Elektronická podpora cvičení předmětu mikropočítače

The electronic support of education in microcontrollers course

Zdeněk Blata

Bakalářská práce
2007

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2006/2007

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Zdeněk BLATA**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Téma práce:	**Elektronická podpora cvičení předmětu mikropočítače**

Zásady pro vypracování:

1.Zpracujte literární rešerši na dané téma.
2.Prostudujte a popište současný stav metod tvorby webových stránek (HTML, PHP, SQL, CSS, atd.)
3.Realizujte webové stránky pro podporu výuky předmětu mikropočítače, které budou zpracovány ve formě výukových lekcí.
4.Zpracujte jednotlivé lekce i ve formě vhodné pro tisk přímo z webových stránek.

Rozsah práce:

Rozsah příloh:

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

Seznam odborné literatury:

1.MC9S08GB/GT Data Sheet, Rev.2.3. Freescale Semiconductor, 2004
2.CPU08 Central Processor Unit Reference manual. Motorola, 2001
3.HCS08 Family Reference Manual, Rev.1. Freescale Semiconductor, 2003
4.Váňa V.: Začínáme s mikrokontroléry Motorola HC08 Nitron. BEN – technická literatura, Praha 2003.
5.KOSEK, J. Téměř vše o WWW [online]. c1999-2006 , 2006/12/13 [cit. 2007-02-02]. Dostupný z WWW: http://www.kosek.cz/.

Vedoucí bakalářské práce: **Ing. Petr Dostálek**
Ústav automatizace a řídicí techniky

Datum zadání bakalářské práce: **13. února 2007**

Termín odevzdání bakalářské práce: **24. května 2007**

Ve Zlíně dne 13. února 2007

prof. Ing. Vladimír Vašek, CSc.
děkan

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Cílem mé bakalářské práce je zpracovat literární rešerši na téma eLearning, popsat současný stav tvorby webových stránek, tzn. pojednat o technologiích HTML, CSS, PHP apod. a vytvořit webové stránky pro podporu výuky cvičení předmětu „Mikropočítače“. Webové stránky jsou zpracovány do jednotlivých výukových lekcí, které jsou ve formě vhodné pro tisk. Každá lekce obsahuje obrázky pro názornější vysvětlení učiva a obrázky se zdrojovými kódy programů, které by studenti měli napsat a odzkoušet v hodinách cvičení. Navíc je v závěru každé lekce odkaz na test, v němž je možné ověřit si znalosti dané látky.

Klíčová slova: eLearning, www, URL, HTTP, HTML, CSS, PHP, mikropočítač

ABSTRACT

The object of my bachelors thesis is elaboration of literature search in subject of eLearning,
describing recent stadium of web sites production, it means enter on technologies HTML, CSS, PHP etc. and making web sites to support education in microcontrollers course. The web sites are worked up into each education lesson, which are in form ready for printing. Each lesson contains pictures for better illustration of the subject and the pictures with the program source codes, which could students write and practice in their practical lessons. At the end of each lesson is the reference to test, in which is possible to prove the knowlage of the subject.

Keywords: eLearning, www, URL, HTTP, HTML, CSS, PHP, microcontroller

Za volnou ruku, toliko prostoru a podpory při práci děkuji vedoucímu mé bakalářské práce Ing. Petru Dostálkovi.

Dostatečně pokročilou technologii nelze odlišit od magie.

Arthur C. Clarke

Až mi bude tak sto deset, přijde Bůh a zmáčkne RESET.

Karel Plíhal

Prohlašuji, že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků, je-li to uvolněno na základě licenční smlouvy, budu uveden jako spoluautor.

Ve Zlíně …….

Podpis diplomanta

OBSAH

ÚVOD

Každá doba si s sebou nese své. Byly doby nejrůznější, s nejrůznějším dopadem na budoucnost. O době, ve které žijeme, o současnosti, hovoříme jako o době počítačů.

Ať si to uvědomujeme či ne, ať si to připouštíme či ne, počítače zasahují do nejrozmanitějších stránek našeho běžného každodenního života. Stěží jsme si dovedli před pár lety představit, kde všude by se počítače mohli vyskytovat, s čím vším by mohli pomáhat, co všechno usnadňovat a kde všude by mohli šetřit energii a čas nebo naopak energii i čas brát, tak jako si dnes jen stěží dokážeme představit kam až povede exponenciálně narůstající technický pokrok a rozvoj.

Pojmem dneška je bezesporu „Internet“. Ovšem když se řekne Internet, tak si drtivá většina naší společnosti nejdříve vybaví pojem webová (internetová) stránka ale už vůbec ne, že se jedná o celosvětovou počítačovou „supersíť“.

Webové stránky sdílejí neuvěřitelné množství informací, a proto s čistým svědomím nazýváme Internet „studnicí informací“. Ovšem množství přijímaných informací je dnes mnohem větší, než jaké jsme schopni pojmout. Takřka denně se setkáváme s nějakým novým neznámým pojmem. Proto je nutné odnášet si ve vědrech ze studnice informací právě ty informace, které pro nás mají větší cenu a jsou důležitější pro náš osobní rozvoj.

Každopádně se lidstvu otevřely nové velké možnosti ve vzdělávání. To dalo za vnik novému oboru, který se nazývá eLearning, pod jehož názvem si můžeme představit elektronické vzdělávání – elektronickou učebnici (eUčebnici).

Tématem této bakalářské práce jedna taková eUčebnice je. Její úkol spočívá v elektronické podpoře cvičení předmětu mikropočítače. Učebnice vtisknutá do prostředí webových stránek opakuje a rozvíjí látku probíranou na hodinách cvičení a to objasněním daného problému za pomoci názorných obrázků a ukázek zdrojových kódů. Správné pochopení nabytých vědomostí je pak kontrolováno v testech. Celkově jde o pochopení funkce a uplatnění mikropočítačů v praxi. Do eKnihovny eUčebnic, do police s popisným štítkem eLearning, by měla mezi ostatní přibýt jedna další.

I. TEORETICKÁ ČÁST

1 E-LEARNING

Pojem eLearning je pojmem mladým, ještě ne tolik rozšířeným a uzrálým, nicméně se nemalou rychlostí dostává do podvědomí okolní společnosti. Jeho jednotná definice ještě není specifikována. eLearning je vzdělávací proces, využívající informační a komunikační technologie k tvorbě kurzů, k distribuci studijního obsahu, komunikaci mezi studenty a pedagogy a k řízení studia. Může se jednat o rozsáhlé kurzy plně distančního charakteru a propracované nástroje kolaborativního učení, naopak ale může jít jen o doplnění prezenční výuky. [1]

1.1 Citované definice

Definice citované z [1]:

1. *eLearning je výuka s využitím výpočetní techniky a internetu.* (Petr Korviny, Moodle (nejen) na OPF, OPF, 2005)
2. *eLearning je v podstatě jakékoli využívání elektronických materiálních a didaktických prostředků k efektivnímu dosažení vzdělávacího cíle s tím, že je realizován zejména/nejenom prostřednictvím počítačových sítí.* (Kamil Kopecký, Základy e-learningu, Net University s.r.o., 2005)
3. *eLearning je vzdělávací proces, využívající informační a komunikační technologie k tvorbě kursů, k distribuci studijního obsahu, komunikaci mezi studenty a pedagogy a k řízení studia.* (Jan Wagner, Nebojme se eLearningu, Česká škola, 2005)
4. *eLearning je forma vzdělávání využívající multimediální prvky - prezentace a texty s odkazy, animované sekvence, video snímky, sdílené pracovní plochy, komunikaci s lektorem a spolužáky, testy, elektronické modely procesů, atd. v systému pro řízení studia (LMS).* (Virtuální Ostravská universita, 2005)

1.2 Výhody

Většina výhod zde jmenovaných bude vztažena k porovnání s klasickou školní vyučovací hodinou, na kterou je potřeba se dostavit osobně, v daný čas, s potřebným učebním materiálem apod.

- Časová nezávaznost
  - danou látku si osoba vzdělávaná může prostudovat v kteroukoli denní dobu.
  - kdykoliv cítí únavu může učení přerušit a následně se k němu vrátit, aniž by mu utekla část výkladu vyučujícího.
- Distanc osobní přítomnosti
- V případě neznalosti termínu použitého při výkladu je možnost dohledat si jeho význam v některých z webových encyklopedií.
- Pokud to dané prostředí dovoluje, je možnost výběru různého designu. Pak má uživatel možnost uzpůsobit si vzhled prostředí tak, aby mu bylo co možná nejsympatičtější (např. různá barevná schémata prezentací či výběr z více definovaných stylů www stránek).
- Kolektivnost - možnost probírání určitého problému např. v diskusních fórech či pomocí e-mailů s lidmi, které daný problém opravdu zajímá a touží po jeho pochopení.
- Pokud se jedná například o výuku programování je vítanou možností kopírování ukázkových zdrojových kódů, které by si jinak uživatel musel pracně přepisovat z učebnice.
- Animované obrázky
- Doprovodný zvuk
- a mnohé další podle typu použitých prostředků k eLearningu

1.3 Nevýhody

Nevýhody můžeme spatřit například v tom, že musíme:

- mít počítač anebo alespoň přístup k němu
- mít potřebné programové vybavení
- být dostatečně gramotní pro práci s počítačem
- mít možnost připojit se k síti Internet - pokud se jedná o online kurzy

1.4 Problémy spojené s tradičním přístupem k eLearningu

Lze nalézt řadu příčin, proč některé eLearningové projekty nepřináší očekávané výsledky. Mezi základní příčiny patří:

- Nízká míra zapamatování výuky (jak pro klasickou tak eLearningovou výuku)
- Standardní kurzy – navržené pro distribuci pro velký počet uživatelů a proto neřešící specifické potřeby jednotlivce.
- Nedostatek interakcí – s ostatními studenty. [3]

