpadá nutnost stálého načítání a ukládání hodnot přes akumulátor. Organizace ALU umožnila snížit počet potřebných hodinových taktů pro vykonání téměř všech instrukcí na 2 takty, načtení plus dekódování a vykonání. Díky tomu, že instrukce vystačí s jedním slovem, byla umožněna implementace jednoduchého překrývání zmiňovaných fází. Pro operaci registr-registr byl snížen počet hodinových taktů na 1 takt. V porovnání se základním instrukčním cyklem řady 8051 dostáváme 12x větší výkon.

Jak už bylo řečeno instrukční sada byla vyvinuta pro použití s vyššími programovacími jazyky, převážně jazykem C. Velmi důležitou podmínkou pro překlad efektivního kódu je existence ukazatelů a realizace zásobníku. Zásobník je realizován programově, je 16-bitový, dekrementační, a ukazuje do paměti RAM. Do struktury byli implementovány ukazatele (registry) pojmenované X, Y a Z, které jsou umístěny v pracovních registrech. Také byly přidány instrukce pro práci s nimi s možností pre-dekrementace a post-inkrementace. Ukazatele jsou 16-bitové, mohou tedy adresovat celou paměť programu a dat. Existence 3 ukazatelů je zvolena z důvodu urychlení provádění kódu při potřebě ukazatele pro zdroj i cíl a třetí ukazatel může být využit jako zásobník. Také byli přidány instrukce pro práci s předchozím příznakem přenosu, využitelné při sčítání a odčítání u větších datových typů např. long integer. U řetězců můžeme provádět odčítání s předchozím příznakem přenosu a při nulovém výsledku jsou si řetězce rovny. První příkaz početní operace samozřejmě použijeme bez využití předchozího příznaku přenosu, jinak by se nám smíchal s předchozím výpočtem, což určitě nechceme.

Rodina 8-bitových mikrokontrolérů AVR se dá rozdělit na 4 skupiny: mikrokontroléry řady **AT90**, **ATtiny, ATmega** a **ATxmega**. Jako první vznikla řada AT90S a její název měl naznačovat, že se jedná o pokračování řady AT89C, tedy nástupce rodiny 8051. U zbylých 2 řad výrobce navrhl jejich názvy podle jejich určení. Řada ATtiny je určena pro menší, jednodušší aplikace, kdežto řada ATmega je určena pro vetší a komplikovanější aplikace. Řada AT90S byla začleněna do řady ATmega a nyní se vyskytují jen mikrokontroléry se specializovanou funkcí s označením AT90. Řada ATxmega

je žhavou novinkou a jedná se o nejvýkonnější mikrokontroléry rodiny AVR doplněné o celou řadu výkonných periferií. Protože jsou mikrokontroléry ATxmega novinkou, nebudu se jimi dále zabývat.

Mikrokontroléry AVR mají 16-bitové časovače/čítače. Vzhledem ke skutečnosti, že samotné mikrokontroléry jsou 8-bitové, vzniká problém při jejich čtení a zápisu. Mezi dobami přečtení dolních a horních 8 bitů si nemůžeme být jisti, zda se horní bity za tuto dobu nezmění. Návrháři tedy vybavili vyšší byte 16-bitových registrů záchytným 8-bitovým registrem. Při čtení nižšího byte se v ten samý okamžik zachytí hodnota vyššího byte v záchytném registru a při následném čtení vyššího byte se hodnota vyčte ze záchytného registru. Tím je zajištěno správné načtení dat. Při zápisu do 16-bitového registru je postup přesně opačný. Nejdříve se zapíše horní byte, který se nezapíše přímo do registru, ale zachytí se opět v záchytném registru a při zápisu dolního byte se zapíše zároveň dolní byte a horní byte ze záchytného registru.

Dále mají jako volitelné zdroje hodinového taktu:

- externí krystal/keramický rezonátor
- externí nízkofrekvenční krystal
- externí RC oscilátor
- kalibrovaný interní RC oscilátor
- externí hodiny taktu.

Tyto volitelné zdroje mají i ty nejmenší z představitelů rodiny AVR, liší se jen v interním RC oscilátoru, kde menší čipy podporují frekvenci 1 MHz a u větších je na výběr z 1, 2, 4 a 8 MHz. Všechny frekvence jsou navrženy pro použití při 5V a 25°C s přesností ±3%. Při použití kalibrace za běhu programu, jak je popsáno v aplikačních poznámkách na webu výrobce (viz [10]), je možné zpřesnit interní oscilátor na ±1% pro různé napájecí napětí a různou pracovní teplotu.

U vstupně/výstupních portů můžeme nastavit zda daný pin použijeme jako vstupní nebo výstupní. Funkce pinu (vstup/výstup) se nastavuje přes registr DDRx, kde x značí port (A, B, atd.). Při funkci výstupu může každý pin přenášet proud 20mA, maximálně však 40mA. Při funkci vstupu může pin pracovat v režimu vysoké impedance nebo s připojeným pull-up rezistorem. Rozdílný je přístup k pinům oproti architektuře 8051. Zatímco u architektury 8051 je z pinů načítalo a zapisovalo přes stejný registr, u architektury AVR se do pinů zapisuje přes registr PORTx a čte přes registr PINx, kde x značí port. Registr PINx ve skutečnosti není registr. Je to adresa, která umožňuje přístup k fyzické hodnotě na portu. Na této adrese jsou přítomny skutečné úrovně na vývodech integrovaného obvodu. Na tuto adresu nejde zapisovat. Adresování vstupně/výstupních portů včetně řídících registrů se provádí pomocí speciálních instrukcí (IN/OUT plus bitové instrukce).

Řadič přerušení je plně vektorový systém, stejně jako u rodiny 8051. Rodina AVR ale nemá plně konfigurovatelný řadič přerušení, existují pouze pevné priority přerušení. S tímto faktem musí programátor počítat a případně uzpůsobit danou aplikaci. Vektory přerušení jsou umístěny na začátku

paměti programu (nejnižší adresy). Čím je adresa vektoru nižší, tím je jeho priorita vyšší. Při obsluze přerušení může být přerušení také přerušeno, pokud se vyskytne požadavek na přerušení s vyšší prioritou.

Další rozdílnou věcí oproti rodině 8051 jsou tzv. Fuses, česky pojistky. U nižších řad se jedná o 1 byte a u vyšších řad se jedná o dolní pojistku (Fuse Low Byte), horní pojistku (Fuse High Byte) a někdy i o rozšířenou pojistku (Extended Fuse Byte). Pojistky se programují odděleně od programování paměti programu. Můžeme tedy programovat zvlášť pojistky a zvlášť program, nejedná se o společný proces. Přes pojistky se nastavuje chování mikrokontroléru a jeho periferií, např. povolení JTAG, zdroj hodinového signálu, vždy aktivovaný hlídací obvod (Watchdog) a další volby. Před začátkem programování je dobré si všechny možnosti projít. Vyhnete se tak pozdějším komplikacím.

Informace jsem čerpal z [3][4][5][7][9][10].

