به نام خدا

آغاز کار با میکروکنترلرهای خانواده AVR شرکت ATMEL.

مقدمه

این مقاله برای افرادی نوشته شده است که هیچ آشنایی نصبت به میکروکنترلرهای AVR ندارند و می‌خواهند کار با این میکروکنترلرها را آغاز نمایند.

من در این مقاله میکروکنترلرهای خانواده AVR شرکت ATMEL و کامپایلر avr-gcc را به شما معرفی می‌کنم. و به صورت قدم به قدم به شما می‌آموزم که چگونه برای میکروکنترلرها برانامه بنویسید، چگونه آن‌ها را برنامه ریزی کنید و چگونه از آن‌ها در مدارات خود استفاده کنید.

معرفی میکروکنترلرهای AVR

میکروکنترلرهای AVR توسط شرکت Atmel طراحی و ساخته شده‌اند. اولین قطعات از میکروکنترلرهای AVR در سال ۱۹۹۳ روانه بازار شد و به زودی جای خود را در قلب طراحان مدارات میکروکنترلری باز کرد. نخستین قطعات که در این خانواده معرفی شدند، میکروکنترلرهای AVR در سری AT90Sxxxx بودند. ولی از آنجایی که ای قطعات نسبت به سایر میکروکنترلرهای AVR که بعدا در سری Mega ارائه شدند امکانات کمتری دارند، به ندرت از آن‌ها استفاده میشود.
شرکت Atmel همزمان با ارائه‌ی میکروکنترلرهای AVR در سری Mega اقدام به طراحی و تولید میکروکنترلرهای AVR در سری Tiny کرده است. این قطعات در بسته‌بندی‌های کوچکتر نسبت به همونه‌های قبلی و با امکانات فوق‌العاده [۱] در جریان‌های مصرفی کم ارائه شده‌اند و زمینه را برای طراحی مداراتی با توان مصرفی فوق‌العاده کم و کارآیی بسیار بالا فراهم کرده‌اند.
من در این مقاله بر روی ATMega16 از سری Mega تأکید بیشتری خواهم داشد. دلیل انتخاب این قطعه وجود بسیاری از قابلیت‌های تمامی سری Mega و پایه بودن آن است.
میکروکنترلرهای AVR دارای درگاه داده ۸ بیتی و از نوع CMOS و با ساختار [۲] Risc هستند و در ساخت آن‌ها معماری نوع Harward به کار برده شده است. در این نوع معماری از باس‌های سه‌گانه مجزا (آدرس – داده – کنترل) برای حافظه برنامه استفاده می‌شود. کاربرد ساختار Risc باعث می‌شود که این قطعات دارای خصوصیات منحصر به فردی باشند. از آن جمله می‌توان به سرعت بالا، سازگاری با کامپایلرهای زبان‌های سطح بالا چون C و امکانات فراوانٍ اشاره کرد.
ساختار Risc برای اولین بار در سال ۱۹۷۰ میلادی برای معماری پردازشگرها معرفی شد. پیش از اینٍ معماری [۳] Cisc متداول‌تر بوده است.بحث در مورد این معماری ها از عهده‌ی این مقاله خارج است.

معماری سخت‌افزاری AVR (افراد مبتدی می‌توانند این بخش را مطالعه نکنند.)

هسته اصلی AVR از یک مجموعه دستورالعمل‌های قوی با ۳۲ ثبات همه منظوره تشکیل شده است [۴]. این ۳۲ ثبات به طور مستقیم به واحد محاسبه و منطق (ALU) متصل بوده و ALU می‌تواند در هر لحظه به طور همزمان به ۲ ثبات از این ۳۲ ثبات دسترسی داشته باشد. در نتیجه بسیاری از دستورالعمل‌های AVR در یک پالس ساعت اجرا می‌شود و این امر باعث می‌شود که سرعت پردازش AVR به بیش از ۱۰ برابر سرعت پردازشگرهای Cisc همتای خود برسد و پهنای باندی برابر [۵] 1MIPS برای هر 1MHz پالس ساعت فراهم آید.
واحد محاسبه و منطق در AVR سه نوع عملیات ریاضی، منطقی و توابع بیتی را پشتیبانی می‌کند. در عین حال در هسته میکروکنترلرهای سری Mega از یک ضرب‌کننده پر سرعت (۲ پالس ساعت) استفاده شده است که توانایی عملیات ضرب بر روی اعداد علامت‌دار، بدون علامت و کسری را دارد.

