

دستور کار آزمایشگاه

درس ریزپردازندۀ ۱

دانشکده مهندسی کامپیووتر

دانشگاه صنعتی امیرکبیر

بِنَامِ خَدَاوَنْدِ جَانِ وَحْسَرِ

خَدَاوَنْدِ نَامِ وَحْسَدِ جَاهِ

خَدَاوَنْدِ كَيهَانِ وَكَرْدَانِ پَهْسَرِ

كَرْزَينِ بَرْ تَرَانْدِ يَشَهِ بَرْ كَنْدَزَدِ

خَدَاوَنْدِ رُوزَيِ دَهِ رسَنْمَاهِ

فَرْ وَزَنْدَهِ مَاهِ وَنَاهِسَدِ دَعْسَرِ

تقدیر و سپاس

بدینویسیله از کلیه کسانی در تهیه این دستورالعمل اینجانب را یاری نموده‌اند یعنی بعضی از مدرسات و دانشجویان آزمایشگاه ریزپردازنده در دانشکده مهندسی کامپیوتر دانشگاه صنعتی امیر کبیر که آزمایشگاه این درس را زیر نظر اینجانب ارائه نموده و یا با هنگام رسانیده‌اند و بخش‌هایی از مطالب این دستورالعمل را تهیه و یا بازبینی نموده‌اند خصوصاً آقایان محمد میرزا آقاتبار، حسین آتشی، خانم حمیده ایزدیار، محمد صالحی، محمدعلی کیوانزاد، حامد رضوانی، فاطمه خادمیان تشکر نموده، برای آنان آرزوی موفقیت در کلیه شرکت‌های زندگی می‌نمایم. جزو حاضر حاوی بخش‌هایی از مطالب مورد نیاز آزمایشگاه ریزپردازنده ۱ می‌باشد و سعی خواهد شد در آینده تدریجاً تکمیل و نوافع آن مرتفع گردد تا بدینویسیله بتواند نیاز دانشجویان و علاقمندان به مبحث ریزپردازنده ۱ را از حیث مطالب علمی و عملی مورد نیاز جهت کسب آشنایی بیشتر و ممارست در این زمینه فراهم آورد. از آنجا که این دستورالعمل مطمئناً دارای نقائص و اشکالاتی است، بدینویسیله از خوانندگان آن تقاضا می‌شود ایرادات آن و نیز پیشنهادات خود را در راستای بهبود هرچه بیشتر آن به اینجانب به یکی از آدرس‌های زیر اطلاع دهند.

تلفن: ۶۴۵۴۲۷۲۲

آدرس الکترونیکی: homayoun@aut.ac.ir

و من الله التوفيق

محمد‌مهدی همایون‌پور

عضو هیات علمی دانشکده مهندسی کامپیوتر

و فناوری اطلاعات دانشگاه صنعتی امیر‌کبیر

فهرست مطالب

۶ مقدمه -۱
۷ ۲- گروه‌بندی دانشجویان، نحوه ارزیابی و نحوه گزارش‌دهی، انجام پروژه پایانی آزمایشگاه
۹ ۳- جلسه اول
۹ ۱-۳- تقدیمه تثیت شده
۱۰ ۲-۲-۳- آشنایی با نرم‌افزار آلریوم
۱۰ ۱-۲-۳- آشنایی با بخش شماتیک
۱۶ ۲-۲-۳- مرور چند نکته
۱۶ ۳-۲-۳- تولید PCB
۲۰ ۴- جلسه دوم
۲۱ ۱-۴- تست اتصالات
۲۲ ۵- جلسه سوم
۲۲ ۱-۵- معرفی پروگرامر STK500
۲۷ ۲-۵- نحوه نصب درایور USB پروگرامر STK500
۳۲ ۳-۵- چند نکته مهم در زمان برنامه‌ریزی میکروکنترلرهای AVR
۳۲ ۴-۵- نحوه نصب نرم‌افزار پروگرامر و تنظیم نوع پروگرامر در آن
۳۳ ۵-۵- معرفی AVR Studio 4
۳۷ ۱-۵-۵- تنظیمات بخش‌های پروگرامر نرم‌افزار AVR Studio
۴۲ ۶-۵- آشنایی با نرم افزار CodeVision
۴۴ ۱-۶-۵- ایجاد فایل جدید پروژه و فایل سورس برنامه
۴۴ ۲-۶-۵- Project کار کردن با
۴۵ ۱-۲-۶-۵- منوی Chip
۴۵ ۲-۲-۶-۵- Port قسمت
۴۶ ۳-۲-۶-۵- LCD قسمت
۴۷ ۳-۶-۵- تنظیم گزینه‌های کامپایلر C
۴۸ ۴-۶-۵- کامپایل کردن پروژه
۴۸ ۵-۶-۵- ساختن پروژه
۴۸ ۶-۶-۵- تنظیمات AVR Chip Programmer
۴۸ ۷-۶-۵- AVR Chip Programmer
۵۰ ۷-۶-۵- معرفی WinAVR
۵۰ ۱-۷-۵- مقدمه
۵۰ ۲-۷-۵- راهنمای نصب
۵۱ ۳-۷-۵- انجام یک پروژه‌ی ساده با WinAVR
۵۲ ۴-۷-۵- نحوه تولید فایل hex

۵۲	-۵-۷-۵-انتقال برنامه به میکروکنترلر
۵۳	۶- جلسه چهارم
۵۴	۶-۱- تولید سیگنال بازنشانی
۵۵	۶-۲- تغییر بیت های فیوز
۵۶	۶-۳- تولید سیگنال ساعت (clock)
۵۹	۷- جلسه پنجم
۶۱	۸- جلسه ششم
۶۳	۹- جلسه هفتم
۶۵	۱۰- جلسه هشتم
۶۷	۱۱- جلسه نهم
۶۹	۱۲- جلسه دهم
۷۱	۱۳- جلسه یازدهم
۷۲	۱۴- جلسه دوازدهم
۷۶	۱۵- جلسه سیزدهم
۸۰	۱۶- مراجع
۸۱	۱۷- پیوست ۱: پایه های میکروکنترلر ATMEGA32
۸۲	۱۸- پیوست ۲ (نحوه ارتباط یک برنامه MICROSOFT VISUAL STUDIO با میکروکنترلر)
۸۴	۱۹- انتقال اطلاعات از میکروکنترلر به کامپیوتر
۸۵	۲۰- انتقال اطلاعات از کامپیوتر به میکروکنترلر
۸۶	۲۱- پیوست ۳ (ارتباط با کامپیوتر از طریق برنامه (HYPER TERMINAL
۸۹	۲۲- پیوست ۴: کد ارتباط به LCD در حالت ۴ بیتی
۹۷	۲۳- جلسه پانزدهم (برقرار ارتباط سریال از طریق رابط (TWI

۱- مقدمه

هدف از درس آزمایشگاه ریزپردازنده ۱، کسب آشنایی عملی با مفاهیم مطرح شده در درس ریزپردازنده ۱ و ممارست در طراحی و ساخت سیستم‌های سخت‌افزاری مبتنی بر ریزپردازنده‌ها و میکروکنترلرهاست. این دستورالعمل آزمایشگاهی با همین هدف تهیه شده است. همانطور که می‌دانیم در درس ریزپردازنده ۱ با مفاهیمی چون معماری ریزپردازنده‌ها و اجزاء آن، انواع حافظه‌های اصلی شامل حافظه‌های ROM، RWM، DRAM، خروجی، دیکود کردن فضای حافظه و فضای پورت‌ها، سرکشی، وقفه و مسائل مربوطه، تولید تأخیر با برنامه، بعضی از ورودی‌ها و خروجی‌های ساده مانند کیبورد و نمایش دهنده هفت قطعه‌ای، ارتباطات سریال و موازی و بعضی مفاهیم دیگر آشنا شدیم. علاوه بر این در درس ریزپردازنده ۱ با معماری و قابلیت‌های فراوانی که میکروکنترلرها در اختیار می‌گذارند مانند استفاده از انواع منابع ساعت، انواع حالات صرفه‌جویی در مصرف توان، چگونگی کار با انواع وقفه‌های خارجی و داخلی، کار با زمان‌سنج‌شمارندها، استفاده از درگاه‌های موازی، برقراری ارتباط سریال توسط واسطه‌هایی مانند USART، SPI و I2C، کار با مقایسه کننده آنالوگ و مبدل آنالوگ به دیجیتال و برنامه‌ریزی میکروکنترلر به زبان‌های اسمنبلی و زبان C برای برنامه‌نویسی و کار با میکروکنترلر آشنا شدیم. در این دستور کار سعی شده است تا با انجام آزمایش‌هایی، بخش عمده‌ای از مفاهیم فوق را بطور عملی تمرین و تجربه کرده و از اسمنبل و کامپایلر برای تهیه برنامه کار ریزپردازنده‌ها و میکروکنترلرها استفاده نماییم. علاوه بر این در این آزمایشگاه با چگونگی استفاده از نرم‌افزار آلتیوم نیز آشنا خواهیم شد تا از آن برای طراحی شماتیک و بورد مدار چاپی و انجام پروژه‌های عملی مبتنی بر استفاده از میکروکنترلرها و ریزپردازنده‌ها استفاده نماییم. پیش‌نیازهای مورد نیاز برای این آزمایشگاه عبارتند از:

- آشنایی با معماری ریزپردازنده‌ها و میکروکنترلرهای
- آشنایی با مبانی ساخت سیستم‌های مبتنی بر ریزپردازنده و میکروکنترلر
- آشنایی با زبان C
- آشنایی با اسمنبلی میکروکنترلرهای خانواده AVR
- آشنایی با کامپایلرهای CodeVision و Atmel Studio
- آشنایی با نرم‌افزار Proteus توصیه می‌شود.

۲- گروه‌بندی دانشجویان، نحوه ارزیابی و نحوه گزارش‌دهی، انجام پروژه پایانی آزمایشگاه

- گروه‌بندی دانشجویان
- بیان چگونگی ارزیابی دانشجویان
- بیان نحوه تهیه پیش‌گزارش و گزارش هر جلسه
- بیان چگونگی انجام پروژه پایانی آزمایشگاه

گروه‌بندی: در اولین جلسه آزمایشگاه گروه‌بندی دانشجویان در گروه‌های ۲ یا حداقل ۳ نفره انجام می‌شود.

ارزیابی: نحوه ارزیابی دانشجویان بر اساس موارد زیر خواهد بود:

- انضباط، رعایت مقررات آزمایشگاه، کار تیمی و حداقل همکاری با استاد آزمایشگاه (۱۰٪)، و سرپرست آزمایشگاه (۵٪).
- کیفیت عملکرد در آزمایشگاه (شامل مطالعه مطلب مورد نیاز قبل از هر آزمایش و تسلط کافی بر مطالب، جستجو و تهیه دیتاشیت قطعات مورد نیاز برای آزمایش، پاسخگویی به سوالات) (۳۵٪)
- تهیه پیش‌گزارش و گزارش کار آزمایش‌ها (۳۰٪)
- پروژه نهایی (۲۰٪)

پیش‌گزارش: پیش‌گزارش به صورت دستنویس تهیه شده و در ابتدای جلسه تحويل استاد آزمایشگاه می‌گردد. پیش‌گزارش باید شامل موارد ذیل باشد.

- صفحه عنوان (شامل عنوان و شماره آزمایش، نام دانشجو، نام استاد آزمایشگاه، تاریخ آزمایش)
- مقدمه و هدف آزمایش
- لیست قطعات بکار رفته در آزمایش
- متن اصلی شامل ارائه شماتیک مدار، توصیف طرز کار مدار، طرز کار قطعات مهم بکار رفته با استفاده از اطلاعات و اشکال برگرفته از دیتاشیت آنها و سایر نکات فنی مهم
- توضیح چگونگی استفاده از ثبات‌های کنترلی و وضعیت

گزارش هر آزمایش می‌باشد بصورت تایپ شده (در محیط Word) و مرتب حداقل تا هفته بعد از آزمایش (غیرقابل تمدید)، به آدرس ایمیل استاد درس آزمایشگاه ارسال یا در سیستم مدیریت دروس دانشکده (moodle) در محل مربوط به آزمایشگاه ریزپردازندۀ بارگذاری گردد. این گزارش شامل موارد زیر می‌باشد:

- برنامه نوشته شده و کامنت‌گذاری بخش‌های مهم آن
- بررسی مشکلات پیش‌آمده در طول آزمایش و نحوه حل و برطرف کردن آنها
- ارائه نظرات و ایده‌های پیشنهادی برای بهبود عمکرد و کارایی مدار آزمایش
- مراجع استفاده شده شامل سایت‌های مفید، کتاب و جزو
- پیوست‌ها

پروژه پایانی آزمایشگاه:

اعضاء هر گروه آزمایشگاه ریزپردازنده در نهایت می‌باشد یک دستگاه مبتنی بر استفاده از میکروکنترلر و سایر ادوات مورد نیاز (شامل سنسورها، عملگرها، ماجول‌های ارتباطی می‌سیم بلاسیم و مانند آن) را به طور کامل طراحی و بسازند. این پروژه مشتمل بر ساخت‌افزار (مدارچایی PCB، قطعات مورد نیاز) و نرم‌افزار مربوطه است. ساخت سخت‌افزار شامل ارائه طراحی مدار پروژه، رسم مدارد پروژه در قالب شماتیک در محیط Altium Designer، تبدیل طرح شماتیک به PCB (تهیه یا طراحی فوتبرینت قطعات، جایابی قطعات placement)، مسیریابی (routing) بین قطعات، ارسال فایل PCB برای ساخت، خرید قطعات از بازار، مونتاژ و لحیم‌کاری قطعات، تست سخت‌افزار، انتقال نرم‌افزار (کد و برنامه دستگاه) بر روی سخت‌افزار، تست تمام سخت‌افزار و نرم‌افزار، تست فانکشن (دستیابی به کار کرد مورد نظر پروژه) و تحويل پروژه به استاد آزمایشگاه می‌باشد.

دانشجویان تا جلسه پنجم آزمایشگاه موظفند با نظر استاد درس آزمایشگاه، موضوع پروژه خود را انتخاب و قطعی نمایند. دانشجویان از همان اوایل ترم نسبت به انجام امور مربوط به پروژه آزمایشگاه خود اقدام نمایند. به عنوان مثال می‌توانند به طور تدریجی نسبت به انجام طراحی پروژه، انتخاب قطعات و مازول‌های مورد نیاز، تهیه مدار سخت‌افزاری هر مازول، انتقال مدارهای سخت‌افزاری در قالب شماتیک به محیط آلریوم، تهیه فوتبرینت قطعات، تهیه کد مربوط به کار با هر مازول اقدام نمایند تا حجم کار پروژه در طول ترم توزیع و به آخر ترم منتقل نشود. به عنوان مثال مواردی از ملزمات اکثر پروژه‌ها، تامین ورودی کیبود و خروجی نمایش‌دهنده LCD است. از انجا که این دو مازول در دو جلسه از جلسات آزمایشگاه طراحی و راهاندازی می‌شوند، لذا دانشجویان همان موقع شماتیک این دو مازول را در محیط آلریوم وارد و کد راهاندازی آنها را مرتب و کتاب بگذارند. در خاتمه ترم با تجمعی این بخش‌های سخت‌افزاری و نرم‌افزاری و نهایتاً اضافه کردن سایر مازول‌های مورد نیاز، پروژه تکمیل می‌گردد.

۳- جلسه اول

هدف از آزمایش:

- ساخت مدار تغذیه ثبیت شده
- طراحی شماتیک و مدار چاپی^۱ به کمک نرم افزار آنتیوم

قطعات مورد نیاز:

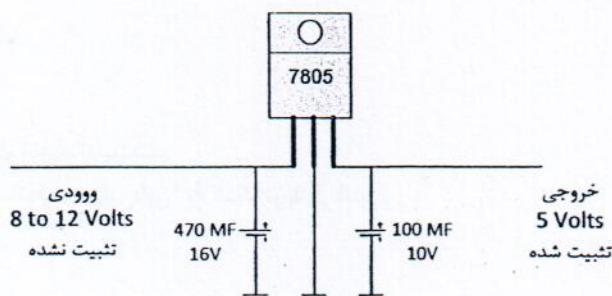
- ترانزیستور BC107: ۱ عدد
- تراشه گیت Nand: ۱ عدد
- رگولاتور ۷۸۰۵: ۱ عدد
- خازن ۴۷۰ میکروفاراد ۱۶ ولتی: ۱ عدد
- خازن ۴۷ یا ۱۰۰ میکروفاراد ۱۰ ولتی: ۱ عدد
- مقاومت ۲۰۰ مگا اهم: ۱ عدد
- مقاومت ۱۰۰ کیلواهم: ۱ عدد
- مقاومت ۱۳۳۰ اهم: ۱ عدد
- مقاومت ۱۸ کیلواهم: ۱ عدد
- ۱ LED

۱-۳- تغذیه ثبیت شده

از آنجا که در ساخت چراغ چشمکزن در این جلسه آزمایشگاه، نیاز به تغذیه ثبیت شده می‌باشد و از طوفی در آزمایش‌های جلسات بعدی با استفاده از میکروکنترلر ATmega32، نیاز به این نوع تغذیه خواهیم داشت، لذا در ادامه نحوه ساخت یک ولتاژ تغذیه ۵ ولتی ثبیت شده ارائه خواهد شد.

برای تامین ولتاژ تغذیه ثبیت شده و حفاظت قطعات دیجیتال در مقابل ولتاژهای بالا، می‌توان از یک رگولاتور یا ثبیت‌کننده ولتاژ استفاده نمود. وظیفه ثبیت‌کننده ولتاژ، دریافت سیگنال با ولتاژ ثبیت نشده در ورودی، ثبیت آن به مقدار ولتاژ موردنظر و تحويل سیگنال با ولتاژ ثبیت شده در خروجی آن است. ولتاژ تغذیه ثبیت شده مورد نیاز در آزمایش‌های این آزمایشگاه از جمله برای تغذیه گیت‌های Nand و تغذیه میکروکنترلر ATmega32، ۵ ولت می‌باشد. لذا برای بدست آوردن ولتاژ تغذیه ثبیت شده ۵ ولتی از تراشه 7805 استفاده می‌نمائیم. همانطور که در شکل ۱-۳ مشاهده می‌شود، ثبیت‌کننده 7805 ۷۸۰۵ دارای سه پایه می‌باشد. پایه اول از سمت چپ، پایه ورودی می‌باشد که سیگنال ورودی به این پایه وارد می‌شود، پایه دوم، به زمین متصل می‌شود و پایه سوم، خروجی رگولاتور است که سیگنال با ولتاژ ثابت ۵ ولت را فراهم می‌کند. شکل ۱-۳ مدار مورد نیاز برای تبدیل یک ولتاژ یکسو شده به یک ولتاژ ثبیت شده را ارائه می‌نماید. خوب است که ولتاژ سیگنال ورودی به رگولاتور در حدود بیشتر از ۲ ولت از سیگنال خروجی آن بیشتر باشد. چنانچه این اختلاف ولتاژ زیاد شود، در زمانیکه جریان قابل توجهی از رگولاتور کشیده می‌شود، رگولاتور گرم می‌شود که گرمای حاصل را تاحدی می‌توان توسط خنک‌کننده‌های آلومینیمی که به بدنه رگولاتور پیچ می‌شوند، کاهش داد. شکل زیر برای حالتی رسم شده است که شماره ثبیتساز از مقابل دیده شود.

PCB (Printed Board Circuit)^۱



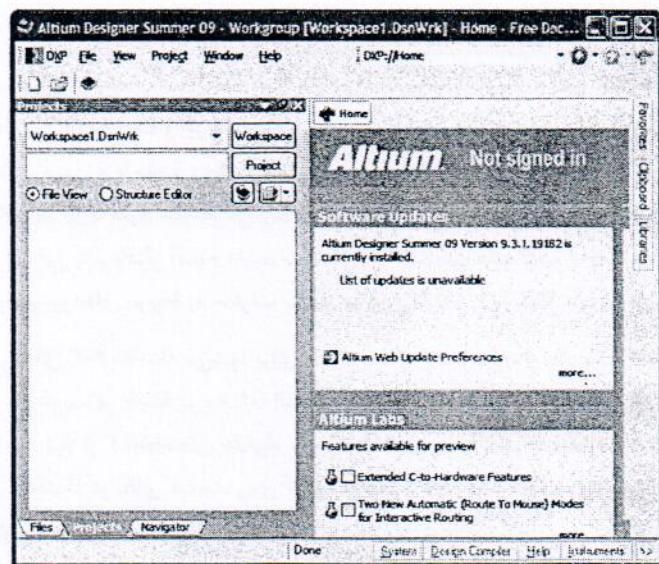
شکل ۲-۳ مدار تقدیم تثبیت شده ۵ ولت با استفاده از تثبیت گفته ۷۸۰۵

۲-۳- آشنایی با نرم افزار آلتیوم

در این جلسه آزمایشگاه می خواهیم با نحوه استفاده از نرم افزار Altium Designer 2009 برای طراحی و تولید شماتیک (schematic) و طرح بورد مدار چاپی پی.سی.بی (PCB) آشنا شویم. به منظور آشنایی عملی با این نرم افزار عملاً به طراحی شماتیک، ساخت پی.سی.بی و مونتاژ و لحیم کاری یک مدار الکترونیکی ساده اقدام خواهیم نمود. این پروژه طراحی یک مدار چشمکزن با یک LED (مدار مولتی ویبراتور بی استابل) است. از آموخته های حاصل از انجام این پروژه در انجام پروژه نهایی آزمایشگاه استفاده خواهد شد. برای این منظور ابتدا در محیط این نرم افزار شکل شماتیک مدار مورد نظر را رسم کرده و سپس بورد پی.سی.بی را به کمک آن مسیردهی و طراحی خواهیم کرد. کار کردن با نرم افزار آلتیوم راحت و به صرفه است. در این بخش تا حد زیادی با این فرایند آشنا خواهیم شد. برای این منظور ابتدا با صفحه شماتیک و رسم مدار با توجه به امکانات کتابخانه ای موجود در آن آشنا شده و سپس به نحوه آماده سازی بورد با استفاده از امکانات مسیردهی می پردازیم.

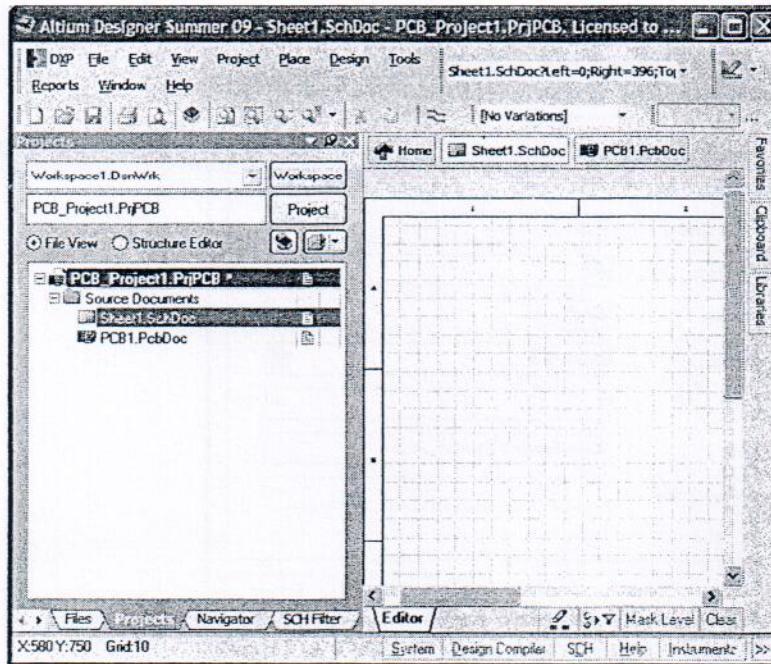
۲-۱- آشنایی با بخش شماتیک

مدار مورد نظر یک مدار چشمکزن^۱ ساده است که در شکل ۲-۳ نشان داده شده است. ابتدا برنامه آلتیوم را اجرا می کنیم که پس از اجرای برنامه مذکور محیطی به صورت شکل ۲-۳ ظاهر می گردد.



شکل ۲-۳. محیط نرم افزار آلتیوم

برای شروع کار از منوی File در بالای صفحه، یک PCB Project جدید را بسازید و به این ترتیب یک پروژه جدید به وجود می‌آوریم. سپس مشابه عمل بالا ابتدا یک فایل Schematic و سپس یک فایل PCB به این پروژه اضافه می‌کنیم. پس از ایجاد پروژه و اضافه کردن فایل‌های مذکور به پروژه، محیط نشان داده شده در شکل ۲-۳ به صورت شکل ۳-۳ دیده می‌شود.

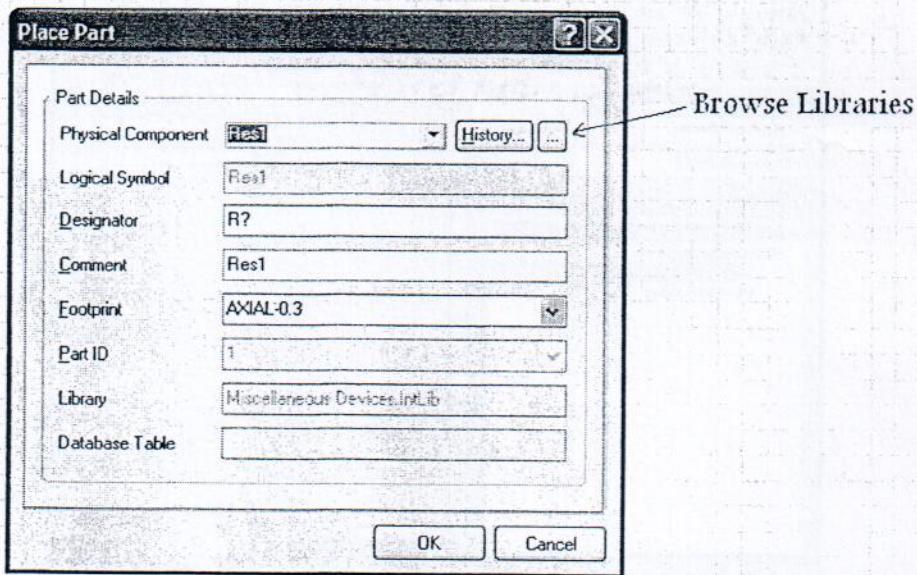


شکل ۳-۳. اضافه کردن فایل Schematic و PCB به پروژه

می‌توان با کلیک بر روی صفحه شماتیک نمایش داده شده و استفاده از دکمه‌های Page Down و Page Up صفحه کلید و یا استفاده از نگهداشتن دکمه Ctrl و چرخاندن اسکرول موس، اندازه این صفحه را به دلخواه کوچک و بزرگ کرد. پیشنهاد می‌کنیم که برای پروژه‌های رسمی و یا مفصل، مشخصات فایل در محل تعییه شده در صفحه شماتیک درج شود. صفحه شماتیک در واقع قسمتی است که مدار بورد مورد نظر در آن رسم و ذخیره می‌شود. شماتیک مدار بورد چشمکزنی که قصد تهیه بورد مدارچاپی آن را با هدف آشنایی با نرمافزار آلتیوم داریم در شکل ۶-۳ نمایش داده است. آشنایی با آلتیوم با این هدف انجام می‌شود که بعد از این بتوانیم بوردهای مربوط به مدارات مبتنی بر ریزپردازنده را به کمک آن طراحی کنیم.

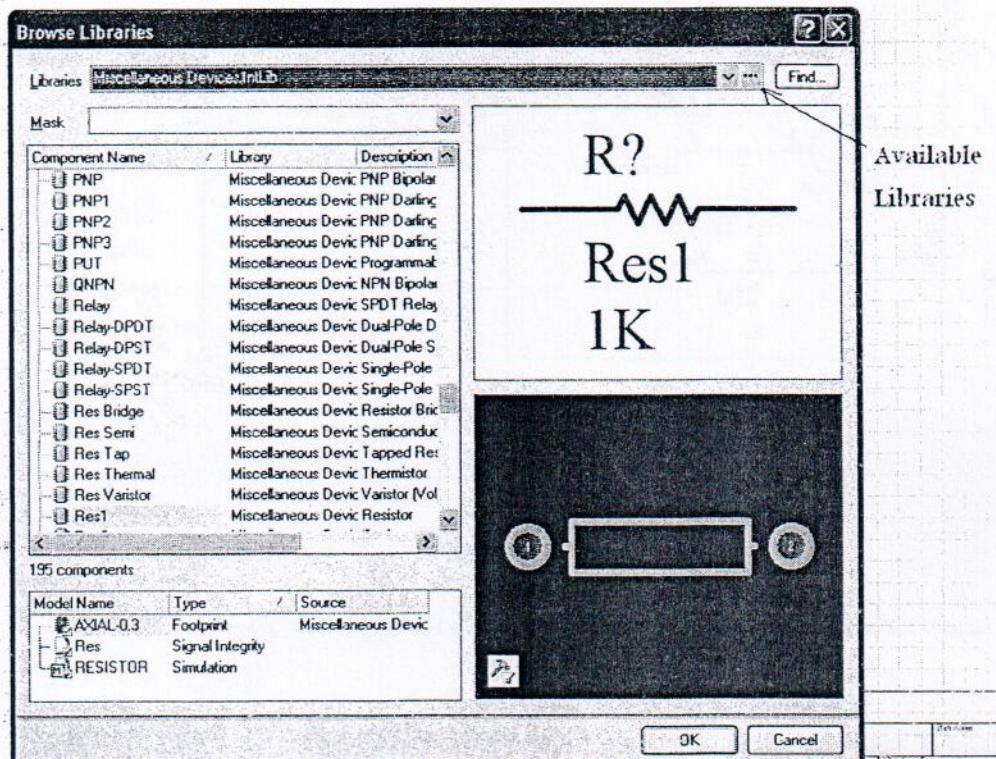
همان طور که در شکل ۶-۳ ملاحظه می‌شود برای طراحی مدار چشمکزن، از یک عدد LED، سه گیت منطقی NAND و تعدادی مقاومت و خازن و ترانزیستور با مقادیر مختلف استفاده شده است. برای رسم این مدار در صفحه شماتیک باید در ابتدا با نحوه وارد کردن عناصر مورد نیاز از کتابخانه‌های دلخواه موجود در آلتیوم و یا سایر کتابخانه‌های قطعات الکترونیکی آشنا شویم. برای این منظور از منوی Design در بالای صفحه، روی گزینه Browse Library کلیک کرده و به این ترتیب صفحه کتابخانه در سمت راست صفحه نمایش ظاهر می‌گردد.