1.4.1 Nízká míra zapamatování výuky

Dle výzkumů Ministerstva obrany US studenti si zachovají jen velmi krátce informaci kterou pouze slyší, uchovají si 40% informace kterou slyší a vidí, a uchovají si 75 % informace, kterou slyší, vidí a současně si mohou vyzkoušet (ověří si vlastní aktivitou). Proto studenti, kteří se pouze pohybují kurzem a čtou a vidí informaci si většinou mnoho obsahu nezapamatují. Rovněž pouhé střídání podávání informace s blokem otázek k zamyšlení se postupně stává stereotypem a nepřináší kýžený efekt. Velký význam pro zapamatování má totiž rovněž poutavé zpracování obsahu – musí zde být výrazná přidaná hodnota oproti možnostem tištěného zpracování, které se za současných technických podmínek čte lépe než jeho obdoba na počítači. Velký význam tedy mají v obsahu různé simulace, příběhy, hry, vhodná kombinace zvuku, obrazu a interaktivního zapojení studenta do kurzu. [3]

1.4.2 Standardní kurzy

Tradičně jsou eLearningové kurzy standardizovány, aby vyhověly co největšímu počtu studentů. Jsou navrženy tak, aby si jak nejzdatnější tak nejméně zdatný student z kurzu něco odnesl. To však na druhé straně dává zbytečně mnoho informací jedněm studentům a nedostatek jiným. Rovněž obsah může být pro jedny moc složitý a pro jiné příliš jednoduchý. Výsledkem jsou kurzy, které nemusí být vysoce efektivní pro konkrétní studenty, protože nemohou naplnit jejich specifické potřeby ve vzdělávání. [3]

1.4.3 Nedostatek interakcí

Kontakt s ostatními studenty ve třídě je klíčový element efektivního učení. Schopnost družit se s ostatními, sdílet zkušenosti a debatovat o tématu je významný prvek úspěšného učení se. Potíž eLearningu bývá někdy nedostatek možnosti těchto interakcí s ostatními. Tohle částečně nahrazují diskusní fóra a posílané zprávy prostřednictvím emailů. Těžko ovšem nahradí přímý kontakt. [3]

1.5 Přínos

Přínos eLearningu může být vysoký, pokud je zvolen správný přístup. Nové pokročilé technologie kombinované se spolehlivými klasickými strategiemi umožňují, aby učení bylo adresný, individuelní, interaktivní a poutavý proces, který je integrován do každodenního života studenta. Tato integrace pak umožňuje skutečné zvyšování efektivity učení. Vždy jde o to, jaký přístup je zvolený jak ze strany tvůrců eLearningu tak ze strany jeho uživatelů. [3]

1.6 Odkaz na eLearningový kurz

Vhodným příkladem eLearningu je webová stránka na adrese www.jakpsatweb.cz, kterou píše Yuhů, Dušan Janovský, a která pojednává o tvorbě, údržbě a vylepšování webových stránek. Denně ji navštěvuje okolo deseti tisíc uživatelů. Mně samotnému velmi pomáhala při tvorbě praktické části bakalářské práce. Je velmi přehledná, dá se v ní snadno a rychle orientovat a je psána výstižně a poutavě. Takové vlastnosti by měla splňovat každá eLearningová pomůcka.

1.7 Budoucnost

Zatím se eLearning spojuje především s osobními počítači. Díky rozvoji nových kategorií výkonných komunikačních prostředků, jako jsou kapesní či osobní počítače či organizéry, ale také nová generace mobilních telefonů, které umožňují připojení k internetu, se začíná hovořit i o m-learningu – mobilním vzdělávání. Dnešní mobilní telefony mají dostatečný výkon i pro přehrávání videopořadů a není důvod, aby nemohly sloužit ke vzdělávání, stejně jako slouží k přístupu k informacím na internetu. [1]

2 SOUČASNÝ STAV METOD TVORBY WEBOVÝCH STRÁNEK

WWW je zkratkou anglického výrazu **World Wide Web**, který se volně překládá jako „***celosvětová pavučina***“. Je to jedna z nejrozšířenějších služeb využívaných v celosvětové počítačové síti nazývající se Internet. Jde o označení pro aplikace internetového protokolu HTTP. Je tím myšlena soustava propojených hypertextových dokumentů.

V češtině se slovo web často používá nejen pro označení celosvětové sítě dokumentů, ale také pro označení jednotlivé soustavy dokumentů dostupných na tomtéž webovém serveru nebo na téže internetové doméně nejnižšího stupně (internetové stránce).

Dokumenty umístěné na počítačových serverech jsou adresovány pomocí URL, jejíž součástí je i doména a jméno počítače. Název naprosté většiny těchto serverů začíná zkratkou www, i když je možné používat libovolné jméno vyhovující pravidlům URL.

Protokol HTTP je dnes již používán i pro přenos jiných dokumentů, než jen souborů ve tvaru HTML a výraz World Wide Web se postupně stává pro laickou veřejnost synonymem pro internetové aplikace. [1]

Služba WWW je kombinací technologií:

- URL – jednoznačná identifikace zdroje v Internetu
- HTTP – komunikační protokol (mezi klientem a serverem)
- HTML – jazyk pro formátování komplexních dokumentů
- CSS – kaskádové styly formátování HTML
- Java, JavaScript, VBScript, … – technologie pro tvorbu dynamických stránek, jejichž skripty se vykonávají na straně klienta
- CGI, PHP, ASP, ... – technologie pro tvorbu dynamických stránek, jejichž skripty se vykonávají na straně web serveru

2.1 URL

URL je zkratka anglického výrazu **Uniform Resource Locator**, jehož překlad zní „***jednoznačné určení zdroje***“. Je to řetězec znaků s definovanou strukturou a je to způsob, jak jednoznačně zapsat umístění souboru na Internetu nebo na intranetu. V HTML se používá jak pro zacílení odkazů, tak pro načítání obrázků a podpůrných souborů. [2]

2.1.1 Jednotlivá pole v URL

protokol, doménové jméno, port, specifikace souboru, parametry

Některá pole jsou nepovinná – buď nemají význam, nebo se předpokládá předdefinovaná hodnota, závislá např. na protokolu (např. pro HTTP je implicitní port 80), nebo na aplikaci (pro webový prohlížeč HTTP). [1]

2.1.2 Příklad

Jako příklad je uvedena URL adresa webové stránky www.jakpsatweb.cz, kdy jsme se přes odkazy dostali až na stránku pojednávající o URL a na které je uvedena tahle tabulka:

http://www.jakpsatweb.cz/html/url.htm#priklad

Tab. 1 Rozložení URL na jednotlivé části adresy [2]

Část adresy	Příklad	Jiné možné hodnoty
protokol	http://	ftp:// , mailto:, atd.
doména 3. úrovně (server)	www.	cokoliv.
doména 2. úrovně	jakpsatweb.	seznam. , google. , apod.
generická doména	cz	com, sk, gov apod.
port	nic	:80 , :*číslo*
cesta (adresáře)	/html/	/, /cokoliv/adresar/
jméno souboru	url.htm	index.html apod.html
záložka	#priklad	#jménozáložky
dotaz	nic	?proměnná=hodnota

2.2 HTTP

HTTP (Hyper Text Transfer Protocol) je internetový protokol určený původně pro výměnu hypertextových dokumentů ve formátu HTML. Používá obvykle port TCP/80. Tento protokol je spolu s elektronickou poštou tím nejvíce používaným a zasloužil se o obrovský rozmach internetu v posledních letech.

V současné době je používán i pro přenos dalších informací. Pomocí rozšíření MIME umí přenášet jakýkoli soubor (podobně jako e-mail), používá se společně s formátem XML pro tzv. webové služby (spouštění vzdálených aplikací) a pomocí aplikačních bran zpřístupňuje i další protokoly, jako je např. FTP nebo SMTP. [1]

2.2.1 Činnost protokolu

Protokol funguje způsobem dotaz-odpověď. Uživatel (pomocí programu, obvykle internetového prohlížeč) pošle serveru dotaz ve formě čistého textu, obsahujícího označení požadovaného dokumentu (URL), informace o schopnostech prohlížeče apod. Server poté odpoví pomocí několika řádků textu (hlavičky) popisujících výsledek dotazu (zda se dokument podařilo najít, jakého typu dokument je atd.), za kterými následují data samotného požadovaného dokumentu.

Pokud uživatel bude mít po chvíli další dotaz na stejný server (např. proto, že uživatel v dokumentu kliknul na hypertextový odkaz), bude se jednat o další, nezávislý dotaz a odpověď. Z hlediska serveru nelze poznat, jestli tento druhý dotaz jakkoli souvisí s předchozím. Kvůli této vlastnosti se protokolu HTTP říká *bezstavový protokol* – protokol neumí uchovávat stav komunikace, dotazy spolu nemají souvislost. Tato vlastnost je nepříjemná pro implementaci složitějších procesů přes HTTP (např. internetový obchod potřebuje uchovávat informaci o identitě zákazníka, o obsahu jeho „nákupního košíku“ apod.). K tomuto účelu byl protokol HTTP rozšířen o tzv. HTTP cookies, které umožňují serveru uchovávat si informace o stavu spojení na počítači uživatele. [1]

2.2.2 Příklad HTTP komunikace

Uživatel si chce například nechat zobrazit stránku http://www.jakpsatweb.cz/server/http-protokol.html

1. Uživatel toto URL zadá do prohlížeče.

2. Prohlížeč(klient) si vyhodnotí doménu, přes DNS si zjistí, jaké IP adresy se má ptát.
3. Přes TCP protokol naváže spojení se serverem na zjištěné IP adrese. Teprve nyní začíná HTTP.
4. Prohlížeč pak pošle na server toto HTTP volání:

```
GET /server/http-protokol.html HTTP/1.1
HOST www.jakpsatweb.cz
```

5. Toto HTTP volání přijde na server, na kterém hostuje doména se jménem www.jakpsatweb.cz. Podle HOST se pozná, kterého virtuálního hostingu se bude požadavek týkat (pokud je jich na serveru více).
6. Server tedy vidí požadavek:

```
GET /server/http-protokol.html HTTP/1.1
```

7. Server se podívá se do kořene dokumentů (první lomítko) do adresáře server/ a vezme soubor http-protokol.html. Vidí, že ho má zpět poslat v protokolu HTTP verze 1.1. Připojí před soubor http hlavičky odpovědi, a pošle to zpět. Symbolicky vypadá odpověď takto:

```
Stavová odpověď
hlavička 1
hlavička 2 atd.
prázdný řádek
samotný html dokument
```

tedy konkrétně například takto:

```
HTTP/1.1 200 OK
Content-type: text/html
Date: Sun, 21 May 2006 17:10:21
GMT

<html>
<head>
<title>stránka</title>
...atd.
```

