VYUŽITIE FOSS PRI PROGRAMOVANÍ MIKROKONTROLÉROV ATMEL

LAJČIAK, Pavol, (SK)

Abstrakt. Článok opisuje použitie otvoreného softvéru pri tvorbe aplikácií pre 8-bitové RISC mikrokontroléry Atmel AVR. Opisuje postup tvorby aplikácií. A to inštaláciu a konfiguráciu potrebných softvérových nástrojov, vytvorenie programu v jazyku C, kompiláciu a napálenie do čipu. Tiež popisuje programátory vhodné na napálenie hotovej aplikácie do mikrokontroléra. Kľúčové slová: mikrokontrolér, Atmel, programátor, jazyk C

Úvod

Algoritmy sú všade okolo nás. Algoritmy používajú rastliny, zvieratá, ľudia, ale aj neživá príroda. Sú do nich vložené samotným Tvorcom. Neživá príroda, rastliny i ľudia sa vyvíjajú podľa určitých algoritmov bez toho, aby si to uvedomovali.

Okrem týchto algoritmov v prírode, existujú algoritmy, ktorými môžeme popísať ľudské konanie a správanie. Existuje napríklad algoritmus na uvarenie čaju, uvarenie chutného obeda atď.

Tiež sa stretávame s algoritmami, ktoré sú zakomponované do strojov a prístrojov, s ktorými sa stretávame v každodennom živote. Sú napríklad súčasťou práčiek, chladničiek, nápojových automatov, riadiaceho systému kúrenia a klimatizácie, a mnohých ďalších elektronických pomocníkov.

Človek sa od nepamäti snažil zjednodušiť a zefektívniť isté činnosti a tak si uľahčiť isté činnosti. A tak vynaliezal rôzne stroje a prístroje, ktoré najskôr sám ovládal. Neskôr tieto stroje automatizoval, aby pracovali sami bez ľudskej obsluhy.

Zo systémového pohľadu sa na stroj môžeme pozrieť ako na spojenie riadiacej jednotky, výkonových členov, senzorickej a akčnej časti. Takúto koncepciu môžeme v podstate nájsť vo všetkých strojoch a prístrojoch.

V minulosti riadiaca časť mohla byť realizovaná mechanickým spôsobom. Príkladom môže byť napríklad hracia skrinka, ktorá ako riadiacu jednotku používala otočný valček, na

ktorom bol systém výčnelkov a pomocou týchto výčnelkov bola naprogramovaná melódia. Zmena melódie bola drahá a zdĺhavá.

Dnes ekvivalent takejto hracej skrinky môžeme nájsť v rôznych kníhkupectvách vo forme hracích pohľadníc.

S nástupom éry počítačov a mikrokontrolérov, sa konštrukcia automatizovaných a riadiacich systémov značne zjednodušila.

Aby takýto systém pracoval, musí mu tvorca, človek vdýchnuť algoritmus, podľa ktorého sa tento systém správa. Pomocou mikrokontroléra, v ktorom je uložený program, v spolupráci s výkonovými členmi, aktormi a senzormi môžeme vytvoriť akýkoľvek stroj alebo prístroj.

1 Vývojové nástroje

Existuje veľa vývojových nástrojov pre radu RISC mikrokontrolérov Atmel AVR pre operačný systém Windows. Samotná firma Atmel dala k dispozícii integrované vývojové prostredie AVR Studio.

AVR Studio spolupracuje s FOSS balíkom WinAVR, ktorý obsahuje nástroje potrebné na vývoj aplikácií platforme Atmel AVR. Obsahuje kompilátor avr-gcc, debuger avr-gdb a programátor avrdude a ďalšie.

Čo sa týka OS GNU/Linux, existuje veľa aplikácií pre vývoj aplikácii na platforme Atmel AVR. V repozitároch distribúcie Ubuntu 10.04 sú obsiahnuté tieto balíčky: ava, avarice, avr-libc, avra, avrdude, avrdude-doc, avrp, avrprog, binutils-avr, gcc-avr, gdb-avr, simulavr, uisp, sdcc, sdcc-doc, sdcc-libraries, sdcc-ucsim, usbprog, usbprog-gui. Môžeme ich nainštalovať pomocou príkazu apt-get install ¡názov_balíčka¿ alebo pomocou správcu balíkov synaptic.