2.1.3 HC11

HC11 je rodina mikrokontrolérů původně firmy Motorola, nyní produkované firmou Freescale Semiconductor. Jedná se o pokročilou modifikaci řady HC08 a HC05 na bázi mikroprocesoru Motorola 6800 (je kompatibilní na bázi zdrojového kódu - směrem vzhůru). Procesory mají Von Neumanovu architekturu a instrukční sadu typu CISC. V textu se budu věnovat převážně jádrům HC11 a HC08.

Šíře instrukčního slova je 8-bitová a šíře adresovací sběrnice je 16-bitová. Frekvence sběrnice je rovna čtvrtině frekvence zdroje hodinového taktu. Pro jeden takt sběrnice jsou tedy zapotřebí 4 takty zdroje hodin. U řady HC08 bylo zapotřebí pro instrukce 1-3 takty sběrnice, u složitějších instrukcí až 7 taktů sběrnice. Řada HC11 má ale složitější instrukce a tak potřebuje 2-7 taktů sběrnice, u složitějších instrukcí až 41 taktů sběrnice. Po připojení na napájení je reset prováděn interními obvody.

Organizace centrální procesorové jednotky je podobná rodině 8051. Na rozdíl od rodiny 8051 zde nenajdete žádné pracovní registry. Řada HC08 má celkem 16 adresovacích módů. Pro výpočetní úkony je zde akumulátor A a index registry H a X, které dohromady tvoří 16-bitový registr H:X. Instrukční sada obsahuje 8-bitové násobící a dělící instrukce. Řada HC11 je v tomto ohledu lépe vybavena. Pro výpočetní úkony zde jsou dva 8-bitové akumulátory A a B, které dohromady tvoří 16-bitový akumulátor D, a dva 16-bitové index registry IX a IY. Existence dvou 16-bitových ukazatelů je značným ulehčením pro kopírovací práce. Instrukční sada obsahuje 8-bitovou násobící instrukci a 16-bitové dělící instrukce. Obě řady mají 16-bitový ukazatel zásobníku, který je dekrementační a ukazuje do paměti RAM, a sčítací instrukce pracující s předchozím příznakem přenosu i bez něj, ale odečítací instrukce obsahují pouze bez předchozího příznaku přenosu.

Pro lepší orientaci ve značení mikrokontrolérů vysvětlím označení mikrokontroléru MC68HC908GP32CFB (popis převzat z [1]):

MC - značení výrobce (MOTOROLA)

68HC - označení základní technologie

908 - FLASH verze, 708 EPROM nebo OTP, 08 jen ROM

GP - „rodina“ procesorů (GP = General Purpose - pro obecné použití), právě tato písmena označují vybavení periferií. Příklady dalších dvojic písmen: **AZ, BD, EY, GP, GR, GT, GZ, JB, JL, KH, KX, LJ, MR, RK, SR, QT** a **QY**.

32 - přibližná velikost hlavní paměti v kB.

CFB - typ pouzdra

Mikrokontrolér obsahuje 16-bitový časovač/čítač, na který je připojeno několik porovnávacích/záchytných jednotek pro práci s ním. Časovač zvyšuje nebo snižuje svoji hodnotu o jedničku při každé periodě hodinového signálu. U některých typů mikrokontrolérů je možnost nastavit takový čítací režim, ve kterém se při čítání směrem nahoru a dosažení maximální hodnoty se čítač nevynuluje, ale čítá směrem dolů. Analogicky se potom při dosažení minimální hodnoty se nenastaví na samé jedničky, ale čítá směrem nahoru. Tento režim je velmi vhodný pro řízení motorů.

O zdroj hodinového taktu se stará modul nazývaný CGM, z anglického Clock Generation Module, česky modul generátoru hodin. Složitost modulu CGM se liší podle určení konkrétního mikrokontroléru. Některé verze modulu CGM dokáží z jednoho zdroje odvodit i více hodinových taktů (z pravidla jde o násobky či zlomky hodinového taktu). Jako zdroje taktu pro CGM přicházejí v úvahu následující možnosti:

- externí krystal o vhodné rezonanční frekvenci
- externí krystal s fázovým závěsem
- externí zdroj hodinového signálu
- vnitřní oscilátor

Při výběru externího krystalu s fázovým závěsem stačí připojit levný a malý krystal, nejčastěji 32kHz, fázový závěs potom vynásobením frekvence oscilátoru zajistí hodinový zdroj v řádu MHz. Tento způsob je velmi užitečný zejména u aplikací, kde si návrhář musí dávat pozor na rušící vlivy. Každý drát na desce plošných spojů působí jako malá anténa. Čím je frekvence krystalu vyšší, tím je vyzařování a rušení ostatních obvodů vyšší. Použitím malého krystalu spolu s fázovým závěsem docílíme slabších hodnot vyzařování do okolí. U vnitřního oscilátoru je výrobní přesnost kmitočtu ±25%. Oscilátor lze ale jemně doladit pomocí kalibračního registru OSCTRIM. Hodnota kalibrační konstanty je hrubě změřena ve výrobě a pro přesnost pod 5% ji lze využít. K dispozici je v paměti FLASH na adrese 0xFFC0. Není však chráněna proti vymazání a při prvním smazání celé FLASH

paměti se vymaže rovněž. Proto před prvním programováním ještě nepoužitého čipu nového mikrokontroléru je dobré si tuto hodnotu z čipu přečíst a zaznamenat.

Vstupně/výstupní porty fungují na podobném principu jako u mikrokontrolérů AVR. Funkce pinu (vstup/výstup) se nastavuje přes registr DDRx, kde x značí port (A, B, atd.). Rozdíl je ale při zacházení s portem. U architektury AVR jsou 2 registry, jeden pro zápis a druhý pro čtení přímo z portu. U mikrokontrolérů HC08 a HC11 se na port přistupuje přes registr PTx, kde x značí port. Mikrokontrolér automaticky při zápisu do registru PTx zapíše hodnotu do vyrovnávací paměti portu a při čtení z registru PTx vrácí fyzické hodnoty z portu. Adresování portů se provádí přes běžné instrukce pro práci s pamětí.

Řadič přerušení je plně vektorový systém. Ve většině případů priorita přerušení nejde nastavit, ale u některých typů mikrokontrolérů je možné prioritu přerušení nastavit. Vektory přerušení jsou umístěny na konci paměti (nejvyšší adresy). Čím vyšší adresa vektoru, tím vyšší priorita. I zde může být obsluha přerušení přerušena přerušením s vyšší prioritou, stejně jako u mikrokontrolérů 8051 a AVR. Při práci s přerušením jsem narazil na jednu zajímavou vlastnost. Před návratem přerušení je zapotřebí přečíst bit, který je zodpovědný za vyvolání přerušení, čímž se odstraní požadavek na přerušení. Když přečtení neprovedeme, bit indikující přerušení zůstane nezměněn a přerušení se provede opakovaně (dojde k neustálému obsluhování přerušení i když příčina už pominula). Tato vlastnost se netýká mikrokontrolérů 8051 a AVR, u nich dojde k odstranění požadavku na přerušení při začátku obsluhy přerušení automaticky, takže se stav indikujícího bitu nemusí číst.