8. Tuto odpověď dostane klient (většinou tedy prohlížeč). Z odpovědi napřed odstraní všechny HTTP hlavičky, čili vše před prvním prázdným řádkem. Pak prohlížeč samotný HTML dokument zpracuje = zobrazí čtenáři. [2]

2.3 HTML

HTML je zkratka z anglického výrazu **HyperText Markup Language**, který překládáme jako „***značkovací jazyk pro hypertext***“. Je jedním z jazyků pro vytváření stránek v systému World Wide Web, který umožňuje publikaci stránek na Internetu. Jazyk HTML navrhli Tim Berners-Lee a Robert Caillau. Jazyk je podmnožinou dříve vyvinutého rozsáhlého univerzálního značkovacího jazyka SGML(Standard Generalized Markup Language). Vývoj HTML byl ovlivněn vývojem webových prohlížečů, které zpětně ovlivňovaly definici jazyka. [1]

2.3.1 Popis jazyka

Jazyk je charakterizován množinou značek a jejich atributů. Mezi značky se uzavírají části textu dokumentu a tím se určuje význam (*sémantika*) obsaženého textu. Názvy jednotlivých značek se uzavírají mezi úhlové závorky („< >“). Část dokumentu uzavřená mezi značkami tvoří tzv. **element** (prvek) dokumentu. Součástí obsahu elementu mohou být další vnořené elementy. Atributy jsou doplňující informace, které upřesňují vlastnosti elementu. Značky (také nazývané **tagy**) jsou obvykle párové. Rozlišujeme počáteční a koncové značky. Koncová značka má před názvem značky znak lomítko („</ >“). [1]

2.3.2 Příklady HTML tagů

Tab. 2 Příklady HTML tagů

tag	párový	význam	příklad
b	ano	tučné písmo	slovo
i	ano	kurzíva	slůvko
sup	ano	horní index	^{slovíčko}
sub	ano	dolní index	<sub>slovíčko</sup>
p	nepovinně	odstavec	<p>začátek nového odstavce
br	ne	zalomení řádku	…konec řádku.
hr	ne	vodorovná čára	<hr>

2.3.3 Struktura dokumentu

Tab. 3 Základní značky vymezující oblasti HTML souboru [2]

tag	párový	význam	výskyt
html	ano	začátek HTML dokumentu	na začátku souboru
head	ano	hlavička stránky	na začátku souboru
body	ano	tělo stránky + definice pozadí	za <head>
<!-- -->	ano	poznámka	kdekoliv
!doctype	ne	specifikace DTD	úplně na začátku souboru

HTML

Začíná a končí celý dokument. Veškerý další obsah musí být uvnitř. Jedná se o značku nepovinnou, většina prohlížečů si ji domyslí. Pokud však mají být soubory v souladu s normou, je vhodné tag používat. Tag html nemá žádné atributy. [2]

HEAD

Hlavička dokumentu, která se nezobrazuje. Obsahuje nepovinně další tagy (title, meta, link, style, script aj.). Pokud se v hlavičce použije prostý text, v některých prohlížečích se zobrazí na začátku stránky! Tag head nemá žádné atributy. [2]

BODY

Tělo dokumentu. Obsahuje veškerý zobrazovaný obsah stránky. Tag body obsahuje mnoho atributů, ale jejich používání se nahradilo využíváním CSS vlastností. [2]

POZNÁMKA

Všechno, co je v HTML souboru obaleno značkami „<!--“ a „-->“, je považováno za poznámku a nezobrazuje se. [2]

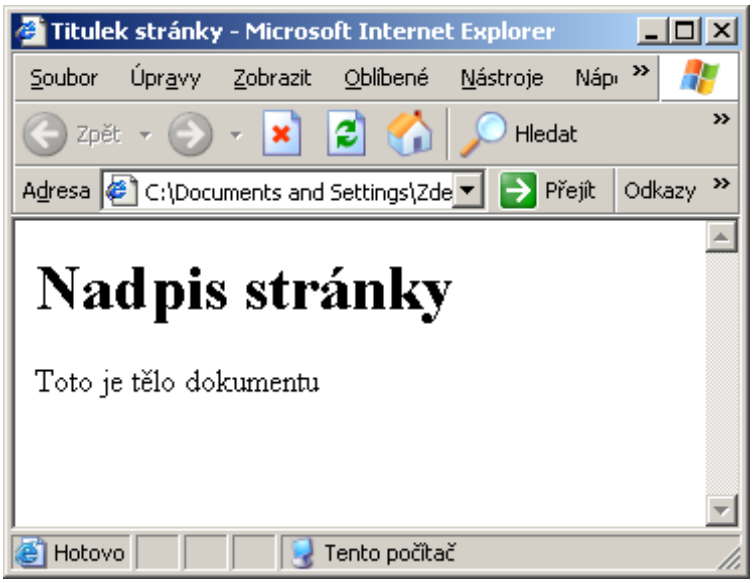
!DOCTYPE

Specifikace DTD. Píše se úplně na začátek souboru, ještě před značku <html>. Není nutné to dělat, ale podle standardu značkovacích jazyků SGML a XML je vhodné strukturovanou formou říci, že tento dokument je HTML dokument. [2]

2.3.4 Příklad zdrojového kódu

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<html>
<!-- toto je komentář -->
   <head>
      <title>Titulek stránky</title>
   </head>
<!-- tělo dokumentu -->
   <body>
      <h1>Nadpis stránky</h1>
      <p>Toto je tělo dokumentu</p>
   </body>
</html>
```

Výstupem takovéhoto zdrojového kódu bude stránka zobrazená na následujícím obrázku:



Obr. 1 Ukázková www stránka

2.4 CSS

CSS je zkratka pro anglický název **Cascading Style Sheets**, česky „***kaskádové styly***“. Kaskádové, protože se na sebe mohou vrstvit definice stylu, ale platí jenom ta poslední. Je to kolekce metod pro grafickou úpravu webových stránek. Pomocí CSS stylů lze definovat kromě barvy, písma a velikosti spoustu dalších věcí jako např. rámeček, podtržení, tučnost, vlnitost, zobrazení, odrážky, okraje atd. Hlavním smyslem je umožnit návrhářům oddělit vzhled dokumentu od jeho struktury a obsahu. Původně to měl umožnit už jazyk HTML, ale v důsledku nedostatečných standardů a konkurenčního boje výrobců prohlížečů se vyvinul jinak. [1], [2], [4]

2.4.1 Výhody CSS

- **oddělení struktury a stylu**
- **konzistentní styl** - na všech stránkách webové prezentace by měly být všechny nadpisy stejné úrovně, seznamy, zdůrazněné části textu apod. stejného stylu. S použitím formátovacích možností HTML je to obtížné – u každého objektu v každém dokumentu se vzhled objektu stále znovu nastavuje. S použitím CSS je to velmi jednoduché. Vytvoří se soubor stylu, který se připojuje k HTML dokumentu. Ve všech dokumentech jsou pak objekty stejného vzhledu.
- **rozsáhlejší možnosti** - CSS nabízí rozsáhlejší formátovací možnosti než samotné HTML.
- **dynamická práce se styly** - provést změnu stylu webu, který pro formátování vzhledu využívá jen možnosti HTML, znamená najít a nahradit všechny značky a změnit atributy mnoha dalších značek. V případě používání CSS znamená změna stylu webu přepsání jediného souboru – souboru stylů.
- **větší kompatibilita alternativních webových prohlížečů**
- **kratší doba načítání stránky** - soubor CSS se uloží do mezipaměti prohlížeče a pokud není změněn, tak se načítá pouze jednou a zobrazení stránek se velmi urychlí. [1]

2.4.2 Trojí použití CSS

1. **Přímo v textu** zdroje u formátovaného elementu
2. **Pomocí „stylopisu“** (angl. „stylesheet“) v hlavičce stránky. Stylopis je jakýsi seznam stylů. Je v něm obecně napsáno, co má být jak zformátováno, například že nadpisy mají být zelené. Do stránky se stylopis píše mezi tagy <style> a </style>.
3. **Použitím externího stylopisu** - to je soubor ***.css**, na který se stránka odkazuje tagem <link>. V souboru je umístěný stylopis. Hlavní výhoda je v tom, že na jeden takový soubor se dá nalinkovat mnoho stránek, takže pak všechny vypadají podobně. Tohle je nejčastější použití. [2]

2.4.3 Syntaxe

Stylový předpis se skládá z posloupnosti pravidel. Každé pravidlo určuje vzhled některého elementu dokumentu, nebo skupiny elementů. Pravidlo začíná tzv. selektorem, který specifikuje („adresuje“) skupinu elementů. Selektor je následován seznamem deklarací, které určují vzhled vybrané skupiny elementů. Celý seznam je uzavřen ve složených závorkách a jednotlivé deklarace jsou odděleny středníkem (tj. za poslední deklarací středník už být nemusí). [1]

2.4.4 Příklad deklarace stylu těla dokumentu, odstavce a nadpisu první úrovně