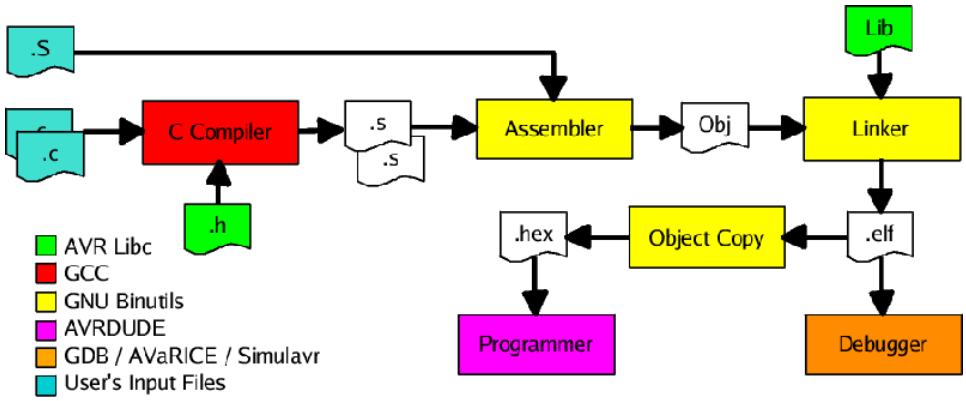
1.1 Asembléry

ava je pokročilý asembler a linker pre 8-bitovú rodinu mikrokontolérov Atmel AVR. Tak ako preprocesor jazyka C poskytuje prácu so segmentmi a virtuálnymi symbolmi. Syntax tohoto asembléra nie je kompatibilná so syntaxou asembléra od firmy Atmel.

avra je asembler pre 8-bitovú rodinu mikrokontolérov Atmel AVR. Z väčšej časti je kompatibilný so syntaxou asembléra od firmy Atmel, ale pridáva nové vlastnosti ako lepšiu podporu makier a prídavné direktívy preprocesora.

1.2 Reťazec vývojových nástrojov

Na obrázku 1 je ukázaný reťazec vývojových nástrojov, ktorý je potrebný pre tvobru aplikácií pre platformu 8-bitových RISC mikrokontolérov pre vývoj jazyku C.



Obrázok 1: Reťazec vývojových nástrojov [1]

1.3 Kompilátor

GCC je skratka pre GNU Compiler Collection. Ide o veľmi flexibilný kompilačný systém. Má rôzne front-endy pre rôzne jazyky. Obsahuje aj veľa back-endov, ktoré generujú kód asembléra pre veľa rôznych architektúr procesoror a hostiteľských operačných systémov. Všetky zdieľajú spoločný middle-end, obsahujúci generickú časť kompilátora, vrátane mnohých optimalizácií.

Z pohľadu GCC je hostiteľským systémom ten, na ktorom kompilátor beží. Cieľovým systém je ten, pre ktorý sá kód prekladá. A zostavovací systém je systém, na ktorom bol kompilátor zostavený zo zdrojového kódu. Ak kompilátor má spoločný hostiteľský a cieľový systém, hovoríme o natívnom kompilátore. Ak hostiteľský a cieľový systém sú rôzne, hovoríme o krížovom kompilátore (cross-compiler).

GCC je odlišný od ostatných kompilátorov. GCC sa zameriava na preklad vysokoúrovňových jazykov do asembléru cieľovej platformy. Balíček gcc-avr je kompilátor GNU C pre cieľovú platformu AVR. AVR GCC obsahuje tri kompilátory - kompilátor pre jazyk C, C++ a Ada. [1]

1.4 GNU AVR Binutils

Programy v balíčku binutils-avr sú používané na manipuláciu s binárnymi a objektovými súbormi, ktoré boli vytvorené pre architektúru Atmel AVR. Skratka GNU Binutils znamené „Binary Utilities“. Obsahuje assembler (gas), linker (ld) a veľa ďalších utilít, ktoré pracujú s binárnymi súbormi a sú vytvorené ako časť reťazca vývojárskeho softvéru.

Nástroje, ktoré boli vytvorené pre prácu s AVR začínajú s prefixom „avr-“. Napríklad, meno pre asembler, ktorý sa natívne volá „as“,(dokonca v dokumentácii GNU asembler je nazývaný ako „gas“). Ale, keď je skompilovaný pre prácu s cieľovou platformou AVR stane sa s neho „avr-as“.