Informace jsem čerpal z [1][2][3][7][9][11].

2.1.4 PIC

Jedná se o mikrokontroléry firmy Microchip Technology Inc. s Harvardskou architekturou a instrukčním souborem RISC, které mají svoji osobitou instrukční sadu. Jeden typ mikrokontroléru je nabízen v několika pouzdrech s malými změnami ve svém vybavení, často se liší jen ve velikostech pamětí. Konstruktér má tedy možnost použít vždy vhodný typ pro daný účel.

V současné době se 8-bitové mikrokontroléry PIC dají rozdělit na 3 skupiny:

- **Low-End** také nazývané Base-Line – základní řada pro ty nejmenší aplikace. Má 12-bitovou šíři instrukčního slova a 2-úrovňový hardwarový zásobník. Patří sem mikrokontroléry s označením PIC10 a PIC12. Řada PIC12 svým označením patří do skupiny Low-Rnage, ale vlastnostmi by měla patřit spíše do skupiny Mid-Range, neboť má 14-bitovou šířku instrukčního slova a 8-úrovňový hardwarový zásobník, což jsou hodnoty charakteristické pro skupinu Mid-Range.
- **Mid-Range** – střední řada pro normální aplikace. Má 14-bitovou šíři instrukčního slova a 8-úrovňový hardwarový zásobník. Patří sem mikrokontroléry s označením PIC14 a PIC16.

- **High-End** – nejvyšší řada pro náročné aplikace. Má 16-bitovou šíři instrukčního slova a 31-úrovňový hardwarový zásobník. Patří sem mikrokontroléry s označením PIC17 a PIC18. Mikrokontroléry PIC17 mají pouze 16-úrovňový hardwarový zásobník.

Když jsem dělal zařazení do skupin, bral jsem v potaz paměť programu typu FLASH. Mikrokontroléry jednotlivých skupin mají shodné periferie. Myslím tím, že počet a typ periferií není stejný, ale pokud mají např. Timer0, jsou u obou mikrokontrolérů stejné. Tento fakt je velká výhoda. Programátor se nemusí dlouze učit jednotlivé periferie různých mikrokontrolérů. A hlavně je přechod od jednoho typu mikrokontroléru k druhému ze stejné skupiny daleko méně náročný než u jiných mikrokontrolérů či mikrokontrolérů jiných výrobců.

Centrální procesorová jednotka je propojena s jediným 8-bitovým akumulátorem, nazývaným W. Oproti jiným mikrokontrolérům mají mikrokontroléry PIC odlišný instrukční soubor. Většina assemblerů (překladač jazyka symbolických adres) obsahuje mnoho makroinstrukcí (makroinstrukce vytvoří z několika instrukcí jednu novou), které suplují neexistující instrukce. Tento postup je vhodný zejména pro začínající programátory, které by například nepřítomnost podmíněných skoků mohla zmást. Místo podmíněných skoků je zde instrukce, která při splnění určité podmínky přeskočí následující instrukci. Následující instrukce může být instrukce skoku a tím je vytvořena alternativa k podmíněným skokům. Instrukční soubor řady Low-End a Mid-Range nemají sčítací a odečítací instrukce pracující s předchozím příznakem přetečení. Řada High-End už je ale obsahuje. Oproti jiným mikrokontrolérům obsahuje instrukční soubor jiný přístup k ukládání výsledku po výpočetní operaci. Po každé výpočetní operaci máme na výběr, zda výsledek bude uležen v pracovním registru W, nebo jestli bude uložen zpět do zdroje. Pro lepší pochopení uvedu dva příklady, které mají sečíst obsah registrů A a B a výsledek uložit zpět do registru A (A = A + B):

```
MOVF    A, W
ADDWF   B, W
MOVWF   A
```

Příklad 1: Sčítání pomocí akumulátoru

```
MOVF    B, W
ADDWF   A, F
```

Příklad 2: Sčítání se znakem F

V příkladu 1 se postupuje známým způsobem z jiných mikrokontrolérů, kdy výpočet probíhá pomocí akumulátoru. Obsah pracovního registru je nakonec uložen na adresu cíle. Příklad 2 využívá také pracovního registru, ale znakem F ve druhé instrukci říkáme, že zdroj je zároveň cíl. Tím ušetříme jednu instrukci a urychlíme výpočet. U instrukčního souboru je zachována kompatibilita směrem vzhůru. Po připojení na napájení je reset prováděn interními obvody.

Adresace paměti závisí na rodině zvoleného mikrokontroléru. Každá série implementuje systém bank, který umožňuje rozšířit adresový prostor. Novější verze procesorů jsou schopny

adresovat celý prostor registrů v jednom okamžiku (nezávisle na zvolené bance), ale starší a základní verze umožňují přístup ke všem registrům jen přes akumulátor. Z hlediska použití vyšších programovacích jazyků je důležitá existence ukazatelů. Řada Low-End a Mid-Range obsahuje jeden 8-bitový ukazatel pojmenovaný FSR (File Select Register, česky registr výběru souboru). Při nepřímém adresování se postupuje stejně jako při adresování přímém, jen se jako cíl použije registr INDF (Indirect Data File, česky nepřímý datový soubor). Registr INDF není fyzický registr. Adresováním INDF registru se značí nepřímé adresování. Adresa zdroje/cíle se nachází v registru FSR. Řada High-End obsahuje tři 12-bitové ukazatele pojmenované FSR0, FSR1 a FSR2. Obsluha je stejná jako u řady Low-End a Mid-Range pomocí registrů INDF0, INDF1 a INDF2. Místo registru INDFx je možno nepřímo adresovat také s post-inkrementací pomocí registru POSTINCx, post-dekrementací pomocí registru POSTDECx, pre-inkrementací pomocí registru PREINCx a adresací s posunutím, které je dáno znaménkovým obsahem pracovního registru W, pomocí registru PLUSWx, kde x je číslo 0, 1 nebo 2. Díky implementaci tří 12-bitových ukazatelů s možností post-inkrementace, post-dekrementace a adresací s posunutím je řada High-End při programování pomocí vyšších programovacích jazyků schopna mnohem kompaktnějšího a rychlejšího kódu.

Každá řada má jako zdroj hodinového taktu na výběr jiné možnosti. Nejmenší zástupci mají často jako zdroj hodinového taktu pouze interní oscilátor. Jako zdroje hodinového taktu přichází v úvahu:

- externí krystal/keramický rezonátor
- externí krystal/rezonátor s fázovým závěsem
- externí RC oscilátor
- interní oscilátor

Interní oscilátor může kmitat na frekvenci 31kHz nebo 8MHz s děličkou na nižší frekvence. Přesnost RC oscilátoru se pohybuje kolem ±20%, takže lze využít pro aplikace, u kterých není vyžadováno precizní časování. Přesnost interního oscilátoru se pohybuje od ±1% až do ±5% v závislosti na napájecím napětí a teplotě.