برای انتخاب مقاوست‌ها از کتابخانه Miscellaneous Devices.Intlib که حاوی انواع مختلفی از عناصر الکتریکی و الکترونیکی است، استفاده می‌کنیم. در سطر دوم از لیست کتابخانه عبارت Res1 را تایپ کرده و سپس با فشردن دکمه Place از همین صفحه عملیات انتقال ۴ مقاومت لازم در مدار را انجام می‌دهیم. قبل از گذاشتن قطعه در صفحه می‌توان با زدن دکمه Tab ابتدا ویژگی‌های قطعه را مشخص و سپس آن را در صفحه قرار داد. برای پایان عملیات کافی است که روی صفحه شماتیک کلیک راست کنیم. روش دیگر برای قرار دادن قطعات استفاده از منوی Place و قسمت Part است. همچنین می‌توان از کلیدهای میانبر و با فشردن p+P استفاده کرد. پس از فشردن دوبار کلید p شکل ۶-۳ ظاهر می‌شود. قسمت History لیست قطعاتی را که قبل اضافه شده‌اند را نشان می‌دهد.



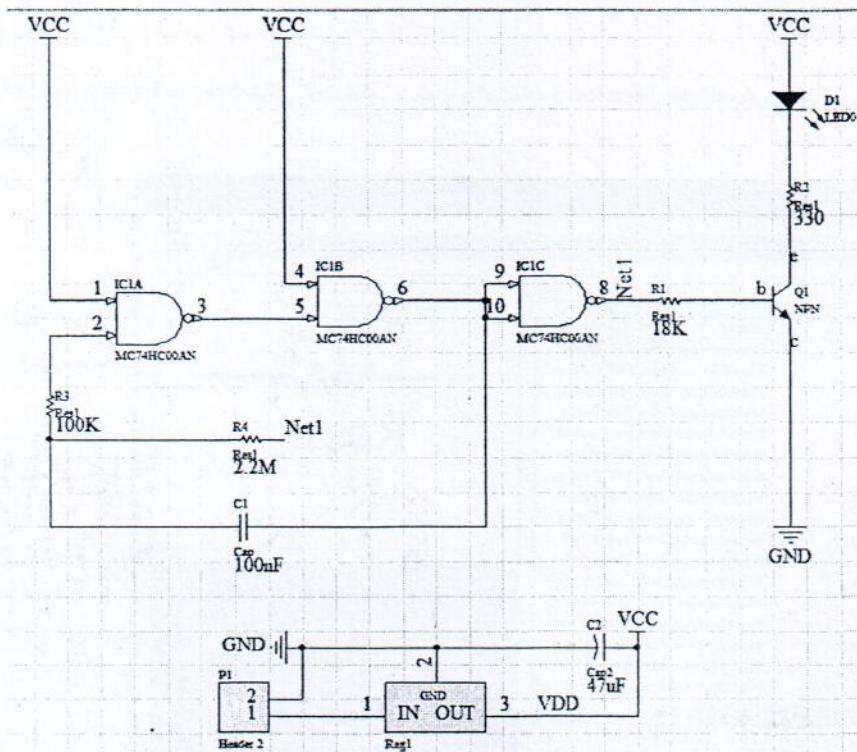
شکل ۶-۳-منوی Place Part

با فشردن شکل ۵-۳ Browse Library ظاهر شده و با استفاده از Available libraries می‌توان کتابخانه‌های مختلف را به پروژه اضافه کرد.



شکل ۵-۳-منوی Browse Libraries

بدين صورت می‌توان تمام قطعات مورد نیاز را به شماتیک اضافه کرد. پس از قرار دادن مقاومت‌ها، به مقداردهی و نامگذاری آن‌ها می‌پردازیم. برای این منظور روی اسم هر مقاومت (R?)، دوبار کلیک کرده و بدين ترتیب صفحه‌ای نمایش داده می‌شود که می‌توان در فیلد Value آن نام دلخواه را وارد کرد. چنانچه روی مقدار مقاومت (1k) دوبار کلیک کنیم، می‌توان در صفحه باز شده مقدار دلخواه را با ذکر واحد آن وارد کرد. البته روش دیگری نیز وجود دارد و آن به این ترتیب است که روی خود قطعه دوبار کلیک کنیم. در این حالت لیست کامل تری باز شده و در قسمت Designator نام عنصر و در بخش Value مقدار آن را وارد کرده و نهایتاً با فشردن دکمه Enter از صفحه کلید، همه تغییرات به یکباره اعمال خواهد شد. دقیقاً همین مراحل را برای انتقال خازن‌های مورد نیاز و ترانزیستور انجام می‌دهیم. این عناصر هم در همان کتابخانه مذکور قرار دارند و این بار به جای عبارت Res1 عبارت Cap و NPN را در صفحه کتابخانه تایپ می‌کنیم.



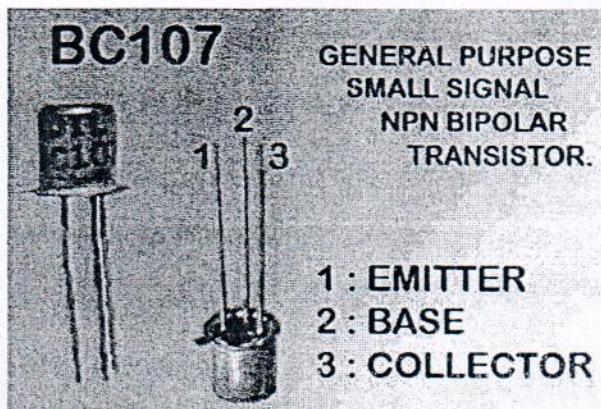
شکل ۳-۶-۳. مدار یک مدار چشمکزن ساده که در صفحه شماتیک آلتیوم رسم شده است

یک LED را از کتابخانه Intlib Miscellaneous Devices وارد صفحه شماتیک می‌کنیم. بعد از نامگذاری این عنصر، روی آن دوبار کلیک می‌کنیم. در صفحه باز شده، روی کلمه Footprint دوبار کلیک کرده و به این ترتیب یک منوی جدید باز می‌شود. در این صفحه گزینه Any از لیست PCB Library را فعال کرده و سپس دکمه Browse را فشار می‌دهیم. در لیست ظاهر شده RB5-1.0.5 را انتخاب می‌کنیم که مناسب‌تری برای LED در مدار مورد نظر ماست.



شکل ۳-۷-۳. مقایسه Footprint اولیه (سمت راست) و نهایی (سمت چپ)

برای ترانزیستور مورد استفاده نیز به سبب نزدیکی پایه‌های آن، باید Footprint آن به شکل دیگری تبدیل شود. در اینجا ما از شکل HDR1X3 از کتابخانه IntLib استفاده نمودیم. همچنین ویژگی‌های ترانزیستور مورد استفاده در این مدار در شکل ۸-۳ آمده است.



شکل ۸-۳. پایه‌های ترانزیستور مورد استفاده در مدار

همان طور که در نقشه مدار دیده می‌شود، به سه گیت NAND احتیاج داریم. برای این منظور از کتابخانه Motorola Logic Gate.IntLib سه قطعه MC74HC00AN انتخاب می‌کنیم. با توجه به این که در IC‌های موجود در بازار ۴ گیت NAND جاسازی شده است، در این مرحله باید اقدام ظرفی انجام دهیم. برای این منظور ابتدا نام هر دو قطعه را یکسان انتخاب می‌کنیم، سپس روی یکی، از دو قطعه دوبار کلیک کرده و در منوی باز شده به کمک فلش‌های موجود، گیت NAND مذکور را به عنوان قسمت دوم از مجموعه (Part 2/4) انتخاب می‌کنیم. البته این کار توسط خود نرم‌افزار نیز به صورت اتوماتیک، انجام می‌شود. در ادامه در قسمت Designator نام قطعه را انتخاب کرده و دکمه Enter را فشار می‌دهیم. به این ترتیب در مرحله مسیردهی، نرم‌افزار هر سه گیت NAND را از مجموعه یک IC انتخاب خواهد کرد. برای رگولاتور استفاده شده در مدار نیز می‌توان از کتابخانه Motorola Power Mgt Voltage Regulator.IntLib را استفاده نمود.

برای اتصال منبع تغذیه و زمین به بورد به یک قطعه هدر (Header) دوتایی احتیاج داریم. این قطعه را از کتابخانه Miscellaneous Connectors.IntLib تحت عنوان Header2 انتخاب کرده و وارد صفحه شماتیک کرده و نام دلخواهی برای آن در نظر می‌گیریم. در مرحله پایانی ورود عناصر، باید تغذیه و اتصال زمین را وارد صفحه شماتیک کنیم. این موارد را می‌توان از منوی عناصر بالای صفحه، انتخاب کرد.

پس از چینش اجزاء نوبت به سیم‌کشی بین آن‌ها می‌رسد. برای این منظور دو مکانیزم وجود دارد. اول این که از منوی Place و یا از لیست اشکال بالای صفحه، گزینه Wire را انتخاب کرده و اتصالات لازم را به وجود آوریم. وقتی که بین دو قطعه سیم می‌کشیم، مکان مناسب برای اتمام سیم جایی است که علامت ضربدر قرمز در آنجا ظاهر می‌شود. در شرایطی که مدار بسیار متراکم و یا فواصل سیم‌کشی طولانی باشد، می‌توان از مکانیزم Net استفاده کرد. برای انجام این کار روی نقاطی که می‌خواهیم به هم متصل شوند، دو NetLabel قرار داده و نام مشابهی را برای آن‌ها برمی‌گزینیم. به عنوان مثال در مدار شماتیک شکل ۶-۳ برای اتصال مقاومت R4 به مقاومت R1، از دو Net استفاده کرده‌ایم. Net را می‌توان از آیکن موجود در بالای صفحه انتخاب نمود.

همچنان که در شکل ۶-۳ مشاهده می‌شود، در قسمتی از مدار از یک Net با نام VDD استفاده شده است. این نام برای نرمافزار از قبل تعریف شده و معنای آن اتصال این نقطه از مدار به پایه تغذیه تراشه‌های استفاده شده می‌باشد. از آنجا که ممکن است در مدار لزوماً VCC همان ولتاژ ۵ ولت نباشد، لذا حتماً باید با گذاشتن این NET، ولتاژ تغذیه برای تراشه را فراهم نمود.

۲-۲-۳- مرور چند نکته

لازم به ذکر است که در انتهای ستون کتابخانه می‌توان footprint عنصر انتخابی را نیز مشاهده کرد و بسته به نیاز مورد مناسب را انتخاب نمود. می‌توان حين Hold کردن یک المان در صفحه شماتیک (با دکمه چپ ماوس) با فشردن کلید space از صفحه کلید، آن را تحت زوایای ۹۰ درجه در صفحه چرخاند. همچنین در زمان Hold بودن قطعه، با زدن دکمه X قطعه حول محور Y و با زدن دکمه Z قطعه حول محور X خواهد چرخید. برای قرار دادن رگولاتور همانند شکل ۶-۳ نیاز به استفاده از این نوع چرخش دارید.

چنانچه نام کامل عنصر مورد نیاز و همچنین کتابخانه مربوط به آن را ندانیم، می‌توانیم از جستجو در کتابخانه‌ها استفاده کیم. برای این منظور در ستون کتابخانه ابتدا اطلاعات موجود در سطر مربوط به انتخاب عناصر را پاک کرده و سپس دکمه search را در بالای لیست فشار می‌دهیم. در صفحه‌ای که باز می‌شود، در قسمت path مسیری را که در آن پرونده library قرار گرفته مشخص می‌کنیم. سپس گزینه Name یا هر ویژگی دیگری از قطعه را از لیست Filters انتخاب و مقدار ویژگی مورد نظر را وارد می‌کنیم. بهتر است operator را نیز به شکل مناسبی برای جستجوی دقیق مشخص نماییم. پس از آن که جستجو به پایان رسید، لیست قطعات پیدا شده قابل انتخاب هستند.

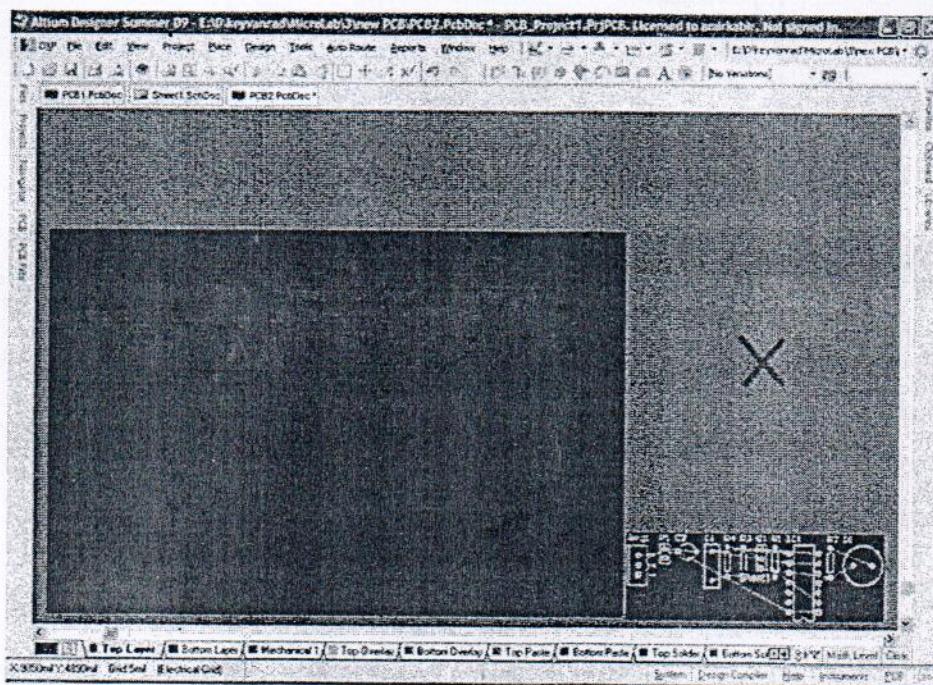
۳-۲-۳- تولید PCB

قبل از تولید PCB حتماً باید فایل‌های مربوط به پروژه را ذخیره نماییم. سپس در صفحه شماتیک در منوی Design از بالای صفحه، گزینه Update PCB Document PCB1.PcbDoc را انتخاب می‌کنیم. بدین ترتیب یک صفحه باز می‌شود. در این صفحه ابتدا دکمه Validate Changes و سپس Execute Changes را فشرده و نهایتاً آن را می‌بندیم. چنانچه در طراحی صفحه شماتیک مشکلی رخ داده باشد، خطاهایی در این صفحه مشاهده خواهد شد. در این شرایط باید ضمن رجوع به صفحه شماتیک و کشف و رفع خطاهای مراحل مذکور را مجدداً تکرار نماییم. در مرحله بعد، روی صفحه PCB کلیک کرده و کلید Page Down از صفحه کلید را آن قدر فشار می‌دهیم تا مدار در پایین صفحه مانند آنچه که در شکل ۹-۳ نشان داده شده است مشاهده گردد. برای رفتن روی محیط بورد در PCB، بر روی صفحه خالی در محیط آلیوم (مثل محل ضربدر قرمز رنگ در شکل ۹-۳) کلیک راست کرده و از گزینه view گزینه Fit board را انتخاب کنید.

مدار را انتخاب کرده و وارد قسمت مشکی رنگ می‌کنیم. سپس با کلید Page Up صفحه را به وضعیت عادی باز می‌گردانیم. بعد روی لایه شفافی که عناصر بر روی آن قرار گرفته‌اند کلیک کرده و آن را پاک می‌کنیم. در این لحظه می‌توان عناصر را به صورت دلخواه در صفحه چید. این بخش تا حد زیادی به ذوق هنری طراح وابسته است چرا که او می‌تواند با چینش مناسب اجزا، مجموعه ای شکل را به عنوان محصول نهایی ارائه کند. پس از آن که اجزا به نحو دلخواه در کنار هم قرار گرفته‌اند، نوبت به تعیین محدوده بورد می‌رسد. برای این منظور از منوی Design گزینه Board Shape را Redefine Board Shape انتخاب کرده و به این ترتیب مستطیلی دور مجموعه عناصر مدار کشیده و در پایان کلیک راست می‌کنیم. مرحله بعد که از اهمیت بالایی هم برخوردار است، تعیین محدوده مسیردهی دور تا دور مدار می‌باشد چراکه نرمافزار را در حین عمل Routing هدایت کرده و آن را از

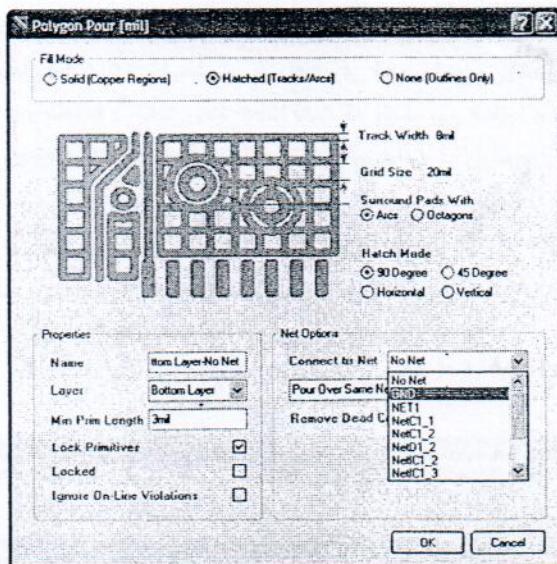
خروج احتمالی از محدوده بورد بر حذر می‌دارد. برای تعیین این محدوده، باید از لایه‌های پایین صفحه Keep Out Layer را انتخاب کرده و سپس دور تادور قطعه را سیم کشی کرد.

با انجام مراحل فوق مقدمات عملیات مسیر دهی توسط نرم‌افزار انجام شده است. در این مرحله از منوی Auto Route گزینه All را انتخاب می‌کنیم. در صفحه‌ای که باز می‌شود، به صورت پیش‌فرض استراتژی مسیردهی Default 2 Layer Board است. بهتر است، در صورت فشرده نبودن مدار و نبود محدودیت ابعاد برای آن، حالت یک لایه را انتخاب کرد تا هزینه ساخت بورد کمتر شود. اگر یک لایه انتخاب شود، باید آن را لایه زیرین در نظر بگیریم تا قطعات را بر روی بورد گذاشته و از زیر لحیم کنیم. در این استراتژی بخشی از سیم‌ها از بالا و بخش دیگر از قسمت زیرین بورد عبور خواهد نمود. حال دکمه Edit Rules را فشار می‌دهیم. در بخش Routing از گزینه Width پنهانی حداقل، حدکثر و ایده‌آل سیم‌ها را انتخاب می‌کنیم. در مدار مورد بررسی ما هر سه مقدار با هم مساوی و برابر ۰.۶ mm (معادل 23.622 mil) انتخاب شدند. یک mil برابر ۰.۰۰۱ اینچ می‌باشد و هر اینچ معادل ۲.۵۴ سانتی‌متر است. پارامتر مهم دیگر Minimum Clearance در بخش Electrical است که فاصله بین خطوط و مسیرها را مشخص می‌کند که در این مدار برابر با ۱ mm (معادل 39.37 mil) انتخاب شده است. این مقدار را می‌توانید در زیرمنوی Clearance در بخش Constraints، که زیرمجموعه منوی Electrical است تغییر دهید. پس از تعیین پارامترها، دکمه All Route را فشار می‌دهیم و عملیات مسیردهی به سرعت صورت پذیرفته و گزارش سیم‌نم از نحوه عملکرد آن، قابل مشاهده خواهد بود. در صورتی عملیات routing با موقوفیت انجام شده است که در انتهای گزارش سیستم، تعداد عدم اتصال (failed to connect) و اتصال کوتاه (connections) برابر صفر باشد. همچنین در بخش Routing Layers در گزینه Routing می‌توان لایه‌های مورد استفاده در پروژه را تعیین نمود. در این مرحله عملاً بخش عمده کار به پایان رسیده است. اما یک سری کارهای تکمیلی باقی مانده که در ادامه آن‌ها را فهرست‌وار مرور می‌کنیم.



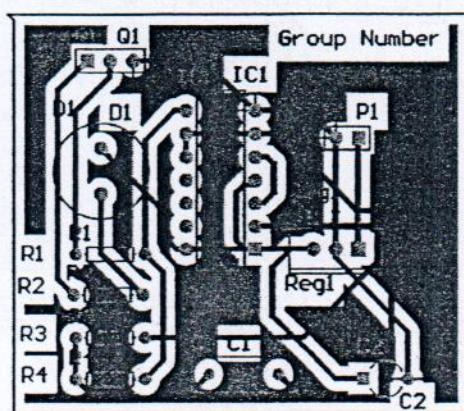
شکل ۹-۳- صفحه PCB مدار چشمکزن

یکی از بخش‌های مهم، نوشتار توضیحات شامل اسم عناصر به کار رفته و مشخصات بورد است. در صورتی که نوشتارها را در Bottom layout قرار می‌دهید، حتماً آنها را mirror کنید تا هنگام تولید PCB، بتوانید به درستی از آنها استفاده کنید. می‌توان برای ضد نویز کردن مدار از لایه‌های فلزی در دو طرف مدار استفاده کرد. در بورد مورد مطالعه، ما صفحه تحتانی را با پوشش فلزی محافظت نمودیم. برای انجام این کار ابتدا باید از قسمت پایین صفحه، Top و Bottom Layer را با توجه به اینکه کدام سمت سیم‌کشی انجام شده است، انتخاب نموده و سپس از منوی Polygon Pour گزینه Place را انتخاب کنیم. در قسمت Net Options و در لیست Connect to Net ضروری است (شکل ۱۱-۳). ضمناً می‌توان صفحه را به صورت یکپارچه و یا مشبک انتخاب کرد.



شکل ۱۰-۳- انتخاب GND در قسمت Net Options

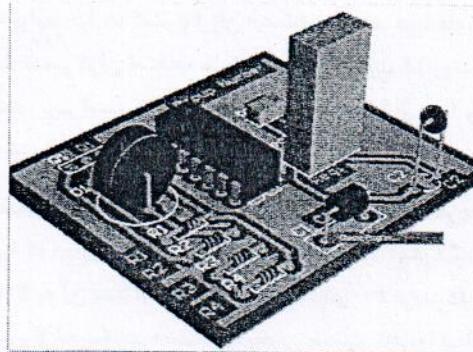
پس از این انتخاب‌ها، دور تا دور محدوده‌ای را که می‌خواهیم صفحه محافظ روی آن قرار بگیرد مشخص کرده و با کلیک راست این عمل به انجام می‌رسد. بورده‌ی که با انجام تمامی مراحل یاد شده حاصل می‌شود، در شکل ۱۱-۳ نمایش داده شده است.



شکل ۱۱-۳- بورد نهایی مدار چشمکزن تولید شده توسط نرم افزار آلتیوم در دولایه

در PCB نیز با Hold کردن قطعه و زدن دکمه Space می‌توان قطعه را چرخاند. میزان چرخش به ازای هر بار زدن دکمه Tools→Component Placement→Auto Placer از قسمت Tools در Preferences می‌توان از نرم‌افزار برای قرار دادن اتوماتیک قطعات، استفاده نمود. نکته قابل توجه دیگر، امکان مشاهده مدار به صورت سه بعدی است. از طریق Tools→Legacy Tools→Legacy 3D View می‌توان قطعات، مورد استفاده و سیم‌کشی انجام شده را مشاهده نمود.

در شکل ۱۱-۲ نمایش سه بعدی مدار طراحی شده مشاهده می‌شود.



شکل ۱۲-۳- مشاهده سه بعدی PCB طراحی شده

در صورتی که در طراحی مدار متوجه مشکلی شده باشیم، ابتدا صفحه‌های محافظت نویز را حذف کرده و سپس از منوی Tools و ردیف Un-Route گزینه All را انتخاب می‌کنیم. به این ترتیب می‌توان تغییرات دلخواه را اعمال کرد. ضمناً چنانچه بخواهیم از طریق فایل شماتیک تغییراتی ایجاد کنیم، می‌توان پس از انجام تغییرات و انجام عملیات Update در صفحات شماتیک و PCB کار را ادامه داد. در پایان کار باید فایل PCB تکمیل شده را با دو فرمت PCB 4.0 Binary File و PCB 3.0 Binary File در فولدری کار ذخیره کرده و برای تولید بورد به مراکز ساخت بورد مدار چاپی تحویل می‌دهیم. قبل از ارسال فایل PCB برای ساخت بورد، کنترل می‌کنیم که PCB طراحی شده شامل تمامی اتصالات بوده و با مدار موجود بر روی شماتیک بطور کامل مطابقت داشته باشد. GND و VCC مربوط به تراشه‌ها نیز خوب است کنترل شوند. مرحله بعد قرار دادن قطعات بر روی بورد مدار چاپی و لحیم‌کاری آنها و نهایتاً تست، رفع ایراد و راهاندازی مدار چشمکزن است. در ادامه با لوازم لحیم‌کاری و نحوه انجام صحیح لحیم‌کاری و نیز با روش تست اتصالات آشنا خواهیم شد.

۴- جلسه دوم

هدف:

• آشنایی با طرز لحیم‌کاری

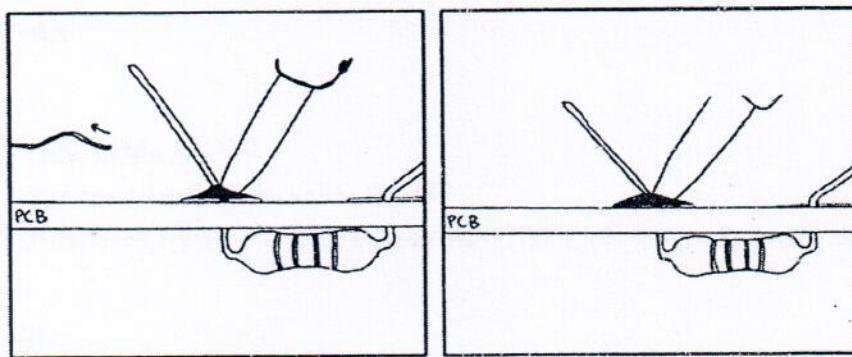
لحیم‌کاری توسط وسیله‌ای بنام هویه انجام می‌شود که در دو نوع قلمی و تفنگی قابل تهیه است. هویه‌های قلمی معمولاً دارای توان مصرفی کمتر و لذا قادر به تولید حرارت کمتر و ارزان قیمت‌تر هستند. هویه‌های تفنگی معمولاً دارای توان مصرفی (وات) بالا و برای کارهای حرفه‌ای مثلاً توسط تعمیرکاران استفاده می‌شوند. برای مدارات الکترونیکی و دیجیتال حاوی قطعات حساس که با حرارت زیاد ممکن است آسیب بینند، بهتر است که از هویه‌ای قلمی با وات ۱۵ الی ۳۰ وات استفاده شود. هویه‌ها را برای اجتناب از سوختن وسائل مجاور مانند سطح میز، فرش، سیم و مانند آن در درون پایه هویه قرار می‌دهند.

لحیم‌کاری علاوه بر هویه به لحیم که آلیاژی از قلع و سرب است نیز نیاز دارد. هرچه قلع لحیم بیشتر باشد کیفیت آن بهتر و لحیم‌کاری راحت‌تر انجام می‌شود چرا که دمای ذوب لحیم پایین‌تر و لذا قطعات کمتر آسیب می‌بینند و لحیم حاصله برآق تر خواهد بود. بطور مثال لحیم با مشخصه ۴۰/۶۰ به آن معنا است که آن لحیم دارای ۶۰ درصد قلع و ۴۰ درصد سرب می‌باشد. سرب یک ماده سمی است. بنابراین بهتر است پس از لحیم‌کاری دست‌ها را بخوبی شست. تجربه، در انجام بهتر لحیم‌کاری نقش بسزایی دارد. لحیم‌کاری موجب اتصال دو قطعه فلزی (مثل اتصال پایه قطعات الکترونیکی و دیجیتال به خطوط مدار چاپی) به کمک ماده لحیم می‌گردد. ماده لحیم پس از ذوب شدن توسط هویه، دو قطعه را بهم متصل می‌نماید. پس از سرد شدن لحیم، این اتصال برقرار می‌ماند.

سطح خطوط مدارچاپی و پایه قطعات همواره در معرض ترکیب با اکسیژن هوا، کثیفی و برخورد با مواد چرب می‌باشند. حرارت بالای ایجاد شده در زمان لحیم‌کاری نیز موجب اکسیده شدن بیشتر این سطوح می‌شود. روغن لحیم ماده‌ای است که چنانچه سطحی که قرار است لحیم گردد بدان آغشته شود، کلوفون موجود در آن موجب حل نمودن و کنار زدن مواد زائد شده و اتصال بهتر لحیم به قطعات می‌گردد. روغن لحیم پس از حرارت دیدن دودی اینجاد می‌کند که بهتر است از آن اجتناب نمود. امروزه اغلب سیمه‌های لحیم، در مغزی خود دارای روغن لحیم نیز می‌باشند. پس از لحیم‌کاری و سرد شدن محل لحیم‌کاری، در صورت تمایل می‌توان به کمک حالاتی مانند بنزین یا تینر و یک برس مثل مسوک، روغن‌های لحیم باقیمانده را شستشو داده و تمیز نمود.

سطح هویه بدلیل حرارت بالا با اکسیژن هوا ترکیب شده و بدلیل اکسیده شدن، سیاه می‌شود. به همین دلیل قبل از هر عمل لحیم‌کاری نوک هویه را با اسفنج مرتبط تماس می‌دهند تا مواد اکسید شده جدا شوند و سپس با اتصال سیم لحیم به نوک هویه، نوک هویه برآق شده و برای لحیم‌کاری آماده می‌شود. در زمان لحیم‌کاری معمولاً بور德 مدارچاپی را توسط گیره‌هایی نگه می‌دارند تا کار لحیم‌کاری تسهیل شود.