Nasleduje zoznam programov, ktoré sú obsiahnuté v balíčku AVR Binutils:

- avr-as – asembler,
- avr-ld – linker,
- avr-ar – vytvára, modifikuje alebo extrahuje z knižníc (archívov),
- avr-ranlib – generuje index obsahu knižnice (archívu),
- avr-objcopy – kopíruje a prekladá objektové súbory do rôznych formátov,
- avr-objdump – zobrazuje informácie z objektových súborov vrátane disasemblovacích informácií,
- avr-size – vypíše veľkosti sekcii a celkovú veľkosť,
- avr-nm – vypíše symboly z objektových súborov,
- avr-strings – vypíše tlačiteľné reťazce zo súborov,
- avr-strip – Discard symboly zo súborov,
- avr-readelf – zobrazí obsah súborov vo formáte ELF,
- avr-addr21ine – konvertuje adresy do súboru a riadku,
- avr-c+-—-filt – Filter na odkódovanie (demangle) kódovaných symbolov C++.

1.5 Knižnica jazyka C

avr-libc je štandardná knižnica jazyka C, používaná na vývoj programov pre mikrokontoléry Atmel AVR, ktorá sa Ubuntu 10.04 LTS nachádza vo verzii 1.6.7. Balíček obsahuje statické knižnice ako aj potrebné hlavičkové súbory. Poskytuje podmnožinu štandardnej knižnice C pre 8-bitové RISC mikrokontoléry Atmel AVR a základný kód potrebný pre štart väčšiny aplikácií.

1.6 Debugovacie nástroje

Balíček gdb-avr bol kompilovaný pre cieľovú architektúru AVR. GDB je debuger na zdrojovej úrovni, schopný prerušiť program na ľubovolnom špecifikovanom riadku, zobraziť hodnoty premenných, a určiť chybu, pokiaľ nastala. V súčasnosti funguje pre jazyky C, C++, Fortran, Modula 2 a programy v jazyku Java. Je povinnou výbavou pre každého seriózneho programátora. Tento balíček je primárne určený pre AVR vývojárov.

Okrem debugera gdb-avr pod OS GNU/Linux môžeme používať balíček avarice, pokiaľ cielový mikrokontolér disponuje JTAG rozhraním a my máme programovací hardvér, pomocou ktorého môžeme tento mikrokontolér pripojiť. Program avarice prekladá príkazy medzi diaľkovým debugovacím protokolom GDB a protokolom AVR JTAG ICE. AVR JTAG sa používa na debugovanie procesora v reálnom aplikácii - doske. Pomocou neho samozrejme tiež môžete naprogramovať AVR mikrokontolér.

1.7 Simulátor

simulavr simuluje rodinu mikrokontrolérov Atmel AVR, emuluje vzdialený cieľ pre gdb a zobrazuje informácie o registroch a pamäti v reálnom čase.

1.8 Programátory

Balíček avrdude je programátor slúžiaci na čítanie/zápis/manipuláciu s obsahom ROM a EEPROM pamäti mikrokontroléra s použitím techník ISP. Balíček avrdude-doc obsahuje dokumentáciu ku konfigurácii a prevádzkovaniu balíčka avrdude.

Balíček avrp je programátor slúžiaci na programovanie FLASH/EEPROM pre 8-bitovú rodinu RISC procerorov Atmel AVR. Tiež umožňuje programovanie rady Atmel AT89. Podporuje minimálne štyri rôzne programovacie zariadenia, vrátane vlastnej vývojovej dosky s ISP programovaním od Atmelu.

Balíček avrprog môže programovať AVR mikrokontroléry a používa paralelný port ako programovacie zariadenie. Zariadenie je programované v ISP móde. Súčasťou balíka je schéma vyžadovaného hardvéru, ktorý je navrhnutý tak, aby bolo efektívny a lacný. `http://avrprog.sourceforge.net`

Programátor uisp je vyžadovaný na programovanie AVR mikrokontolérov objektovým kódom vytvoreným asemblérom/linkerom ava, gas alebo gcc. Podporuje programovanie ISP pomocou vývojovej/prototypovej dosky STK500 a mnohé ďalšie extrémne lacné programáry pripojitelné na paralelný port. Tiež može byť použitý na programovanie mikrokontolérov Atmel AT89S51 a AT89S52.

usbprog je programovací nástroj používaný na nahradenie firmvéru v programovacom zariadení USBprog. Dokáže automaticky získať zoznam dostupných firmvérov z internetu. Vybraný frrmvér dokáže stiahnuť a priamo nahrať do programovacieho zariadenia USBprog.