```
body {                              /* styl těla dokumentu*/
      color: black;                 /* černá barva textu  */
      background-color: white;      /* bíla barva pozadí  */
     }

  h1 {                    /* vzhled nadpisu první úrovně */
      margin: 5px;        /* okraj šířky 5 pixelů        */
      font-size: 12pt     /* velikost fontu 12 bodů      */
     }

   p {                      /* styl odstavce       */
      text-align: center;   /* text centrovat      */
      line-height: 10pt;    /* výška řádku 10 bodů */
     }
```

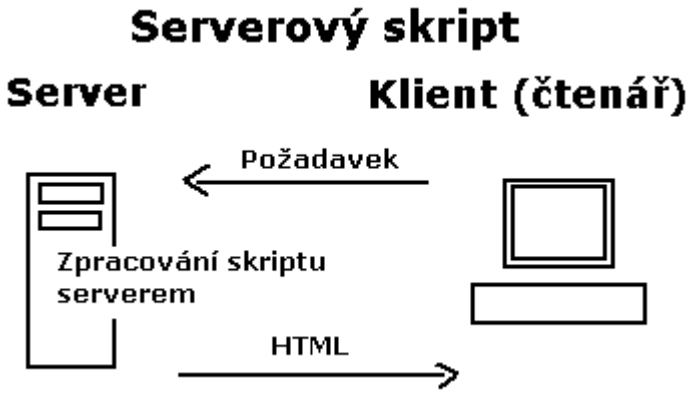
2.5 PHP

PHP je skriptovací jazyk fungující na straně serveru. Zkratka PHP pochází původně z anglického výrazu **Personal Home Page**, nyní se jedná o rekurzivní zkratku z výrazu **PHP: Hypertext Preprocesor**, která se dá přeložit jako „***PHP: Hypertextový preprocesor***“. Jazyk je určený především pro programování dynamických internetových stránek. Nejčastěji se začleňuje přímo do struktury jazyka HTML, což je velmi výhodné pro tvorbu webových aplikací, lze jej ovšem také použít i k tvorbě konzolových a desktopových aplikací. PHP se stalo velmi oblíbeným především díky jednoduchosti použití a tomu, že kombinuje vlastnosti více programovacích jazyků a nechává tak vývojáři částečnou svobodu v syntaxi. Syntaxe jazyka kombinuje hned několik programovacích jazyků a sice Perl, C, Pascal a Java. [1]

2.5.1 Historie

PHP je nástupcem staršího produktu, nazvaného PHP/FI, který vytvořil Rasmus Lerdorf v roce 1995, na počátku jako jednoduchou sadu skriptů v jazyce Perl pro zpracování záznamů o přístupech k jeho webu. Tuto sadu nazval „Personal Home Page Tools“. Protože byla třeba větší funkčnost, napsal Rasmus mnohem rozsáhlejší implementaci v C, která byla schopna komunikovat s databázemi a umožňovala uživatelům vyvíjet jednoduché dynamické aplikace pro Web. Rasmus se rozhodl uvolnit zdrojový kód PHP/FI pro všechny, takže kdokoli ho může používat, stejně jako opravovat chyby a vylepšovat kód. První oficiální verzí PHP, která se velmi blížila takovému PHP, jak ho známe dnes , byla verze PHP 3.0. Přes verzi PHP 4.0, vydanou v zimě roku 1998, přišla v červenci roku 2004 verze PHP 5.0, která je nyní verzí poslední. [5]

2.5.2 Provádění skriptů



Obr. 2 Zpracování serverového skriptu [2]

PHP skripty jsou prováděny na straně serveru, k uživateli je přenášen až výsledek jejich činnosti. Uživatel pošle serveru požadavek, ten se na serveru vykoná a k uživateli přijde už zpracovaný výsledek, tzn. příslušná HTML stránka.

2.5.3 Platformová nezávislost

PHP je nezávislý na platformě, skripty fungují bez úprav na mnoha různých operačních systémech (Windows, Linux, …), což je další vskutku velkou výhodou. [1]

2.5.4 Knihovny funkcí

PHP obsahuje rozsáhlé knihovny funkcí pro zpracování textu, grafiky, práci se soubory, přístup k většině databázových serverů (např. MySQL) a podporu celé řady internetových protokolů (např. HTTP, FTP, SMTP, POP3).[1]

Tab. 4 Ukázka funkcí pro práci s databází MySQL

funkce	popis
mysql_connect();	připojí se k MySQL databázi podle zadaných parametrů
mysql_select_db();	vybere příslušnou databázi
mysql_query();	funkce pro databázoví dotaz
mysql_error();	vypíše případnou chybu

2.5.5 Začlenění kódu PHP do dokumentu HTML

HTML soubory, které mají v sobě začleněný PHP kód musí mít příponu *.php. Dříve se také používalo *.php3 nebo *.php4. PHP kód může být vložen na libovolném místě HTML dokumentu. Vkládá se mezi značky „<?“ (popř. „<?php“) a „?>“. [2]

```
<?php
 /*tohle je poznámka psaná v PHP kódu*/
?>
```

2.5.6 Příklad zdrojového kódu

Ukázka kódu je vložena do těla HTML dokumentu. [2]

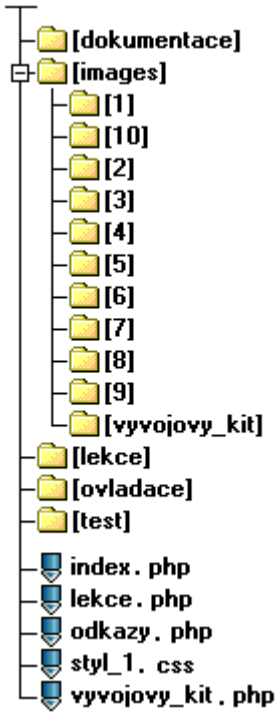
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<html>
<!-- toto je komentář -->
   <head>
      <title>Titulek stránky</title>
   </head>
<!-- tělo dokumentu -->
   <body>
      <h1>Nadpis stránky</h1>

      <?php
       echo "ahoj světe";      //vypíše textový řetězec v uvozovkách
      ?>

   </body>
</html>
```

II. PRAKTICKÁ ČÁST

3 WWW STRÁNKY E-LEARNINGOVÉHO KURZU

Praktická část bakalářské práce spočívala ve vytvoření webových stránek pro podproru výuky předmětu „Mikropočítače“. Výukový materiál opakuje a doplňuje informace získané na hodinách cvičení.

3.1 Adresářová struktura



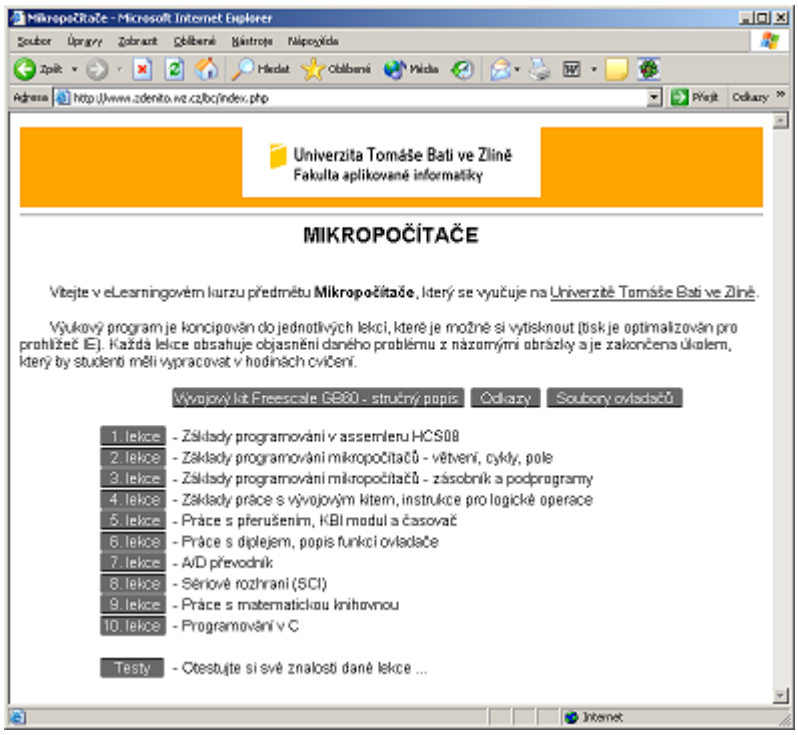
Obr. 3 Adresářový strom www stránek

Popis jednotlivých složek:

- **dokumentace** – obsahuje soubor *CPU_MC9S08GB60.pdf*, který je dokumentací k CPU HCS08.
- **images** – je složka obsahující použité obrázky, které jsou roztříděné podle jednotlivých lekcí. Převážná jejich většina je ve formátu PNG.
- **lekce** – je složka obsahující soubory s obsahem jednotlivých lekcí.
- **ovladace** – je složka obsahující zkomprimované soubory ovladače pro displej a sériové komunikační rozhraní a ovladače pro práci s matematickou knihovnou.
- **test** – je složka obsahující soubory testovacích otázek k jednotlivým lekcím.

Popis jednotlivých souborů:

- **index.php** – je hlavní soubor, úvodní stránka, rozcestník. Obsahuje odkazy na jednotlivé lekce, testy, soubory ovladačů, popis vývojového kitu a na stránku s odkazy.



Obr. 4 Hlavní stránka – index.php

- **lekce.php** – je soubor, kde je definováno záhlaví stránek s jednotlivými lekcemi, ve kterém se zobrazuje nadpis (téma dané lekce) a odkazy pro možnost přecházení mezi jednotlivými lekcemi či návratu na hlavní stránku. Do této stránky se načítá obsah jednotlivých lekcí pomocí PHP funkce „include();“.
- **odkazy.php** – stránka s odkazy na dokumentaci k CPU HCS08 a na stránku, kde je možnost stáhnout si vývojového prostředí CodeWarrior. Ke stáhnutí je na stránce napsán i krátký návod.
- **styl_1.css** – externí stylopis - soubor s nadefinovaným formátováním www stránek. Jakákoli změna ve stylu stránek se provádí změnou v tomto souboru.
- **vyvojovy_kit.php** – je soubor obsahující stručný popis vývojového kitu. Popsány jsou základní vlastnosti, periferie, napojení periferií na porty, registry.

3.2 Formátování

Veškeré formátování je nadefinováno v externím souboru *styl_1.css*. Definuje styl těla dokumentu, konkrétně barvu pozadí a barvu textu, záhlaví stránek, nadpisy a odkazy. Pokud bychom chtěli něco ve stylu stránek změnit, udělali bychom to předefinováním prvků v tomto souboru.

3.2.1 Pozadí a text

Celkové prostředí www stránek tvoří bílé pozadí a černá barva textu. Font písma je Arial.

```
body {
      background-color: White; /* Bílá barva pozadí */
      color:Black;             /* Černá barva písma */
      font-family: Arial;      /* Font písma Arial  */
     }
```

3.2.2 Záhlaví

Každá stránka má svoje záhlaví oranžové barvy s odpovídajícím nadpisem. Hlavní stránka, stránka Odkazy, stránka Ovladače a stránka Testy mají v oranžovém záhlaví zobrazené logo fakulty Aplikované informatiky. Dále pak už jednotlivé lekce a testy mají v záhlaví nadpis odpovídající probírané látce.