امکانات داخلی ATMega16

امکانات موجود در هر دو نوع بسته‌بندی تقریباً مشابه همدیگر بوده [۶]. و به طور خلاصه شامل این موارد است:

۱) انواع حافظه داخلی
- حافظه برنامه یا FLASH به اندازه 16KB
- حافظه داده یا [۷] RAM به اندازه 1KB
- حافظه ماندگار [۸] EEPROM به اندازه 512Byte
- 32 ثبات همه منظوره

۲) انواع درگاه سریال
- [۹] USART: درگاه سریال سنکرون و آسنکرون همه منظوره استاندارد
- TWI و یا I2C: درگاه سریال ۲ سیمه سنکرون
- SPI: درگاه سریال سنکرون با سرعت بالا برای ارتباط دو پروسسوری و چند پروسسوری

۳) درگاه JTAG برای تست میکروکنترلر و اشکال‌زدایی از نرم‌افزارهای نوشته شده در سیستم واقعی. از این درگاه همچنین برای برنامه‌ریزی حافظه‌های FLASH و EEPROM نیز استفاده می‌شود.
۴) یک مبدل آنالوگ به دیجیتال ۱۰ بیتی با ۸ کانال ورودی
۵) دو تایمر ۸ بیتی و یک تایمر ۱۶ بیتی با امکاناتی چون تقسیم کننده، ورودی شکار و خروجی‌های PWM.
۶) یک مقایسه‌گر آنالوگ
۷) شش حالت خواب
۸) امکان پیکربندی میکروکنترولر با فیوز بیت‌های اختصاصی برای استفاده از اسیلاتور داخلی یا کریستال خارجی و یا

شبکه RC برای تولید پالس ساعت برای میکروکنترلر.

درگاه‌های میکروکنترلرهای AVR

هر درگاه در میکروکنترلرهای AVR از سه ثبات ایجاد شده است، که وضعیت، مقدار و جهت درگاه را تعیین می‌کند. ثبات‌های هر درگاه به شرح زیر است:

DDRx: با صفر بودن هر بیت از این ثبات، بیت متناظر، درگاه، ورودی و با یک بودن آن، بیت متناظر درگاه به صورت خروجی تعریف می‌شود. برای مثال در زبان C دستورات زیر درگاه B را به صورت خروجی و پین 2 از درگاه C را به صورت ورودی تعریف می‌کند:

```
DDRB=0xFF;
DDRC &= ~(1<<2);
```

PINx: این ثبات برای خواندن از درگاه استفاده می‌شود. با نوشتن بر روی ای ثبات، هیچ مقداری به درگاه منتقل نخواهد شد.

PORTx: این ثبات خروجی درگاه است و با نوشتن بر روی این ثبات، داده‌ی نوشته شده بر روی پین متناظر با آن منعکس می‌شود. البته اگر پین به صورت خروجی تعریف شده باشد. ولی اگر پین، به صورت ورودی تعریف شده باشد، یک کردن هر بیت از این ثبات باعث pull-up دار شدن پین متناظر و صفر کردن آن باعث tri-state شدن آن می‌شود.

معرفی کامپایلر avr-gcc

کامپایلر avr-gcc یک کامپایلر باز متن زبان C قدرتمند برای میکروکنترلرهای خانواده‌ی AVR شرکت Atmel است.
avr-libc یک کتابخانه باز متن زبان C با کیفیت بالا است برای استفاده با [۱۰] gcc برای میکروکنترلرهای Atmel AVR.
مجموعه ی نرم‌افزارهای avr-binutils ، avr-gcc و avr-libc زنجیره ای از نرم‌افزارهای آزادی را برای میکروکنترلرهای Atmel AVR تشکیل می‌دهند.
نرم‌افزارهای اضافه دیگری هم همراه آنها وجود دارد. نرم‌افزارهای برنامه‌ریزی uisp و avrdude نرم‌افزار شبیه‌سازی simulavr و نرم‌افزارهای اشکال‌زدایی avr-gdb و AvaRICE. برای اطلاعات بیشتر به سایت راهنمای کاربر avr-libc مراجعه نمایید [۱۱].