Môžete jednoducho nainštalovať rôzny firmvér priamo cez USB. Programovacie zariadenie dokáže programovať a debugovať AVR a ARM mikrokontoléry ako USB-RS232 prevodník, ako JTAG rozhranie alebo ako jednoduchý vstupno-výstupné rozhranie s 5 linkami. Je to program ovládaný z príkazového riadku. Balíček usbprog-gui poskytuje grafické rozhranie pre balíček usbprog.

2 Ďalšie nástroje

sdcc je kompilátor jazyka C pre rodiny mikrokontolérov Intel MCS51, AVR, HC08, PIC a Z80. Obsahuje kompilátory, asembléry and linkery. Balíček sdcc-doc poskytuje dokumentáciu a príklady a balíček sdcc-libraries obsahuje hlavné knižnice pre balíček sdcc.

sdcc-ucsim je simulátor mikrokontolérov. Obsahuje rozšíriteľnú podporu pre rôzne rodiny mikrokontolérov. V súčasnosti podporuje rodiny mikrokontolérov Intel MCS51, AVR, HC08, PIC a Z80.

Okrem týchto balíčkov existujú balíčky integrovaných vývojových prostredí KontrollerLab a cdkjavr. Podľa aktivity projektu sa zdá, že balíček cdkavr sa už dlhšiu dobu nevyvíja. O trocha lepšie je na tom balíček KontrollerLab, ktorý síce nie je v stabilnej verzii, no jeho posledná verzia je z roku 2008.

3 Programovanie mikrokontrolérov

Programovanie mikrokontolérov Atmel AVR môžeme prevádzať rôznymi spôsobmi. Prvým spôsobom je programovanie technikou ISP (In Circuit Serial Programing), ktorú podporujú všetky procesory z tejto rodiny. Táto technika umožňuje programovanie priamo v obvode, bez toho, aby sme museli mikrokontrolér vytiahnuť zo zariadenia. Programovanie sa vykonáva pomocou ISP konektora, ktorý môže byť 6 pinový alebo 10 pinový. Oba konektory obsahujú signály MOSI, MISO, RST, CLK, VCC, GND. Pri návrhu zariadenia, v ktorom mikrokontrolér chceme programovať pomocou ISP, je potrebné dbať na to, aby na pinoch, ktoré sa používajú na ISP programovanie, nebola žiadna nízkoimpedančná záťaž, alebo aby sa dala počas programovania odojiť.

Existuje aj druhý spôsob programovania pomocou rozhrania JTAG. Toto rozhranie poskytuje väčšie možnosti ako ISP programovanie. No jeho nevýhodou je, že rozhranie JTAG obsahujú viacpinové a drahšie mikrokontroléry. Preto sa týmto programovaním sa nebudeme naďalej zaoberať, nakoľko je to nad rámec tohoto článku.

4 Programátory – softvér

Na to, aby sme aplikáciu, ktorú pre cieľový mikrokontrolér vytvoríme, mohli napáliť do pamäti FLASH/EEPROM mikrokontroléra, potrebujeme programovací softvér a programovací hardvér. V literatúre sa obyčajne pre obe veci používa rovnaký pojem programátor.

Máme dve krajné možnosti vybrať si vhodný programovací softvér a zostrojiť k nemu programovací hardvér. Alebo vybrať si programovací hardvér a nájsť k nemu vhodný programovací softvér.

Moja situácia bola taká, že som chcel používať viacero programovacích hardvérov a tak som hľadal vhodný programovací softvér, ktorý by mi to umožňoval a je priebežne udr-

žiavaný. Mojim požiadavkám najviac vyhovoval balíček avrdude. Môžeme ho využívať z príkazového riadku.

Pokiaľ vám nevyhovuje práca v príkazovom riadku existujú k balíčku avrdude rôzne GUI (Graphics User Interface), ktoré prácu s ním zjednodušujú. Asi najlepším GUI pre OS GNU/Linux je AVR8 Burn-O-Mat, ktorý je napísaný v jazyku Java a tým pádom ide o multiplatformný nástroj. Na domácej stránke tohoto projektu môžete nájsť aj online nástroj na výpočet poistiek mikrokontoléra. Okrem neho existujú ďalšie GUI pre OS GNU/Linux ale nie sú také vyspelé, alebo nie sú aktuálne. Ide o balíčky avrdude-gui a gnome-avrdude. Pre operačný systém Windows existuje Khazama AVR Programmer, ktorého kvality sú porovnateľné s aplikáciou AVR8 Burn-O-Mat.