3.2.3 Nadpisy

Látka jednotlivých lekcí je oddělována nadpisy druhé úrovně (h2), jejichž styl je definován:

```
h2 {
    background-color: DimGray; /* Šedé pozadí       */
    color:white;               /* Bílá barva písma  */
    text-decoration:none;      /* Text bez dekorace */
    text-align:center;         /* centrovaný        */
   }
```

Obr. 5 Ukázkový nadpis používaný ve www stránkách

3.2.4 Podnadpisy

Podnadpisy jsou realizovány nadpisy třetí úrovně (h3) v podobě centrovaného tučně zvýrazněného textu bez dekorace.

3.2.5 Odkazy

Odkazy na hlavní stránce, stránce Odkazy a na stránce Ovladače jsou realizovány formou tlačítek s měnící se barvou pozadí. Tlačítka jsou slabě rámována černou barvou. Barva pozadí neaktivního tlačítka je šedá. Po najetí kurzorem myši na takovéto tlačítko se barva jeho pozadí mění na oranžovou.

Obr. 6 Tlačítko

Definice stylu odkazu pro docílení tlačítkové podoby je následující:

```
a.l          {                          /* ODKAZ TŘÍDY „l“       */
              text-decoration: none;     /* Text bez dekorace     */
              border: 1px solid black;   /* Černě rámovaný        */
             }
a.l:link     {
              color: White;              /* Bílá barva textu      */
              background-color: DimGray; /* Šedá barva pozadí     */
             }
a.l:visited  {                          /* NAVŠTÍVENÝ ODKAZ      */
              color: White;              /* Bílá barva textu      */
              background-color: DimGray; /* Šedá barva pozadí     */
             }
a.l:active   {                          /* AKTIVNÍ ODKAZ         */
              text-decoration: none;     /* Text bez dekorace     */
              color: White;              /* Bílá barva textu      */
              background-color: Orange;  /* Oranžová barva pozadí */
             }
a.l:hover    {
              text-decoration: none;     /* Text bez dekorace     */
              color: White;              /* Bílá barva textu      */
              background-color: Orange;  /* Oranžová barva pozadí */
             }
```

Všechny ostatní odkazy jsou ve formě podtrženého textu černé barvy.

3.3 Struktura lekce

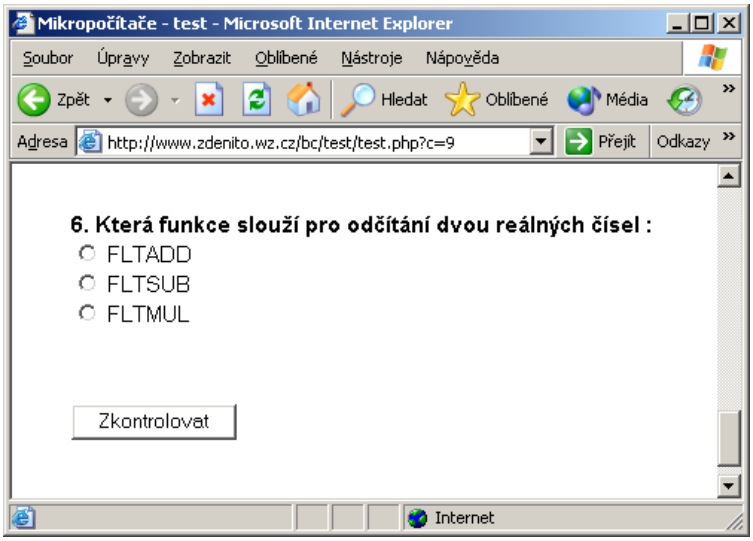
Lekce jsou strukturovány podle jedné „šablony“, kterou bychom mohli názorně popsat následujícími body:

- Úvod do probírané látky

- Objasnění základních pojmů
- Vysvětlení látky
- Úkol k vytvoření programu
- Ukázkové řešení úkolu včetně vývojového diagramu a zdrojového kódu programu
- Zadání samostatné práce
- Závěr (shrnutí probrané látky) s odkazem na test a na další lekci

3.4 Test

Test je realizován pomocí PHP skriptu. Počet testovacích otázek se pohybuje kolem šesti. Jedná se o krátké otázky nepříliš složitého charakteru, jejichž hlavním účelem je upevňovat právě nabyté vědomosti. Je kontrolována znalost nejdůležitějších pojmů a poznatků dané lekce. Testovací otázky mají vždy pouze jednu správnou odpověď. Na výběr je ze tří, někdy jen ze dvou, odpovědí.



Obr. 7 Ukázková www stránka s částí testu

3.4.1 Inicializace proměnných

Odpovědi na jednotlivé otázky jsou odesílány formulářem. Formulář otázek je typu „radio“, což znamená, že uživatel má možnost označit vždy jednu odpověď.

```
<FORM action="test.php" METHOD=GET>
   <input type="radio"  name="o1" value="1">
   <input type="radio"  name="o1" value="2">
</FORM>
```

Ze zdrojového kódu plyne, že pokud uživatel označí za správnou první nabízenou odpověď bude proměnné „o1“ přiřazeno číslo 1. Volba odpovědi je metodou „GET“ poslána zpět do téhož skriptu, kde se s ní následně pracuje – je vyhodnocována.

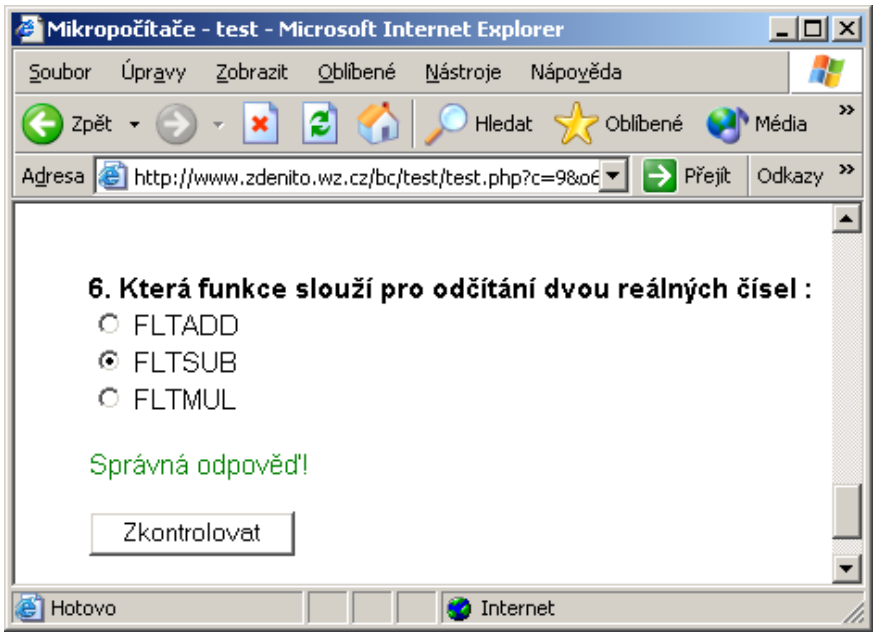
```
$o1=$_GET['o1']; //odpověď na první otázku je přiřazena do proměnné „o1“
```

3.4.2 Kontrola odpovědi

Zda je číslo odpovědi správné je kontrolováno podmínkou „if“.

```
if ($o1=="2") {
                echo'<font color="green">Správná odpověď!</font>';
               }
else if($o1==true) {
                    echo'<font color="red">Špatná odpověď!</font>';
                   }
```

Tento ukázkový příklad nám říká, že pokud je číslo první odpovědi (o1) rovno číslu 2, tzn. že jako správná odpověď byla ve formuláři zvolena druhá nabízená odpověď, je odpověď správná a pod příslušnou otázku s označenou odpovědí se vypíše zelenou barvou text: „Správná odpověď!“. Pokud byla zvolena jiná odpověď, proměnná obsahuje jiné číslo, a pokud proměnná existuje (je „pravda“), je vypsán červenou barvou text: „Špatná odpověď!“. Kontrola zda proměnná existuje je použita z důvodu, aby se pod otázkou nevypisoval žádný text v případě, že by ještě nebyly odeslány odpovědi, neboť vyhodnocování odpovědí, jak už bylo zmíněno probíhá v tomtéž skriptu, ze kterého jsou odpovědi posílány.



Obr. 8 Ukázková www stránka s částí testu po vyhodnocení

4 REALIZACE TISKU JEDNOTLIVÝCH LEKCÍ

Kvůli odlišné interpretaci zobrazení stránek různých prohlížečů a následně odlišného formátu při tisku, nebylo možné optimalizovat tisk pro více prohlížečů. Zvolil jsem proto optimalizaci pro prohlížeč Internet Explorer.

Důvodem volby tohoto prohlížeče, byl fakt, že je nainstalován na každém počítači s operačním systémem Windows, a proto ho bude moci využívat drtivá většina uživatelů.

Jednotlivé lekce jsou formátovány tak, aby byly ve vhodné formě pro tisk na papír formátu A4. To obnášelo především to, aby se použité obrázky tiskly v celé své velikosti, a aby text byl co nejvíce soudržný.

4.1 Soudržnost textu

Tisknuté lekce musí mít potřebnou úroveň a musí být přehledné, aby se v nich dalo snadněji a pohodlněji orientovat. Například není vhodné, aby nadpis byl na jedné stránce a text až na stránce nadcházející. Proto bylo potřeba, v místech, kde se tohle dělo, upravovat daný text a to buď přidáváním či ubíráním volných řádků.

4.2 Formát obrázků

Největším problémem byla velikost obrázků. Obrázky velkých rozměrů se netiskly v celé své velikosti, ale byly ořezané zprava. Proto bylo potřeba používat takovou velikost obrázků, aby se vešly na formát papíru A4.

Průměrná maximální šířka použitých obrázků se pohybuje okolo 650 pixelů.

ZÁVĚR

Ve své bakalářské práci jsem se zaměřil na popis pojmu eLearning a pokusil jsem se popsat jeho výhody a nevýhody. Dále jsem popsal současný stav tvorby www stránek. Nastínil jsem použití některých webových technologií jako jsou URL, HTTP, HTML, CSS a PHP.

V praktické části jsem popsal adresářovou strukturu mnou vytvořených www stránek, jejich formátování, strukturu jednotlivých lekcí a princip na kterém je založeno vyhodnocování testů pomocí PHP skriptu. A konečně jsem popsal realizaci tisku jednotlivých lekcí.