برای انجام لحیم‌کاری، پس از گرم شده هویه و روغن‌کاری محل لحیم، نوک هویه را با اسفنج خیس تمیز کرده و با ذوب کمی لحیم بر روی نوک هویه، نوک هویه را برآق و برای لحیم‌کاری آماده می‌کنیم. پس از آن نوک هویه را با محل لحیم‌کاری تماس می‌دهیم تا آن محل گرم شود. سپس نوک سیم لحیم را در مجاورت محل لحیم‌کاری، با نوک هویه تماس داده و نوک هویه را بر روی سیم لحیم فشار می‌دهیم. اینکار موجب می‌شود که لحیم ذوب شده و در محل لحیم‌کاری که به اندازه کافی گرم شده، جاری شده و قطعات را به یکدیگر متصل نماید. از نگه داشتن بیش از حد نوک هویه در محل لحیم‌کاری باید اجتناب نمود، چرا که اینکار موجب انتقال بیش از حد حرارت به قطعه الکترونیکی از طریق پایه‌های آن و در نتیجه آسیب دیدن قطعه می‌گردد. یک اتصال خوب پس از لحیم‌کاری صاف، یکدست و مقعر است. زمان نگه داشتن هویه در محل اتصال به وسعت سطح لحیم‌کاری، جنس لحیم، توان مصرفی هویه (برحسب وات) بستگی دارد.

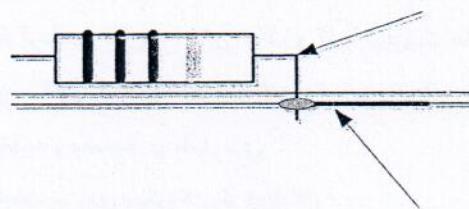


شکل ۱-۴- روش لحیم کاری

بر عکس عمل لحیم کاری، عمل لحیم زدایی در موقعی چون تعمیر مدارات الکترونیکی به منظور تعویض قطعه آسیب دیده و یا تعویض یک قطعه در زمان طراحی لازم می شود. لحیم زدایی توسط مکنده لحیم یا توسط فتیله لحیم انجام می شود. مکنده لحیم وسیله‌ای است که پس از گرم کردن لحیم توسط هویه، با مکش خود، لحیم را از محل لحیم کاری جدا می کند. فتیله لحیم نیز عملاً یک مس باقته شده است که پس از آنکه محل لحیم را به اندازه کافی گرم کردیم، اتصال فتیله لحیم به لحیم ذوب شده موجب جذب لحیم به فتیله و برداشته شدن لحیم از سطح لحیم می شود.

۱-۴- تست اتصالات

پس از دریافت بورد و لحیم کاری قطعات، برای اطمینان از لحیم کاری صحیح، می توان با انجام تست اهمی (یا تست بوق) از اتصال کامل قطعات به بورد مدار چاپی بعد از لحیم کاری مطمئن گردید. برای این منظور همانند شکل ۲-۴، یک سر مولتی‌متر را بر روی پایه قطعه یا پایه تراشه و سر دیگر آن را بر روی مدار متصل به پایه بر روی بورد قرار می دهیم. چنانچه مقاومت صفر بود و یا بوق اتصال شنیده شد، می توان از صحت لحیم کاری مطمئن گردید. خوب است که در صورت امکان قطعات را قبل از نصب و لحیم کاری تست و از صحت کار کرد آنها مطمئن شد.



شکل ۲-۴- کنترل صحت اتصال پس از لحیم کاری

۵- جلسه سوم

هدف:

- معرفی پروگرامر STK500
- نحوه نصب درایور USB برای پروگرامر STK500
- معرفی اسمبلر و کامپایلرهای میکروکنترلرهای خانواده AVR و برنامهنویسی میکروکنترلر از طریق آنها

لوازم مورد نیاز:

- پروگرامر برد آزمایشگاهی AVR
- نرمافزار USB Driver (NUS112v7.1 Driver Setup-v1.8.0)
- نرمافزارهای اسمبلر و کامپایلر مانند WINAVR و CodeVision ، AVRStudio6 ، AVR Studio4

۱-۵- معرفی پروگرامر STK500

برای برد آموزشی پروگرامر USB مدل STK500 ساخت شرکت نوآوران الکترونیک تهیه گردیده است. این پروگرامر به وسیله کابل USB به کامپیوتر وصل می‌گردد. با وصل آن پس از چند ثانیه کامپیوتر سخت‌افزار جدید را شناسایی می‌کند. برد پروگرامر STK500 ساخت شرکت نوآوران الکترونیک در شکل ۱-۵ مشاهده می‌شود.

قابلیت‌های این پروگرامر به شرح زیر است:

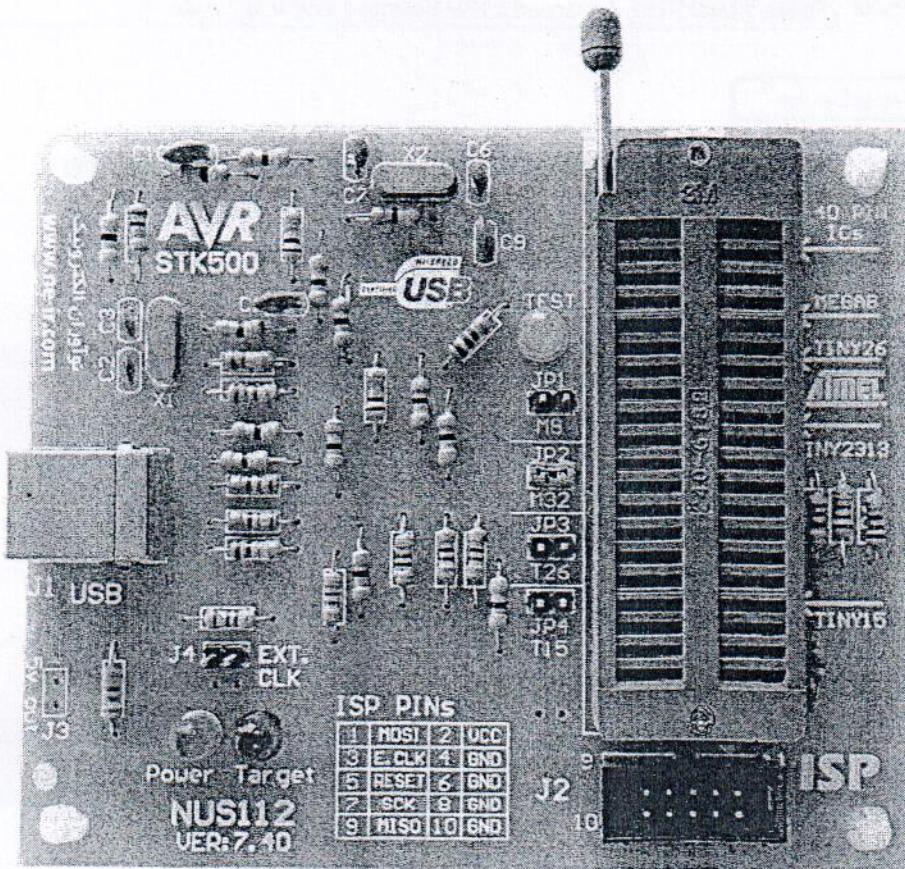
- برنامه‌ریزی تمامی میکروکنترلرهای AVR بر روی ZIF سوکت (ATmega , AT90 , ATtiny Series)
- سازگاری کامل با نرمافزارهای AVR Bascom AVR و AVR Studio و CodeVision AVR
- ارتباط از طریق پورت USB2.0 و USB1.1
- امکان تغییر و بازگرداندن فیوزبیت‌های کلاک و کلیه فیوزبیت‌ها (غیر از فیوزبیت SPIEN)
- مجهز به کانکتور و کابل رابط ISP (In System Programming)
- نمایشگر ولتاژ تنذیه و وضعیت بر نامه‌ریزی
- برنامه‌ریزی اطلاعات بر روی میکروکنترلر (Write)
- خواندن اطلاعات از روی میکروکنترلر (Read)
- مقایسه اطلاعات برنامه‌ریزی شده با فایل اصلی (Verify)
- محرومانه کردن اطلاعات برنامه‌ریزی شده (Lock bit)
- قابلیت پروگرام کردن کلیه فیوزبیت‌های کلاک
- سازگاری با تمامی کامپیوترهای PC و Laptop

• سازگاری با تمامی ویندوزهای ۳۲ بیتی و ۶۴ بیتی (WIN10, WIN8.1, WIN8, WIN7, WIN VISTA, ...)

(WIN XP, WIN 2000)

• عدم نیاز به منبع تغذیه جداگانه (تغذیه از طریق پورت USB)

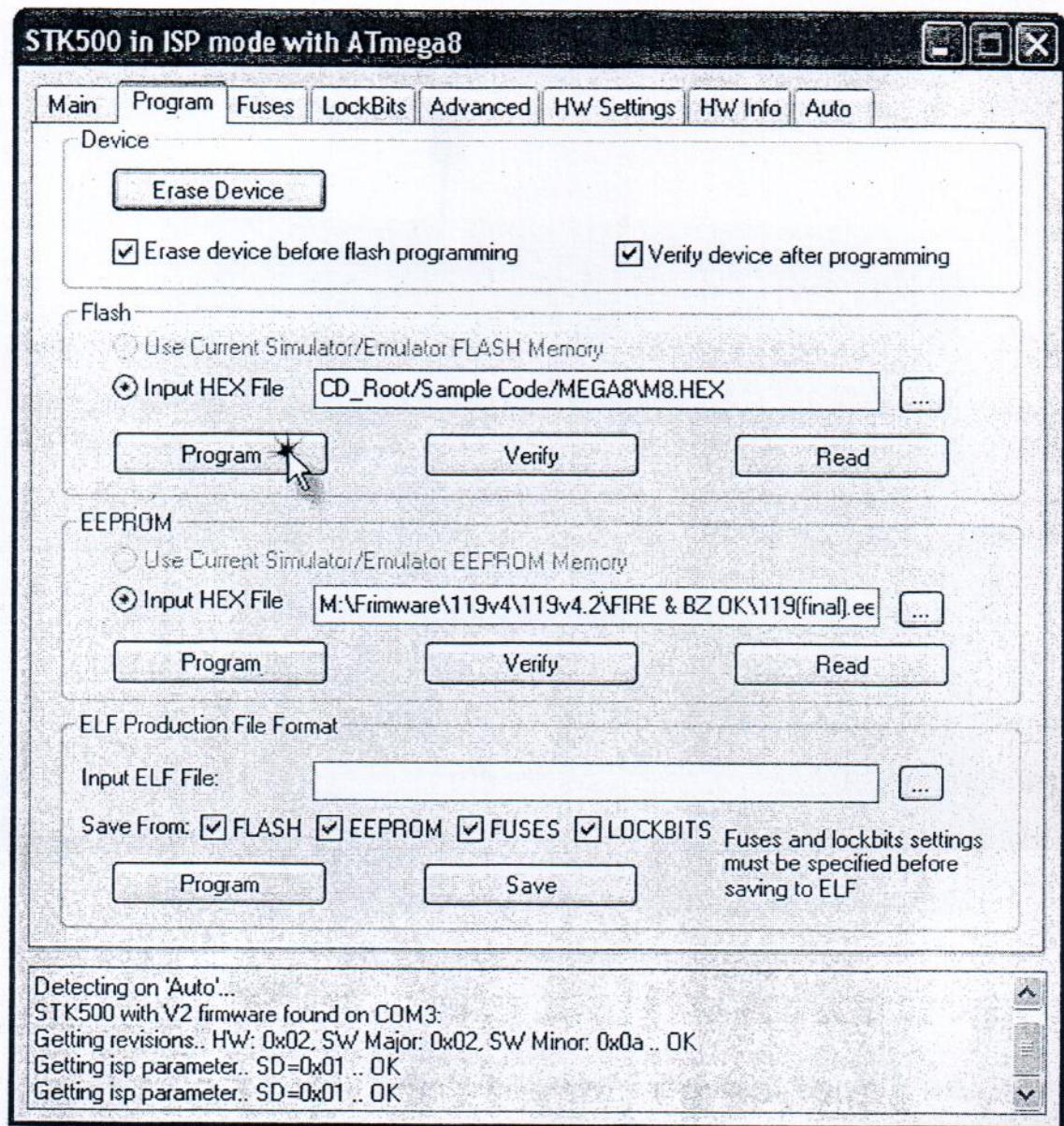
• به همراه کابل رابط USB



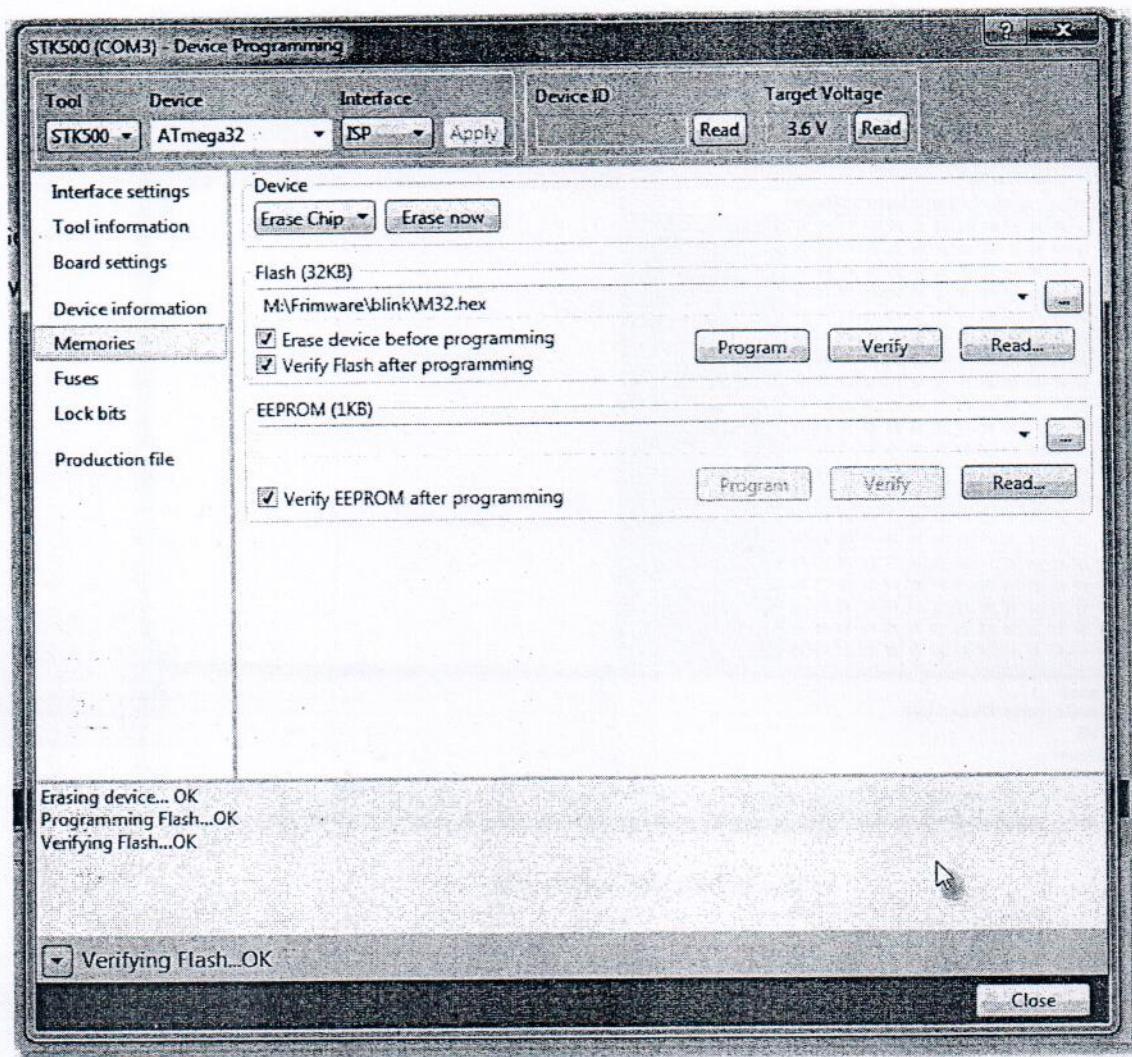
شکل ۱-۵- پروگرامر STK500 ساخت شرکت نوآوران الکترونیک

این پروگرامر توانایی کار در نرمافزارهای AVR Studio4 و AVR Studio6، بسکام (Bascom AVR) و کدویژن (Codevision AVR) را در همه نسخه‌ها، دارا می‌باشد. این توانایی به شما این امکان را می‌دهد که بعد از نوشتن برنامه و کامپایل آن، بدون هیچ وقفه‌ای و بدون اینکه بخواهد نرمافزار دیگری را باز نمایید، بالاصله دکمه Program را زده و نتیجه را در میکروکنترلر خود مشاهده کنید. این کار سرعت کار شما را بسیار افزایش می‌دهد و شما دغدغه کار با یک نرمافزار ناآشنای دیگر را نخواهید داشت. شما می‌توانید از نرمافزارهای AVR Studio4 و AVR Studio6، بسکام (Bascom AVR) و کدویژن (Codevision AVR) در زمان دسترسی به برد پروگرامر STK500 را به ترتیب در شکل ۲-۵ الی شکل ۵-۵ مشاهده می‌نمایید.

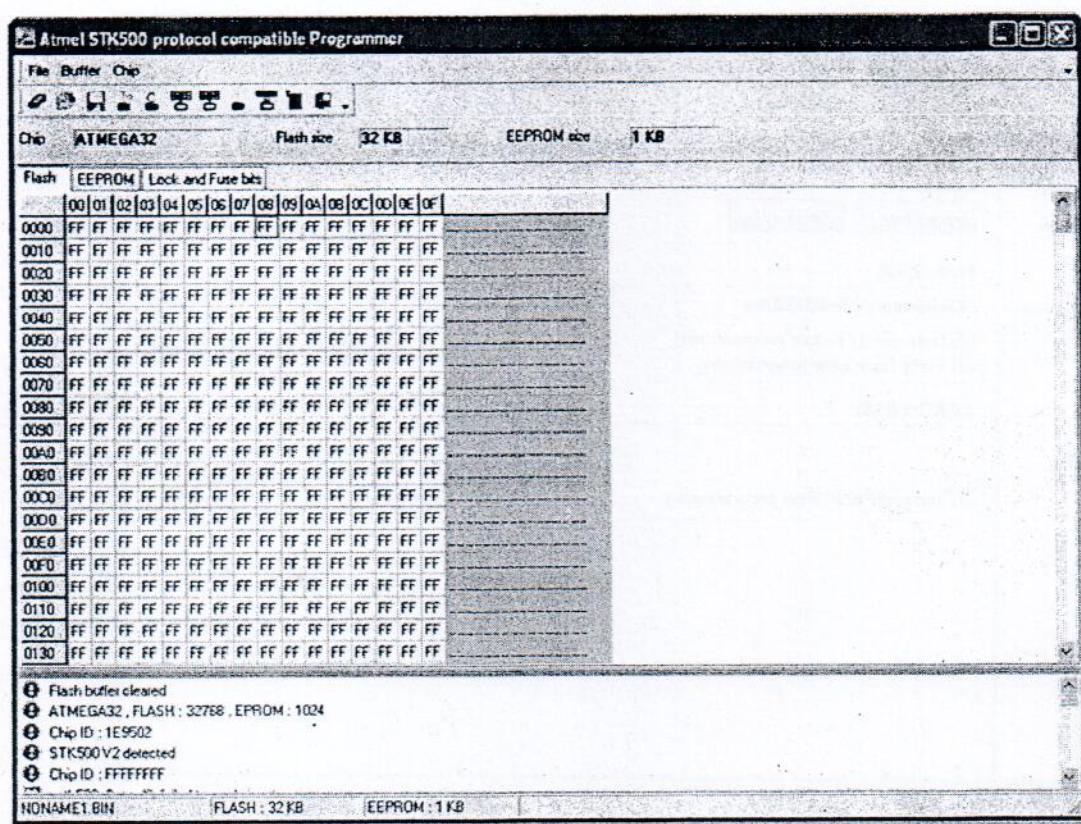
این پروگرامر اکثر میکروکنترلرهای AVR مدل Tiny و Mega از جمله میکروکنترلرهای ATmega16، ATmega32 و ATmega128 را پشتیبانی می‌نماید.



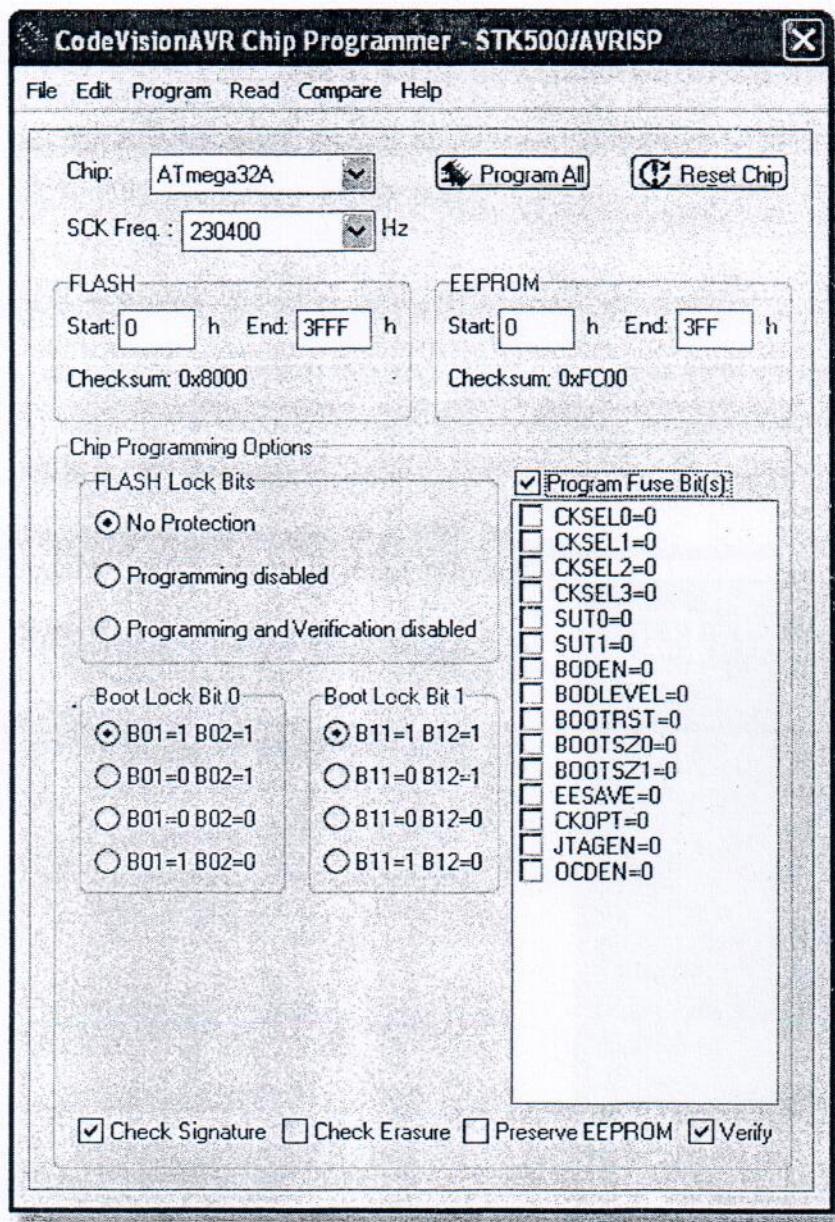
شکل ۲-۵- نرم افزار ۴



شكل ٣-٥ - نرم افزار 6 AVR Studio



شکل ۴-۵ - نرم افزار Bascom AVR



شکل ۵-۵ - نرم افزار CodeVision AVR

۲-۵- نحوه نصب درایور USB پروگرامر STK500

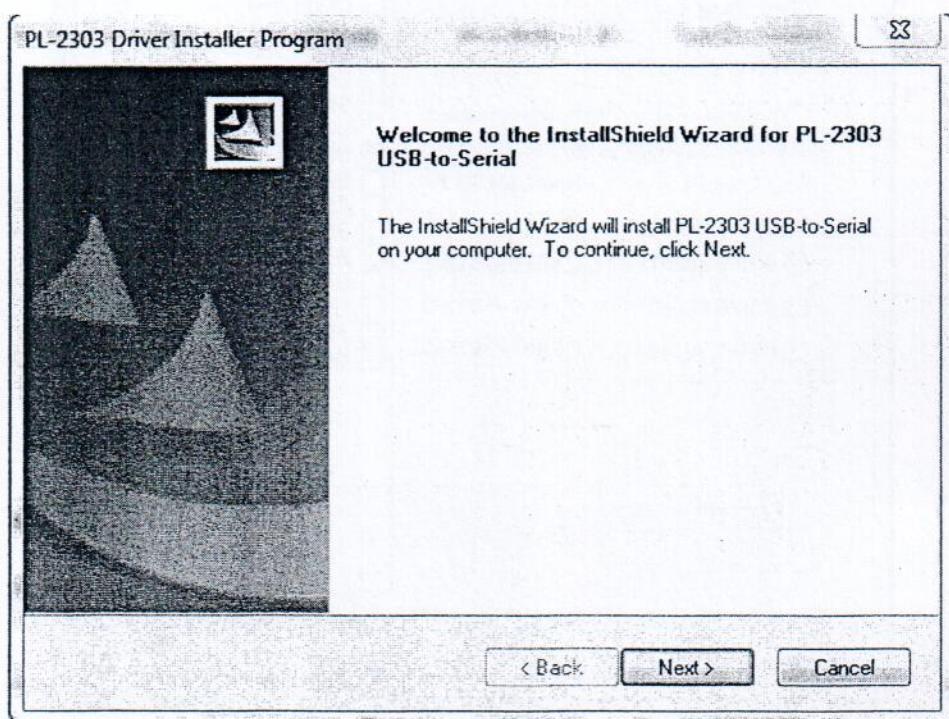
مراحل نصب و کار با پروگرامر STK500 :

۱ - نصب درایور پروگرامر (فقط بار اول)

- ۲- نصب نرم افزار پروگرامر (فقط بار اول)
- ۳- تنظیم نوع پروگرامر در نرم افزار آن (فقط بار اول)
- ۴- رفتن به بخش پروگرام نرم افزار
- ۵- انجام تنظیمات نوع میکرو کنترلر و فیوز بیت های آن
- ۶- برنامه ریزی میکرو کنترلر

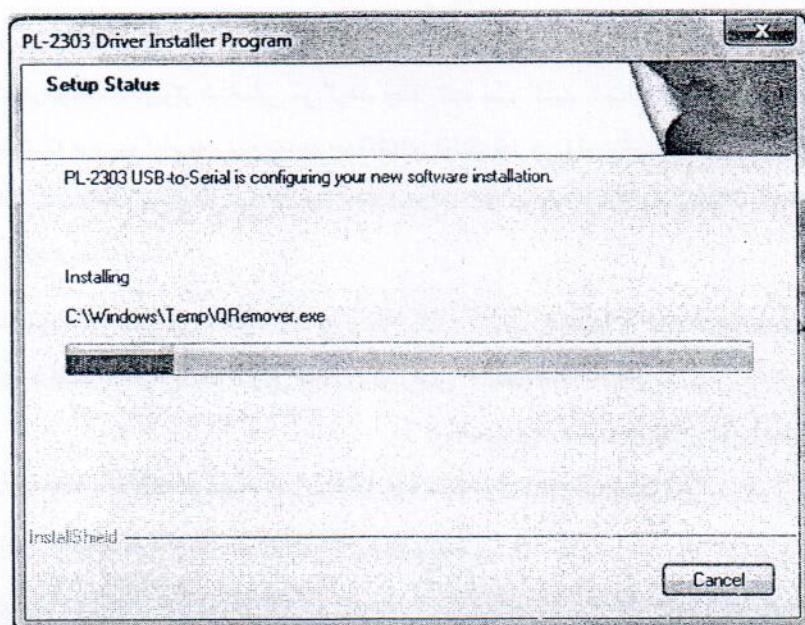
برای نصب درایور USB برای پروگرامر در تمامی ویندوز ها اقدامات زیر را انجام دهید:

در این مرحله کابل USB را به پروگرامر متصل نکنید. فایل NUS112v7.1 Driver Setup-v1.8.0 را از فolder Driver در سی دی محصول پروگرامر ساخت شرکت نوآوران الکترونیک اجرا کنید. با اجرای این فایل پنجره ای مطابق شکل زیر باز خواهد شد. حال بر روی Next کلیک کنید.



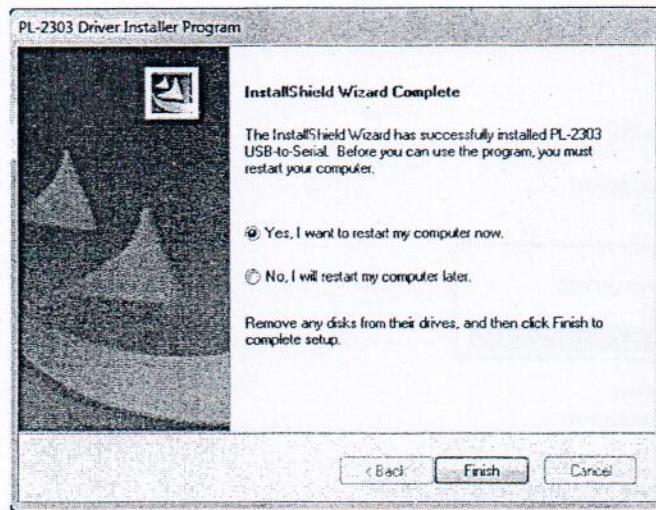
شکل ۵-۶- شروع نصب درایور USB پروگرامر

در این مرحله سیستم در حال کپی کردن فایل های مورد نیاز برای درایور پروگرامر است. متنظر باشد تا این مرحله سپری شود.



شکل ۵-۷- روند نصب درایور USB پروگرام

روی دکمه Finish کلیک و سپس کامپیوتر را Restart نمایید و به مرحله بعد بروید.