Okrem toho existujú ďalšie programovacie softvéry, ktoré môžu dobre poslúžiť pre programovanie mikrokontolérov v spojení so rôznym programovacím hardvérom. Ide o programy avrp, avrprog, uisp, usbprog, usbprog-gui.

5 Programátory – hardvér

Programátorov existuje veľké množstvo. Keď si pozrieme dokumetáciu k programovaciemu softvéru (avrdude, avrp, avrprog, uisp), zistíme, že softvér podporuje veľké množstvo proprietárnych i open source programátorov.

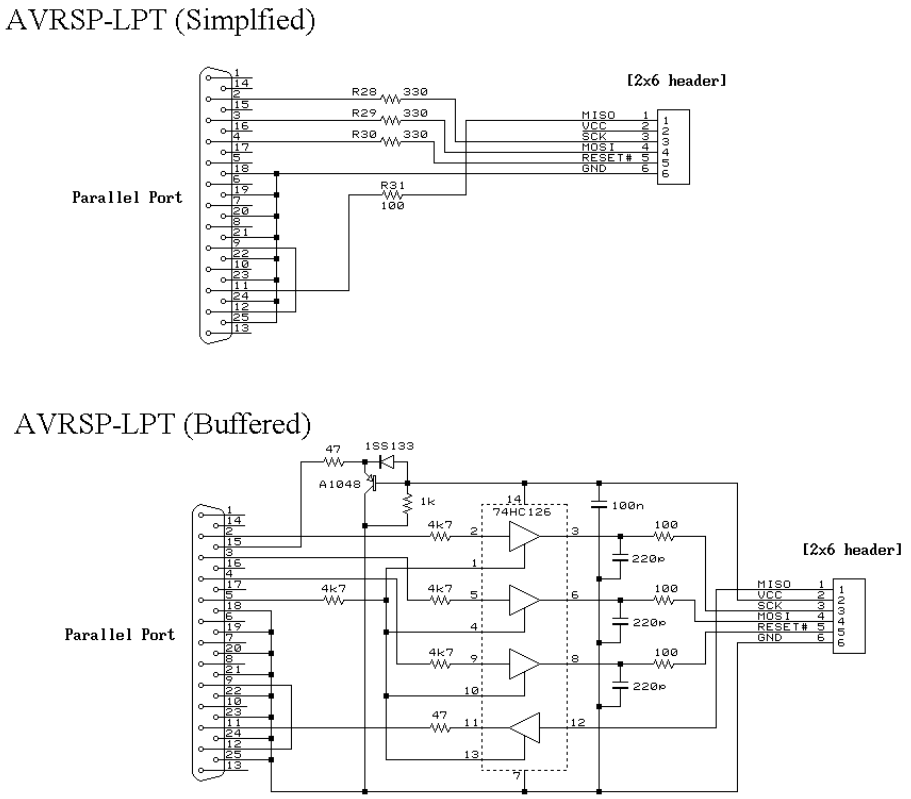
Existujú programátory pripojiteľné na sériový, paralelný a USB port. Jednoduché programátory obsahujú len niekoľko málo diskrétnych súčiastok. Najednodnoduchší na paralelný port obsahuje len 4 rezistory, 25 pinový LPT konektor a 6 alebo 10 pinový ISP konektor. Prípadne ISP konektor môžeme vynechať a vodiče/signály pripojiť priamo na príslušné piny mikrokontroléra.

Na nasledujúcom obrázku je ISP programátor na paralelný port, ktorého autorom je Elm-Chan. [3] Jeho výhodou je extrémne malá zložitosť. Programátor obsahuje len konektor na paralelný port a štyri rezistory a ISP konektor. ISP konektor môžeme vynechať, pokiaľ vodiče pripojíme priamo na vývody mikrokontoléra.

Jeho hlavnou nevýhodou je veľká citlivosť paralelného portu na preťaženie a skrat. Vyvarujte sa prípadnému skratu a pretaženiu, lebo paralelný port sa veľmi ľahko zničí. Napájanie je potrebné riešiť z iného zdroja. Ja osobne používam napájanie z USB portu.