لینک این نرم‌افزارها و نحوه‌ی نصب آن‌ها در سیستم عامل گنو/لینوکس (به زبان فارسی) را می‌توانید در وبلاگ من پیدا کنید [۱۲].

مجموعه‌ی این نرم‌افزارها برای سیستم عامل مایکروسافت ویندوز نیز وجود دارد. WinAVR یک پکیج کامل از مجموعه‌ی این نرم‌افزارها برای مایکروسافت ویندوز است [۱۳].

پروگرامر

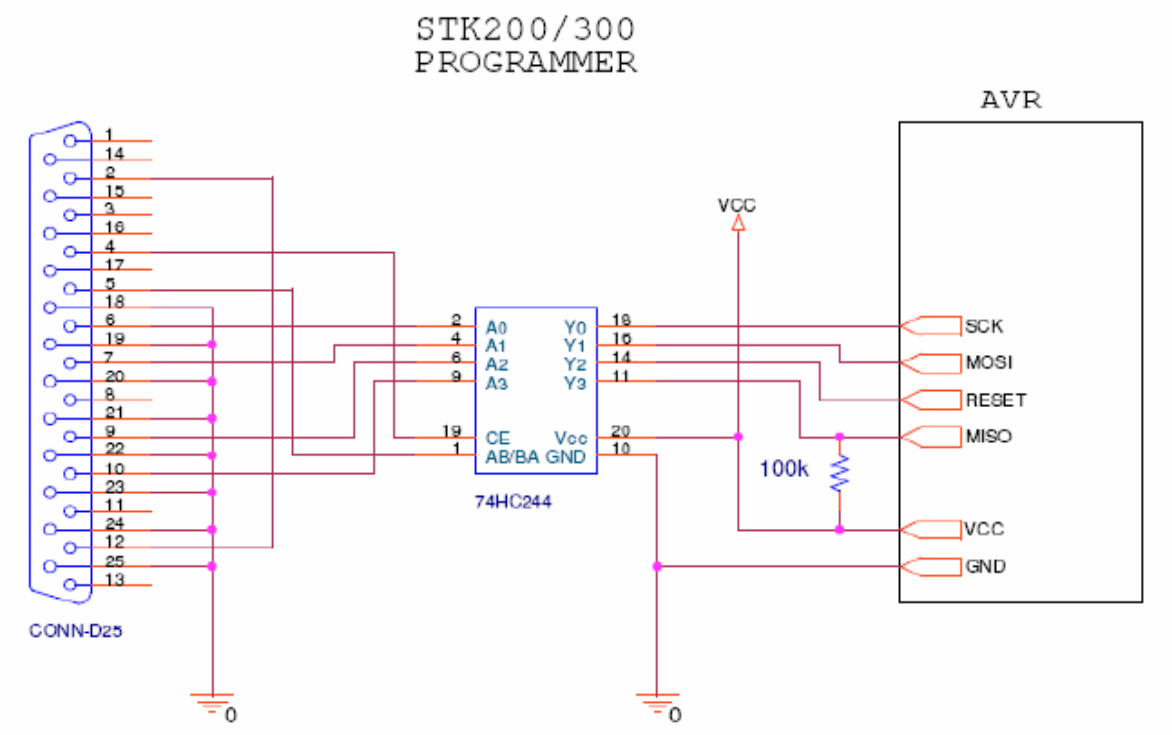
پروگرامر وسیله‌ای است که شما می‌توانید به‌وسیله‌ی آن میکروکنترلرها را برنامه‌ریزی نمایید. برای تهیه‌ی پروگرامر دو راه وجود دارد.

۱) خودتان آن را بسازید.