Stránky by měly být přínosem pro výuku předmětu mikropočítače. Hned v úvodní stránce je celkový náhled na probíranou látku během celého kurzu cvičení v podobě rozpisu lekcí. Lekce obsahují názorné obrázky pro lepší pochopení probírané látky. Vítaný je jistě animovaný obrázek principu fungování zásobníku. Studenti jistě ocení možnost dohledat si potřebné informace přímo při hodinách cvičení, například při samostatné práci, při tvorbě vlastního programu. Vyzkoušením si testu minulé lekce si můžou zopakovat učivo minulé látky a jít připraveni do další hodiny nebo se tím připravovat na případné písemné práce.

ZÁVĚR V ANGLIČTINĚ

In my bachelors thesis I tried to describe the conception of eLearning and its advantages and disadvantages. Thereinafter I described recent stadium of web sites production. Than I outlined using of some web technologies as URL, HTTP, HTML, CSS and PHP. In the practical part I described the directory tree of the web sites I made, its formating, the structure of each lesson and the principle on which is the evaluation of the test using the PHP script based. And at last I described the realization of each lesson printing.

The web sites should be benefit for education in the subject microcomputers. Right in the introduction page is the whole preview to the subject during the education course in the form of lesson specification. The lessons contain some objective pictures for better understanding of the subject. Welcomed is also the animated picture of function holder princip. The students will also appreciate the possibility to find out the needed informations right during the practical lesson, e.g. during autonomon working, or in production of own program. By testing out the last lesson test they can recapitulate the last lesson theme and get ready for next lesson.

SEZNAM POUŽITÉ LITERATURY

[1] Wikipedia – Internetová encyklopedie [online]. [cit. 5.22.2007]. Dostupný z WWW: http://wikipedia.org

[2] Jak psát web – O tvorbě internetových stránek [online]. [cit. 5.22.2007]. Dostupný z WWW: http://www.jakpsatweb.cz/

[3] Pavel Zídek: Mixování tradičního přístupu s novými technikami pro zvýšení efektivity v e-Learning [online]. [cit. 5.22.2007]. Dostupný z WWW: http://www.e-learn.cz/soubory/blendingapproaches.pdf

[4] Tvorba webu – Tvorba www stránek [online]. [cit. 5.22.2007]. Dostupný z WWW: http://www.tvorba-webu.cz/

[5] Historie PHP a souvisejících projektů [online]. [cit. 5.22.2007]. Dostupný z WWW: http://www.php.net/manual/cs/history.php

[6] Kosek Jiří : Téměř vše o www [online]. c1996-2006 , 2006/12/13 [cit. 5.22.2007]. Dostupný z WWW: http://www.kosek.cz/

[7] HCSO8 Family Reference Manual, Rev.1. Freescale Semiconductor, 2003

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

LMS	**L**earning **M**anagement **S**ystem
US	**U**nited **S**tates
WWW	**W**orld **W**ide **W**eb („celosvětová pavučina“)
URL	**U**niform **R**esource **L**ocator („jednotné označování objektů“)
HTTP	**H**yper **T**ext **T**ransfer **P**rotocol
SMTP	**S**imple **M**ail **T**ransfer **P**rotocol
MIME	**M**ultipurpose **I**nternet **M**ail **E**xtensions
FTP	**F**ile **T**ransfer **P**rotocol
XML	e**X**tensible **M**arkup **L**anguage („rozšířitelný značkovací jazyk“)
HTML	**H**yper **T**ext **M**arkup **L**anguage („značkovací jazyk pro hypertext“)
CGI	**C**ommon **G**ateway **I**nterface
ASP	**A**ctive **S**erver **P**ages
PHP/FI	**P**ersonal **H**ome **P**age / **F**orms **I**nterpreter
PHP	**P**HP **H**ypertext **P**reprocesor („PHP Hypertextový Preprocesor“)
CSS	**C**ascading **S**tyle **S**heets („kaskádové styly“)
MySQL	**My S**tructured **Q**uery **L**anguage
POP3	**P**ost **O**ffice **P**rotocol version **3**
DNS	**D**omain **N**ame **S**ystem
IP	**I**nternet **P**rotokol
TCP	**T**ransmission **C**ontrol **P**rotocol
PNG	**P**ortable **N**etwork **G**raphics

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I:

Příklad vytisknuté lekce, testu a testu s vyhodnocením

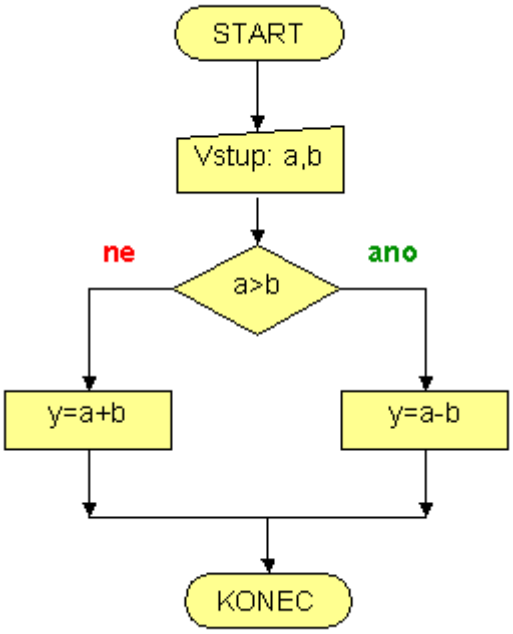
Úvod - základní pojmy

Mikroprocesor (CPU - central procesor unit) je základní částí **mikropočítače**, nazývanou "srdcem" nebo taky "mozkem" mikropočítače, která čte z paměti instrukce a na jejich základě vykonává program. Protože procesor, který by vykonával program zapsaný v nějakém vyšším programovacím jazyku by byl příliš složitý, má každý procesor svůj vlastní jazyk - tzv. strojový kód, který se podle typu procesoru skládá z jednodušších nebo složitějších instrukcí.

Součásti mikroprocesoru jsou:

- **řadič** nebo **řídicí jednotka**, která řídí tok programu, tj. načítání instrukcí, jejich dekódování, načítání operandů instrukcí z operační paměti a ukládání výsledků zpracování instrukcí
- **registry** - velmi malé, ale velmi rychlé paměti, které slouží k uchování operandů a mezivýsledků
- **aritmeticko logická jednotka (ALU)**, která provádí s daty aritmetické a logické operace

Procesor: logický automat pro zpracování instrukcí (centrální jednotka počítače / mikropočítače).
Instrukce: specifikace jednoduché akce, kterou má a umí provést procesor.
Program: posloupnost instrukcí, které říkají procesoru co má dělat.
Programování: zápis programu (v nějakém programovacím jazyku).
Algoritmus: je přesný návod či postup (kuchařka), kterým lze vyřešit daný typ úlohy.
Vývojový diagram: grafické znázornění algoritmu.



Tímto ukázkovým vývojovým diagramem říkám mikropočítači:

- Po **START**u programu načti dvě proměnné **a** a **b**. Potom vyhodnoť podmínku, zda je proměnná **a** větší jak **b**.
- Pokud **ano**, tak proměnné **y** přiřaď výsledek odčítání **a** - **b**.
- Pokud **ne**, tak proměnné **y** přiřaď výsledek součtu čísel **a** + **b**.
- Po provedení výpočtu program **ukonči**.

Zde je ukázka některých značek pro tvorbu vývojových diagramů.

Assembler - jazyk symbolických adres

To, co chceme, aby mikropočítač vykonával, mu říkáme programem, který pro něj napíšeme a následně nahrajeme do paměti.
My budeme pro zápis programu používat programovací jazyk, který se nazývá **assembler (jazyk symbolických adres)**.

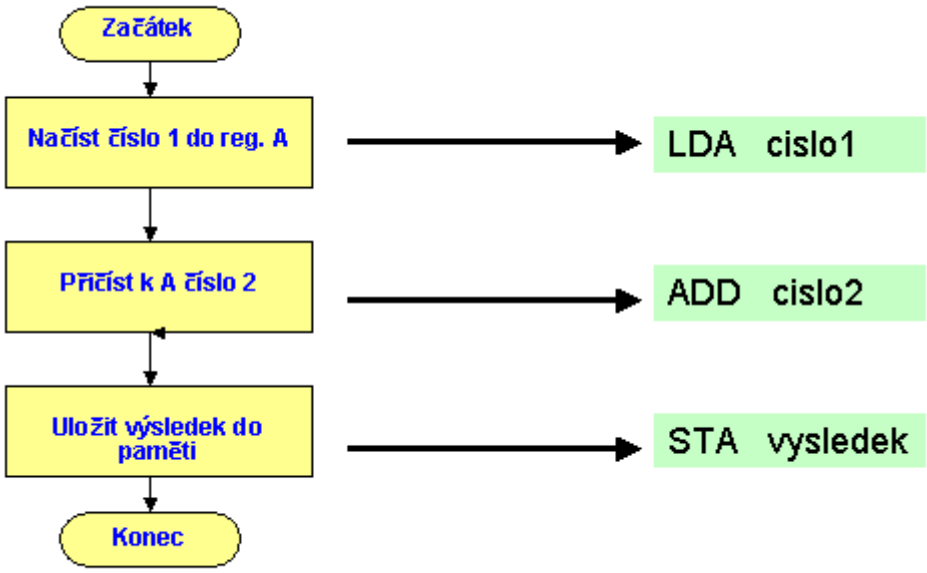
Programovací jazyk:	prostředek pro zápis algoritmů, které mohou být provedeny na (mikro) počítači.
Assembler:	programovací jazyk blízký strojovému kódu. Taky se nazývá jazyk symbolických adres. Název assembler se používá i pro překladač, který tento jazyk překládá do strojového kódu.
Strojový kód:	je posloupnost instrukcí procesoru vyjádřená čísly.

Struktura programového řádku v jazyce symbolických adres:

```
NÁVĚŠTÍ  INSTRUKCE  OPERAND  KOMENTÁŘ
VYPOCET  LDA        #4       ; načti do registru A hodnotu 4
         ADD        CISLO2   ; pričti k registru A obsah paměťové buňky CISLO2
         RTS
                             ; tuhle strukturu je nutné dodržovat! INSTRUKCE musí být
                             odsazena od NÁVĚŠTÍ a OPERAND od instrukce.
; Komentář může být vložen kdekoliv
```

1.Program : Součet čísel

Náš první program bude velmi jednoduchý. Jeho úkolem bude sčítat dvě čísla a ukládat výsledek. Pro proměnné čísla ***(cislo1, cislo2)*** i výsledek ***(vysledek)*** rezervujeme místo v paměti mikropočítače. Pro sčítání použijeme instrukci ***ADD***. Z grafického znázornění funkce této instrukce plyne, že musíme proměnnou ***cislo1*** nahrát do registru (A) sečíst s proměnnou ***cislo2*** a výsledek, který bude uložen v registru (A) uložit do proměnné ***vysledek***.