V domovskom priečinku je potrebné vytvoriť súbor .avrduderc, ktorý je používateľským konfiguračným programom pre avrdude. Treba do neho zapísať nasledujúce riadky. Týmto sa pridá definícia programátora s názvom avrsplpt a nadefinuje priradenie pinov pre port LPT. Následne takto nadefinovaný programátor možeme používať pomocou avrdude.

```
programmer
  id    = "avrsplpt";
  desc  = "Elm-chan parallel programmer";
  type  = par;
  reset = 4;
```



Obrázok 2: Elm-Chan ISP LPT programátor

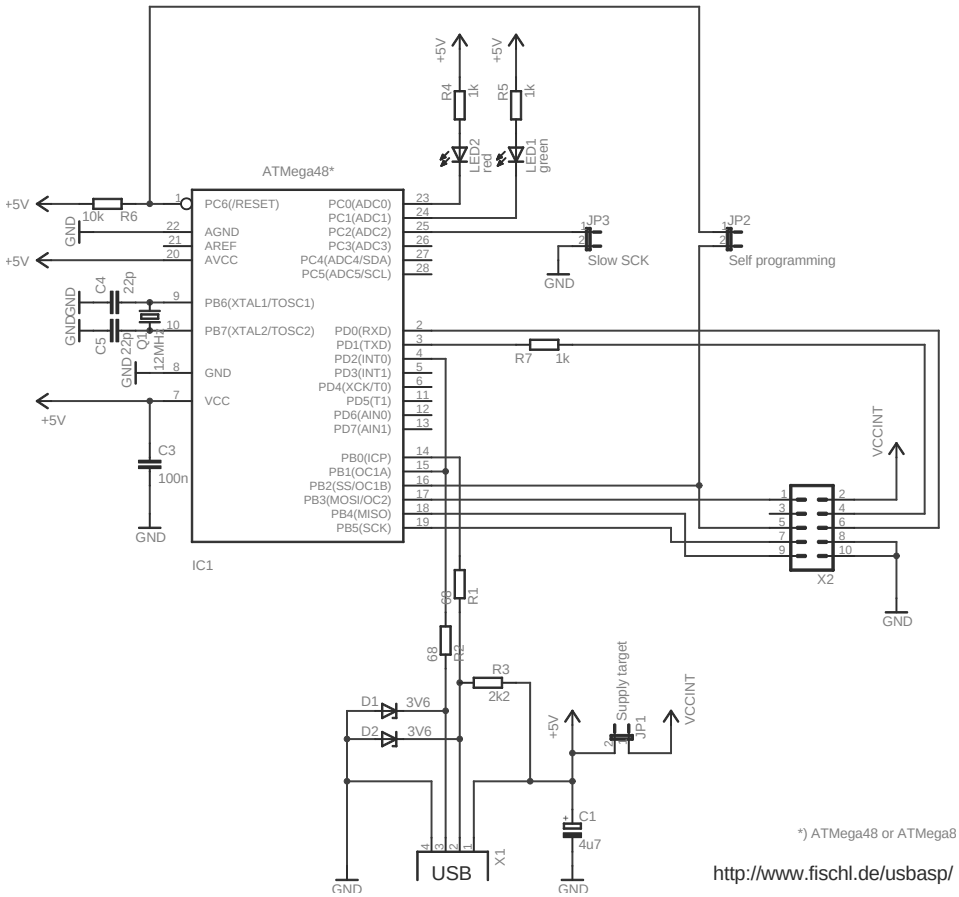
```
  sck   = 2;
  mosi  = 3;
  miso  = 11;
;
```

Programovanie mikrokontroléra pomocou takéhoto programátora na paralelnom porte vykonáme príkazom.

```
avrdude -p m16 -c avrsplpt -U flash:w:name.hex
```

Ďalšou možnosťou je použitie programátora USBasp. Ide o programátor pripájaný na USB port. Jeho srdcom je ATmega8. Ak si tento programátor chceme zostrojiť doma, potrebujeme naprogramovať ATmega8A. Na jeho programovanie môžeme použiť vyššie zmienený ISP programátor na paralelný port.

Ak vás zaujímajú podrobnosti, ako niektorý takýto programátor zostaviť, informácie nájdete na mojej internetovej stránke `http://pavolmaria.org` v sekcii elektronika. [4]



Obrázok 3: ISP programátor USBasp

6 Zapojenie ISP konektora a umiestnenie ISP pinov na mikrokontoléroch

V tab. 1 vidíme umiestnenie ISP pinov na jednotlivých vybraných mikrokontoléroch, v tab. 2 zapojenie 6 pinového a v tab. 3 zapojenie 10 pinového ISP konektora.