Vstupně/výstupní porty jsou podobné jako u mikrokontrolérů HC11. Registrem TRISx nastavujeme funkci portu (vstup/výstup) a registrem PORTx provádíme čtení/zápis, kde x značí port. Při zápisu do registru PORTx mikrokontrolér automaticky hodnotu zapíše do záchytného registru portu a při čtení registru PORTx jsou vráceny fyzické hodnoty z portu. Adresace portů se provádí přes běžné instrukce pro práci s registry.

Mikrokontroléry PIC mají velmi jednoduchý přerušovací systém. Volání podprogramu na přerušení je prováděno z jedné pevné adresy, bez ohledu na to, která periferie přerušení vyvolala. Toto řešení má své výhody i nevýhody. Výhodou je jednodušší návrh mikrokontroléru, systém je průhledný a programátor má vše pod kontrolou. Nevýhodou je, že se programátor musí postarat

o programovou detekci periferie, která žádá o přerušení. Tato programová detekce bude vždy pomalejší než detekce prováděná mikrokontrolérem (plně vektorový systém). V Řadě PIC18 najdeme přerušovací vektory 2, lišící se svojí prioritou. To umožní vybrat přerušení (jedno nebo i více), na které klademe nároky na rychlou odezvu.

Mikrokontroléry PIC mají velkou uživatelskou podporu a tak není divu že existují jejich klony, např. mikrokontroléry Parallax SX.

Informace jsem čerpal z [3][7][9][12].

2.1.5 ST7

Mikrokontroléry distribuované firmou STMicroelektronics jsou založeny na Von Neumannově architektuře. Jedná se o nástupce řady ST6, která je založena na harvardské architektuře a je zaměřena na ty nejjednodušší a nejlacinější aplikace. Od architektury s hardwarovým zásobníkem, až po instrukční soubor, kde nejsou např. instrukce rotace doprava nebo početní instrukce pracující s příznakem přenosu. V této řadě není žádný zástupce vybaven pamětí typu Flash, proto se dále budu zabývat jen jejím nástupcem.

Jádro mikrokontroléru používá pro výpočty jednoho akumulátoru A. Dále jsou k dispozici dva 8-bitové ukazatele X a Y a 16-bitový ukazatel zásobník odkazující do paměti RAM. Instrukční soubor má celkem 63 instrukcí, široké množství adresování a možnost využít dvou index registrů.

Jako zdroje hodinového taktu jsou k dispozici:

- externí krystal/keramický rezonátor
- externí RC oscilátor
- interní RC oscilátor
- externí zdroj hodinového taktu
- záložní hlídací zdroj hodinového taktu

Pokud je povolen záložní hlídací zdroj hodinového taktu, tak při výpadku hlavního zdroje hodinového taktu mikrokontrolér přejde na záložní zdroj, který neustále kmitá na pomalé frekvenci uvnitř mikrokontroléru.

U Vstupně/výstupních portů se přes registr PxDDR nastavuje funkce (vstup/výstup), přes PxOR nastavuje upřesnění funkce, které je někdy pro každý pin různé a přes PxDR se přistupuje na port, kde x značí port. Při čtení z registru PxDR se vrací fyzické hodnoty na portu a při zápisu do PxDR se zapisuje hodnota do záchytného registru portu.

Řadič přerušení je plně vektorový systém. Vektory přerušení jsou umístěny na konci paměti. Čím má vektor přerušení vyšší adresu, tím má vyšší hardwarovou prioritu.

Informace jsem čerpal z [13].

2.2 Přehled vícebitových mikrokontrolérů

2.2.1 4-bitové mikrokontroléry

Mohlo by se zdát, že trh směřuje ke stále komplikovanějším mikrokontrolérům poskytující vyšší výkon a univerzálnější využití. Částečně je to také pravda. Nesmíme ale zapomínat na zařízení jako jsou CD přehrávače, displeje kalkulaček, teplotní senzory, malá bezdrátová zařízení apod. Řada těchto věcí potřebuje nějakým způsobem ovládat a zároveň musí zůstat malé s malou spotřebu elektrické energie. Proto výrobci jako Atmel, SANYO a další investovali finanční prostředky pro vývoj 4-bitových mikrokontrolérů. Např. řada MARC4 výrobce Atmel (viz [10]) je založena na instrukčním souboru typu RISC s dvou-cyklovými instrukcemi a mají nízkou spotřebu energie. Je tedy perfektní pro bezdrátové aplikace jako je elektronický klíč, imobilizéry, bezdrátová klávesnice pro PC a multimédia, bezdrátová čidla a každá další datové aplikace využívající IR a RF. Speciální periferie pro tyto aplikace jsou plně integrovány.

2.2.2 16-bitové mikrokontroléry

Situace na poli 16-bitových mikrokontrolérů je nejistá. Když uživateli přestane vyhovovat 8-bitový mikrokontrolér, ať už z důvodu výkonu nebo nedostačujících periférií, má možnost přejít na mikrokontroléry 16-bitové nebo 32-bitové. Modernizace výrobních technologií způsobila, že levnější 32-bitové mikrokontroléry klesly na cenu výkonnějších 16-bitových mikrokontrolérů. Řada uživatelů volí rovnou 32-bitové mikrokontroléry, protože jim do budoucna poskytují vysoký výpočetní výkon. Řada výrobců má tedy zastoupení 32-bitových mikrokontrolérů rozsáhlejší než 16-bitových.

Z řad 16-bitových mikrokontrolérů jmenujme např. HC16, S12 a S12X firmy Freescale Semiconductor, ST10 firmy STMicroelektronics, PIC24 firmy Microchips a podrobněji si představíme mikrokontrolér **MSP430** vyráběný firmou Texas Instruments (viz [15]). MSP je anglická zkratka Mixed-Signal Processors, česky smíšený signálový procesor, který dokáže zpracovávat analogové a digitální signály. Jedná se o mikrokontroléry s von Neumannovou architekturou a instrukčním souborem RISC, které jsou orientované na práci se zásobníkem. Mají široké spektrum periferií a velmi nízkou spotřebu, která může být ještě více redukována díky mnoha módům snížení spotřeby. Jsou výborným řešením pro bateriově napájené měřící aplikace. Typicky jsou využívány v měřících jednotkách, přenosných měřících přístrojích, inteligentních senzorech a spotřební elektronice.

2.2.3 32-bitové mikrokontroléry

Jak už bylo řečeno, 32-bitové mikrokontroléry jsou více perspektivní než 16-bitové mikrokontroléry. Mají lepší výběr na poli operačních systémů a další výhody spojené s vyšším výpočetním výkonem, jako jsou náročnější periferie. Na druhou stranu může být přechod na 32-bitové mikrokontroléry náročnější, protože jsou jejich architektury odlišné od předchozích architektur. I instrukční soubory jsou odlišné, a proto někteří výrobci nabízí 32-bitové mikrokontroléry s rozšířením pro lepší přechod z 8-bitových mikrokontrolérů.

Jader 32-bitových mikrokontrolérů je celá řada. Jmenujme např. AVR32 firmy Atmel, PIC32 firmy Microchips, C2000 firmy Texas Instruments a široce používaná jádra ARM s instrukčním souborem RISC.