Popis instrukcí, které použijeme

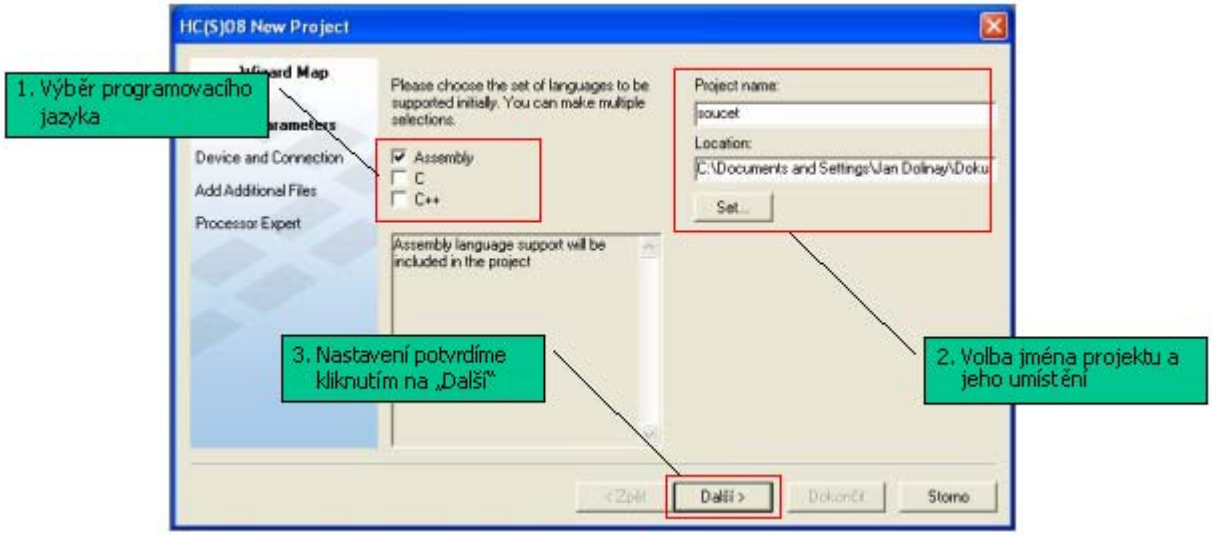
Grafické znázornění	Instrukce	Popis
A <- (M)	**LDA**	- nahraje číslo uložené v paměti (M) do registru (A)
A <- A + (M)	**ADD**	- sečte obsah registru (A) s obsahem paměťové buňky (M) a výsledek uloží do registru (A)
M <- (A)	**STA**	- uloží obsah registru (A) do paměti (M)
M cíl <- M zdroj	**MOV**	- přesun

Nyní, když máme sestavený algoritmus pro řešení daného úkolu, zapíšeme kód programu. Naším programovací prostředím bude **Freescale CodeWarrior**, které si můžete stáhnout zde.

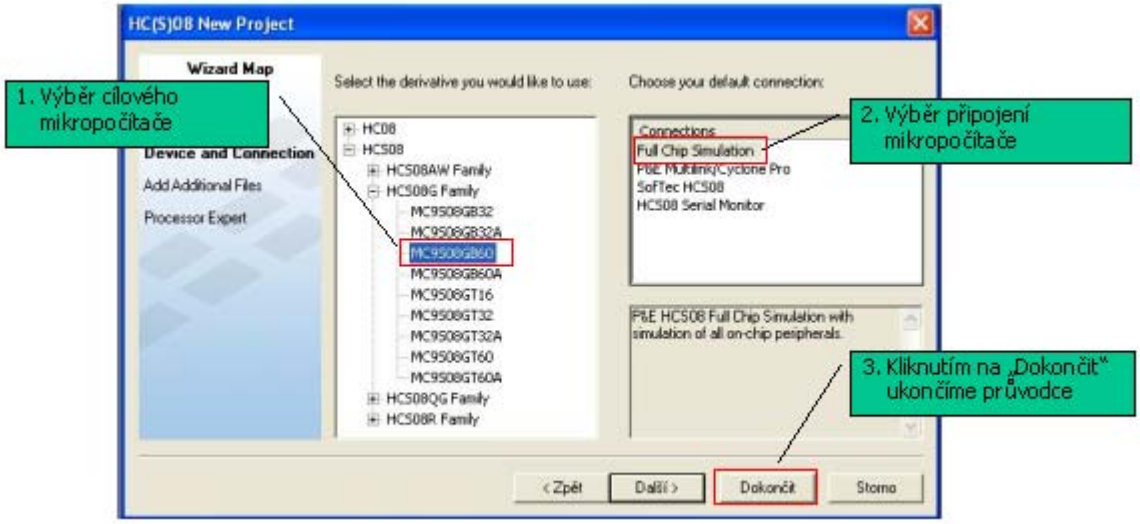
Freescale CodeWarrior

VYTVOŘENÍ NOVÉHO PROJEKTU

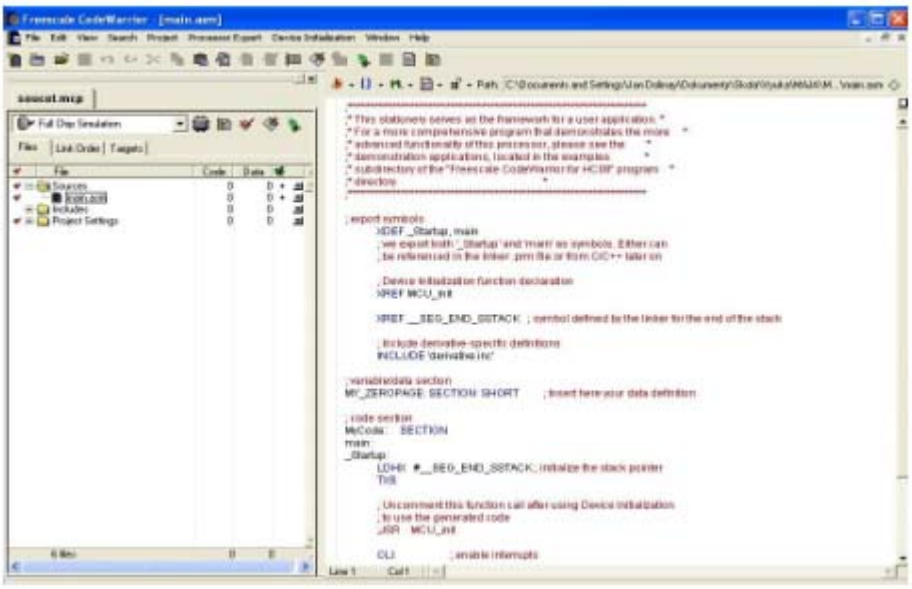
- Po spuštění CodeWarrioru zvolíme **Create new project** (vytvoření nového projektu).
- Zvolíme programovací jazyk (v našem případě zaškrtneme políčko *Assembly*) a zadáme název a umístění vytvářeného projektu.



- Dále vybereme příslušný typ mikropočítače (MC9508**GB60**) a připojení (Full Chip Simulation):

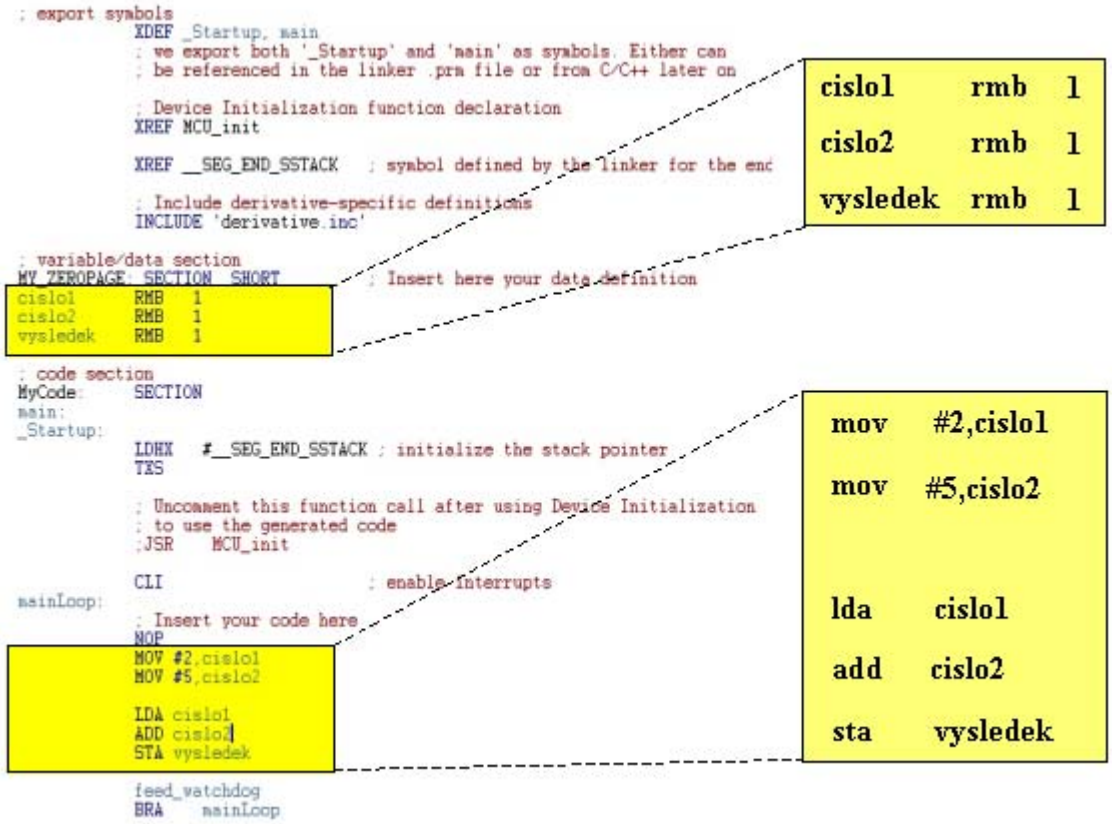


- Po kliknutí na tlačítko **Dokončit** nám CodeWarrior automaticky vygeneruje kostru programu (soubor **main.asm**):