7 Vybrané mikrokontoléry

Čo sa týka výberu mikrokontroléra, zvažoval som rôzne faktory. Vybral som si tieto mikrokontoléry: pre malé projekty mikrokontolér ATtiny2313A, pre stredne veľké projekty mikrokontolér ATmega8A a pre veľké projekty mikrokontolér ATmegal6A. Samozrejme, že

Tabuľka 1: Umiestnenie ISP pinov na mikrokontoléroch

	ATtiny2313A	ATmega8A	ATmegal6A
RESET	1	1	9
SCK	19	19	8
MISO	18	18	7
MOSI	17	17	6
VCC	20	7	10
GND	10	8	11

Tabuľka 2: Zapojenie 6 pinového ISP konektora ATMEL

1	MISO	2	VCC
3	SCK	4	MOSI
5	RST	6	GND

Tabuľka 3: Zapojenie 10 pinového ISP konektora ATMEL

1	MOSI	2	VCC
3	LED	4	GND
5	RST	6	GND
7	SCK	8	GND
9	MISO	10	GND

je možné použiť aj ďalšie mikrokontoléry. No je potrebné, aby boli podporované jednotlivými aplikáciami, ktoré na vývoj používate.

Ak chcete váš vývoj uskutočňovať v jazyku C, je potrebné, aby podpora daného mikrokontoléra bola zahrnutá v knižnici avr-libe. Zoznam podporovaných mikrokontolérov nájdete v dokumentácii ku knižnici avr-libc.

Ak chcete váš vývoj uskutočňovať v asembléri je potrebné, aby podpora konkrétneho mikrokontroléra bola zahrnutá v balíčku s asemblérom avra, ava.

8 Vzorový príklad

Ak chceme začať vývoj pre 8-bitové mikrokontoléry Atmel AVR potrebujeme mať nainštalované príslušné nástroje. Potrebujeme mať programovací hardvér. Potrebujeme mať mikrokontolér, pre ktorý budeme vyvíjať aplikácie. Pre demonštračný príklad som vybral procesor ATmegal6A. Tento mikrokontolér som vybral preto, lebo má dostatočne veľa vý-

Tabuľka 4: Parametre vybraných mikrokontolérov

Mikrokontrolér	ATtiny2313A	ATmega8A	ATmegal6A
Flash (Kbytes)	2	8	16
EEPROM (Bytes)	128	512	512
SRAM (Bytes)	128	1024	1024
PDIP Pins	20	28	40
F.max (MHz)	20	16	16
Vcc(V)	1.8-5.5	2.7-5.5	2.7-5.5

vodov a periférii a tak sa dá v budúcnosti dá skúšať široká paleta najrôznejších úloh. Cena tohoto mikrokontoléra je približne 6 Eur.

Ak by sme chceli používať tento postup vo vyučovaní, treba zvážiť, či nie je vhodnejšie použitie mikrokontoléra ATmega8A, ktorého cena je približne 2 Eur

```
/* ledblink.c, an LED blinking program */
#include<avr/io.h>
#include<util/delay.h>

void sleep(uint8_t millisec)
    {while(millisec)
        {_delay_ms(1);/* 1 ms delay */
         millisec--;
        }
    }
main()
    {DDRC |=1<<PC2;  /* PC2 will now be the output pin */
        while(1)
            {PORTC &= ~(1<<PC2);/* PC2 LOW */
             sleep(100);/* 100 ms delay */

             PORTC |=(1<<PC2); /* PC2 HIGH */
             sleep(100);/* 100 ms delay */
            }
    }
```

Vzorový príklad je prevzaný z [5]
Preklad súboru zo zdrojového kódu do objektového kódu spravíme pomocou:

```
avr-gcc -mmcu=atmega16 Os ledblink.c o~ledblink.o
```

Vytvorenie .hex súboru urobíme príkazom

```
avr-objcopy -j .text -j .data -O ihex  ledblink.o  ledblink.hex
```

Na naprogramovanie mikrokontroléra Atmega16A pomocou USBasp a avrdude použijeme príkaz

```
avrdude -p m16 -c usbasp -e -u flash:w:ledblink.hex
```

avrdude môžeme použiť na čítanie/zápis/kontrolu poistiek mikrokontroléra, EEPROM a FLASH pamäti.