3 Metodika tvorby aplikačních programů

Při programování větších projektů programátor používá různé algoritmy, řetězení kódu a spoustu dalších věcí, které se bez řádného popisu neobejdou. V této kapitole se budu věnovat základním popisům při tvorbě aplikačních programů.

3.1 Slovní popis

Slovní popis (viz [6]) je jeden z nejzákladnějších popisů vůbec, protože vychází z přirozeného jazyka. Dá se jím popsat celá aplikace i jednotlivé části programu, záleží jen na hloubce výkladu popisované věci. Slovní popis má mnoho podob. Jednou z podob je souvislý text (příklad 3), který ale při popisu technických problémů, jako třeba algoritmů, není vhodný. Při čtení souvislého textu se vyžaduje soustředění pozornosti na text, což čtenáře odvádí od soustředění na problém. Je tedy dobré souvislý text převést na grafickou formu (vývojový diagram, diagram datových toků) nebo upravit jeho formu. Vhodnou formou je strukturovaný text, kdy jednotlivé kroky popisujeme na samostatných řádcích, číslovaný seznam (příklad 4) a pseudokód.

> Programu je předáno číslo `n`, o kterém víme, že je větší než číslo `2`. Máme zjistit, zda je toto číslo prvočíslem. V intervalu `<2, n-1>` nesmí existovat žádný dělitel čísla `n`.

Příklad 3: souvislý text příkladu pro zjištění prvočísla

1. načteme číslo `n`
2. přiřadíme `i=2`
3. je-li `i` dělitelem `n`, jdeme na krok 7
4. `i` zvýšíme o `1`
5. je-li `i` menší než `n`, jdeme na krok 3
6. zaznamenáme, že `n` je prvočíslo, jdeme na krok 8
7. zaznamenáme, že `n` není prvočíslo
8. konec výpočtu

Příklad 4: Číslovaný seznam příkladu pro zjištění prvočísla

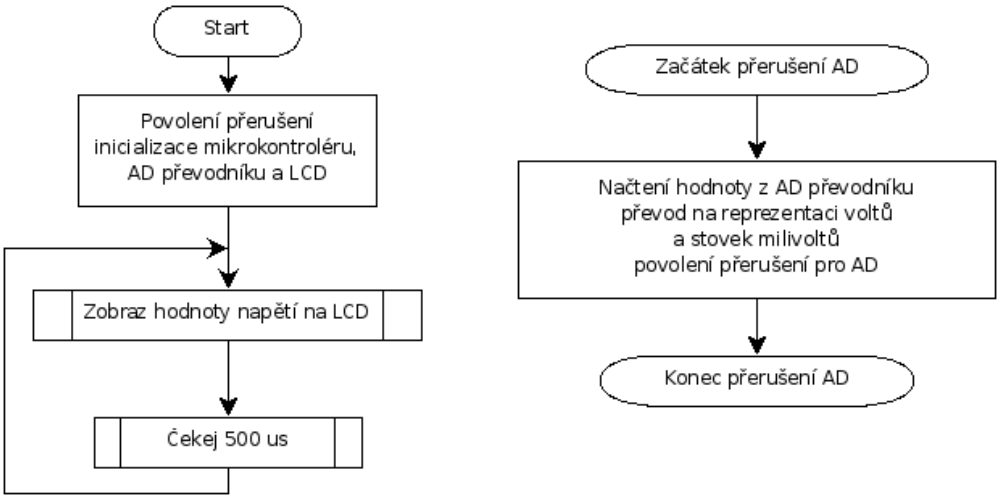
3.2 Vývojový diagram

Algoritmus nebo větvení programu je možné znázornit graficky pomocí vývojového diagramu (viz [6]). Vývojové diagramy zobrazují posloupnost operací, které se mají provádět v průběhu transformace vstupních dat na data výstupní. Jsou definovány normou.

Vývojové diagramy obsahují:

- symboly pro vlastní operace zpracování včetně symbolů definujících stanovený tok, který má být dodržen při zachování logických podmínek
- symboly spojení, které indikují tok řízení
- zvláštní symboly pro usnadnění čtení a zápisu vývojového diagramu.

Výhodou vývojového diagramu je jeho názornost a přehlednost. Nevýhodou je pracnost a složitost konstrukce. Složité diagramy se nevejdou na jednu stránku a stávají se méně přehlednými. Navíc jsou malé možnosti pro pozdější úpravy. Příklad vývojového diagramu je na obrázku 2.

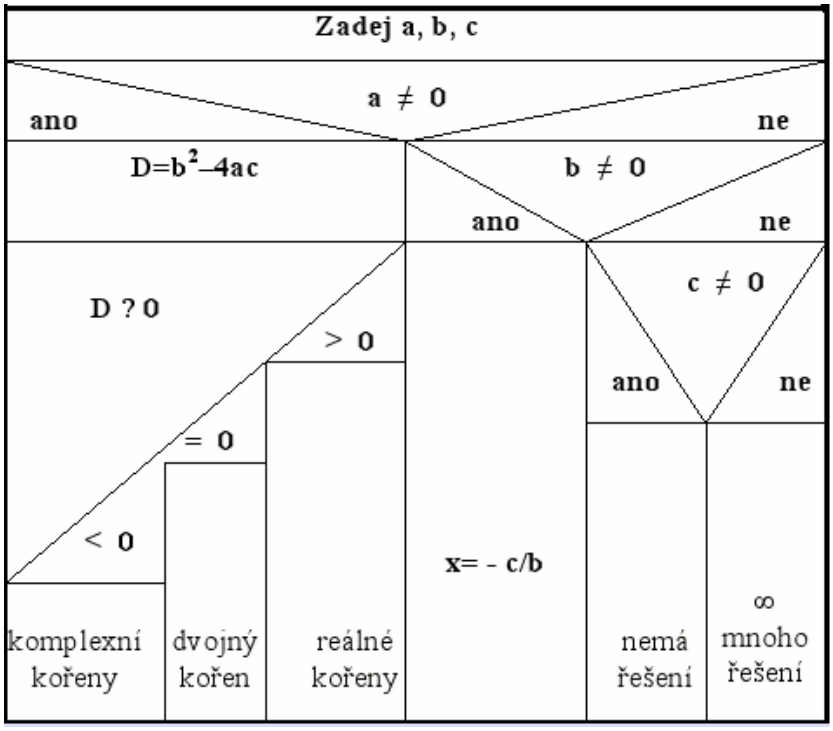


Obrázek 2: Vývojový diagram jednoduchého voltmetru

3.3 Strukturogramy

Strukturogramy (viz [6]), někdy také nazývané Nassi Shneiderman diagramy nebo N-S diagramy, jsou obdobou vývojových diagramů. Nejsou ale definovány normou. Představovaly určitý pokus zhutnit grafickou interpretaci výpočetního postupu. Jsou tvořeny obdélníkovou tabulkou, kde do řádků zapisujeme postup kroků symbolickou či slovní formou v pořadí, v jakém budou prováděny.

Záhlaví tabulky obsahuje název algoritmu nebo dílčího kroku. V praxi se však neujaly a dnes je jejich praktické použití bezvýznamné. Příklad strukturogramu je na obrázku 3.