شکل ۵-۸- خاتمه نصب درایور USB پروگرام

در صورتی که در حین کار به مشکل برخوردید یا به پیغام خطایی برخوردید به لینک زیر مراجعه کنید:

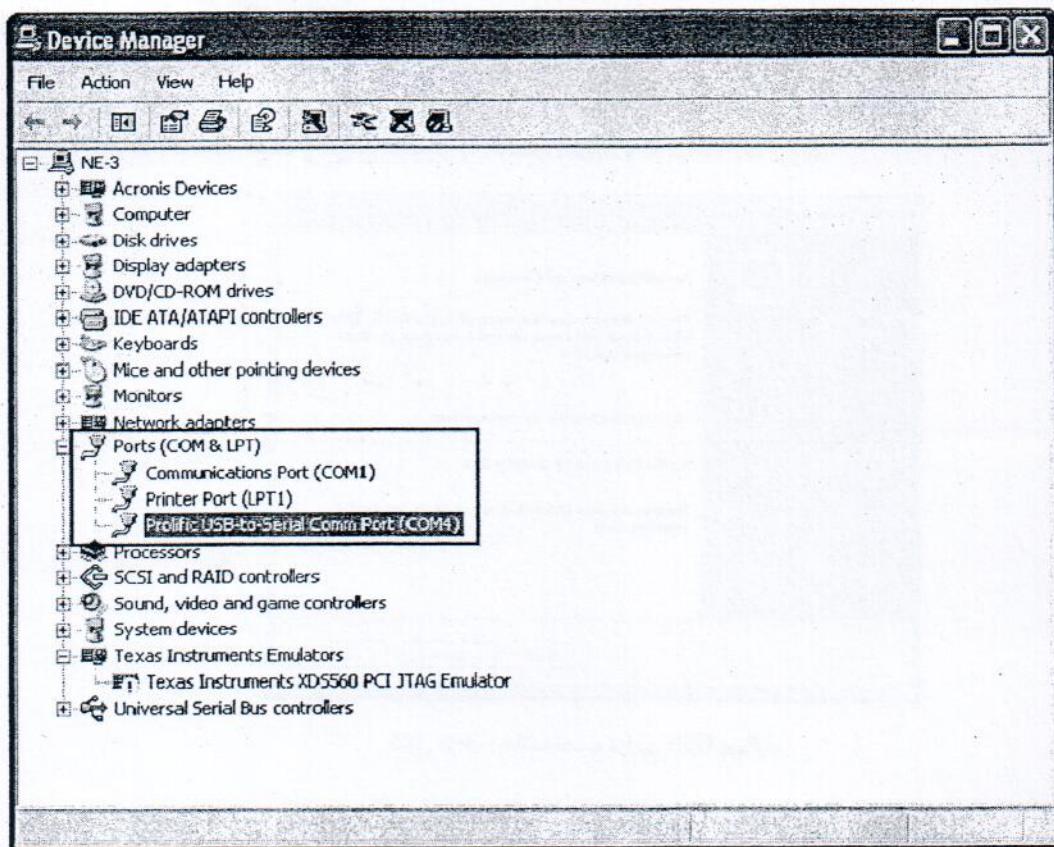
<http://www.ne-ir.com/>

در این مرحله کابل USB به پروگرام و کامپیوتر خود متصل کنید. بعد از اتصال کابل USB به کامپیوتر برای اولین بار، برد پروگرام توسط کامپیوتر شناسایی می‌گردد. چند ثانیه صبر کنید تا سخت‌افزار توسط ویندوز شما شناسایی شود. لازم به ذکر است که برد پروگرام به منبع تغذیه جداگانه‌ای (ولتاژ خارجی) احتیاج ندارد و تغذیه خود را از روی پورت USB تأمین می‌نماید. با مشاهده پیغام‌های *Your new Hardware is installed and ready to use* و *Found New Hardware*، مراحل نصب به خوبی انجام شده است.

با اتمام موفق این مراحل دستگاه شما به عنوان یک پورت سریال مجازی (Virtual Com Port) شناخته می‌شود. به منظور مشاهده و تغییر شماره پورت سریال مجازی در بخش Device Manager از مسیر زیر کار را دنبال کنید.

Control Panel/System/Hardware/Device Manager

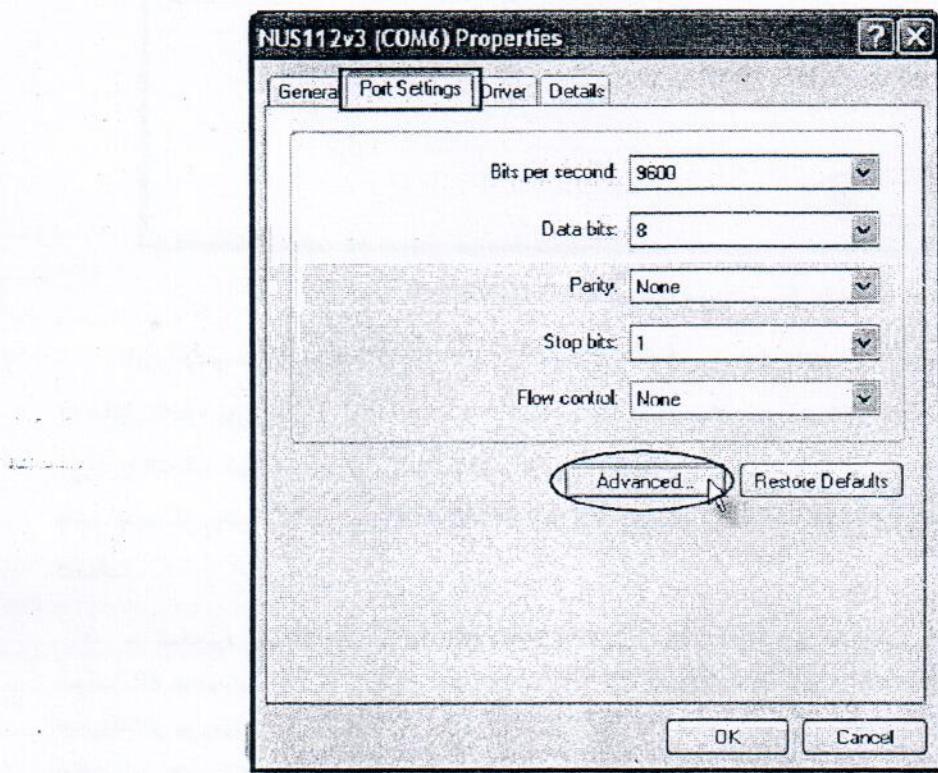
در بخش (COM & LPT) نام نشان‌دهنده معرفی درست دستگاه به کامپیوتر و نشان‌دهنده شماره پورت سریال (مجازی) می‌باشد.



شکل ۹-۵- تخصیص پورت سریال به پورت USB برای اتصال به پروگرام

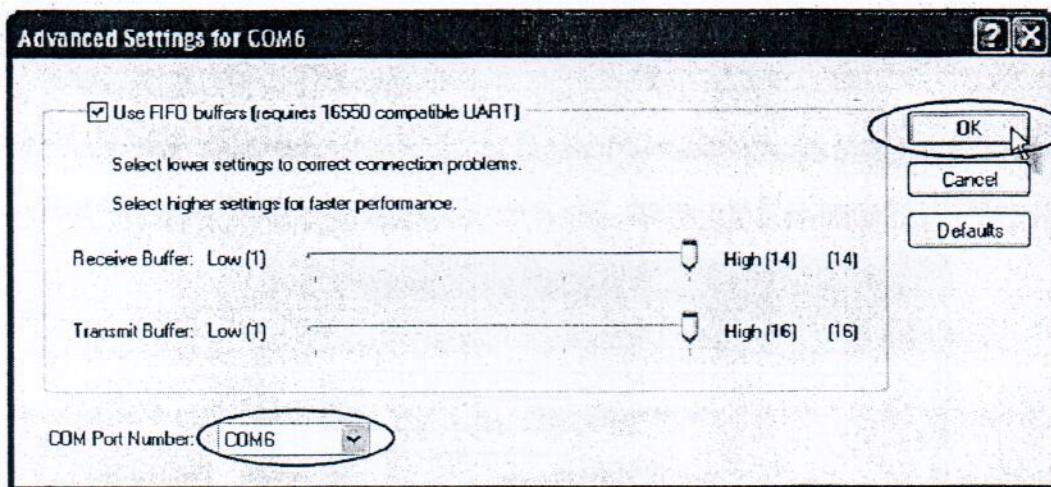
امکان دارد شماره پورت سریال (مجازی) شما از عدد ۹ تجاوز نماید. به علت اینکه این موضوع در نرم افزارهای پروگرام پیش بینی نشده است شما باید با اعمال تغییراتی شماره آن را دلخواه انتخاب کنید.

در بخش (Prolific USB-to-Serial Comm Port (COM X) Ports (COM&LPT) بر روی نام Properties کلیک راست نموده و گزینه Port Settings را انتخاب نمایید. از منوی باز شده در بخش Advanced گزینه Port Settings را انتخاب نمایید.



شکل ۱۰-۵ - تغییر شماره پورت سریال

در این منو در بخش Com Port Number شماره پورت سریال (مجازی) خود را بین عدد ۴ تا ۹ به طور دلخواه انتخاب نموده و بر روی کلید OK فشار دهید و پیغام ظاهر شده را تائید نمایید و پنجه Properties را بیندید. دستگاه پروگرام را از USB جدا نموده و مجدداً وصل نمایید. حال شماره پورت سریال (مجازی) شما به عدد دلخواه شما تغییر کرده است.



شکل ۱۱-۵ - تغییر شماره پورت سریال (ادامه)

این امر در صورتی مقدور می‌باشد که دستگاه دیگری از آن پورت استفاده ننماید. در صورتی که عبارت (is used) در مقابل شماره پورت سریال نوشته شده بود و شما مطمئن هستید که هیچ دستگاهی از آن استفاده نمی‌کند، آن شماره پورت را انتخاب کنید. در صورتی که شماره پورت سریال (مجازی) شناخته شده توسط دستگاه شما COM1 و COM2 باشد، به وسیله روش بالا آن را بین COM3 تا COM9 قرار دهید. دو پورت COM1 و COM2 معمولاً توسط کامپیوتر رزرو شده‌اند.

به کمک برد پروگرام می‌توان یک میکروکنترلر را به دو صورت پروگرام کرد: یکی با قرار دادن میکروکنترلر مورد نظر در سوکت ZIF موجود بر روی برد پروگرام و دیگری به روش In System Programming میکروکنترلر بر روی برد اصلی خود قرار داشته و با اتصال یک کابل مخصوص بین پروگرامر (کانکتور J2) و کانکتور مخصوص برنامه‌ریزی روی برد میکروکنترلر، می‌توان برنامه را بدون جدا کردن میکروکنترلر از برد اصلی به حافظه فلاش میکروکنترلر منتقل نمود. در هنگام قرار گیری میکروکنترلر در سوکت ZIF و یا اتصال ISP به برد پروگرام این LED روشن خواهد شد.

۵-۳- چند نکته مهم در زمان برنامه‌ریزی میکروکنترلرهای AVR

برای برنامه‌ریزی میکروکنترلر خود ابتدا مراحل زیر را با دقت انجام داده و سپس میکروکنترلر خود را برنامه‌ریزی نمایید.

- ۱ - برد پروگرام را از طریق کابل USB به پورت USB کامپیوتر وصل کنید.
- ۲ - در روش استفاده از سوکت ZIF، آی‌اسی میکروکنترلر مورد نظر خود را داخل سوکت ZIF قرار دهید.
- ۳ - یا در روش برنامه‌ریزی به روش ISP، برد پروگرام را از طریق کابل متصل به کانکتور J2 به کانکتور programming موجود بر روی برد حاوی میکروکنترلر متصل نمائید.
- ۴ - حال با نرم‌افزار پروگرام که در بخش قبل توضیح داده شد، برنامه‌ریزی را انجام دهید.

اگر در هنگام پروگرام کردن برنامه پروگرامر بیغام خطای را نشان داد، موارد زیر را دوباره بررسی نمایید:

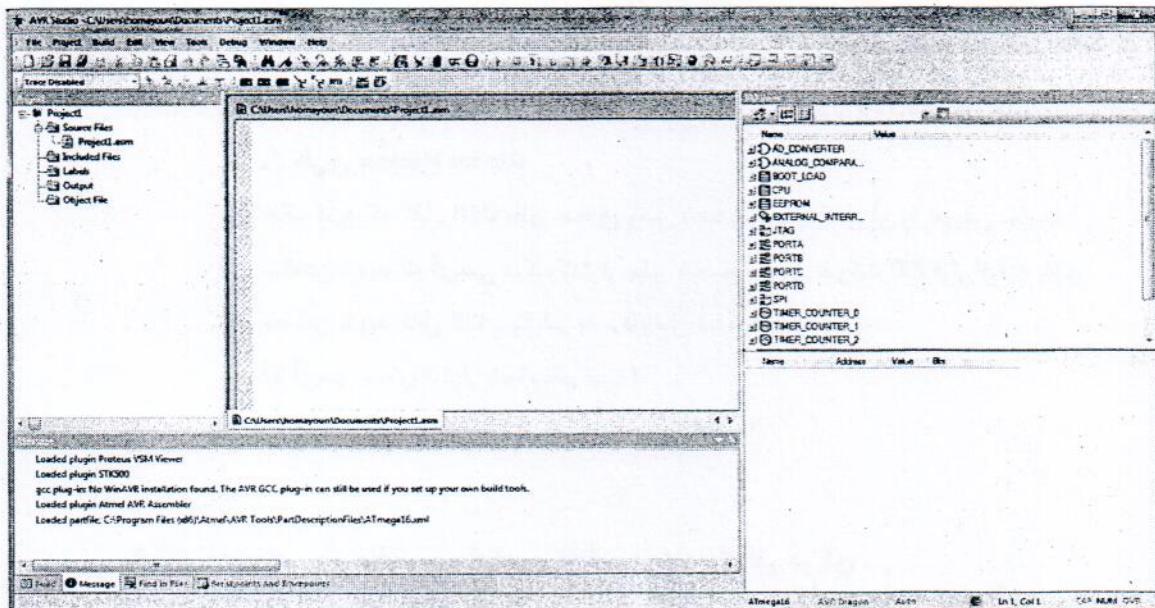
- ۱- مطمئن شوید که درایور پروگرامر نصب شده باشد و سختافزار پروگرامر برای کامپیوتر شناسایی شده باشد
- (از طریق Device Manager)
- ۲- چک کنید که کابل USB بطور صحیح وصل شده باشد و یا کابل آن را تعویض نمایید.
- ۳- مطمئن شوید که آی‌سی میکروکنترلر بطور صحیح بر روی سوکت ZIF قرار گرفته باشد.
- ۴- مطمئن شوید کابل ISP پروگرامر به برد وصل شده باشد.
- ۵- آیا آی‌سی میکروکنترلر شما سالم است؟
- ۶- آیا فیوزبیت‌های میکروکنترلر را خود، تغییر نداده‌اید؟

۴-۴- نحوه نصب نرمافزار پروگرامر و تنظیم نوع پروگرامر در آن

این پروگرامر توانایی کار با نرمافزارهای Bascom AVR و AVR Studio و Code Vision و AVR Studio را دارد. به این نرمافزارها می‌توانید از طریق CD محصول پروگرامر یا اینترنت دسترسی یافته و نصب نمایید. بعد از انجام مراحل فوق دستگاه آماده استفاده می‌باشد. در بخش‌های زیر استفاده از بعضی از نرمافزار پرکاربرد فوق که پروگرامر STK500 را نیز پشتیبانی می‌کنند معرفی و نحوه پروگرام کردن میکروکنترلر در زمان بیان هر یک از این نرمافزارها بیان شده است.

۴-۵- معرفی AVR Studio 4

- به منظور برنامه‌نویسی اسمبلی برای میکروکنترلرهای خانواده AVR می‌توان از اسمبلرهای مختلفی استفاده نمود. یکی از این اسمبلرها AVR Studio نام دارد. این نرمافزار برای نوشتن و دیباگ کردن برنامه اسمبلی برای میکروکنترلرهای خانواده AVR بکار می‌رود. محیط نرمافزار AVR Studio 4 در شکل زیر نشان داده شده است.



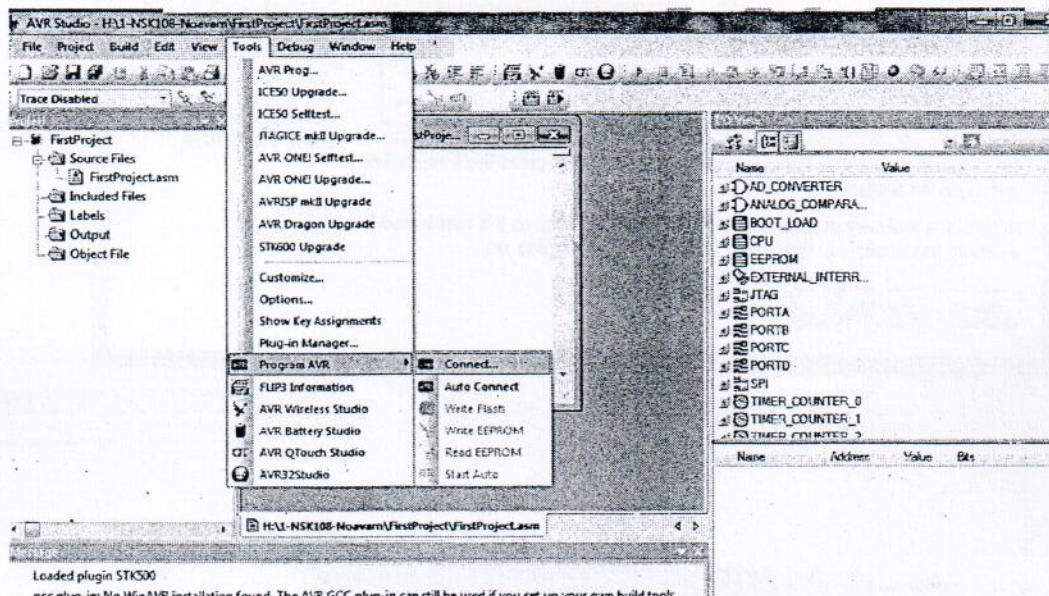
شکل ۱۲-۵ - محیط نرم افزار Visual Studio

توسط این نرم افزار می توان کدهای کامپایل شده مثل کدهای COFF تولید شده توسط کدویژن^۱ را نیز دیباگ کرد. بعضی از قابلیت های این نرم افزار عبارتند از:

- می توان یک پروژه جدید ایجاد نمود. در این صورت می توان یک فایل با پسوند .asm. با هر نامی ایجاد کرد. با تیک زدن Create Initial File، فایل اسembلی با همان نام پروژه ایجاد می شود.
- می توان فایل های از پیش نوشته شده و حتی فایل های COFF تولید شده توسط سایر نرم افزارها را باز کرد.
- می توان مسیر پروژه را مشخص نمود و یا با تیک زدن Create Folder، فولدری با همان نام پروژه در آن مسیر ایجاد نمود.
- با فشردن کلید Next، می توان برنامه را دیباگ یا آنرا شبیه سازی کرد.
- می توان برنامه اسembلی را کامپایل و خطاهای آنرا شناسایی و رفع کرد.
- در صورت انتخاب امکان دیباگ کردن، باید میکروکنترلر مورد نظر و سکوی دیباگ (مثل JTAG ICE AVR Simulator و ...) را انتخاب کرد.
- در صورت انتخاب شبیه ساز AVR Simulator، این شبیه ساز بطور نرم افزاری برنامه شما را دیباگ می کند و قادر به شبیه سازی تمامی میکروکنترلرهای AVR می باشد. بقیه شبیه سازها سخت افزاری و اصطلاحاً Emulator می باشند و هر کدام تعداد خاصی از میکروکنترلرها را پشتیبانی می کنند. در صورت داشتن JTAG ICE می توانید JTAG ICE را انتخاب کنید این یک سخت افزار است که توسط آن می توان برنامه را داخل میکروکنترلر خط به خط اجرا نمود و به کمک آن تمام متغیرها و محتوای ثبات ها را مشاهده کرد. قبل از شبیه سازی باید یکبار برنامه کامپایل شود.

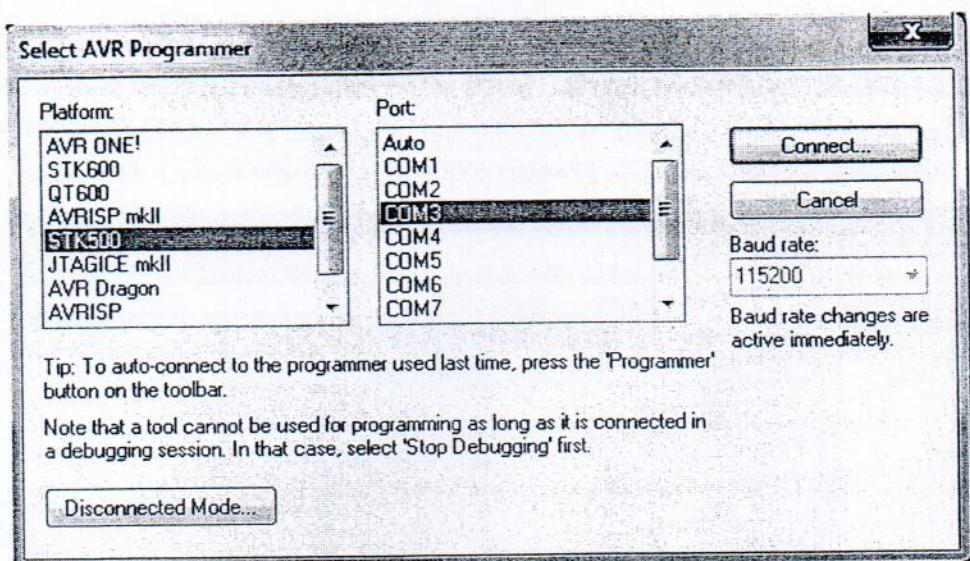
¹ CodeVision

برای برنامه‌ریزی میکروکنترلرهای AVR در محیط نرم‌افزار 4 AVR Studio، باید کابل رابط را به درگاه USB کامپیوتر خود وصل کنید و سر دیگر کابل را به مدار پروگرامر (کانکتور J1) وصل کنید. در این صورت LED سبز زنگ باید روشن شود. همانطور که گفته شد میکروکنترلر را به دو روش قرار دادن روی سوکت ZIF برد پروگرامر و یا به روش ISP می‌توان پروگرام نمود. از منوی Tools در برنامه 4 AVR Studio، گزینه Connect و سپس گزینه Program AVR را انتخاب نمایید.



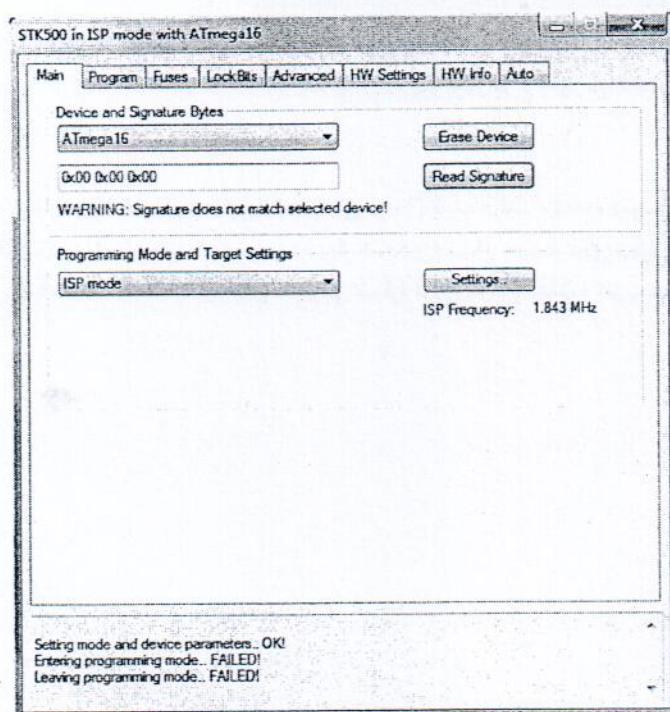
شکل ۱۳-۵ - فرآیند اتصال به پروگرامر در نرم‌افزار 4 Visual Studio

در این منو شماره پورت سریال (مجازی) که در طی مراحل نصب درایور مشاهده نموده‌اید را انتخاب نموده و سپس بر روی گزینه Connect کلیک کنید. امکان دارد شماره پورت سریال (مجازی) شما از عدد ۹ تجاوز نماید که چون این موضوع در نرم‌افزارهای پروگرامر پیش‌بینی نشده است، شما باید با اعمال تغییراتی شماره آن را به نحو مناسب تغییر دهید تا بین ۳ تا ۹ باشد.



شکل ۱۴-۵ - انتخاب پروگرامر و پورت مربوطه

با بازشدن صفحه زیر، نرمافزار به مدار پروگرامر متصل شده است.



شکل ۱۵-۵ - منوی Main در نرمافزار AVR Studio

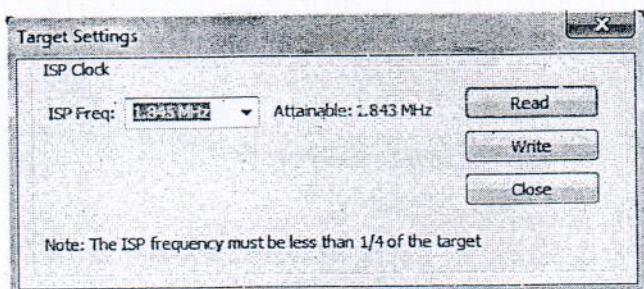
۱-۵-۵- تنظیمات بخش‌های پروگرامر نرم‌افزار AVR Studio

تنظیم بخش (سرعت برنامه‌ریزی) Programming Mode and Target Settings

در صورتی که میکروکنترلر را تازه خریداری کرده‌اید، تنظیمات فیوزبیت ساعت این میکروکنترلر به روش استفاده از مدار RC داخلی کالیبره شده با فرکانس 1MHz می‌باشد. در بخش Programming Mode Target Settings در شکل قبلی گزینه ISP را انتخاب نمائید.

تنظیمات گزینه Settings

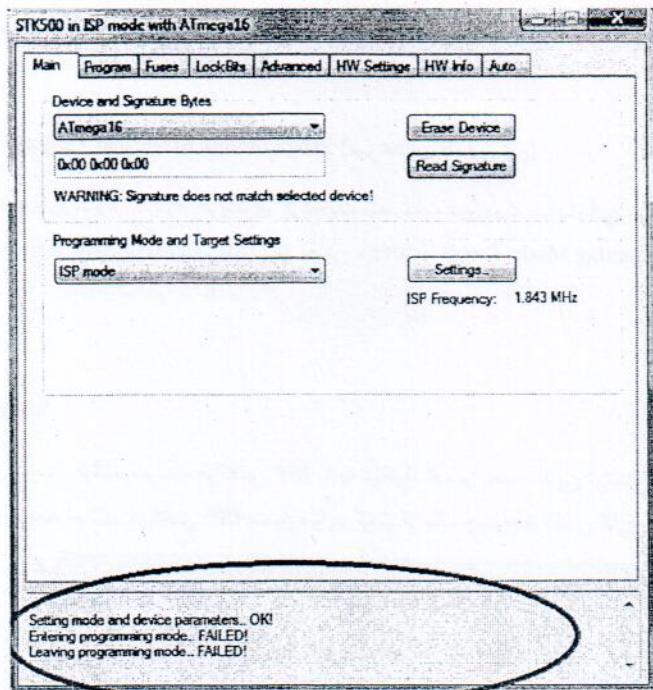
به کمک این منو می‌توانید سرعت فرکانس ISP را به دلخواه تغییر دهید. بدین ترتیب که عدد مورد نظر را انتخاب نموده و بر روی دکمه write کلیک نمائید. فرکانس ISP همواره باید کمتر از یک چهارم فرکانس کاری میکروکنترلر باشد.



شکل ۱۶-۵- تنظیم ساعت ISP

Main منوی

به کمک این منو امکان انتخاب میکروکنترلر، پاک کردن حافظه برنامه، خواندن اطلاعات میکروکنترلر میکروکنترلر فراهم می‌باشد.



محل نمایش پیغام ها

شکل ۵-۱۷-۵ - انتخاب نوع میکروکنترلر و مود برنامه ریزی در منوی Main

Eraser Device: این گزینه حافظه داخلی میکروکنترلر را بطور کامل پاک می کند.

Read Signature: این گزینه مشخصات آی سی میکروکنترلر موردنظر را از داخل تراشه میکروکنترلر می خواند.

با زدن دکمه Read Signature اطلاعات شماره آی سی توسط دستگاه پروگرامر خوانده شده و در کادر روپرتوی آن نمایش داده می شود. در صورتی که شماره آی سی انتخاب شده با اطلاعات خوانده شده برابر باشد، پیغام Sogniture matches selected در زیر آن نمایش داده می شود.

Program منوی

به کمک این منو امور مربوط به پروگرام کردن میکروکنترلر انجام می وشد. یکی از این امور انتخاب مسیر فایل (.hex) برنامه ای است که می خواهید در حافظه فلاش میکروکنترلر پروگرام نمایید یا مسیر فایل هگزادسیمال داده ای است که می خواهید در EEPROM میکروکنترلر قرار دهید.

Eraser Device: این گزینه حافظه داخلی میکروکنترلر را بطور کامل پاک می کند.

Read: این گزینه اطلاعات داخل حافظه Flash میکروکنترلر را می خواند.

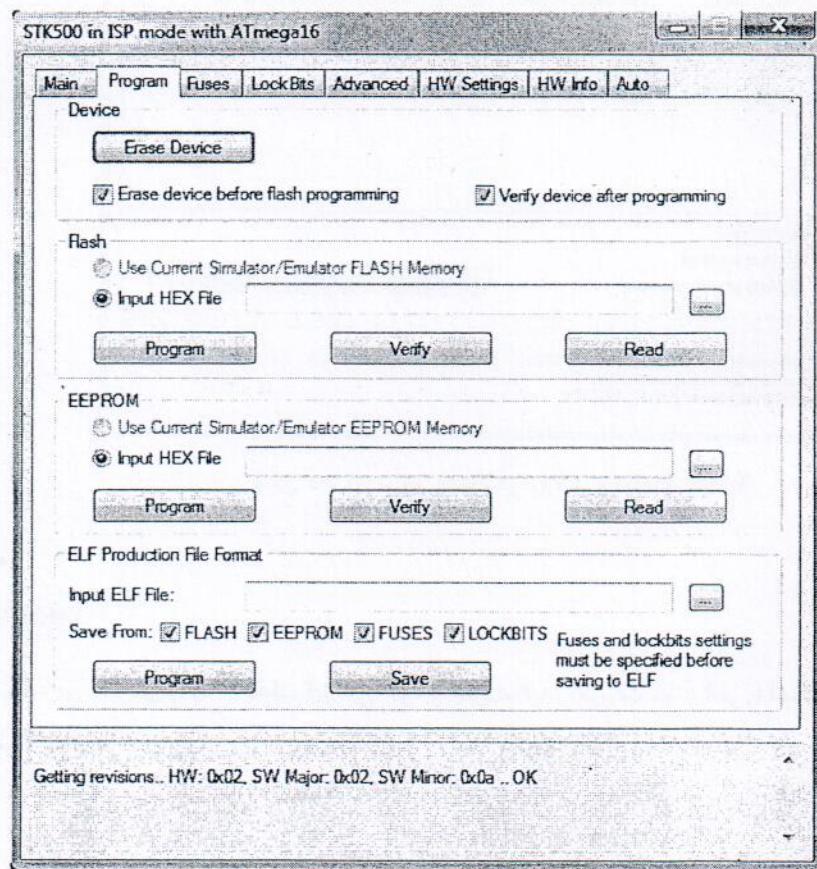
: این گزینه فایل هنگز مورد نظر شما که قبل با گزینه Load Flash انتخاب کرده‌اید را بر روی حافظه میکروکنترلر انتقال می‌دهد.