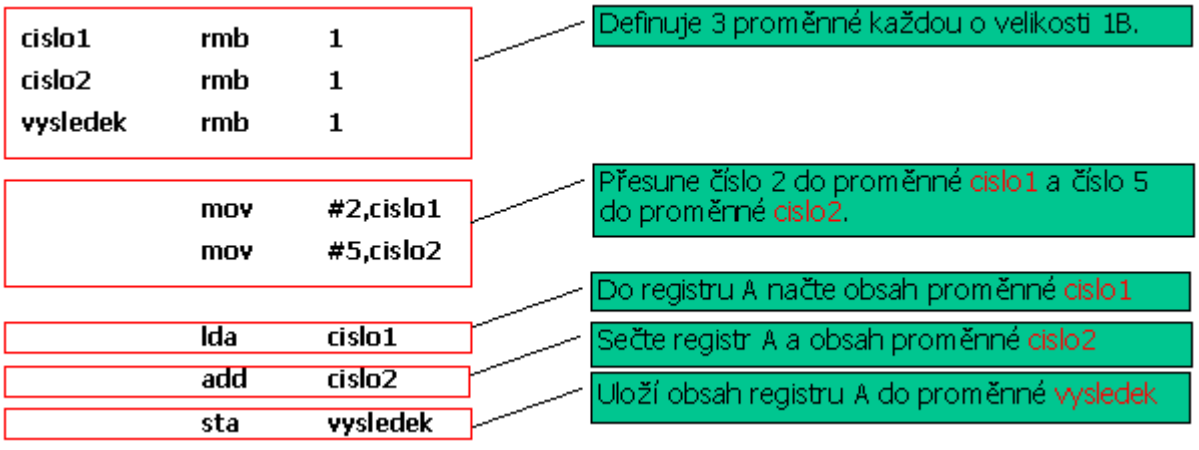


ZÁKONITOSTI PŘI PSANÍ PROGRAMU

- Definice proměnných (Rezervování místa v paměti)
  - proměnné definujeme v části **MY_ZEROPAGE:**, čemuž napovídá komentář tohoto řádku **; Insert here your data definition**
- Kód - program
  - píšeme v části pod návěštím **mainLoop:** s komentářem **; Insert your code here**



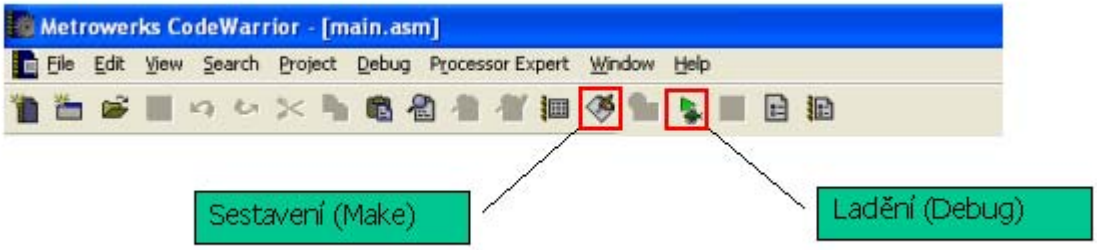
ROZBOR KÓDU PROGRAMU



Direktiva **rmb** slouží k rezervování místa v paměti (definice proměnných).
Jméno RMB [počet byte]

PŘEKLAD A SPUŠTĚNÍ KÓDU PROGRAMU

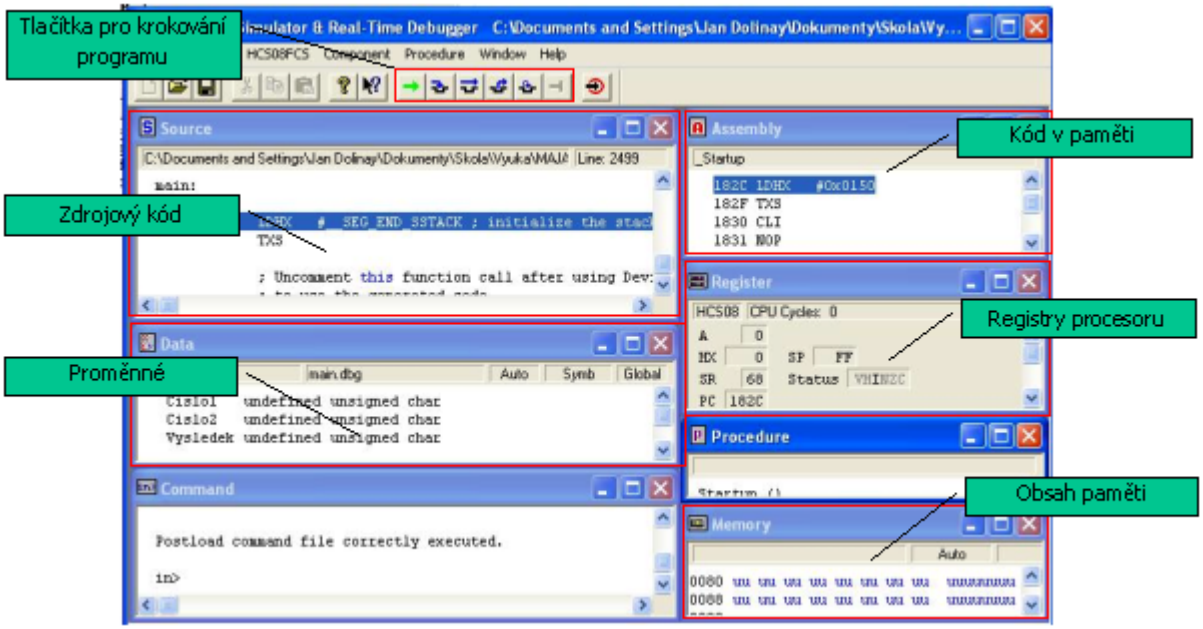
Teď, když máme program napsaný, nám nezbývá než jej **přeložit** (MAKE) a následně **spustit** (DEBUG).



Kompilace: překlad algoritmu zapsaného v programovacím jazyce do strojového kódu.

Spuštění a ladění: zavedení programu do cílového mikropočítače (nebo simulátoru - náš případ) a jeho následné spuštění

BĚH PROGRAMU



Pokud budeme program krokovat po jednom kroku ***(SINGLE STEP)***, můžeme:

- V okně ***Data*** sledovat, jak se jednotlivým proměnným přiřazují hodnoty.
- V okně ***Register*** můžeme vidět, jak se po provedení isntrukce LDA do registru A nahraje hodnota proměnné cislo1 a po provedení instrukce ADD výsledek sčítání.
- v okně ***Memory*** zase můžeme sledovat, na které adrese paměti se ukládá.

Zavřením okna simulátoru se vrátíme do prostředí CodeWarrior.

Závěr

Tímto jsme zdárně dokončili **1. lekci**, ve které jsme si objasnili některé základní pojmy a naučili jsme se vytvoření nového projektu v programovém prostředí **CodeWarrior**. Vypracovali jsme první úkol, pro který jsme sestavili **algoritmus** pro jeho řešení a ten zakreslili **vývojovým diagramem**. Napsali jsme si svůj první program v **jazyku symbolických adres (Assembleru)**, který jsme sestavili a spustili. V okně simulace jsme pak sledovali, co se děje v jednotlivých krocích provádění programu mikropočítačem.

Zkuste test z 1. lekce nebo jděte na 2. lekci

1. Základy programování mikropočítačů

◉ Test 1. lekce

1. Program je :
- ○ grafické znázornění algoritmu.
- ○ posloupnost instrukcí, které říkají procesoru co má dělat.
- ○ direktiva, která slouží k rezervování místa v paměti mikropočítače.

2. Algoritmus je :
- ○ přesný návod či postup (kuchařka), kterým lze vyřešit daný typ úlohy.
- ○ velmi malá, ale velmi rychlá paměť, která slouží k uchování operandů a mezivýsledků
- ○ direktiva, která slouží k rezervování místa v paměti mikropočítače.

3. Assembler je :
- ○ programovací jazyk blízký strojovému kódu, též se nazývá jazyk symbolických adres.
- ○ velmi malá, ale velmi rychlá paměť, která slouží k uchování operandů a mezivýsledků
- ○ direktiva, která slouží k rezervování místa v paměti mikropočítače.

4. Která instrukce nahraje číslo uložené v paměti (M) do registru (A) ?
- ○ STA
- ○ ADD
- ○ LDA

5. Která instrukce sečte obsah registru (A) s obsahem paměťové buňky (M) a výsledek uloží do registru (A) ?
- ○ DIV
- ○ ADD
- ○ LDA

Zkontrolovat

1. Základy programování mikropočítačů

◉ Test 1. lekce

1. Program je :
○ grafické znázornění algoritmu.
◉ posloupnost instrukcí, které říkají procesoru co má dělat.
○ direktiva, která slouží k rezervování místa v paměti mikropočítače.

Správná odpověď!

2. Algoritmus je :
◉ přesný návod či postup (kuchařka), kterým lze vyřešit daný typ úlohy.
○ velmi malá, ale velmi rychlá paměť, která slouží k uchování operandů a mezivýsledků
○ direktiva, která slouží k rezervování místa v paměti mikropočítače.

Správná odpověď!

3. Assembler je :
○ programovací jazyk blízký strojovému kódu, též se nazývá jazyk symbolických adres.
○ velmi malá, ale velmi rychlá paměť, která slouží k uchování operandů a mezivýsledků
◉ direktiva, která slouží k rezervování místa v paměti mikropočítače.

Špatná odpověď!

4. Která instrukce nahraje číslo uložené v paměti (M) do registru (A) ?
○ STA
○ ADD
◉ LDA

Správná odpověď!

5. Která instrukce sečte obsah registru (A) s obsahem paměťové buňky (M) a výsledek uloží do registru (A) ?
○ DIV
◉ ADD
○ LDA

Správná odpověď!

Zkontrolovat