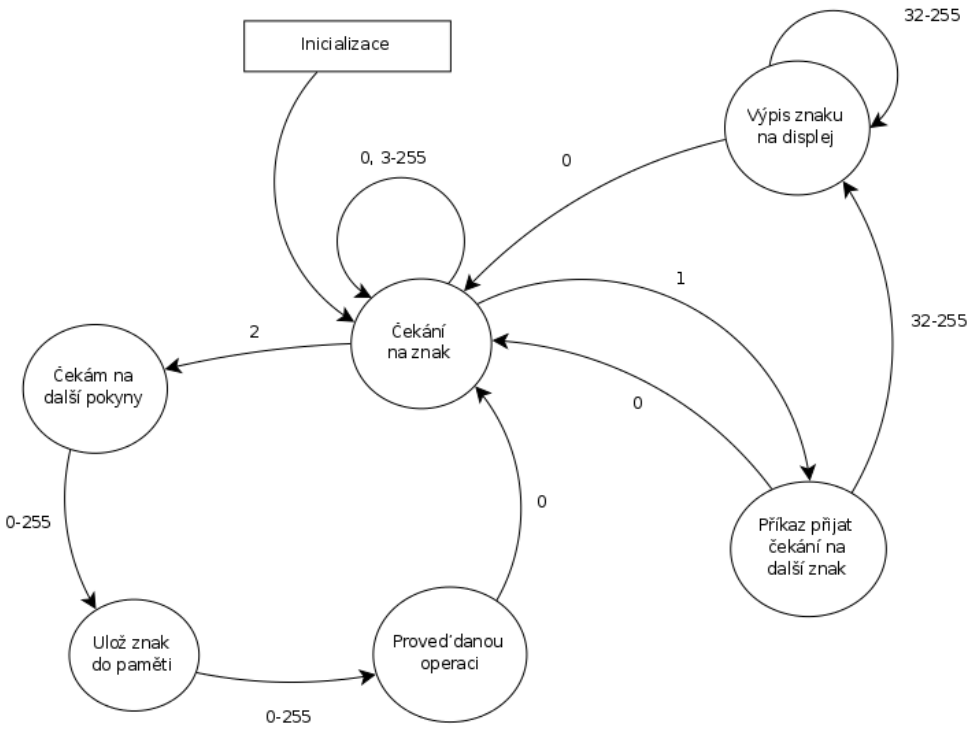


Obrázek 3: Strukturogram algoritmu řešení kvadratické rovnice

3.4 Konečný automat

Konečný automat (viz [16]) je jednoduchý výpočetní model, používaný v informatice pro studium vyčíslitelnosti. Je popsán konečnou množinou stavů, mezi kterými přechází na základě vstupních signálů. Na začátku se automat nachází v definovaném počátečním stavu. Při příchodu vstupního signálu automat přejde do stavu daného přechodovým diagramem. Přechod do dalšího stavu závisí na současném stavu automatu a na vstupním signálu.

Přechod mezi stavy automatu se zobrazuje šipkami. Pokud může automat z jednoho stavu přecházet do více dalších stavů, je nutno vyznačit hodnoty vstupních signálů, při kterých nastávají jednotlivé přechody. Příklad konečného automatu je zobrazen na obrázku 4.

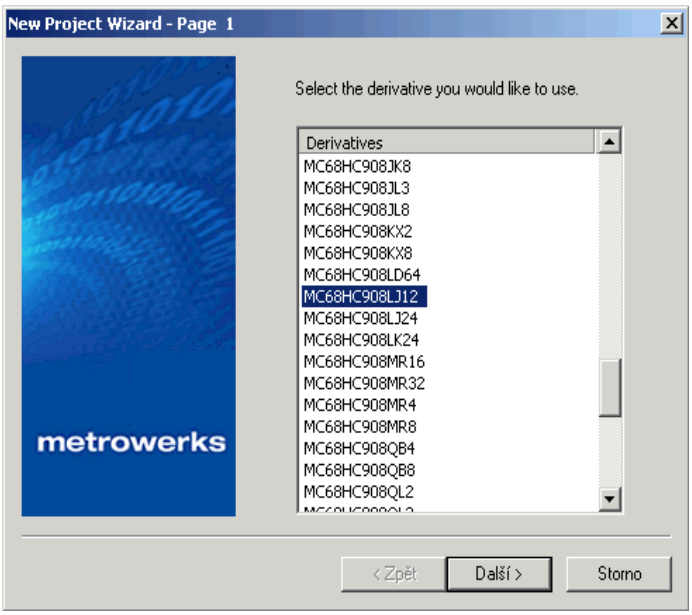


Obrázek 4: Konečný automat obsluhující SCI

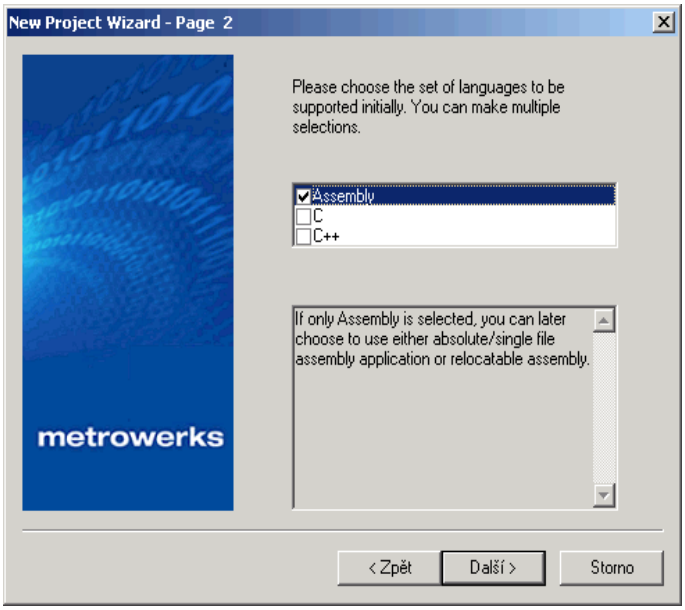
4 Návrh sady úloh

Úlohy jsou k bakalářské práci přiloženy na disku CD-ROM. Na disku jsou přiloženy vývojové diagramy k laboratorním cvičením z předmětu *IMP - Mikroprocesorové a vestavěné systémy*, které jsou umístěny ve složce *Vývojové diagramy k laboratořím z IMP*. Příklady k mikrokontrolérům jsou členěny podle označení jednotlivých mikrokontrolérů a řazeny do odpovídajících složek.