برای ساخت پروگرامر، چیز زیادی لازم نمی‌باشد. من به شما یاد میدهم که چطور یک پروگرامر STK200/300 بسازید. قطعات زیر را تهیه نمایید:

۱. 1m کابل ۶ رشته‌ای
۲. کانکتور پورت Parallel
۳. یک عدد مقاومت 100KΩ
4. یک عدد آی‌سی بافر 74HC244

شماتیک مدار:



طبق این شماتیک مدار را بسازید و به کامپیوتر خود متصل نمایید.

۲) آن را خریداری نمایید.

بهترین کار خریداری یک پروگرامر آماده است که در فروشگاه‌های قطعات الکترونیکی موجود می‌باشد. و بدون هیچ مشکلی کار می‌کند.

آغاز کار با میکروکنترلرهای AVR

برای شروع ابتدا نرم‌افزارهای توضیح داده‌شده در بالا را نصب نمایید. یک پروگرامر تهیه کنید و به کامپیوتر خود متصل نمایید، سپس یک میکروکنترلر ATMega16 تهیه نمایید و کار برنامه‌نویسی را آغاز نمایید.

کامپایلر avr-gcc یک کامپایلر تحت خط فرمان است **[۱۴]**. این به این معناست که avr-gcc دارای **[۱۵]** GUI نمی‌باشد. و باید دستورات را در خط فرمان اجرا کنید.

ابتداء یک ویرایشگر متن، مانند gedit ، emacs (در گنو/لینوکس) و یا notepad یا pn (در مایکروسافت ویندوز) ویا... را باز کنید. سپس این کد ها را درآن تایپ کنید (دقت کنید که زبان C به کوچک و بزرگی حروف حساس است):

```
//Blinking LED

#include <avr/io.h>

#include <util/delay.h>

//wait for 1 second

void
delay_1s(void)
{
        int i;

        for (i = 0; i < 100; i++)
         _delay_ms(10);
}

int
main(void)
{
        //set DDRB.1 high
        DDRB |= 1<<1;
```

```
        while(1)
        {
         //set PORTB.1 high
         PORTB |= 1<<1;

         //call delay1s() function.
         delay_1s();

         //set PORTB.1 low
         PORTB &= ~(1<<1);

         //call delay1s() function.
         delay_1s();
        }
        return (0);
}
```

سپس یک دایرکتوری بسازید و برنامه را در آن با نام main.c ذخیره کنید.

این برنامه پین شماره یک پورت B طی فواصل زمانی ۱ ثانیه‌ای ۰ و ۱ می‌کند. و در نتیجه اگر به این پایه یک دیود نوری یا همان LED متصل کنید، LED شروع به روشن و خاموش شدن می‌کند (چشمک میزند).

شرح برنامه:

خط اول توضیحات مربوط به برنامه است. در زبان C ، توضیحات با // شروع می‌شوند. توضیحات کامپایل نمی‌شوند و هیچ تاثیری بر برنامه ندارند.
خط بعدی هدر avr.h را به برنامه اضافه می‌کند. این فایل تمامی ثبات‌های AVR را تعریف می‌کند.
خط بعدی فایل هدر delay.h را به برنامه اضافه می‌کندُ که شامل توابعی برای ایجاد تاخیرهای زمانی میان دستورات برنامه است.
خط بعدی: توضیحات.
دو خط بعدی، تابع delay_1s را تعریف می‌کنند.
خط بعد به معنای آغاز تابع است.
بعدی متغیر i را از نوع Integer تعریف می‌کند.
بعدی یک حلقه ایجاد می‌کند که برای ۱۰۰ بار دستور:

_delay_ms(10);