9 Využitie mikrokontrolérov a hotové projekty

Mikrokontroléry môžeme využiť na rozličné činnosti. Prvá úloha, ktorá býva pomocou mikrokontrolérov realizovaná je blikajúca LED. Potom to môžu byť rôzne svetelné efekty, ovládanie tlačítok a maticovej klávesnice, čítanie hodnot z A/Č prevodníka, riadenie PWM a mnohé ďalšie. Na internete nájdeme aj množstvo projektov realizovaných pomocou AVR mikrokontrolérov a to napríklad projek LCD2USB alebo aj USBasp, ktoré ako svoje jadro používajú Atmega8A mikrokontrolér.

Zaujímavé projekty nájdeme na stránke Objective Developemet `http://www.obdev.at/products/vusb/index.html`. Táto firma je autorom virtuálneho USB portu V-USB (firmvér), ktorý slúži na priame pripojenie USB bez iných obvodov.

Záver

Cieľom tohto článku bolo ukázať možnosti FOSS pri vývoji aplikácií pre cieľovú platformu Atmel AVR. Pri vývoji bolo ukázané použitie kompilátora gcc-avr, príkazu avr-copy z balíčka avr-binutils a programovacieho softvéru avrdude v spojení s programovacím hardvérom USBasp a Elm-Chan programátorom na paralélny port.

Tento článok bol napísaný až teraz, no myšlienky naň dozrievali asi päť rokov. Sú za nim skryté hodiny práce a to čítanie dokumetácie, zostavovanie programovacieho hardvéru a testovanie jednotlivých FOSS nástrojov, vhodných na vývoj pre platformou Atmel AVR tak, aby bol vývojový cyklus uzavretý.

Literatúra

[1] Documentation:AVR GCC/AVR GCC Tool Collection
`http://www.avrfreaks.net/wiki/index.php/Documentation:AVR_GCC/AVR_GCC_Tool_Collection`

[2] AVR Libc user manual
http://savannah.nongnu.org/download/avr-libc/avr-libc-user-manual-1.6.5.pdf.bz2

[3] Elm-Chan ISP LPT programmer
http://elm-chan.org/works/avrx/avrx_lpt.png

[4] USBASP - Sériový programátor procesorov Atmel AVR na USB port
http://www.pavolmaria.org/index.php?id=elektronika/avrspusbasp

[5] AVR Microcontrollers in Linux HOWTO
http://tldp.org/HOWTO/Avr-Microcontrollers-in-Linux-Howto/x207.html

Kontaktná adresa

Pavol LAJČIAK (Ing.),
Katedra informatiky PF KU v Ružomberku, Hrabovská cesta 1,
034 01 Ružomberok, pavol.lajciak@pavolmaria.org

Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
1.–4. júla 2010, Žilina, Slovensko

Organizátori: Miloš Šrámek, Spoločnosť pre otvorené informačné technológie
Tatiana Šrámková, Katedra fyziky, FEI STU Bratislava
Michal Kaukič, Aleš Kozubík, Tomáš Majer, Žilinská univerzita
Lýdia Gábrišová, Ľubica Michálková, Žilinská univerzita
Juraj Bednár, Digmia, Slovensko
Miloslav Ofúkaný, GeoCommunity, Slovensko
Peter Mráz, Kremnica
Slavko Fedorik, SOŠ elektrotechnická, Poprad
Peter Štrba, Spojená škola/Gymnázium M. Galandu, Turčianske Teplice
Ladislav Ševčovič, FEI, Technická univerzita v Košiciach

Editori: Michal Kaukič
Miloš Šrámek
Slavko Fedorik
Ladislav Ševčovič

Recenzenti: Mgr. Juraj Bednár
Mgr. Rudolf Blaško, PhD.
RNDr. Ján Buša, CSc.
Ing. Slavko Fedorik
Ing. Karol Grondžák, PhD.
Mgr. Michal Kaukič, CSc.
Ing. Tomáš Kliment
RNDr. Aleš Kozubík, PhD.
Mgr. Juraj Michálek
doc. RNDr. Štefan Peško, CSc.
Ing. Pavel Stříž, PhD.
RNDr. Ladislav Ševčovič
Ing. Michal Žarnay, PhD.

Vydavateľ: Spoločnosť pre otvorené informačné technológie – SOIT, Bratislava

ISBN 978-80-970457-0-8

Sadzba programom pdfTEX Ladislav Ševčovič