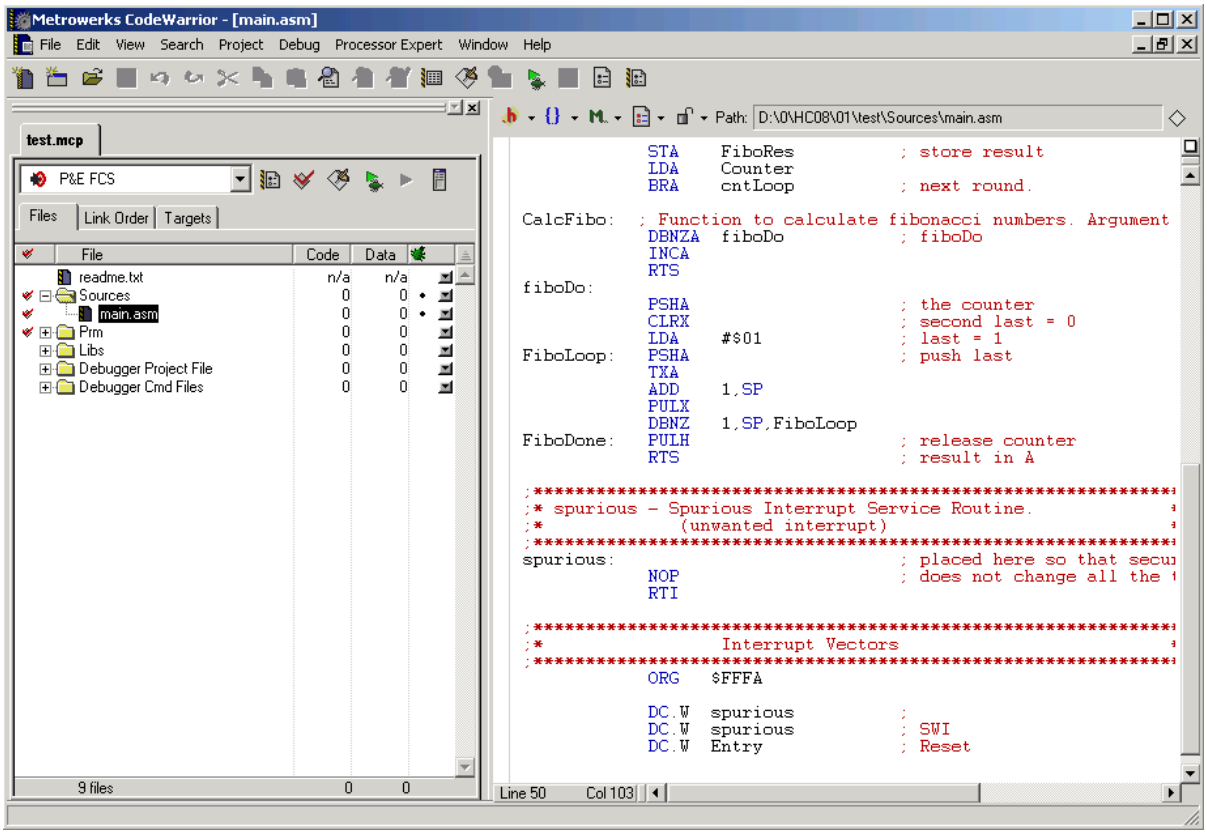
Úlohy pro mikrokontrolér 68HC908LJ12 byly odladěny ve vývojovém prostředí CodeWarrior. Toto prostředí nabízí podporu řady mikrokontrolérů (obrázek 5), programovací jazyky assembler, C, C++ (obrázek 6) a spoustu dalších funkcí. Na obrázku 7 je vidět hlavní okno vývojového prostředí. Na obrázku 8 je simulátor a real-time debugger (ladicí program v reálném čase), který pracuje přímo s mikrokontrolérem. Při simulaci můžeme nastavovat vstupní hodnoty periferiím, jakou jsou např. porty (obrázek 9 a obrázek 10). Podpora umístění bodu přerušení (breakpoint, viz obrázek 11) je samozřejmostí. Vývojové prostředí CodeWarrior je precizně propracované a nabízí uživateli spoustu možností a funkcí.