را اجرا می‌کند که ۱۰ میلی‌ثانیه تاخیر در روند برنامه به وجود می‌آورد. در نتیجه ۱ ثانیه تاخیر ایجاد می‌شود.
خط بعدی پایان تابع delay_1s را تعریف می‌کند.
دوخط بعدی تابع main را که تابع اصلی برنامه است تعریف می‌کنند. برنامه از تابع main آغاز می‌شود.
خط بعد به معنای آغاز تابع است.
خط بعدی: توضیحات.
خط بعدی پایه‌ی شماره ۱ از درگاه B را به صورت خروجی تعریف می‌کند.
خط بعدی یک حلقه‌ی بی انتها تعریف می‌کند. کدهای این حلقه تا بی‌نهایت تکرار می‌شوند.
خط بعدی آغاز حلقه را اعلام می‌کند.
خط بعدی: توضیحات.
خط بعدی پین شماره ۱ از درگاه B را یک میکند.
خط بعدی: توضیحات.
خط بعدی تابع delay_1s را صدا میزند که یک تاخیر ۱ ثانیه‌ای ایجاد می‌کند.
خط بعدی: توضیحات.
خط بعدی پین شماره ۱ از درگاه B را صفر میکند.
خط بعدی: توضیحات.
خط بعدی تابع delay_1s را صدا میزند که یک تاخیر ۱ ثانیه‌ای ایجاد می‌کند.
خط بعدی پایان حلقه را اعلام می‌کند.
خط بعدی مقدار صفر را از تابع بر می‌گرداند.
خط بعدی پایان تابع main را اعلام می‌کند.

حال Makefile ی را که ضمیمه‌ی این مقاله هست را در دایرکتوریی که برنامه را در آن ذخیره کرده‌اید کپی نمایید. (دقت کنید که نام Makefile را تغییر ندهید. Makefile هیچ پسوندی ندارد)

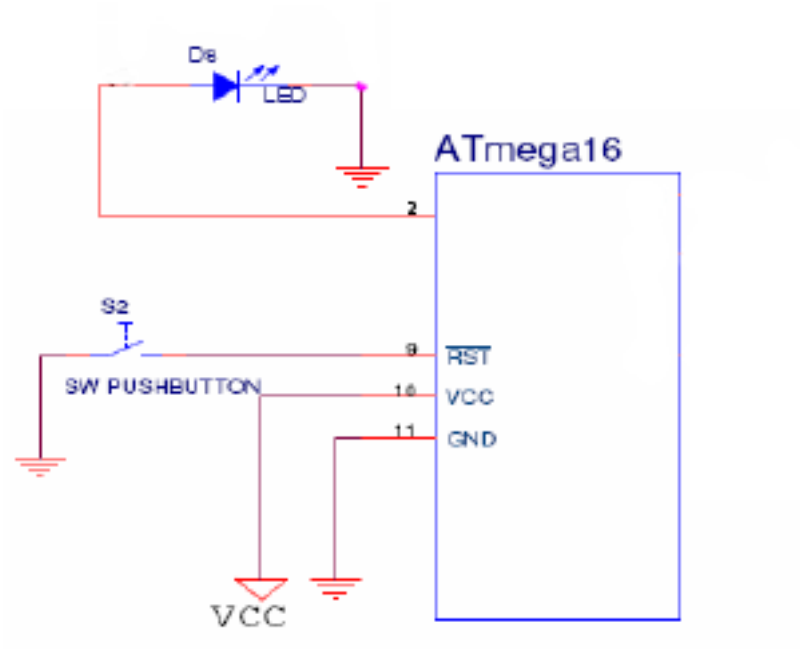
سپس در خط فرمان گنو/لینوکس یا مایکروسافت‌ویندوز با دستور cd به دایرکتوری برنامه بروید و دستور زیر را اجرا کنید: (# را تایپ نکنید)

 # make all

خوب حال اگر در دایرکتوری فایلی به نام main.hex تشکیل شد، میکروکنترلر را به پروگرامر متصل کنید و دستور زیر را اجرا کنید:

 # make program

در صورتی‌که پیغام خطایی مشاهده نکردید، میکروکنترولر شما با موفقیت برنامه‌ریزی شده است.
میکروکنترلر را از پروگرامر جدا کنید و مانند مدار زیر ببندید:



ولتاژ ۵ ولت DC را به آن متصل نمایید. همان‌طور که مشاهده می‌کنید، LED شروع به چشمک زدن می‌کند.