: این گزینه یک بار اطلاعات ریخته شده بر وری حافظه Flash را می‌خواند و با اطلاعات فایل هنگز انتخاب شده مقایسه می‌کند، به عبارت دیگر چک می‌کند که تمامی اطلاعات ریخته شده بر وری حافظه Flash به طور صحیح منتقل شده باشند.

: این گزینه فایل هنگز موردنظر شما را روی حافظه EEPROM میکروکنترلر انتقال می‌دهد.

: این گزینه اطلاعات داخل حافظه EEPROM میکروکنترلر را می‌خواند.

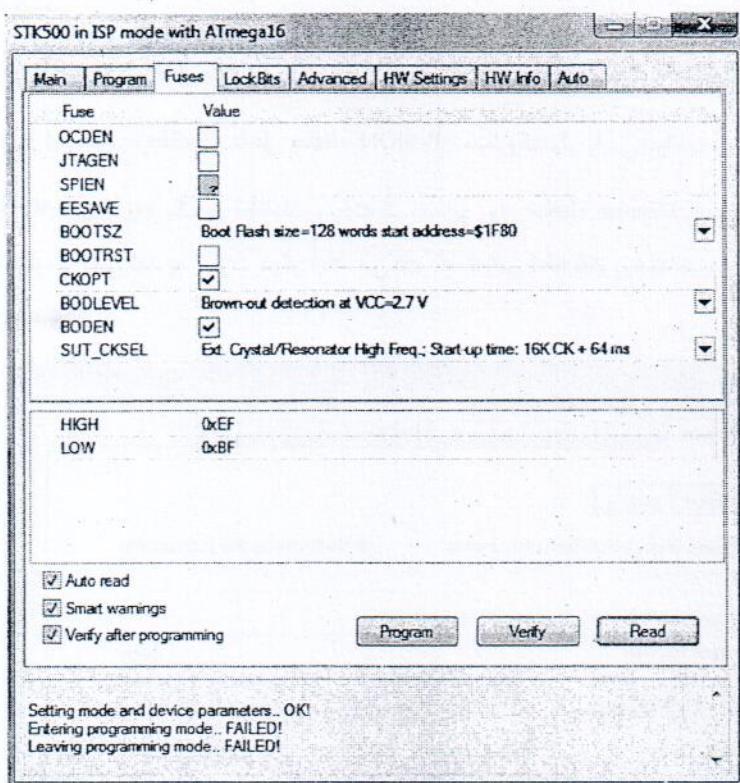
: این گزینه یک بار اطلاعات ریخته شده بر وری حافظه EEPROM را می‌خواند و با اطلاعات فایل هنگز انتخاب شده مقایسه می‌کند به عبارت دیگر چک می‌کند که تمامی اطلاعات ریخته شده بر وری حافظه EEPROM به طور صحیح منتقل شده باشند.



شکل ۱۸-۵ - منوی Program در نرم‌افزار 4 Visual Studio

Fuse منوی

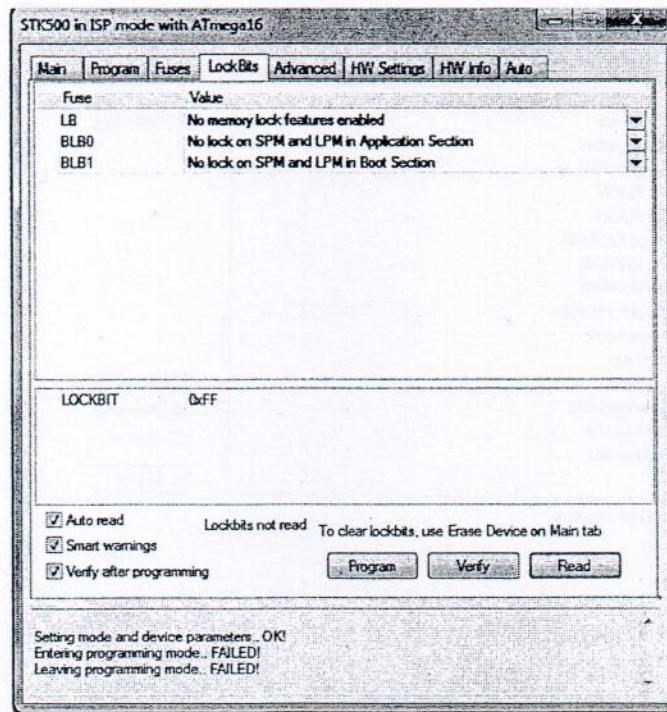
به کمک این منو می‌توانید مقادیر بیت‌های فیوز درون میکروکنترلر را تغییر دهید. تغییرات مورد نظر را برای بیت‌های فیوز اعمال کرده و بر روی گزینه Program کلیک نمائید.



شکل ۱۹-۵ - منوی Fuses در نرمافزار Visual Studio 4

LOCKBITS منوی

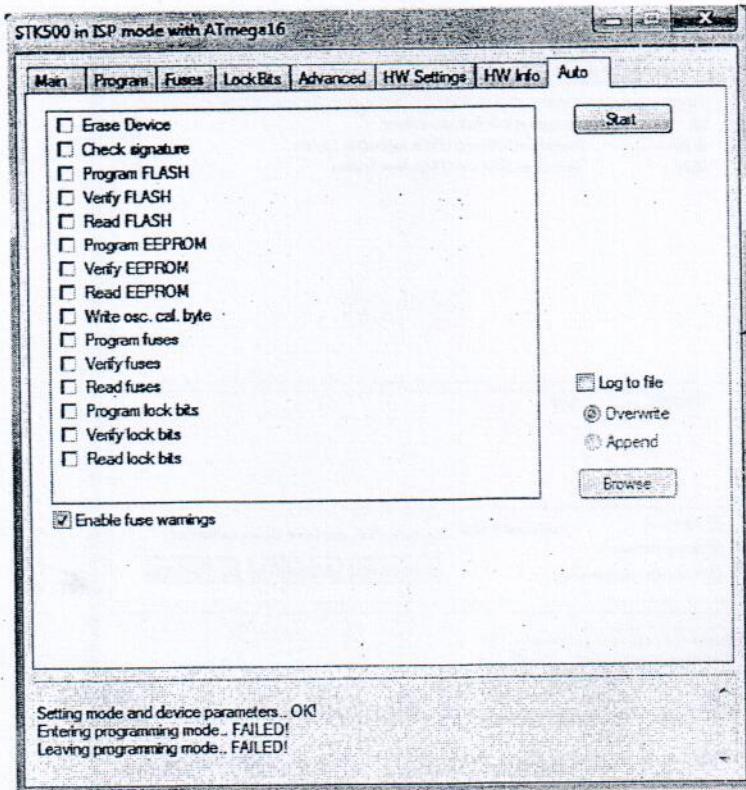
به کمک این منو می‌توانید اطلاعات آی‌سی خود را قفل نمائید. تغییرات مورد نظر را اعمال کرده و بر روی گزینه Program کلیک نمائید.



شکل ۲۰-۵ - منوی Lock Bits در نرم افزار ۴ Visual Studio

منوی Auto

به کمک این منو می توانید یک یا چند مورد از عملیاتی که در منوهای قبلی قابل انجام بودند را انتخاب و آنها را به ترتیب اجرا نمود. هر کدام از مراحلی را که می خواهید طی یک عملیات برنامه ریزی انجام دهید را تیک زده و سپس بر روی گزینه Start کلیک نمائید.



شکل ۲۱-۵ - منوی Auto در نرم‌افزار ۴ Visual Studio

۶-۴- آشنایی با نرم افزار CodeVision

نرم‌افزار CodeVisionAVR یک کامپایلر زبان C می‌باشد، محیط توسعه مجتمع (Integrated Development Environment) و تولید کننده برنامه بصورت خودکار است. نرم‌افزار CodeVisionAVR برای میکروکنترلرهای خانوادهی Atmel AVR طراحی شده است. این نرم‌افزار تقریباً تمام عناصر زبان ANSI C را تا آنجا که معماری AVR اجازه می‌دهد، پیاده‌سازی کرده است. همچنین برای استفاده از مزیت‌های معماری AVR و برآوردن نیازهای سیستم‌های نهفته، خصوصیات بیشتری به آن اضافه شده است. فایل‌های کامپایل شده COFF object را می‌توان با مشاهده متغیرها، با استفاده از Atmel AVR Studio 3.50 و یا نسخه جدیدتر، اشکال زدایی نمود.

این IDE نرم‌افزار برنامه‌ریزی AVR را در خودش دارد که انتقال خودکار برنامه به تراشه میکروکنترلر را فراهم می‌کند. قبل از این کار، ابتدا باید برنامه را کامپایل و اسembل کرد. این نرم‌افزار برای کار کردن با بردهای Atmel STK500، Kanda Systems Micro Tronics ATCPU و Vogel Elektronik VTEC_ISP، Dontronics DT006، STK200+/300 طراحی شده است. برای اشکال‌زدایی سیستم‌های نهفته که از ارتباطات سریال استفاده می‌کنند، این IDE دارای ترمینال مشخصی است. نرم‌افزار CodeVision AVR شامل ویزارد (Wizard) های زیر برای تولید خودکار برنامه می‌باشد که به شما امکان می‌دهد که در طول چند دقیقه، تمام کد مورد نیاز برای پیاده‌سازی توابع زیر را بنویسید:

- تنظیمات دسترسی به برنامه خارجی
- شناسایی منبع بازنشانی تراشه
- مقداردهی اولیه در گاههای ورودی/خروجی
- مقداردهی اولیه وقفهای خارجی
- مقداردهی اولیه زمان سنج/شمارندها
- مقداردهی اولیه زمان سنج نگهبان
- مقداردهی اولیه USART و ارتباط سریال بافر شده مبتنی بر وقفه
- مقداردهی اولیه مبدل آنالوگ به دیجیتال
- مقداردهی اولیه مقایسه کننده آنالوگ
- مقداردهی اولیه واسط SPI
- مقداردهی اولیه گذرگاه I2C، سنسور دمای LM75، ترمومتر DS1621
- تراشه‌های ساعت واقعی DS1307 و DS1302، PCF8583 و PCF8563
- مقداردهی اولیه گذرگاه ۱ سیمه و سنسورهای دمای DS1820/DS18S20
- مقداردهی اولیه نمایش دهندهای LCD

بعضی از قابلیت‌هایی که توسط محیط توسعه مجتمع CodeVisionAVR IDE قابل دسترسی هستند، عبارتند از:

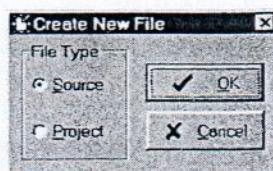
- می‌توان یک فایل برنامه (سورس) جدید را ایجاد، ذخیره و پرینت نمود.
- می‌توان فایل برنامه (سورس) را از نو باز نمود، تغییر نام داد و یا ادیت نمود.
- می‌توان یک پروژه جدید را ایجاد نمود. در این زمان از کاربر سوال می‌شود که آیا مایل است از امکان ویزارد استفاده نماید یا خیر.
- پروژه را ذخیره نمود یا پروژه قبلی را باز نمود.
- می‌توان فایل سورسی را به یک پروژه اضافه یا کم کرد.
- می‌توان option پروژه را تعیین نمود (شامل نوع میکروکنترلر و فرکانس ساعت آن، مدل حافظه، فرمت فایل خروجی، بهینه‌سازی از جهت سرعت و حجم کد، حجم پشته، حجم RAM داخلی و خارجی، نوع برنامه Boot Loader یا Application)، تعیین مسیر فایل‌های کتابخانه و include
- پاک کردن حافظه تراشه و امکان چک کردن خالی بودن حافظه EEPROM و FLASH
- برنامه‌ریزی حافظه FLASH تراشه مورد نظر و چک کردن صحت انتقال برنامه به آن
- برنامه‌ریزی حافظه EEPROM تراشه مورد نظر و چک کردن صحت انتقال برنامه به آن
- برنامه‌ریزی بیت‌های فیوز و قفل
- اجرای برنامه منتقل شده به تراشه
- امکان دیباگ کردن برنامه و تعیین مسیر آن
- امکان ادیت کردن بافر داده EEPROM و FLASH
- دارای یک ترمینال برای برقراری ارتباط سریال
- دارا بودن امکان پیکربندی اسپلر
- امکان تنظیم پارامترهای محیط IDE

- انجام تنظیمات برنامه‌ریز تراشه برای انتخاب نوع برنامه‌ریز درون سیستمی و تعیین درگاهی از کامپیوتر که به برنامه‌ریز درون سیستمی متصل است. STK500 از جمله برنامه‌ریزهای درون سیستمی است که از درگاه سریال مانند درگاه COM3 استفاده می‌نماید.

در ادامه قابلیت‌های نرم‌افزار کدویژن با جزئیات بیشتری ارائه خواهد شد.

۵-۱-۶-۱- ایجاد فایل جدید پروژه و فایل سورس برنامه

با استفاده از CodeVision AVR IDE، شما می‌توانید هر نوع فایل متنی که توسط کامپایلر C تولید و یا استفاده می‌شود را مشاهده و یا ویرایش کنید. شما می‌توانید یک فایل سورس جدید را از طریق منوی File|New و یا با فشردن دکمه Create new file بر روی نوار ابزار، ایجاد کنید. در شکل ۲۲-۵ یک جعبه‌ی دیالوگ مشاهده می‌کنید که باید در آن گزینه source را انتخاب کنید.

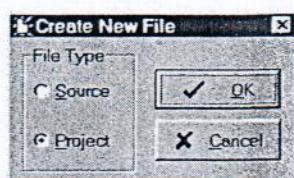


شکل ۲۲-۵- ایجاد فایل جدید

پنجه ویرایشی برای فایلی که جدیداً ایجاد شده است، آشکار می‌شود. این فایل را با نام دلخواه در دایرکتوری مورد نظر، با پسوند *.c ذخیره نمایید.

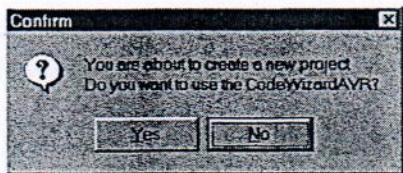
۵-۲-۶-۲- کار کردن با Project

در واقع Project، فایل‌های source و تنظیمات کامپایلر را که برای ساختن برنامه‌ای خاص به آن نیاز دارید را یکپارچه می‌نماید. برای ایجاد یک پروژه جدید، می‌توانید از منوی File|New و یا با فشردن دکمه Create new file بر روی نوار ابزار، اقدام به این کار کنید. در این صورت یک جعبه‌ی دیالوگ مشاهده خواهد کرد که باید File Type|Project را در آن انتخاب و دکمه OK را فشار دهید (همانند شکل ۲۳-۵).



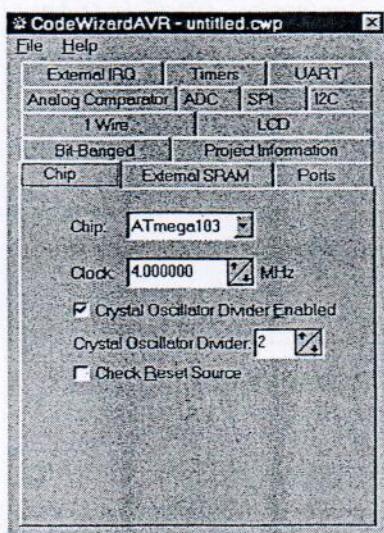
شکل ۲۳-۵- ایجاد

دیالوگی مانند شکل زیر باز می‌شود تا از شما سوال کند که آیا مایل به استفاده از CodeWizardAVR برای ایجاد پروژه جدید هستید یا خیر. در صورت عدم تمايل، گزینه‌ی NO را انتخاب کنید.



شکل ۲۴-۵-ایجاد پروژه جدید

پس از تائید ایجاد پروژه جدید، پنجره‌ی CodeWizardAVR که در شکل زیر آمده است، نمایش داده می‌شود.



شکل ۲۵-۵-پنجره‌ی CodeWizardAVR

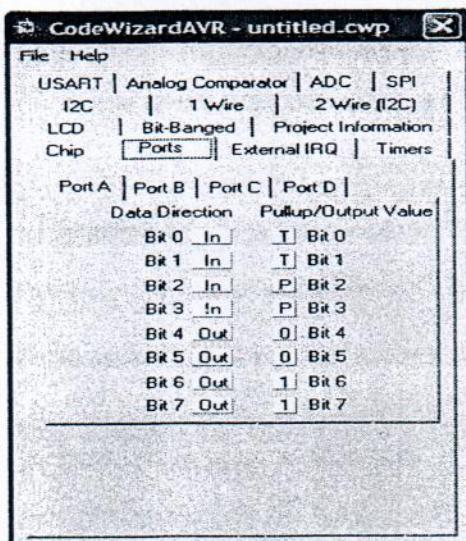
Chip ۱-۲-۶-۵

در قسمت Chip، که در شکل بالا نیز نمایش داده شده است، شما می‌توانید گزینه‌های تراشه مورد استفاده را تنظیم کنید. نوع میکروکنترلر، با استفاده از Chip list box مشخص می‌شود. فرکانس پالس ساعت تراشه بر حسب MHZ با استفاده از Clock box مشخص می‌گردد. برای تراشه‌های AVR که دارای تقسیم کننده‌های crystal oscillator می‌باشند، گزینه اضافه晶振分频器 مشاهده می‌شود. این گزینه، این امکان را به شما می‌دهد تا تقسیم کننده‌های crystal oscillator را فعال و یا غیر فعال کنید. چنانچه crystal oscillator فعال شود، می‌توانید با استفاده از Crystal Oscillator Divider Enabled نسبت تقسیم را مشخص کنید.

Port ۲-۲-۶-۵

برای تنظیمات پورت‌های ورودی و خروجی، باید گزینه Port را انتخاب نمود. در این صورت جعبه دیالوگی مانند شکل ۲۶-۵ مشاهده خواهد شد. برای مشخص کردن پورت‌های مورد استفاده در برنامه، می‌توانید پورت مورد نظر را انتخاب کرده و سپس ورودی/خروجی بودن هریک از بیت‌های آنرا در قسمت Data Direction تعیین نمایید. با کلیک کردن بر روی Pull up/Output value، می‌توانید تنظیمات زیر را انجام دهید:

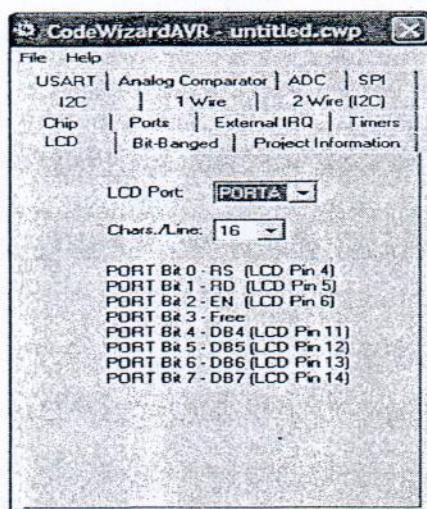
- اگر بیت ورودی باشد، تعیین می‌کنید که بیت مورد نظر، (T) و یا (P) Pull up باشد.
- اگر بیت خروجی باشد، مقدار اولیه ۰ یا ۱ را می‌پذیرد.



شکل ۲۶-۵- پنجره‌ی تنظیم پورت‌ها

LCD ۳-۲-۶-۵ قسمت

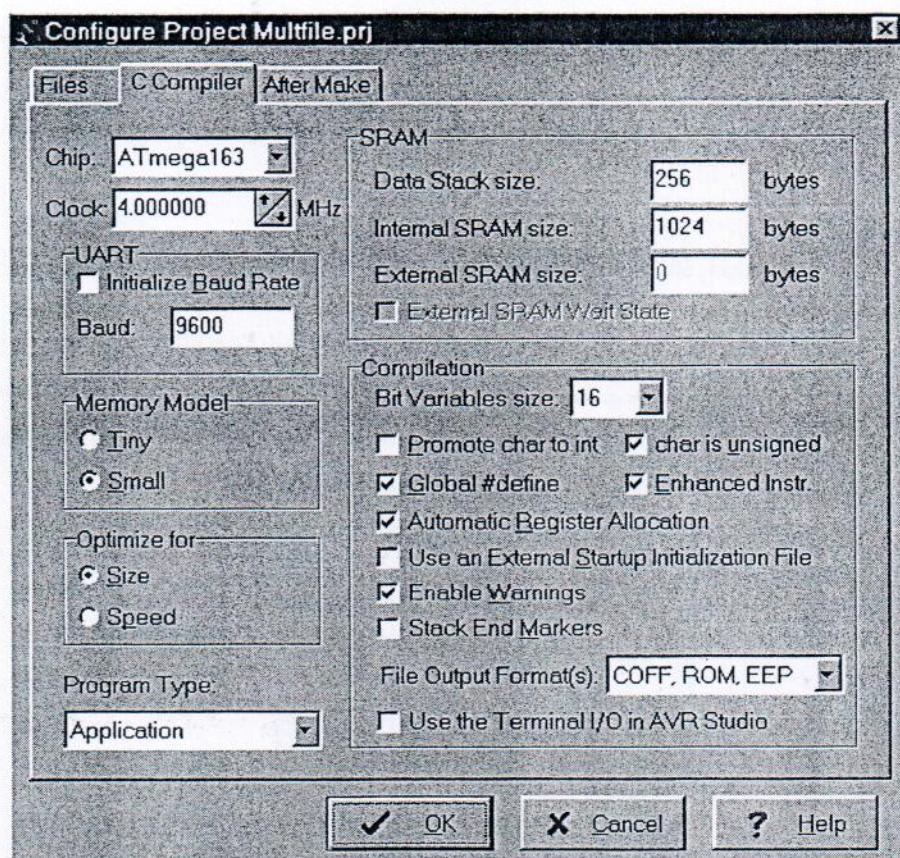
در این قسمت می‌توان پورتی را که LCD بر روی آن قرار می‌گیرد، را مشخص نمود. گزینه دیگری که باید مشخص نمود می‌باشد، که تعداد کاراکترها را در هر خط LCD مشخص می‌کند. در شکل ۲۷-۵ ابتدا پورت A و سپس تعداد کاراکترها در هر خط LCD را ۱۶ انتخاب کرده‌ایم.



شکل ۲۷-۵- پنجره‌ی تنظیم LCD

۳-۶-۵- تنظیم گزینه های کامپایلر C

برای تنظیم گزینه های کامپایلر C برای پروژه ای که در حال حاضر باز می باشد، باید از منوی Project | Configure استفاده کنید و گزینه Compiler C را انتخاب کنید (به شکل ۲۸-۵ توجه کنید).



شکل ۲۸-۵- تنظیم گزینه های کامپایلر C

تراسه مورد نظر را در قسمت Chip انتخاب کنید. فرکانس پالس ساعت پردازنده نیز باید مشخص شود که در Delay Dallas Semiconductor DS1820/DS1822 Temperature Sensors Function، Function مقداردهی اولیه UARTها مورد استفاده قرار می گیرد. اگر برنامه از ارتباط سریال استفاده می کند، می بایست گزینه Initialize I Wire Protocol Functions، Function را انتخاب نموده و Baud را نیز تعیین کنید. کامپایلر بطور خودکار کد start-up را برای مقداردهی اولیه UART Baud Rate، تولید می کند. مدل حافظه مورد نیاز را در قسمت Memory Model مشخص نمایید. مدل حافظه به دو صورت Tiny و Small می باشد. برنامه کامپایل شده، را برای داشتن حجم کم، سرعت اجرای بالا با استفاده از گزینه های Optimize for|Size، Optimize for|Speed می توان بهبود بخشید. برای برنامه هایی که خود برنامه ریز (Self Programming) هستند، نوع برنامه به دو صورت می تواند انتخاب شود:

- Application •
- Boot Loader •

در صورت انتخاب Application، کد تولید شده بعنوان برنامه مورد نظر قابل استفاده است. در صورت انتخاب Boot Loader یک امکان اضافی برای debug کردن در انتخاب‌های AVR Options فراهم می‌شود که در صورت انتخاب، کامپایلر کد اضافه‌ای برای این منظور تولید می‌کند.

۴-۶-۵- کامپایل کردن پروژه

برای کامپایل کردن پروژه، می‌توانید منوی Project|Compile را انتخاب کنید و یا دکمه F9 را فشار دهید. کامپایلر CodeVisionAVR C اجرا خواهد شد و فایل اسمبلی با پسوند .asm را ایجاد خواهد کرد.

۵-۶-۵- ساختن پروژه

برای ساختن پروژه، منوی Project|Make را انتخاب کنید و یا دکمه Shift+F9 را فشار دهید.

۶-۶-۵- تنظیمات AVR Chip Programmer

با استفاده از منوی Settings|Programmer، می‌توانید نوع Programmer مورد استفاده را انتخاب کنید (STK500) و همچنین پورت کامپیوتری که برنامه‌ریز به آن متصل است را نیز مشخص کنید (شکل ۲۹-۵).



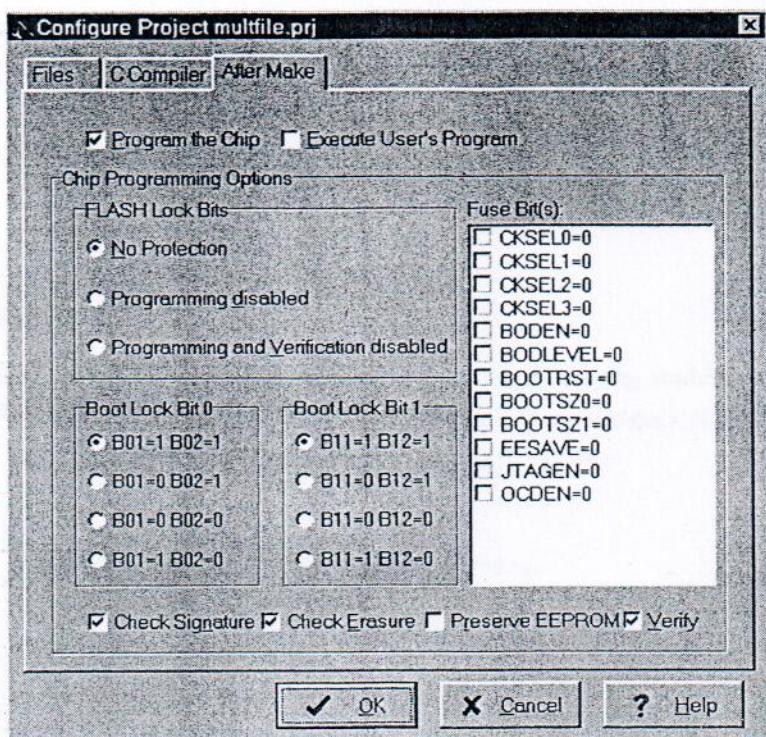
شکل ۲۹-۵- تنظیمات AVR Chip برنامه‌ریز

نسخه فعلی CodeVisionAVR، برنامه‌ریزهای زیر را پشتیبانی می‌کند:

- Kanda Systems STK200+ and STK300
- Atmel STK500
- Dontronics DT006
- Vogel Elektronik VTEC-ISP
- Micro Tronics ATCPU and Meag2000

۷-۶-۵- AVR Chip Programmer

اگر در پنجره Project Configure، قسمت After Make را انتخاب کنید، این گزینه فعال می‌شود (شکل ۳۰-۵).



شکل ۳۰-۵- انتقال برنامه‌ی به تراشه‌ی AVR

با انتخاب گزینه Program the Chip، پس از کامپایل و اسambil کردن، برنامه بطور خودکار با استفاده از نرمافزار Built_in برنامه‌ریزی، به تراشه AVR منتقل خواهد شد. هر یک از موارد ذیل یا همه با هم از طریق گزینه Program the Chip قابل اجرا خواهند بود.

۱. پاک کردن حافظه‌های تراشه
۲. بررسی کردن خالی بودن حافظه‌های EEPROM و FLASH
۳. برنامه‌ریزی حافظه FLASH و تائید محتوای آن
۴. برنامه‌ریزی حافظه EEPROM و تائید محتوای آن
۵. برنامه‌ریزی بیت‌های فیوز و قفل

شما می‌توانید تراشه‌ای را که می‌خواهید برنامه‌ریزی کنید از قسمت Chip انتخاب کنید. برای محافظت از کپی کردن برنامه‌تان، باید گزینه‌های مرتبط را با استفاده از FLASH Lock Bits انتخاب کنید.

در هنگام پروگرام کردن با استفاده از نرم افزار CodeVision، باید به این نکته دقت داشت که با زدن گزینه All، بیت‌های فیوز در حالتی که مقادیر درستی برای آن‌ها تنظیم نشده است، در میکروکنترلر مقداردهی نشوند. اینکار می‌تواند موجب قرار گرفتن میکروکنترلر در وضعیت ناخواسته گردد. به عنوان مثال تنظیم بیت‌های فیوز برای حالتی که میکروکنترلر از ساعت کریستالی استفاده نماید، موجب می‌شود که میکروکنترلر دیگر با ساعت داخلی خود کار نکند و تا اتصال کریستال و خازن‌های مربوطه به مکیروکنترلر امکان استفاده از میکروکنترلر فراهم نخواهد بود. بنابراین بهتر است همواره برای پروگرام کردن از گزینه Program -> Erase Chip استفاده کنید.

و سپس گزینه Flash->Program استفاده گردد تا بدین ترتیب فقط برنامه در حافظه فلاش قرار گیرد. در صورت نیاز به تغییر بیت‌های فیوز حتما در ابتدا مقادیر فعلی بیت‌های فیوز را از قسمت Read->Fuse Bits خوانده و سپس بیت‌های فیوز مورد نظر را تغییر داده و نهایتاً از قسمت Program آنها را برنامه‌ریزی نمائید. در صورت استفاده از پروگرامر برد آزمایشگاهی NSK108، طبق روند گفته شده برای برنامه‌ریزی توسط آن پروگرامر عمل نمائید.