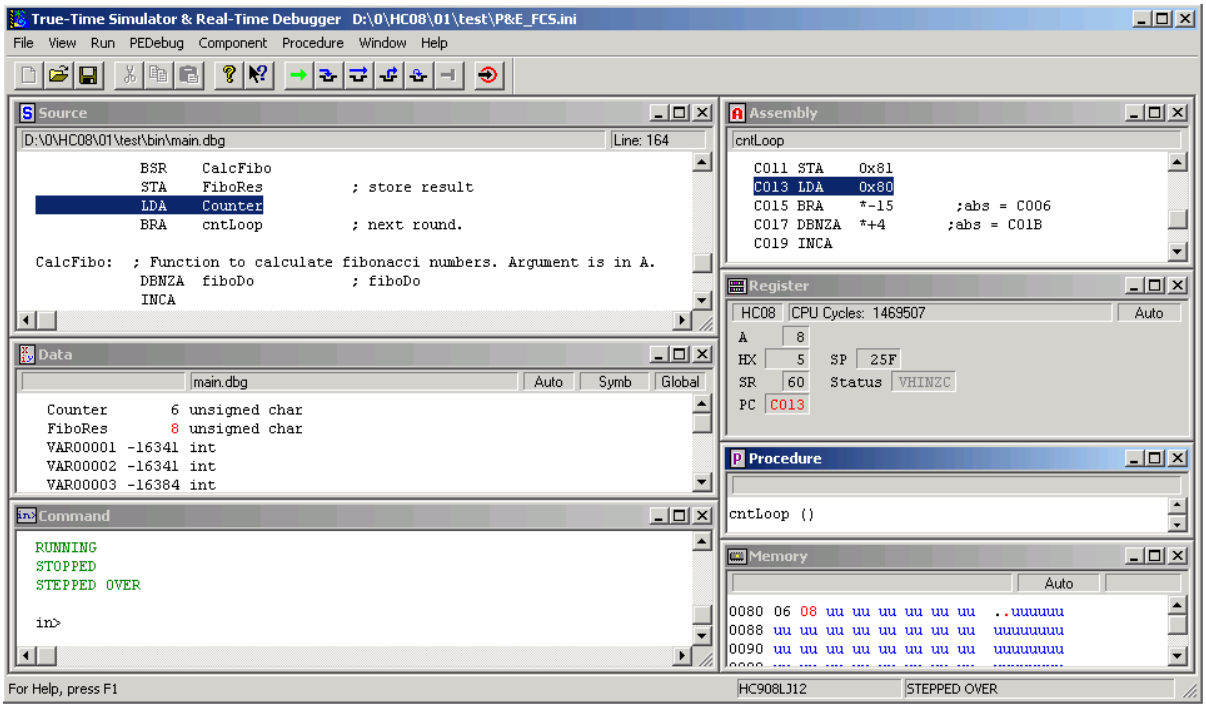
Obrázek 5: Výběr typu mikrokontroléru



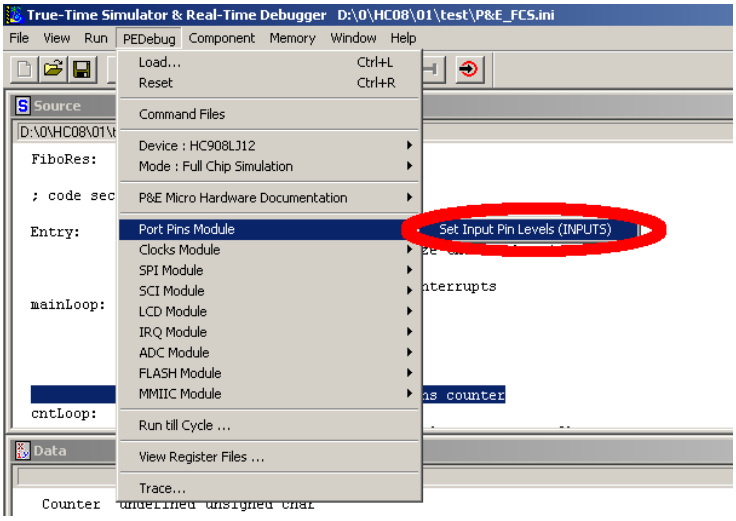
Obrázek 6: Výběr programovacího jazyka



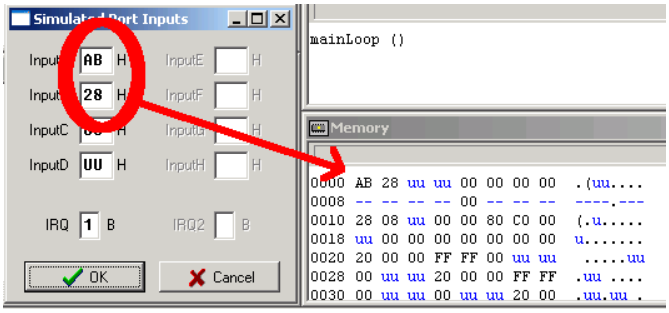
Obrázek 7: Hlavní okno vývojového prostředí



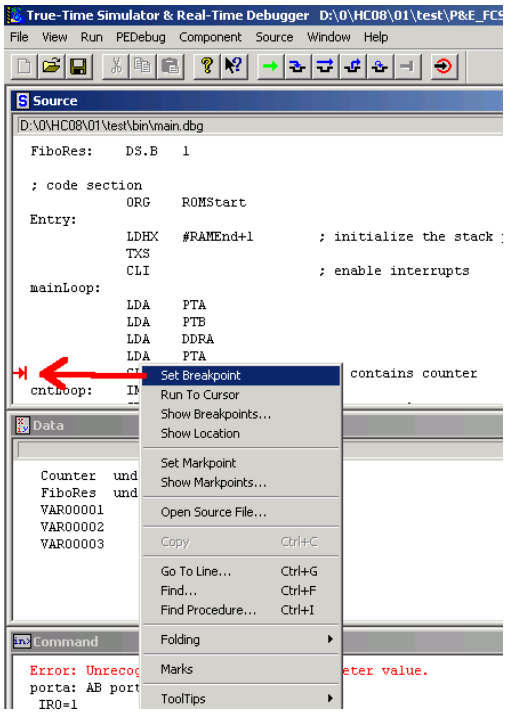
Obrázek 8: Simulátor a real-time degugger



Obrázek 9: Výběr editace hodnot na portu při simulaci



Obrázek 10: Změna hodnot na portech A a B



Obrázek 11: Umístění bodu přerušení

5 Závěr

Tato práce je zaměřena na vlastnosti nejznámějších jader 8-bitových mikrokontrolérů. Popisuje jejich architekturu a typické vlastnosti. Po přečtení práce by uživatel měl být schopen vybrat si jedno jádro, které mu vyhovuje z hlediska předchozích zkušeností nejvíce. Navržená sada úloh má napomoci uživateli k porozumění ovládání periferií mikrokontroléru M68HC908LJ12. Úlohy jsou odladěny ve vývojovém prostředí CodeWarrior.

Situace na poli jednočipových mikrokontrolérů je stále dynamická. U vícebitových mikrokontrolérů se stále častěji setkáváme se smíšenými signálovými procesory a programovatelnými logickými obvody CPLD a FPGA. Architektura vícebitových mikrokontrolérů je rozdílná od 8-bitových mikrokontrolérů. Objevují se stále komplikovanější a výkonnější periferní obvody. K dispozici je také celá řada operačních systémů.

Další vývoj práce může být zaměřen na 32-bitové mikrokontroléry, které jsou stále používanější. Dále může být rozšířena sada úloh o jiné typy mikrokontrolérů, jiné vývojové prostředí nebo o jiné programovací jazyky. Zajímavá je také oblast programovatelných logických obvodů CPLD a FPGA a oblast operačních systémů mikrokontrolérů.

Literatura

[1] Váňa, Vladimír: *Začínáme pracovat s mikrokontroléry Motorola HC08 NITRON*, Praha 2003, nakladatelství BEN, ISBN 80-7300-124-1.

[2] Rozehnal, Zdeněk: *Mikrokontroléry Motorola HC11*, Praha 2001, nakladatelství BEN, ISBN 80-86056-77-5.

[3] Predko, Myke: *Handbook of Microcontrollers*, McGraw-Hill, ISBN 0079137164, 1998.

[4] Váňa, Vladimír: *Mikrokontroléry Atmel AVR – popis procesoru a instrukční soubor*, Praha 2003, nakladatelství BEN, ISBN 80-7300-083-0

[5] Váňa, Vladimír: *Mikrokontroléry Atmel AVR – assembler*, Praha 2003, nakladatelství BEN, ISBN 80-7300-93-8

[6] Kreslíková, Jitka: *Skripta pro předmět IZP - Základy programování*, FIT VUT Brno, 2006

[7] Schwarz, Josef, Růžička, Richard, Strnadel, Josef: *Skripta pro předmět IMP – Mikroprocesorové a věstavěné systémy*, FIT VUT Brno 2006

[8] Sekanina, Lukáš: *Skripta pro předmět INP – Návrh počítačových systémů*, FIT VUT Brno, 2006

[9] Kopelet, Jiří: *KONSTRUKČNÍ ELEKTRONIKA A RADIO*, ročník VIII, 2003, číslo 1, strana 3

[10] stránka korporace Atmel, výrobce mikrokontrolérů s jádry 8051 a AVR [online].
Dostupný na URL: http://www.atmel.com/ (květen 2008)

[11] stránka firmy Freescale Semiconductor, výrobce mikrokontrolérů s jádry HC08 a HC11 [online].
Dostupný na URL: http://www.freescale.com/ (květen 2008)

[12] stránka firmy Microchip Technology Inc., vyrobce mikrokontrolérů s jádrem PIC [online].
Dostupný na URL: http://www.microchip.com/ (květen 2008)

[13] stránka firmy STMicroelectronics, výrobce mikrokontrolérů s jádrem ST7 [online].
Dostupný na URL: http://www.st.com/ (květen 2008)

[14] stránka firmy Intel [online].
Dostupný na URL: http://www.intel.com/ (květen 2008)

[15] stránka firmy Texas Instruments [online].
Dostupný na URL: http://www.ti.com/ (květen 2008)

[16] Wikipedie: Otevřená encyklopedie: Konečný automat [online].
Dostupný na URL: http://cs.wikipedia.org/wiki/Kone%C4%8Dn%C3%BD_automat (květen 2008)

Seznam příloh

Příloha 1. CD-ROM