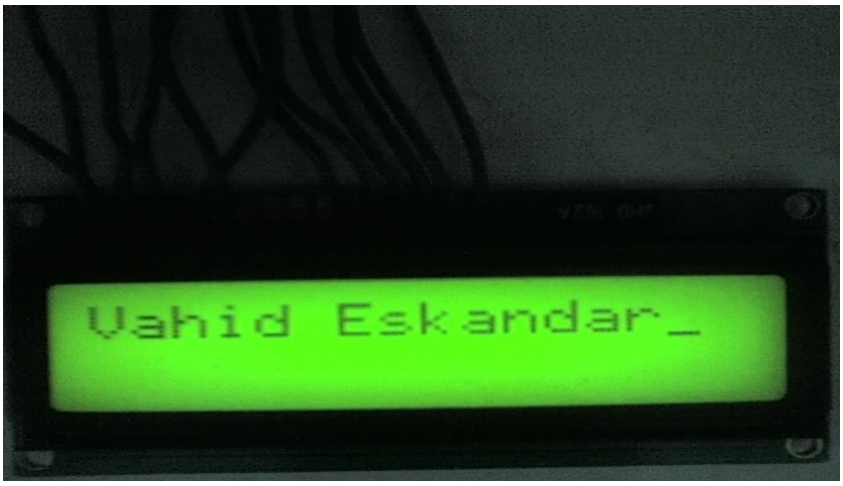
متصل کردن [۱۶] LCD به میکروکنترلر

برنامه‌ای را که در پوشه‌ی LCD قرار دارد را کامپایل کنید و میکروکنترلر را پروگرام کنید.
LCD را به صورت زیر به میکروکنترلر متصل نمایید:

RS -------> PORTB.0
RW -------> PORTB.1
E -------> PORTB.2
D4 -------> PORTB.4
D5 -------> PORTB.5
D6 -------> PORTB.6
D7 -------> PORTB.7

Vcc +5V
Vss GND

حالا مدار را متصل کنید.



شما می‌توانید با تمرین زیاد و یادگیری کامل زبان C برنامه‌های پیشرفته‌تری بنویسید.

شما می‌توانید اطلاعات بیشتر را از سایت Atmel دریافت نمایید. [۱۷]
در صورت مشاهده‌ی اشکال در مقاله می‌توانید از طریق E-mail به من اطلاع دهید.

نویسنده:

وحید اسکندر

:Email

vahid.eskandar@gmail.com

:Weblog

http://atmel.blogfa.com

برای اطلاعات بیشتر درباره‌ی میکروکنترلرها، از وبلاگ من دیدن کنید.

منابع:

AVRLibc
Atmel
Atmel.blogfa.com
میکروکنترلرهای AVR سری Mega

پانویس‌ها:

[۱] امکاناتی که کمتر از سری Mega و حدوداً برابر با سری 90Sxxxx هستند.
[۲] Reduced Instruction Set Computer
[۳] Complex Instruction Set Computer
[۴] این برخلاف پردازنده‌های مشابه Cisc است که معمولاً دارای یک ثبات آکومولاتور بوده و تمام دستورات منطقی روی آن عمل می‌کند.
[۵] Million Instruction Per Second – یک میلیون دستورالعمل در هر ثانیه
[۶] این دو نوع بسته‌بندی در موارد جزئی با یکدیگر اختلاف دارند. از جمله آن‌ها این است که عملکرد مد دیفرانسیلی در ADC بر روی بسته‌بندی Dip تضمین نشده است.
[۷] Random Access Memory – حافظه با دستیابی دلخواه
[۸] Electrically Erasable Programmable Read Only Memory – حافظه فقط خواندنی برنامه پذیر پاکشوی الکتریکی
[۹] Universal Synchronous Asynchronous Receiver / Transmitter
[۱۰] GNU C Compiler – کامپایلر زبان C گنو
[۱۱] AVR Libc
[۱۲] http://atmel.blogfa.com
[۱۳] WinAVR Project
[۱۴] Command Line Compiler
[۱۵] Graphical User Interface - رابط کاربر گرافیکی
[۱۶] Liquid Crystal Display - نمایشگر کریستال مایع
[۱۷] Atmel

The End