WinAVR -۷-۵ معرفی

۱-۷-۵ مقدمه

WinAVR یکی از ساده‌ترین محیط‌های برنامه‌نویسی منبع باز است. یکی از توزیع‌های Eclipse به نام Wascana است که می‌توان بر روی این IDE افزونه WinAVR را نصب نمود. امکانات فراوان CodeVision را ندارد، اما مزایایی دارد که عبارتند از:

- دسترسی کامل به ثبات‌ها و بالا بردن تسلط برنامه‌نویس بر روی آنها
- رایگان بودن
- داشتن محیط توسعه مجتمع IDE رایگان برای برنامه‌نویسی
- نداشتن برخی مشکلات کامپایلر Codevision
- Customization
- انعطاف‌پذیری و امکان

۲-۷-۵ راهنمای نصب

نصب Wascana مانند نصب برنامه‌های معمولی تحت ویندوز می‌باشد. پس از نصب آن، باید WinAVR را بر روی سیستم خود نصب نمایید تا به وسیله‌ی کامپایلر آن بتوان پروژه‌ها را کامپایل نمود و خروجی Hex تولید نمود. این کار مشابه روال نصب هر برنامه تحت ویندوز دیگری است. چگونگی نصب پلاگین AVR بر Wascana در ادامه بیان گردیده است. مراحل نصب پلاگین AVR روی محیط Wascana عبارتند از:

- پس از نصب محیط Wascana و برنامه‌ی WinAVR، محیط Wascana را باز کرده و در منوی Help، منوی Software Update، گزینه‌ی Find and Install را انتخاب کنید.
- در پنجره‌ی باز شده (Install/Update)، گزینه‌ی دوم (Search for new features to install) را انتخاب کنید و دگمه‌ی Next را کلیک کنید.
- در پنجره‌ی باز شده (Install) از قسمت سمت راست، دگمه‌ی New Local Site را کلیک کنید.
- در پنجره‌ی باز شده (Browse for Folder) پوششی پلاگین AVR (پوششی شامل پوشش‌های features و plugins) را انتخاب کنید و دگمه‌ی OK را کلیک کنید.
- پس از بازگشتن به صفحه‌ی Install پلاگین اضافه شده را با تیک زدن چکباکس کنار آن، انتخاب کنید و دگمه‌ی Finish را در پایین پنجره کلیک کنید.
- پنجره‌ی جدیدی با عنوان Update باز می‌شود. در این پنجره، شما می‌توانید زیر مجموعه‌های بسته‌ی این پلاگین را مشاهده کنید.

- زیرمجموعه‌ی پلاگین مورد نظر (eClipse Plugins for AVR/...) را بوسیله‌ی کلیک کردن روی علامت مثلث‌شکل کنار نام پلاگین باز کنید و دو زیرمجموعه‌ی موجود در دسته‌ی Other را انتخاب کنید و دکمه‌ی Next را در پایین پنجره کلیک کنید.
- سپس در پنجره‌ی باز شده (Install) گزینه‌ی I accept the terms in the license agreement را انتخاب کرده و دکمه‌ی Next را در پایین پنجره کلیک کنید.
- اکنون در پنجره‌ی Install باید در قسمت Features to install دو عنوان AVR Eclipse Plugin... قابل مشاهده باشد. در این صورت دکمه‌ی Finish را در پایین صفحه کلیک کنید.
- سپس در پنجره‌ی باز شده (Verification) با کلیک کردن دکمه‌ی All ، نصب پلاگین را تایید کنید.
- پس از اتمام نصب پلاگین، محیط Wascana برای اضافه کردن پلاگین جدید به امکانات قابل استفاده برای کاربر، باید restart شود. پس با کلیک کردن دکمه‌ی Yes در جواب پیغام نرمافزار، فرآیند نصب را تکمیل کنید. بعد از آغاز دوباره‌ی محیط Wascana ، امکان ایجاد پروژه‌ی AVR به محیط اضافه شده است.

۳-۲-۵- انجام یک پروژه‌ی ساده با WinAVR

پس از نصب plugins مربوطه از منوی File قسمت C -> New را انتخاب می‌کنیم. در پنجره‌ی باز شده نام پروژه را در Project Name وارد کرده و تنظیمات زیر را اعمال می‌نماییم:

Project Type: AVR Cross Target Application

Toolchain: AVR-GCC Toolchain

در صفحه‌ی AVR Target Properties نوع تراشه AVR و سرعت ساعت مورد نظر را انتخاب و سپس دگمه Finish را کلیک می‌کنیم تا پروژه ساخته شود. حال، روی پروژه ساخته شده راست کلیک می‌کنیم. از منوی باز شده از گزینه‌ی هتو، روی گزینه‌ی Source File کلیک می‌کنیم و نام فایلی که ایجاد می‌شود را main.c می‌گذاریم. برای شروع، کد پایه زیر را می‌نویسیم:

```

main.c
#include <avr/io.h>
#include <avr/interrupt.h>

int main(){
    return 0;
}

```

شکل ۳-۱۵- برنامه main.c

کتابخانه‌ی io.h دارای ثبات‌های ورودی و خروجی برای AVR و Interrupt.h برای دسترسی به وقفه‌ها می‌باشد. کتابخانه Interrupt.h در این پروژه استفاده نمی‌شود و صرفا برای معرفی آورده شده است. باید توجه داشت که در اینجا برخلاف CodeVision ویزاردی برای مقداردهی به ثبات‌های AVR وجود ندارد، لذا باید تسلط کافی بر استفاده از روی ثبات‌های کنترلی و وضعیت داشته باشیم. کد لازم برای چشمک زدن یک LED که به پایه PA0 میکروکنترلر متصل است در ذیل آورده شده است.تابع delay_ms() از کتابخانه‌ی delay.h برای ایجاد تأخیر زمانی استفاده شده است.

```

#include <avr\io.h>
#include <avr\interrupt.h>

#include <util\delay.h>

int main(){
    DDRA |= (1<<PA0);

    while(1){
        PORTA |= 1;
        _delay_ms(1000);
        PORTA &= ~1;
        _delay_ms(1000);
    }

    return 0;
}

```

شکل ۳۲-۵- برنامه چشمکزن LED

۴-۷-۴- نحوه تولید فایل hex

برنامه نوشته شده به زبان C، برای اجرا بر روی میکروکنترلر باید کمپایل شده و به فایل hex تبدیل شود. مراحل تولید فایل hex به شرح زیر است:

- برای تولید فایل Hex ابتدا باید محل تولید آن را از Debug به Release تغییر داد و سپس کامپایل نمود.
- برای این کار روی پروژه راست کلیک می‌کنیم و از منوی باز شده گزینه properties را انتخاب می‌کنیم.
- از پنجره باز شده در قسمت C/C++ Build از گزینه Manage Configurations کلیک می‌کنیم.
- در پنجره گزینه release را انتخاب کرده و سپس بر روی گزینه Set Active و نهایتاً OK کلیک می‌کنیم.
- حال از منوی Project گزینه all Build را انتخاب می‌کنیم. در پایین صفحه در قسمت console خطاهای برنامه در صورت وجود نوشته می‌شود. مشاهده عبارت Finish Building به معنای آن است که برنامه بدون خطا کامپایل شده است و فایل hex اکنون در شاخه Release (که زیرشاخه محلاً ساخته شدن پروژه می‌باشد) قرار گرفته است.

۴-۷-۵- انتقال برنامه به میکروکنترلر

در صورت استفاده از برد پروگرامر، به کمک پروگرامر آن، فایل hex تولید شده را به میکروکنترلر برآورده آموزشی منتقل نمائید.

۶- جلسه چهارم

هدف از آزمایش:

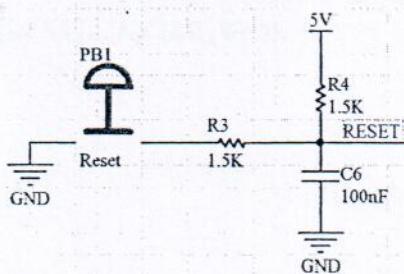
- نصب میکروکنترلر بر روی برد برقی و برقراری اتصالات به صورت دستی^۱
- تولید سیگنال بازن Shanی
- چگونگی برنامه ریزی بیت های فیوز
- تولید سیگنال ساعت (clock)

قطعات مورد استفاده:

- برد برقی: ۱ عدد
- میکروکنترلر ATmega32 یا ATmega16: ۱ عدد
- مقاومت ۱.۵K: ۲ عدد
- خازن ۱۰۰nf: ۱ عدد و خازن ۲۲ یا ۲۷ پیکوفاراد: ۲ عدد
- کلید فشاری: ۱ عدد
- کریستال کوارتز: ۱ عدد

۶-۱- تولید سیگنال بازن Shanی

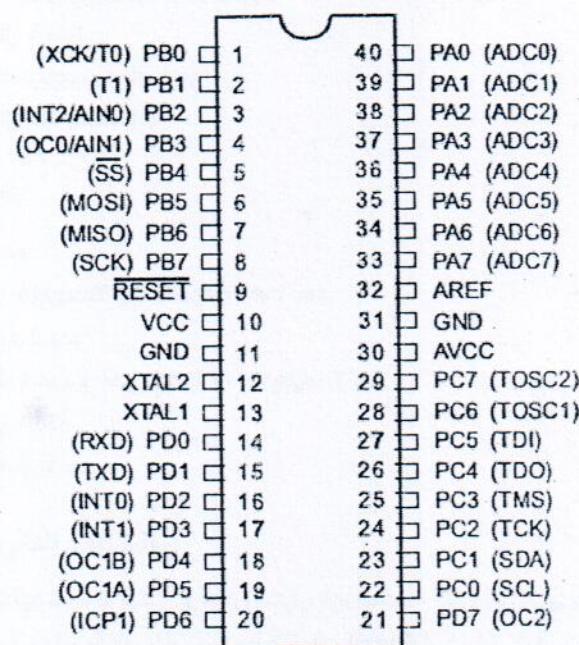
برای بهبود عملکرد سیستم مبتنی بر میکروکنترلر، لازم است که مداری نیز برای بازن Shanی کردن آن ظراحتی و استفاده نمائیم. می دانیم که میکروکنترلر دارای پایه ای بنام ریست یا بازن Shanی (Reset) است. این پایه در میکروکنترلر ATmega32 فعال روی سطح پائین بوده و وظیفه آن بردن میکروکنترلر به وضعیت بازن Shanی است. این بازن Shanی بلطفاً پس از وصل تغذیه میکروکنترلر و یا بصورت دستی قابل انجام می باشد. برای این منظور، مدار بازن Shanی زیر پیشنهاد می شود. پایه RESET میکروکنترلر را به مدار بازن Shanی ارائه شده در شکل ۶-۱ متصل می کنیم. با فشردن کلید PB1 عمل بازن Shanی بصورت دستی انجام می شود. این مدار همچنین قادر است در زمان وصل نمودن تغذیه میکروکنترلر، میکروکنترلر را بازن Shanی نماید.



^۱ در بعضی از جلسات اولیه آزمایشگاه، اتصال بین پایه های میکروکنترلر و قطعات جانبی بر روی برد برقی و توسط دانشجو انجام می شود. اینکار موجب آشنایی بیشتر دانشجویان با پایه های میکروکنترلر و اتصال قطعات جانبی به آنها می گردد. در آزمایشات بعدی برای بالا بردن سرعت انجام آزمایش ها از برد آموزشی استفاده خواهد شد.

شکل ۱-۶- مدار بازن Shanai

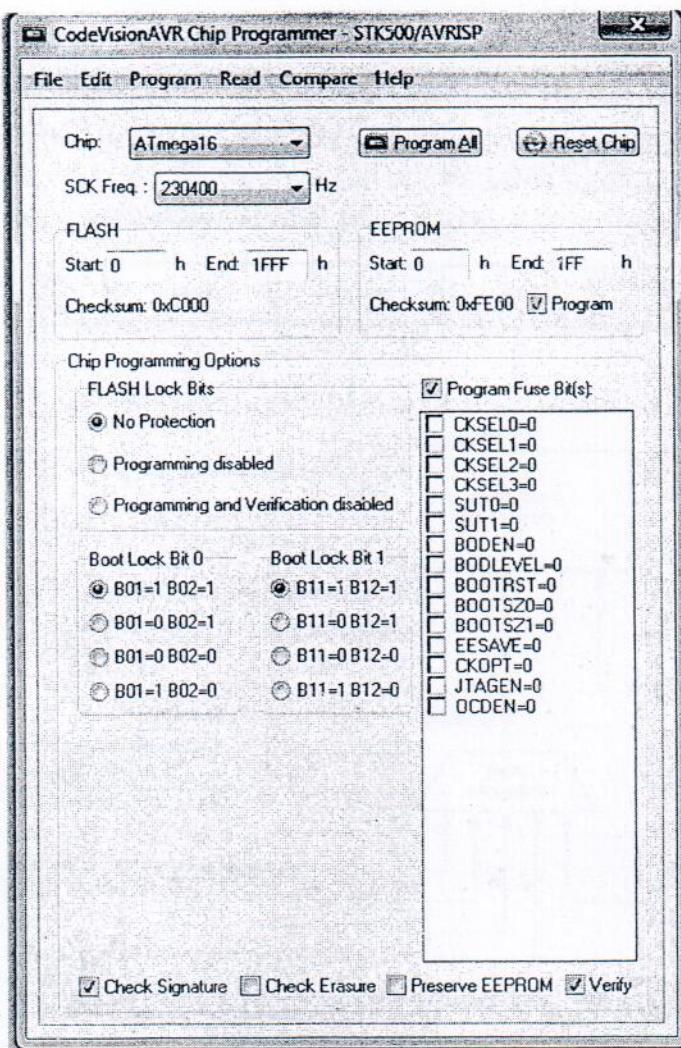
شکل و ترتیب پایه‌های میکروکنترلر ATmega16 برای استفاده در این آزمایش و آزمایشات بعد، در شکل زیر ارائه شده است. پایه‌های میکروکنترلر ATmega32 نیز تقریباً مشابه پایه‌های میکروکنترلر ATmega16 هستند.



شکل ۲-۶- شکل و ترتیب پایه‌های ATmega32 و ATmega16

۲-۶- تغییر بیت‌های فیوز

اگر در نرم‌افزار CodeVision بخواهید نسبت به برنامه‌ریزی بیت‌های فیوز اقدام نماید، پنجره Tools، قسمت Chip را انتخاب کنید، این گزینه فعال می‌شود (شکل ۳۰-۵).

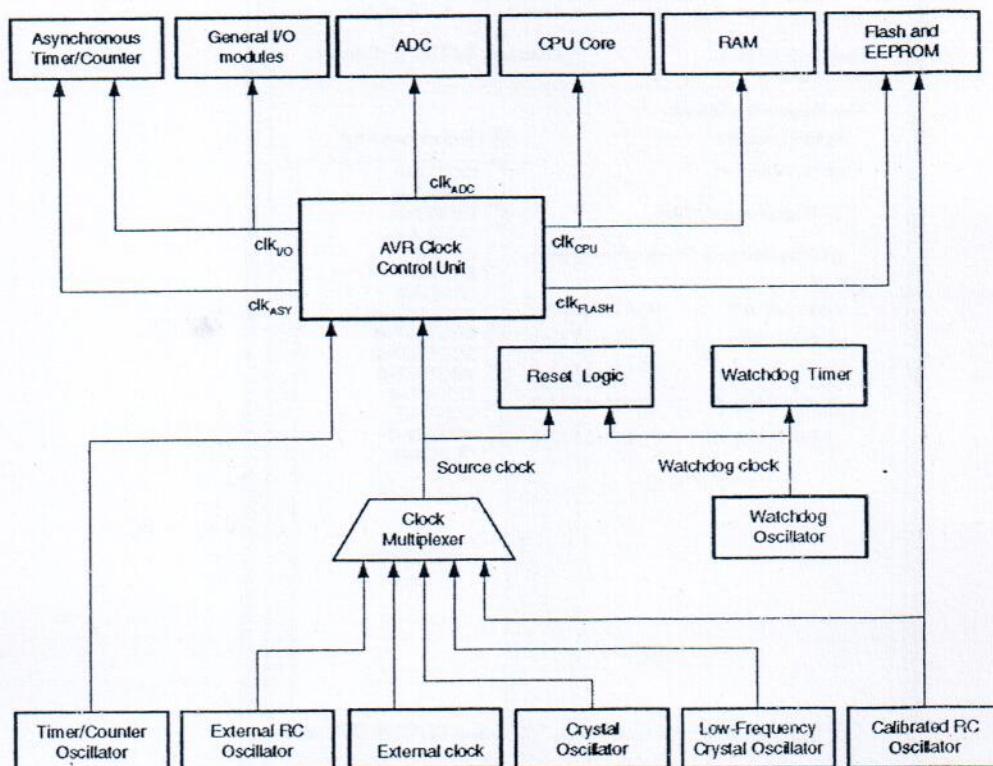


شکل ۳-۶- تغییر مقادیر بیت‌های فیوز

بیت‌های فیوزی را که می‌خواهیم برنامه‌ریزی (0 شوند) را تیک می‌زنیم و با انتخاب گزینه Program Fuse bit مورد نظر را برنامه‌ریزی می‌نمائیم. در هنگام پروگرام کردن با استفاده از نرم افزار CodeVision، باید به این نکته دقت داشت که با زدن گزینه Program All، بیت‌های فیوز در حالتی که مقادیر درستی برای آن‌ها تنظیم نشده است، در میکروکنترلر مقداردهی نشوند. اینکار می‌تواند موجب قرار گرفتن میکروکنترلر در وضعیت ناخواسته گردد. به عنوان مثال تنظیم بیت‌های فیوز برای حالتی که میکروکنترلر از ساعت کریستالی استفاده نماید، موجب می‌شود که میکروکنترلر دیگر با ساعت داخلی خود کار نکند و تا اتصال کریستال و خازن‌های مربوطه به مکیروکنترلر امکان استفاده از میکروکنترلر فراهم نخواهد بود. بنابر این بهتر است همواره برای پروگرام کردن از گزینه Program -> Erase Chip و سپس گزینه Program -> Flash استفاده گردد تا بدین ترتیب فقط برنامه در حافظه فلاش قرار گیرد. در صورت نیاز به تغییر بیت‌های فیوز حتماً در ابتدا مقادیر فعلی بیت‌های فیوز را از قسمت Read -> Fuse Bits خوانده و سپس بیت‌های فیوز مورد نظر را تغییر داده و نهایتاً از قسمت Program آنها را برنامه‌ریزی نمائید.

۳-۶- تولید سیگنال ساعت (clock)

تولید سیگنال ساعت به روش‌های متعددی در میکروکنترلرهای خانواده ATmega امکان‌پذیر است. شکل ۴-۶ سیستم‌های ساعت اصلی و توزیع آنها را در میکروکنترلرهای AVR نشان می‌دهد. در یک زمان مشخص، نیازی به فعال بودن همه منابع ساعت نمی‌باشد. برای کاهش مصرف توان، پالس‌های ساعتی که به مازول‌های غیرفعال می‌روند را می‌توان به طور کامل متوقف یا توسط روش‌های مختلف مدیریت توان و حالت‌های خواب^۱، در وضعیت خواب قرار داد.



شکل ۴-۶- توزیع ساعت در میکروکنترلر ATmega32

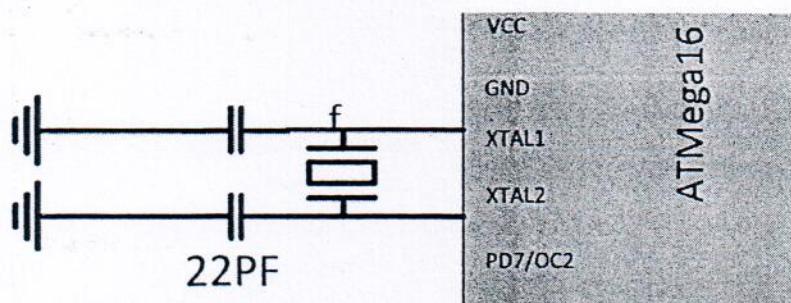
همانطور که ملاحظه می‌شود منابع مختلفی به عنوان سیگنال ساعت می‌تواند در میکروکنترلرهای AVR استفاده گردد. این منابع ساعت عبارتند از:

- اسیلاتور RC داخلی کالیبره شده
- اسیلاتور کریستالی
- کلک خارجی
- اسیلاتور RC خارجی

^۱ sleep

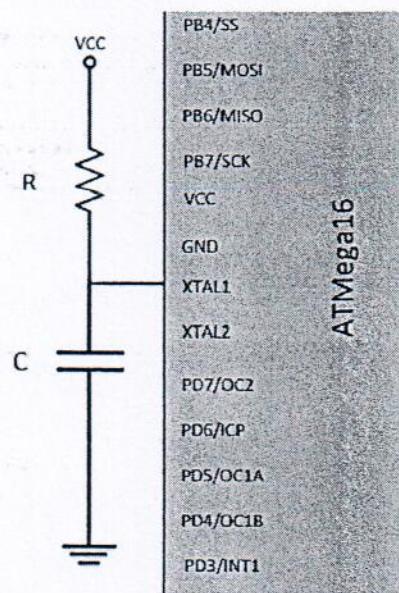
• اسیلاتور کریستالی فرکانس پائین (32.768 کیلوهرتز)

الف- با استفاده از دیتاشیت، میکروکنترلر ATmega32 و برنامه‌ریزی بیت‌های فیوز میکروکنترلر، سیگنال ساعت را به روش استفاده از کریستال تولید نمایید. شکل ۵-۶ محل قرار گرفتن کریستال با فرکانس f (بسته به کریستال موجود در آزمایشگاه که حداقل می‌تواند ۱۶ مگاهرتز باشد) و خازن‌های 22PF مورده نیاز برای تولید ساعت به این روش را نشان می‌دهد.



شکل ۵-۶- مدار تولید ساعت به روش استفاده از کریستال

ب- با استفاده از دیتاشیت میکروکنترلر ATmega32 یا ATMEGA16 و برنامه‌ریزی بیت‌های فیوز میکروکنترلر، سیگنال ساعت را به کمک روش اسیلاتور RC خارجی تولید نمایید (شکل ۶-۶). تغییرات فرکانس ساعت با تغییر مقادیر مقاومت و خازن را با استفاده از یک فرکانس‌متر یا اسیلوسکوپ مشاهده نمایید. فرکانس ساعت بر روی کدام پایه میکروکنترلر قابل مشاهده است؟



شکل ۶-۶- مدار تولید ساعت به روش استفاده از مدار RC خارجی

- ج- با استفاده از دیتاشیت ATmega32 و برنامه‌ریزی بیت‌های فیوز میکروکنترلر، سیگنال ساعت را به کمک روش اسیلاتور RC داخلی کالیبره شده تولید نمایید. از تنظیم ثبات OSCCAL برای کالیبره کردن ساعت RC داخلی تولید شده استفاده نمایید. تغییرات فرکانس ساعت با تنظیم ثبات OSCCAL را با استفاده از یک فرکانس‌متر یا اسیلوسکوپ مشاهده نمایید. فرکانس ساعت بر روی کدام پایه میکروکنترلر قابل مشاهده است؟
- د- با اتصال یک کریستال فرکانس پایین با فرکانس 32.768 کیلوهertz به پایه‌های TOSC1 و TOSC2 ساعت میکروکنترلر را تامین نمایید. فرکانس تولید شده را توسط فرکانس‌متر یا اسیلوسکوپ مشاهده نمایید.

۷- جلسه پنجم

هدف از آزمایش:

- کار با درگاهها
- سرکشی (polling) یک پایه از یک درگاه
- کار با رله و بازر
- تولید تأخیر نرمافزاری
- کار با زمانسنج نگهبان

قطعات مورد نیاز:

- برد برد: ۱ عدد
- میکروکنترلر ATmega16 یا ATmega32: ۱ عدد
- مقاومت ۵۶۰ اهم: ۲ عدد
- مقاومت ۲.۲ کیلواهم: ۴ عدد
- مقاومت ۱.۵K: ۱ عدد (برای مدار ریست)
- خازن 100nf: ۱ عدد (برای مدار ریست)
- کلید فشاری: ۲ عدد
- LED: ۲ عدد
- کلید فشاری ۲ عدد
- ترانزیستور 2N3804: ۲ عدد
- رله ۵ ولت: ۱ عدد

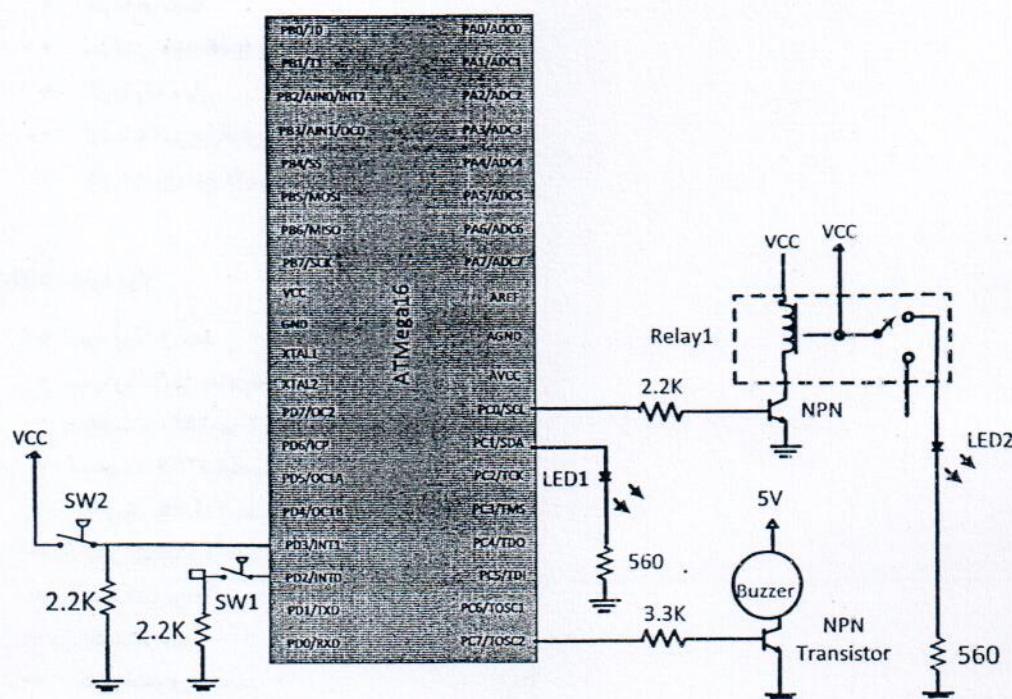
میخواهیم به اهداف این آزمایش در قالب ساخت یک مدار برای روشن و خاموش کردن یک LED دست یابیم. با توجه به شکل ۱-۷ اقدامات زیر را انجام دهید. در این آزمایشات تولید ساعت میکروکنترلر به روش استفاده از مدار RC داخلی و با فرکانس ۱MHz باشد. مدار ریست را نیز مشابه مدار شکل ۱-۶ در همه آزمایشاتی که با برداشت کار میکنید بیندید.

الف- کلید SW1 را به پایه PD2 وصل کنید. این پایه را به عنوان ورودی برنامه‌ریزی کنید و آنرا بالاکش کنید. یک عدد LED و یک عدد مقاومت را مطابق شکل زیر به پایه PC1 وصل کنید. این پایه را در وضعیت خروجی قرار دهید. برنامه‌ای به زبان اسمبلی بنویسید که با فشردن کلید SW1 دیود LED1 روشن شود. اطلاع از وضعیت پایه PD2 متصل به SW1 به روش سرکشی انجام می‌شود.

ب- یک بازر را با واسطه یک ترانزیستور همانند شکل زیر به پایه PC7 متصل کنید. مشابه بند قبل برنامه‌ای بنویسید که با فشار کلید SW1 بازر روشن و با برداشتن دست از روی کلید، بازر خاموش شود. برای روشن شدن بازر پایه PC7 باید high شود.

ج- مشابه بند قبل کاری کنید که با هر بار فشردن کلید SW1 رله Relay1 وصل و LED2 روشن و با فشار مجدد کلید SW1 رله قطع و LED2 خاموش شود. رله با واسطه یک ترانزیستور به پایه PC0 وصل است. برای روشن شدن رله میبایست پایه PC0 در

وضعیت high قرار گیرد. در صورت high کردن پایه PC0، ترانزیستور روش و سپس رله وصل می‌شود. با روشن و خاموش شدن رله دیود نوری LED2 نیز روشن و خاموش می‌شود. مدار مربوطه را بر روی پردازه‌سازی و برنامه اسکلی مربوطه را بنویسید.



شکل ۱-۷-اتصال قطعات به میکروکنترلر

د- برنامه‌ای بنویسید که با هر بار فشردن کلید SW1 LED1، PD2 متصل به پایه INT1 به تعداد ۵ بار و هر بار به مدت تقریبی ۵، ۰ ثانیه روشن و ۵، ۰ ثانیه خاموش شود. تولید زمان تاخیر ۵، ۰ ثانیه‌ای را به صورت نرم‌افزاری و بدون استفاده از زمان‌سنج انجام دهید. اطلاع از وضعیت پایه PD2 متصل به INT1 به روش سرکشی انجام می‌شود.

ه- زمان‌سنج نگهبان را برای حالتی که در صورت عدم بازنشانی آن تا زمان ۲،۱ ثانیه، زمان‌سنج نگهبان اقدام به بازنشانی میکروکنترلر نماید را مقداردهی اولیه (initialization) نمایید.

و- برنامه‌ای بنویسید که پس از روشن شدن میکروکنترلر، با یکبار فشردن کلید کلید SW1 روشن گردد و در همان حال باقی بماند. پس از آن چنانچه تا قبل از زمان ۲،۱ ثانیه اقدام به فشردن کلید INT1 ننمایید، زمان‌سنج نگهبان میکروکنترلر را بازنشانی نماید و در نتیجه LED1 خاموش گردد. ولی چنانچه شما مرتباً در زمان‌های کمتر از ۲،۱ ثانیه کلید SW1 را فشار دهید، هر بار زمان‌سنج نگهبان از نو بازنشانی شده و فرصت بازنشانی میکروکنترلر را بدست نخواهد آورد و بدین ترتیب LED1 همچنان روشن خواهد ماند.

۸- جلسه ششم

هدف از آزمایش:

- تهیه برنامه وقفه بازن Shan
- کار با وقفه های خارجی
- کار با درگاه ها
- کار با نمایش دهنده 7-Seg

قطعات مورد نیاز:

- میکروکنترلر ATmega32 یا ATmega16 ۱ عدد
- مقاومت ۵۶۰ اهم: ۲ عدد
- مقاومت ۲،۲ کیلواهم: ۴ عدد
- کلید فشاری: ۲ عدد
- LED: ۲ عدد
- کلید فشاری: ۲ عدد
- ترانزیستور NPN: ۲ عدد
- ترانزیستور PNP: ۱ عدد
- نمایش دهنده 7-Seg: ۱ عدد
-

می خواهیم به اهداف این آزمایش در قالب ساخت یک مدار برای روشن و خاموش کردن یک LED با استفاده از وقفه دست باییم. اقدامات زیر را انجام دهید. در این آزمایشات تولید ساعت میکروکنترلر به روش استفاده از مدار RC داخلی و با فرکانس ۱MHz باشد.

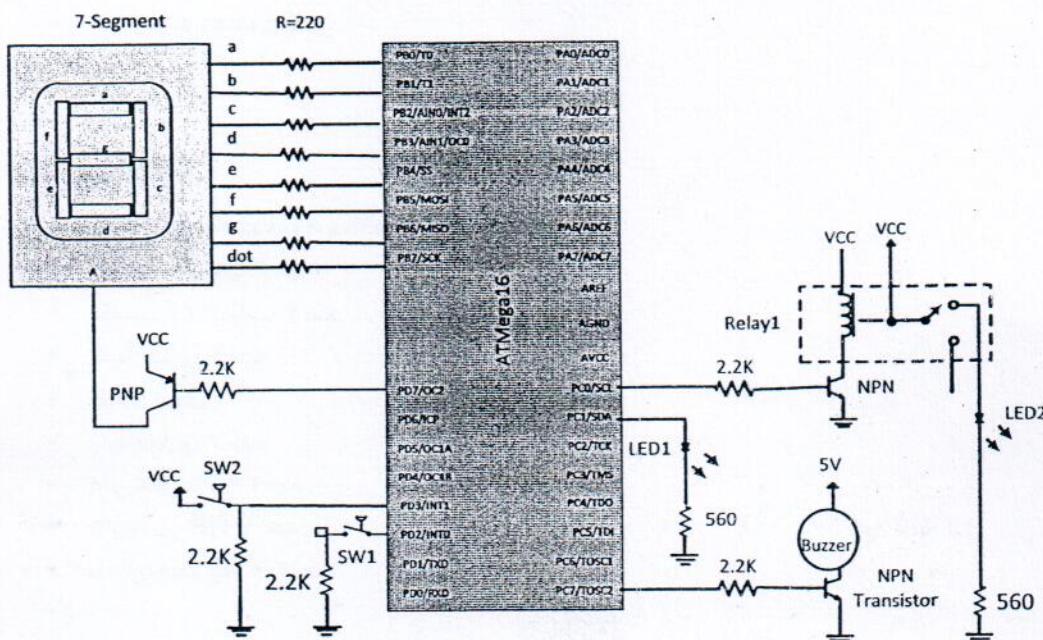
الف- روال وقفه بازن Shan را نوشه و در آن مقداردهی ثبات اشاره گر پشته را انجام دهید. حالات مختلفی که در آنها بتوان بردار وقفه و نیز روال وقفه را در بخش رامانداز (boot) یا بخش کاربردی (application) حافظه فلاش قرار داد را امتحان نمایید.

ب- پایه PD2 که کلید SW1 به آن متصل است، ورودی وقفه خارجی INT0 نیز می باشد. می خواهیم با یک بار فشردن کلید SW1 دیود نوری LED1 در شکل ۱-۸ روشن و با فشردن همین کلید برای بار دوم، LED1 خاموش گردد. روشن و خاموش کردن LED در داخل روتین وقفه خارجی INT0 انجام گردد. حالات مختلف ممکن حساسیت وقفه به لبه و حساسیت به سطح را برای ورودی وقفه امتحان کنید. برنامه مربوطه را شامل تنظیمات اولیه ثبات های کنترلی مورد استفاده، برنامه روتین وقفه و سایر اقدامات لازم ارائه نمایید.

ج- پایه PD3 که کلید SW2 به آن متصل است، ورودی وقفه خارجی INT1 نیز می باشد. می خواهیم با یک بار فشردن کلید INT1، LED1 مدت ۲ ثانیه روشن و سپس خاموش گردد. به منظور بررسی امکان داشتن وقفه تودر تو، در هنگام ورود به وقفه INT0، بیت وقفه سراسری I را ۱ نمایید و امکان پذیرش وقفه در حین اجرای روتین وقفه INT0 را فراهم سازید. بدین ترتیب چنانچه

بعد از فشردن کلید INT0، در کمتر از ۵ ثانیه کلید INT1 نیز فشرده شود، LED1 به جای ۵ ثانیه برای مدت ۷ ثانیه (مجموع زمان در نظر گرفته شده برای روشن شدن LED1 در وقفه INT0 و INT1) روشن خواهد شد.

د- یک عدد Seg-7 را به پایه‌های پورت B میکروکنترلر وصل کنید و تغذیه آند آنرا نیز از طریق ترانزیستور متصل به پایه PD7 تامین نمائید. برنامه‌ای بنویسید که با فشردن کلید SW1 مقادیر ۰ الی ۹ با تأخیر ۵، ۰ ثانیه‌ای به ترتیب بر روی نمایش‌دهنده Seg-7 نمایش داده شوند و اینکار مرتبًا تکرار شود تا اینکه کلید SW2 فشرده شود.



شکل ۱-۸- اتصال قطعات به میکروکنترلر

۹- جلسه هفتم

هدف از آزمایش:

- کار با وقفه خارجی
- راهاندازی یک کیبورد ماتریسی
- نمایش کاراکترها توسط نمایش دهنده ۷ قطعه‌ای

قطعات مورد نیاز:

- میکروکنترلر ATmega16
- کیبورد
- نمایش دهنده ۷ قطعه‌ای
- ترانزیستور PNP
- مقاومت
- دیود
- کلید فشاری
- LED

در این آزمایش هدف طراحی یک کیبورد ماتریسی و نمایش مقادیر دریافتی از کیبورد توسط نمایش دهنده ۷ قطعه‌ای است. سختافزار شکل ۹-۱ شامل یک کلید، یک LED، یک کیبورد ماتریسی و یک کیبورد دارای تعداد سطر است که به پایه‌های PC4 الی PC7 متصل هستند. ستون‌های کیبورد با واسطه تعدادی مقاومت به پایه‌های PC0 الی PC3 متصل هستند. همین ستون‌ها از طرف دیگر با واسطه تعدادی مقاومت و دیود به پایه INT0 میکروکنترلر متصل هستند.

پایه‌های PC0 الی PC3 که به ستون‌های کیبورد وصل هستند در حالت ورودی و بصورت بالاکش و سطرهای کیبورد (پایه‌های PC7) را در حالت خروجی قرار داده‌ایم. ستون‌ها از طریق ۴ عدد دیود بصورت "و سیمی" (wired and) به هم مرتبط شده و با فشرده شدن هر یک از کلیدها، یکی از ستون‌ها در سطح منطقی ۰ قرار می‌گیرد. بدین ترتیب دیود مربوط به آن ستون در بایاس مستقیم قرار گرفته و اتصال آند دیودها که به ورودی وقفه خارجی INT0 متصل هستند در سطح منطقی ۰ قرار گرفته و می‌تواند باعث بروز وقفه گردد. بهتر است که پایه INT0 را در وضعیت ورودی بالاکش قرار دهیم. بدین ترتیب با فشردن یک کلید این پایه از وضعیت منطقی ۱ به وضعیت ۰ می‌رود. برای این منظور می‌توانید حساسیت پایه INT0 را در حالت حساسیت به لبه پایه رونده قرار دهید.

از آنجا که هر میکرو به طور پیش‌فرض به گونه‌ای تنظیم شده که بیت فیوز مربوط به JTAGEN در آن یک شده است، لذا پایه‌هایی از پورت C در وضعیت استفاده برای منظور JTAG قرار دارند و نمی‌توانند به عنوان پایه درگاه بصورت عادی استفاده شوند. لذا برای استفاده معمولی از این پورت باید ابتدا فیوز بیت JTAGEN را غیر فعال نماییم (حالت فیوز بیت برنامه‌ریزی نشده). در صورت استفاده از CodeVision، به منظور آزادسازی بیت‌های پورت C و استفاده از آنها به عنوان پایه‌های درگاه برای استفاده‌های عادی، بعد از خواندن بیت‌های فیوز (در قسمت Read->Fuse Bits)، مقدار بیت فیوز JTAGEN را غیر فعال نموده و سپس مقادیر بیت‌های فیوز را در میکروکنترلر مقداردهی نمائید (در قسمت Program->Fuse Bits).

اقدامات زیر را انجام دهید:

الف- سخت‌افزار شکل ۱-۹ را بر روی پرینت بورد طراحی کنید.

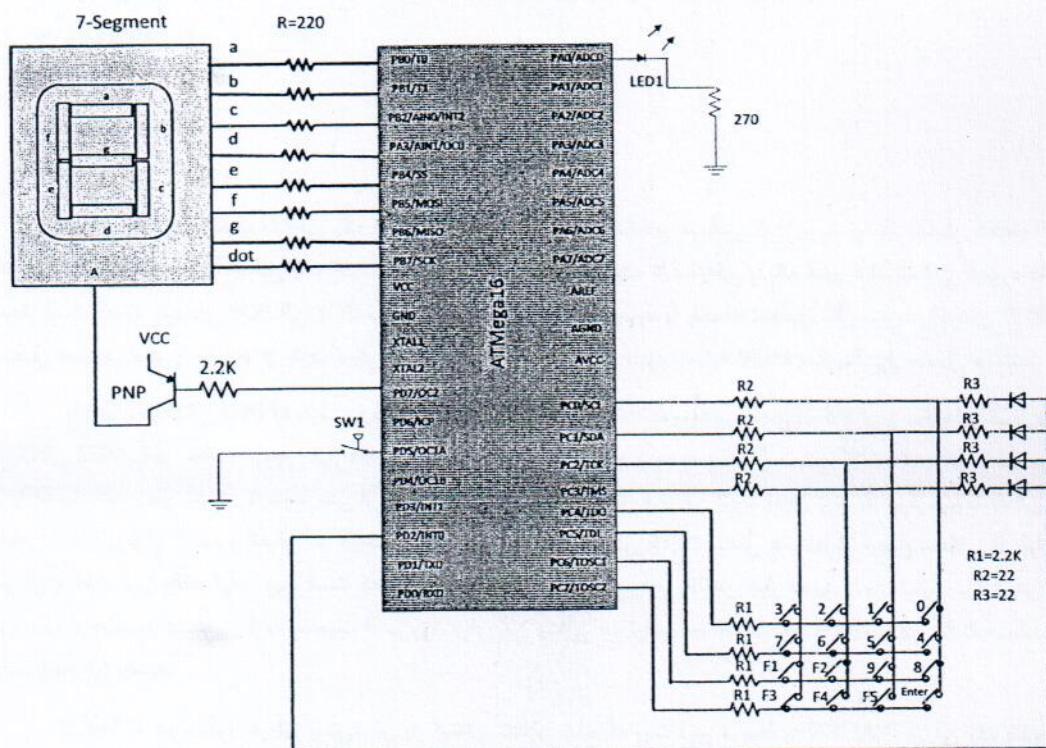
ب- برنامه‌ای بنویسید که یک مقدار بین ۰ الی ۹ و A تا F موجود در ثبات R0 را دریافت و آن را بر روی نمایش دهنده ۷ قطعه‌ای نمایش دهد. معادل هگزادسیمال برای نمایش ارقام ۰ الی ۹ و نقطه ممیز اعشاری از چپ به راست عبارتند از:

`0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80,0x90`

معادل هگزادسیمال برای نمایش ارقام حروف الفبای A الی F به صورت A,b,c,d,E,F توسط نمایش دهنده ۷ قطعه‌ای را نیز خود بدست آورید.

ج- برنامه‌ای بنویسید که با فشردن هر یک، از کلیدهای کیبورد وارد روتین وقفه INT0 شده و در آنجا LED1 روشن گردد.

د- برای راهاندازی بخش کیبورد، در روتین وقفه INT0، زیر روالی بنام KeyFind را صدا بزنید که شماره کلید را محاسبه و آنرا در ثبات R0 برگرداند. سپس به کمک برنامه بند ب، آن مقدار را توسط نمایش دهنده ۷ قطعه‌ای نمایش دهید. تنظیمات اولیه ثبات‌های کنترلی مورد استفاده، برنامه روتین وقفه، برنامه زیر روال KeyFind و سایر اقدامات لازم را ارائه نمائید.



شکل ۱-۹- کیبورد ماتریسی

۱۰- جلسه هشتم

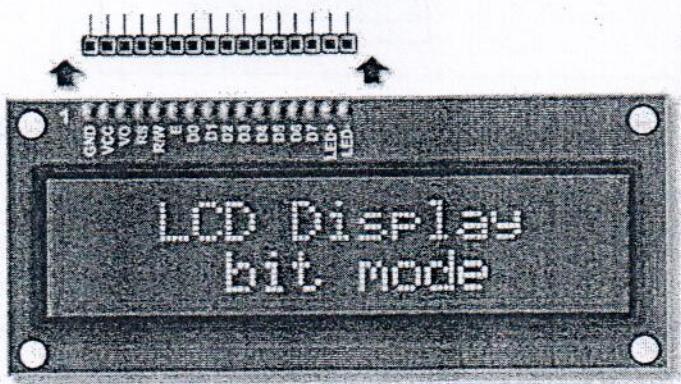
هدف از آزمایش:

- اتصال LCD به میکروکنترلر و نمایش اطلاعات دریافتی از کیبورد توسط آن

قطعات مورد نیاز:

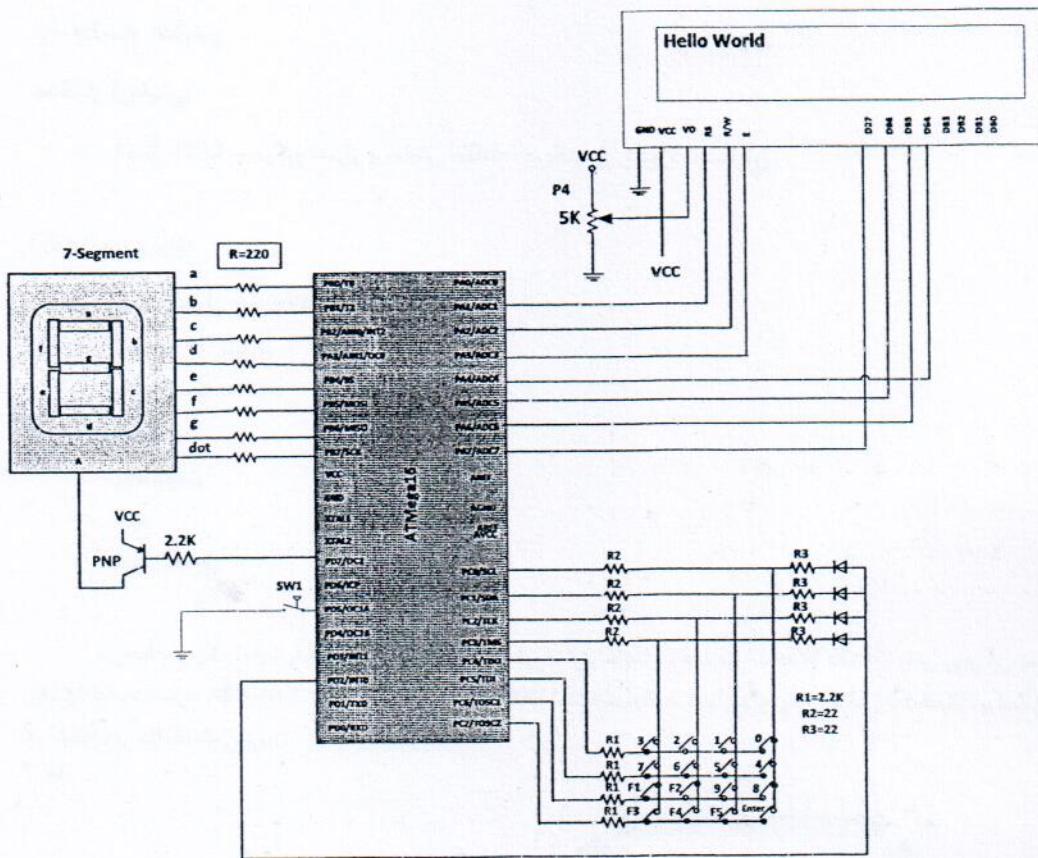
- میکروکنترلر ATmega16
- کیبورد ۱۶ کلیدی
- LCD نمایش دهنده
- مقاومت
- پتانسیومتر
- دیود
- کلید فشاری

می خواهیم یک نمایش دهنده کاراکتری را به میکروکنترلر متصل و عبارت "Hello World" را بر روی آن نمایش دهیم. از یک LCD به شماره A LM02L، TS1620A و یا BONA2002 استفاده نمائید. شما ای از یک LCD و مشخصات و اسمی پایه های آن در شکل زیر ملاحظه می گردید.



شکل ۱-۱۰- نمایش دهنده کاراکتری (LCD) و مشخصات پایه های آن

LCD را همانند شکل ۲-۱۰ به درگاه A میکروکنترلر متصل کنید. برای تنظیم کنترast LED یک پتانسیومتر با مقدار 5K اهم بین پایه VO و GND مطابق شکل قرار دهید. کد اسambilی ارتباط با LCD کاراکتری از طریق اینترنت قابل دسترسی است. یکی از این برنامه ها کد اسambilی LCD_4bit.asm است که از طریق سایت های مشخص شده در مراجع [۱] و [۲] قابل دسترسی است. این مراجع در انتهای این بخش معرفی شده اند. یکی از توابع این برنامه تابع lcd_putchar است که کاراکتر موجود در یکی از ثبات های رجیستر فایل را بر روی LCD نمایش می دهد. سایر توابع مهم موردنیاز در این کد lcd_putchar، lcd_init، LCD_wait، LCD_getaddr و LCD_getchar است. لذا در صورت استفاده از این کد، می بایست درگاه D را در این کد به درگاه A تغییر نام دهید.



شکل ۲-۱۰ - نمایش دهنده کاراکتری (LCD)

الف- عبارت "Hello World" را بر روی این LCD نمایش دهد.

ب- زیرروالی بنام LCD بنویسید که با هر بار صدا زدن، کاراکترهایی را که در یک بلوک از حافظه Flash به آدرس شروع LCDTABLE قرار دارند را نمایش می‌دهد. بایت اول در این بلوک حافظه تعداد کاراکترهاست. بدیهی است که چنانچه بخواهیم از این زیر روال استفاده کنیم، می‌بایست از قبل کاراکترهایی را که باید چاپ شوند و تعداد آنها را (اولین بایت) در این بلوک حافظه قرار داده باشیم.

ج- برنامه‌ای بنویسید که اطلاعات دریافتی از کیبورد را دریافت و پشت سر هم بر روی LCD نمایش دهد.

مراجع:

[1] Christoph Redecker, Using An LCD In 4 bit Mode, <http://www.avrbeginners.net/interfacing/>

[44780_lcd/4bit.html](#)

[2] Christoph Redecker, m8_LCD_4bit.asm code, <http://www.avrbeginners.net/interfacing/>

[44780_lcd/m8_lcd_4bit.asm](#).

11- جلسه نهم

هدف از آزمایش:

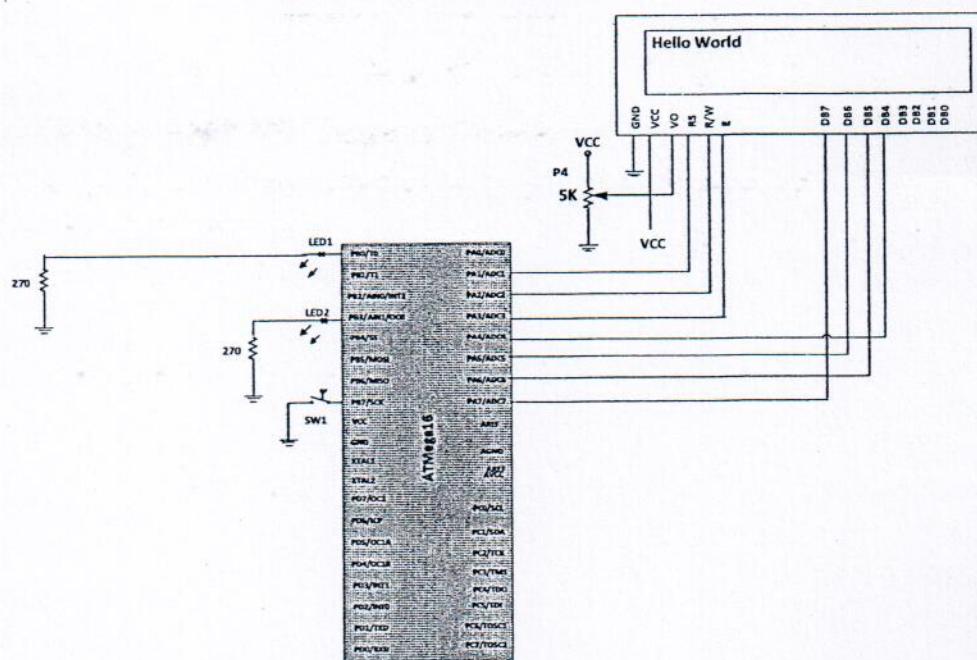
- کار با زمان‌سنج/شمارنده • میکروکنترلر در حالت عملکرد عادی
- کار با زمان‌سنج/شمارنده • میکروکنترلر در حالت CTC

قطعات مورد نیاز:

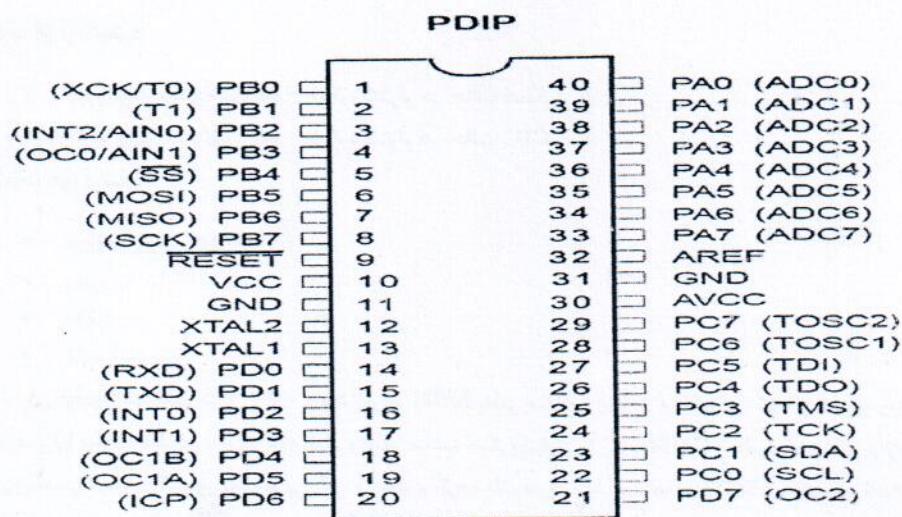
- میکروکنترلر ATmega16
- مقاومت
- LED
- کلید فشاری

الف- می‌خواهیم در مدار شکل ۱-۱۱ دیود نوری LED1 بطور متناوب روشن و خاموش شود و هر بار روشن شدن ۱ ثانیه بطول بیانجامد (با سیکل وظیفه ۰.۵۰٪). برای این منظور ساعت میکروکنترلر را روی ۱MHZ تنظیم کنید. اگر از زمان‌سنج • استفاده کرده و پیش‌ تقسیم کننده را روی تقسیم بر ۱۰۲۴ تنظیم کرده باشید، با ۴ بار مشاهده TOV0، زمان یک ثانیه سپری شده است. چرا؟ تنظیمات ثبات‌های کنترلی زمان‌سنج/شمارنده • و برنامه کار سیستم را بنویسید. از زمان‌سنج/شمارنده • در حالت عملکرد عادی استفاده کنید.

ب- می‌خواهیم که در مدار شکل ۱-۱۱ دیود نوری LED2 با فرکانس ۲ هرتز روشن و خاموش شود. برای این منظور ساعت میکروکنترلر را روی ۱MHZ تنظیم کنید. از زمان‌سنج/شمارنده • در حالت پاک کردن زمان‌سنج در برابر مقایسه (مود CTC) استفاده کرده و با استفاده از وقفه ناشی از مقایسه و مطابقت (compare-match)، دیود نوری LED2 را با فرکانس ۲ هرتز روشن و خاموش کنید. اسمی پایه‌های میکروکنترلر از جمله پایه OC0 که به پایه PB3 متصل است در شکل ۱-۱۱ ارائه شده است.



شکل ۱-۱۱- اتصال LED‌ها به درگاه B میکروکنترلر



شکل ۲-۱۱- اسامی پایه‌های میکروکنترلر

در صورت داشتن فرصت کافی می‌توان آزمایش بعدی را در همین جلسه انجام داد.

۱۲- جلسه دهم

هدف از آزمایش:

- کار با زمان سنج/شمارنده ۰ و ۱ میکروکنترلر
- شمارش پالس های موجود بر روی پایه T1
- اندازه گیری فرکانس

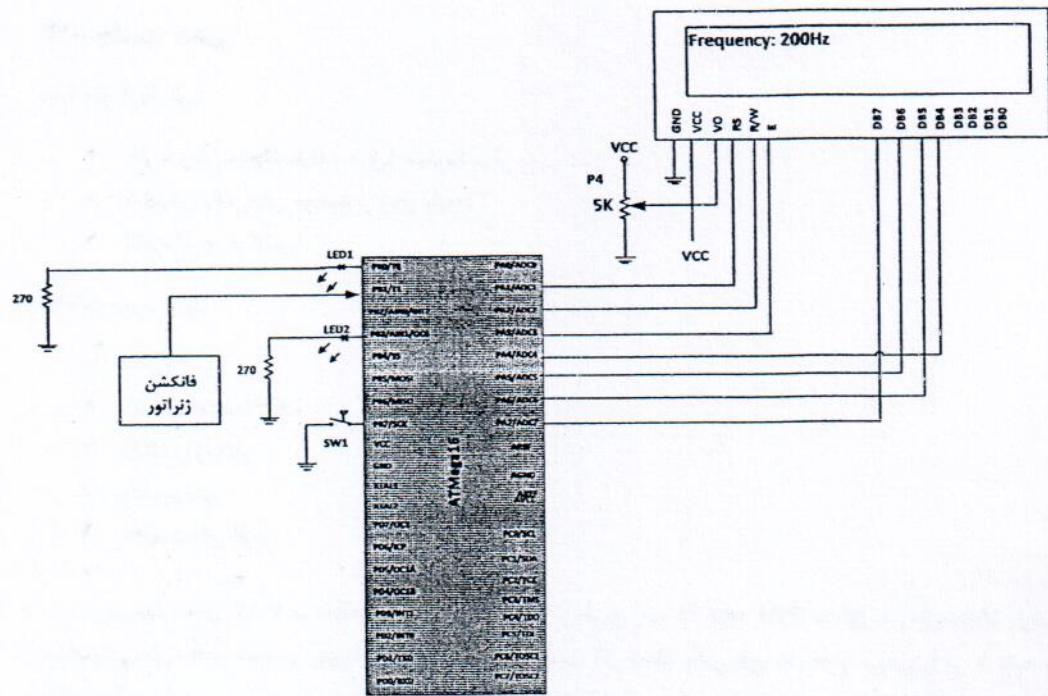
قطعات مورد نیاز:

- میکروکنترلر
- نمایش دهنده LCD
- فانکشن ژنراتور
- پتانسیومتر
- مقاآمت های لازم

در مدار شکل ۱-۱۲ می خواهیم یک موج مربعی را از طریق پایه T1 (پایه PB1) متعلق به زمان سنج/شمارنده ۱ دریافت و تعداد پالس های ظاهر شده بر روی این پایه را شمارش کنیم. اگر تعداد یانس های یک موج مربعی را در ۱ ثانیه شمارش کنیم فرکانس آن موج مربعی بدست خواهد آمد. از زمان سنج/شمارنده ۰ برای تولید زمان ۱ ثانیه استفاده می کنیم. با راه اندازی زمان سنج/شمارنده ۰، شمارش پالس های دریافتی روی ورودی T1 را آغاز می کنیم و هنگامی که زمان ۱ ثانیه سپری شد، مقدار موجود در ثبات زمان سنج/شمارنده ۱ و ۱ پس از تبدیل به یک عدد ددهدی بر روی نمایش دهنده LCD نمایش می دهیم. برای سهولت فرض کنید که فرکانس موج ورودی حداکثر ۲۵۵ هرتز می باشد. با اتصال یک عدد فانکشن ژنراتور و تولید یک موج مربعی با دامنه تغییرات ۰ و ۵ ولت و اتصال آن به پایه T1، مدار فوق را امتحان نمائید.

اقدامات زیر را انجام دهید:

- الف- سخت افزار برد آموزشی را مطابق شکل ۱-۱۲ برای انجام این آزمایش آماده کنید.
- ب- تنظیمات ثبات های کنترلی زمان سنج/شمارنده های شماره ۰ و ۱ را انجام دهید.
- ج- برنامه کار سیستم را بنویسید و سیستم را راه اندازی و روند آزمایش و مشاهدات خود را گزارش نمایید.



شکل ۱-۱۲ - مدار کار با زمان سنج/شمارندها

۱۳- جلسه یازدهم

هدف از آزمایش:

- کار با زمان سنج/شمارنده • در حالت PWM سریع
- کار با زمان سنج/شمارنده • در حالت PWM با فاز صحیح
- تنظیم شدت روشنایی LED یا موتور DC ساده توسط موج PWM

قطعات مورد نیاز:

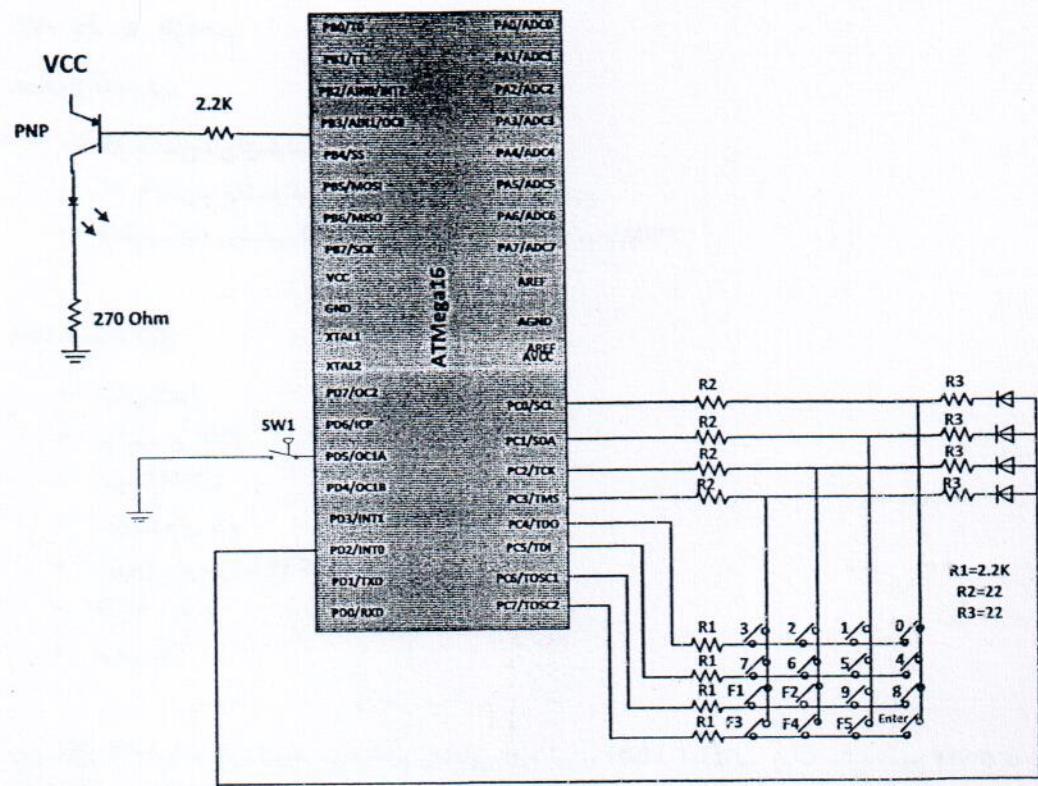
- میکروکنترلر
- ترانزیستور PNP 1N4148
- دیود
- مقاومت‌های لازم
- کیبورد ماتریسی با ۱۶ کلید
- LED: ۱ عدد
- موتور DC

مدار شکل ۱-۱۳ را درنظر بگیرید. می‌خواهیم روشنایی دیود نوری LED1 را با تغییر چرخه وظیفه موج PWM که از طریق پایه OC0 متصل به پایه PB3 تولید می‌شود تنظیم نمائیم می‌خواهیم برنامه کار این مدار را به گونه‌ای بنویسید که متناسب با رقمی که توسط کیبورد وارد می‌نمایید، میزان روشنایی LED بین حداقل مقدار (بازه رقم ۰) و حداکثر مقدار (بازه رقم ۹) تغییر نماید. خواسته فوق را در دو حالت زیر به انجام رسانید: با زیاد شدن سیکل وظیفه موج ایجاد شده، آیا شدت نور LED زیاد می‌شود یا کم؟ چرا؟

الف- تنظیمات ثبات‌های کنترلی زمان سنج/شمارنده • و برنامه کار سیستم را برای حالتی که از زمان سنج/شمارنده • در مود PWM سریع استفاده شده باشد انجام دهید.

ب- تنظیمات ثبات‌های کنترلی زمان سنج/شمارنده • و برنامه کار سیستم را برای حالتی که از زمان سنج/شمارنده • در مود PWM با فاز صحیح استفاده شده باشد انجام دهید.

ج- در صورت در اختیار داشتن یک موتور الکتریکی DC کوچک آنرا به جای دیود نوری بین کلکتور ترانزیستور و پایه زمین قرار داده و دور موتور را به کمک موج PWM در دو حالت فوق کنترل کنید. برای این منظور مقاومت ۲۷۰ اهم را تا حد مناسب کرده یا حذف کنید.



شكل ١-١٣ - كنترل شدت نور LED توسط موج PWM

۱۴- جلسه دوازدهم

هدف از آزمایش:

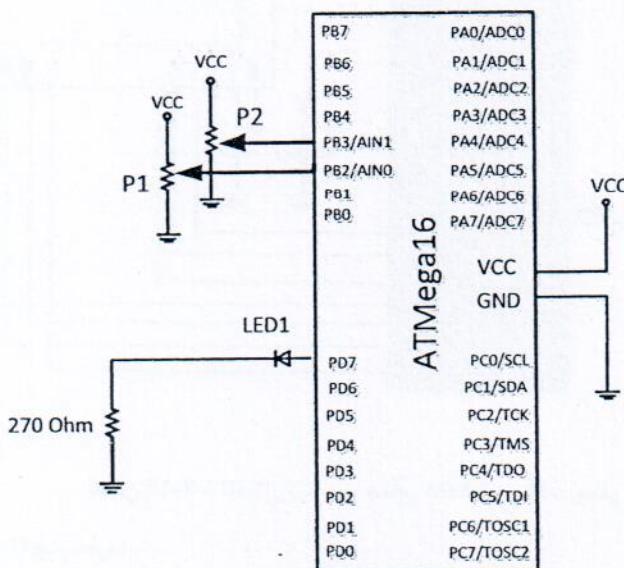
- کار با مقایسه کننده آنالوگ میکروکنترلر
- کار با مبدل آنالوگ به دیجیتال میکروکنترلر
- اندازه‌گیری دما و نمایش بر روی LCD

قطعات مورد نیاز:

- میکروکنترلر
- سنسور دمای LM35
- نمایش دهنده LCD
- پتانسیومتر
- LED

کار با مقایسه کننده آنالوگ:

در این آزمایش ابتدا می‌خواهیم با نحوه کار مقایسه کننده آنالوگ آشنا شویم. برای این منظور مدار شکل ۱-۱۴ را بر روی یک عدد برد بیندید. در اینجا از دو عدد پتانسیومتر ۵ یا ۱۰ کیلواهم استفاده شده است. سر بالای هر پتانسیومتر به VCC سریائین به زمین و سر وسط به پایه AIN0 یا AIN1 متصل شده است.



شکل ۱-۱۴ - اتصال پتانسیومرهای P1 و P2 به ورودی‌های مقایسه کننده آنالوگ

از ساعت کالیبره شده داخلی با فرکانس ۱MHz استفاده نمایید. مطابق شکل ۱-۱۴ سر وسط یک پتانسیومتر P1 را به پایه AIN0 (سر مثبت مقایسه کننده آنالوگ) و سر وسط پتانسیومتر P2 را به پایه AIN1 (سر منفی مقایسه کننده آنالوگ) متصل

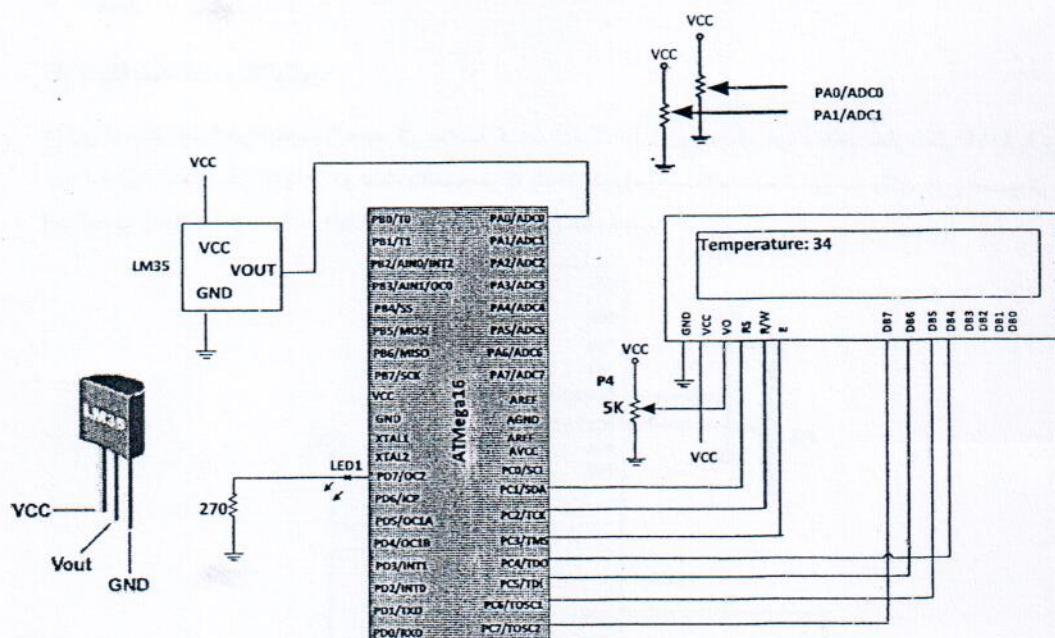
نمایند. می خواهیم چنانچه ولتاژ ورودی به پایه AIN0 مقایسه کننده آنالوگ از ورودی به پایه AIN1 بیشتر شود، در روتین وقفه مقایسه کننده آنالوگ، دیود نوری LED1 متصل به پایه PD7 روشن و در غیر اینصورت خاموش شود.

الف- ثبات های کنترلی مربوط به مقایسه کننده آنالوگ را برنامه ریزی نماید.

ب- برنامه کار سیستم فوق را بنویسید.

کار با مبدل آنالوگ به دیجیتال:

می خواهیم از مبدل آنالوگ به دیجیتال برای اندازه گیری دما توسط یک سنسور در شکل ۲۴-۵ استفاده نماییم. این سنسور در شکل نشان داده شده است. ولتاژ دریافتی بر روی سر وسط این سنسور مناسب با دمای محیط تغییر می نماید. با جستجو در اینترنت و یافتن دیتابیس این سنسور دما، با مشخصات آن آشنا شوید و برسی کنید که بازاء هر درجه سانتیگراد ولتاژ خروجی آن چند میلی ولت تغییر می نماید. خروجی سنسور دما را به پایه ADC0 (پایه PA0) میکروکنترلر متصل کنید. از LCD برای نمایش دمای بین ۰ الی ۹۹ درجه سانتیگراد استفاده نمایید.



شکل ۲-۱۴- اتصال سنسور دمای LM35 و پتانسیومتر به میکروکنترلر

اقدامات زیر را انجام دهید:

الف- ثبات های کنترلی مربوط به مبدل آنالوگ به دیجیتال را برنامه ریزی نماید.

ب- تنظیمات ثبات های کنترلی مورد نیاز و نیز برنامه کار سیستم را برای اندازه گیری دما و نمایش آن بر روی LCD بنویسید.

ج- در یک آزمایش دیگر سنسور دما را از پایه PA0 جدا کرده و سرهای وسط پتانسیومترهای P1 و P2 را مطابق شکل ۲۴-۵ به ورودی‌های ADC0 (PA0) و ADC1 (PA1) میکروکنترلر متصل نمائید و اختلاف ولتاژ بین آنها را توسط مبدل آنالوگ به دیجیتال اندازه‌گیری و در صورت مثبت یا منفی بودن این اختلاف ولتاژ، LED متصل به پایه PD7 را روشن و خاموش نمائید. با ثابت نگه داشتن پیج تنظیم پتانسیومتر P1 و چرخانیدن پیج تنظیم پتانسیومتر P2 می‌توانید اختلاف ولتاژها را منفی و مثبت نمائید و مقادیر اختلاف ولتاژ را بر روی LCD نمایش دهید. نظمیمات ثبات‌های کنترلی مورد نیاز و نیز برنامه کار سیستم را بنویسید.

۱۵- جلسه سیزدهم

هدف از آزمایش:

- کار با واسط USART میکروکنترلر
- آشنایی با تراشه ۲۲ MAX232
- برقراری ارتباط بین یک میکروکنترلر و یک کامپیوتر از طریق ارتباط RS232 با استفاده از واسط USART در طرف میکروکنترلر و USART Serial Connection .NET Component یا ترمینال CodeVision یا هر ترمینال مناسب دیگر در طرف کامپیوتر

قطعات و برنامه‌های مورد نیاز:

- میکروکنترلر
- مقاومت‌های لازم
- خازن‌های مورد نیاز
- کیبورد ماتریسی با ۱۶ کلید
- تراشه ۲۲ MAX232
- کانکتور DB9
- مبدل USB به RS232
- LCD
- برنامه ترمینال

برای برقراری ارتباط همگام^۱ یا ناهمگام^۲ مابین دو وسیله مختلف می‌توان از واسط ارسال و دریافت سریال همگام -ناهمگام USART^۳ استفاده نمود، یعنی فرستنده/گیرنده‌ای که قادر است داده‌ها را بطور سریال و بصورت همگام یا ناهمگام ارسال و دریافت کند. ارتباط سریال همگام به آن نوع ارتباط سریال گفته می‌شود که فرستنده همراه با بیت‌های داده ارسال شده پالس ساعت را نیز ارسال کند و در حالت گیرنده، همگام با پالس ساعت، بیت داده دریافت می‌شود. پس در این نوع ارتباط علاوه بر خطوط ارسال و دریافت داده خط پالس ساعت هم لازم است. در ارتباط سریال ناهمگام پالس ساعت حذف می‌شود و با تنظیم یکسان سرعت انتقال داده در فرستنده و گیرنده، فقط یک خط برای ارسال و یک خط برای دریافت داده لازم است. ارسال روی پایه TxD و دریافت روی پایه RXD انجام می‌شود و در صورت عملکرد همگام پالس ساعت روی پایه XCK جابجا می‌شود. داده‌ها به ترتیب از بیت کم ارزش به سمت بیت پرازش مبادله می‌شوند.

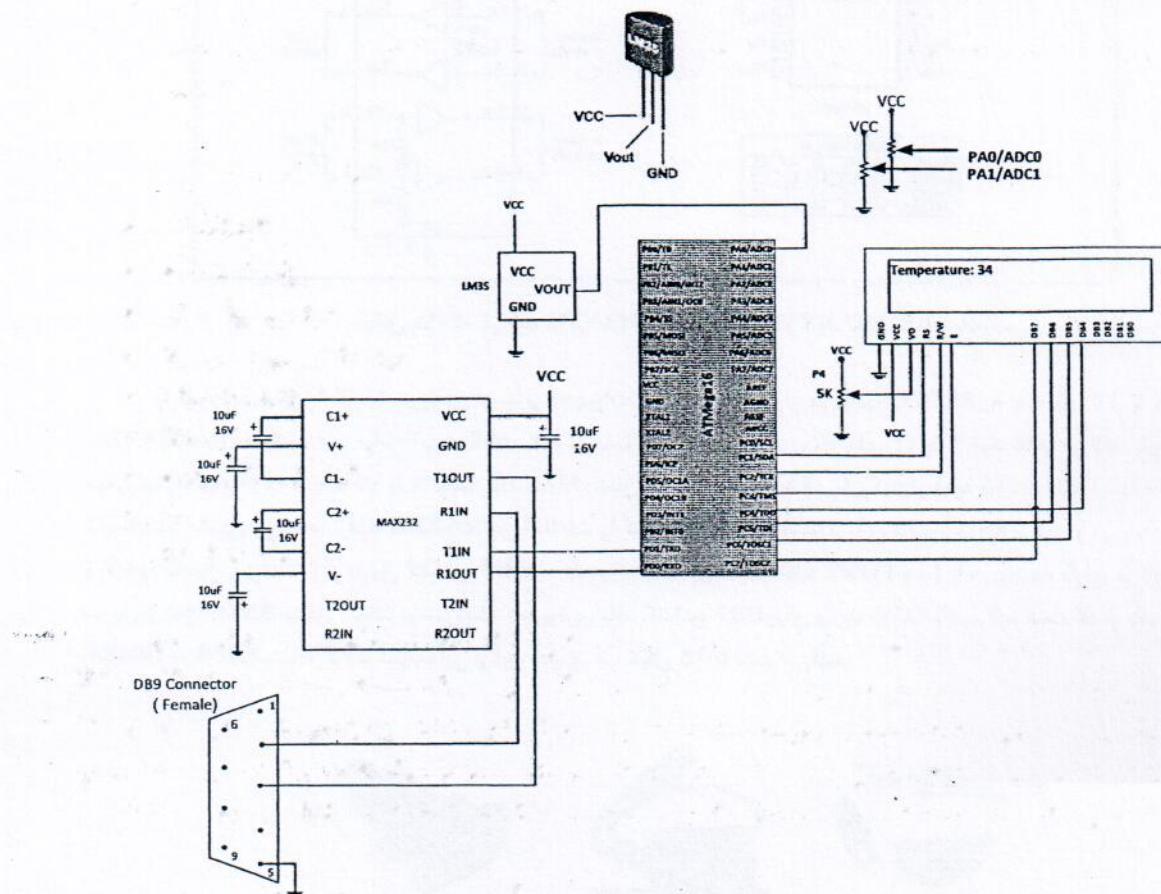
انتخاب مد همگام و یا ناهمگام به عهده طراح می‌باشد که با توجه به نوع دستگاه جانبی که هدف برقرار ارتباط سریال با آن است، صورت می‌پذیرد. یکی از کاربردهای رایج واسط ارسال و دریافت سریال همگام -ناهمگام، دریافت اطلاعات محیط از طریق سنسورها و ارسال این اطلاعات به رایانه به منظور انجام پردازش‌های لازم بر روی این داده‌ها می‌باشد. این ارتباط از طریق پایه‌های ۱۴ و ۱۵ میکروکنترلر ATmega16 یا ATmega32 انجام می‌شود. این پایه‌ها با نام (RXD) و (TXD) مشخص شده‌اند.

^۱ Synchronous

^۲ Asynchronous

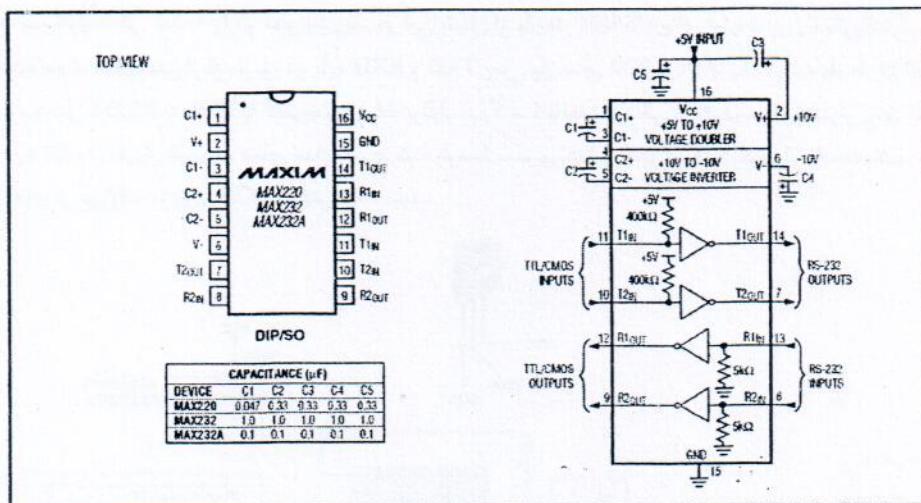
^۳ UniversalSynchronous/Asynchronous Receiver/Transmitter (USART)

سخت افزار شکل ۱-۱۵ را در نظر بگیرید. در این شکل از واسط USART برای ارتباط بین میکروکنترلر و یک کامپیووتر استفاده شده است. میکروکنترلر از طریق دو پایه RXD و TXD یعنی پایه های PD0 و PD1، یک تراشه واسط RS232، یک کانکتور DB9 و یک مبدل RS232 به USB با کامپیووتر ارتباط برقرار می کند. اطلاعات ارسالی از طرف میکروکنترلر برای کامپیووتر می تواند داده های وارد شده از طریق کیبورد، دمای اندازه گیری شده توسط سنسور دما و یا هر دنباله دیگری از داده ها باشد. داده های ارسالی از طرف کامپیووتر می تواند بر روی LCD نمایش داده شود.



شکل ۱-۱۵ - ارتباط سریال از طریق واسط USART

به منظور برقراری ارتباط سریال برای ارسال و دریافت همگام-ناهمگام اطلاعات بین تجهیزات مختلف از تراشه هایی جهت تبدیل اطلاعات از قالب TTL به قالب هایی مانند قالب R232C استفاده می شود. یکی از این تراشه ها تراشه MAX232 می باشد. ساختار تراشه MAX232 در شکل ۲-۱۵ آمده است.



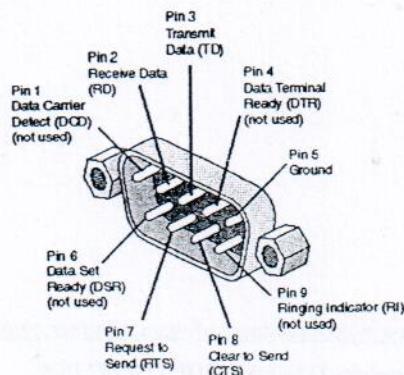
شکل ۲-۱۵- تراشه MAX232 برای تبدیل TTL به RS232 و بالعکس

با توجه به شکل، اطلاعات دریافتی از سایر تجهیزات مثل یک کامپیوتر در قالب RS232C به پایه‌های ۱۳ یا ۸ تراشه MAX232 اعمال می‌شود. این تراشه این اطلاعات را به قالب TTL تبدیل و بر روی پایه‌های ۱۲ یا ۹ خود ظاهر می‌کند. پایه RXD میکروکنترلر (پایه ۱۴) به پایه ۱۲ یا ۹ تراشه MAX232 متصل می‌شود. پایه TXD میکروکنترلر (پایه ۱۵) به پایه ۱۱ یا ۱۰ تراشه MAX232 متصل می‌گردد. تراشه MAX232 این اطلاعات را که در قالب TTL هستند را به قالب RS232 تبدیل و در پایه‌های ۱۴ و ۷ خود تحویل می‌دهد تا از طریق کانکتور DB9 به کامپیوتر (مثلًا پورت COM2، COM1 و ...) یا هر وسیله دارای درگاه ارتباط سریال از نوع RS232 منتقل شوند. درصورتیکه کامپیوتری فاقد کانکتور DB9 برای پورت COM باشد، لازم است که از یک آداپتور USART به استفاده گردد. نمونه‌هایی از این آداپتور در شکل ۳-۱۵ دیده می‌شود.



شکل ۳-۱۵- نمونه‌هایی از آداپتور USART به USB

مشخصات پایه‌های کانکتور DB9 از نوع Male در شکل ۴-۱۵ نشان داده شده است.



شکل ۴-۱۵ پایه‌های کانکتور ۹ از نوع DB9 Male

به منظور برقراری ارتباط از طریق کامپیوتر با واسط سریال، USART، می‌توانید از کامپوننت‌هایی که برای این منظور بصورت آماده موجود است استفاده نمایید. برای این منظور می‌توانید از کامپوننت USART Serial Connection .NET Component در محیط برنامه‌نویسی .NET استفاده و برنامه‌ای برای ارسال و دریافت سریال از طریق واسط USART بنویسید و بدین ترتیب بین کامپیوتر و میکروکنترلر ارتباط برقرار نمایید. جهت اطلاعات بیشتر به پیوست ۲ مراجعه نمایید. راه دیگر برقراری ارتباط با کامپیوتر با استفاده از ارتباط USART استفاده از برنامه Hyper Terminal یا بخش ترمینال در نرمافزار Codevision است. توضیحات لازم برای استفاده از این برنامه در پیوست ۳ آمده است.

می‌خواهیم ارتباط بین میکروکنترلر و کامپیوتر بصورت ناهمگام و با نرخ بیت ۴۸۰۰ بیت در ثانیه و با فریم‌های داده حاوی یک، بیت شروع، ۷ بیت، داده، توازن، فعال از نوع زوج و ۱ بیت پایانی انجام گردد. فرکانس ساعت میکروکنترلر $f_{osc} = 1 \text{ MHz}$ باشد و با ساعت کالیبره شده داخلی تولید شود. در اینصورت اختلاف بین نرخ بیت تولید شده و نرخ بیت واقعی ۴۸۰۰ بیت در ثانیه در حدود ۲٪ است. به منظور برقراری ارتباط بین میکروکنترلر و کامپیوتر اقدامات زیر را انجام دهید.

الف- ثبات‌های کنترلی مربوط به واسط USART میکروکنترلر را برنامه‌ریزی نمایید.

ب- برنامه لازم برای ارسال اطلاعات تایپ شده از طریق کیبورد کامپیوتر و نمایش آنها بر روی نمایش‌دهنده LCD را بنویسید.

ج- برنامه لازم برای ارسال اطلاعات تایپ شده از طریق کیبورد متصل به میکروکنترلر به کامپیوتر و نمایش آنها بر روی مانیتور کامپیوتر را بنویسید.

١٦- مراجع

- [1] ATmega32 microcontroller datasheets.
- [2] CodeVisionAVR C compiler, User manual.
- [3] AVR Assembler, Atmel.
- [4] AVR Studio User Guide, Atmel.
- [5] Codevision software software, 2008.
- [6] <http://www.kpsec.freeuk.com>
- [7] Khazam AVR Programmer, <http://khazama-avr-programmer.software.informer.com>.
- [8] DS1307 RTC, http://www.wvshare.com/datasheet/DALLAS_PDF/DS1307.PDF
- [9] LM75A temperature sensor datasheet, http://www.nxp.com/documents/data_sheet/LM75A.pdf
- [10] 24LC64 serial EEPROM, http://www.datasheetcatalog.com/datasheets_pdf/2/4/L/C/24LC64.shtml
- [11] Christoph Redecker, Using An LCD In 4 bit Mode, http://www.avrbeginners.net/interfacing/44780_lcd/4bit.html
- [12] Christoph Redecker, m8_LCD_4bit.asm code, http://www.avrbeginners.net/interfacing/44780_lcd/m8_lcd_4bit.asm.

۱۷- پیوست ۱: پایه‌های میکروکنترلر ATMega32

شکل زیر پایه‌های میکروکنترلر ATMega32 را نشان می‌دهد. پایه‌های درگاه‌های A، B، C و D هر کدام به یک رنگ مشخص شده‌اند.

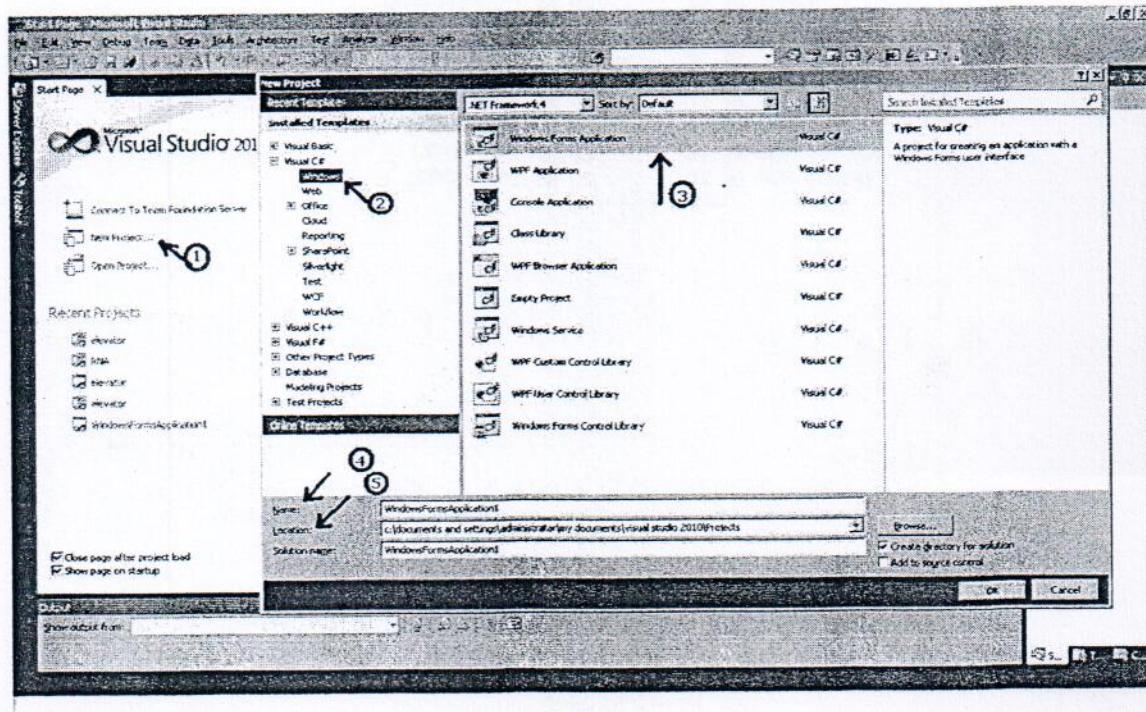
(XCK/T0) PB0	1	40	PA0 (ADC0)
(T1) PB1	2	39	PA1 (ADC1)
(INT2/AIN0) PB2	3	38	PA2 (ADC2)
(OC0/AIN1) PB3	4	37	PA3 (ADC3)
(SS) PB4	5	36	PA4 (ADC4)
(MOSI) PB5	6	35	PA5 (ADC5)
(MISO) PB6	7	34	PA6 (ADC6)
(SCK) PB7	8	33	PA7 (ADC7)
RESET	9	32	AREF
VCC	10	31	GND
GND	11	30	AVCC
XTAL2	12	29	PC7 (TOSC2)
XTAL1	13	28	PC6 (TOSC1)
(RXD) PD0	14	27	PC5 (TDI)
(TXD) PD1	15	26	PC4 (TDO)
(INT0) PD2	16	25	PC3 (TMS)
(INT1) PD3	17	24	PC2 (TCK)
(OC1B) PD4	18	23	PC1 (SDA)
(OC1A) PD5	19	22	PC0 (SCL)
(ICP1) PD6	20	21	PD7 (OC2)

۱۸- پیوست ۲ (نحوه ارتباط یک برنامه Microsoft Visual Studio با میکروکنترلر)

به منظور اینکه داده‌های مابین میکروکنترلر و یک محیط نرم افزاری رد و بدل شوند می‌توان از Microsoft Visual Studio استفاده نمود. به طور خاص تر می‌توانیم از زبان برنامه نویسی C# در این محیط نرم افزاری استفاده نماییم. نحوه ایجاد یک پروژه C# در 2010 Visual Studio و یک برنامه ساده که برای ارتباط با میکروکنترلر باید نوشته شود، در ادامه بیان می‌شود.

برای برقراری ارتباط همگام یا غیرهمگام مابین واحد پردازشگر مرکزی و دستگاه‌های جانبی از واسط ارسال و دریافت سریال هم‌زمان-غیرهم‌زمان USART استفاده می‌شود. برای این که برنامه نرم افزاری بتواند با میکروکنترلر ارتباط برقرار کند باید امکان داشته باشد که بتواند با پورت سریال ارتباط برقرار کند. به این منظور از یک component پورت سریال در برنامه استفاده می‌کنیم.

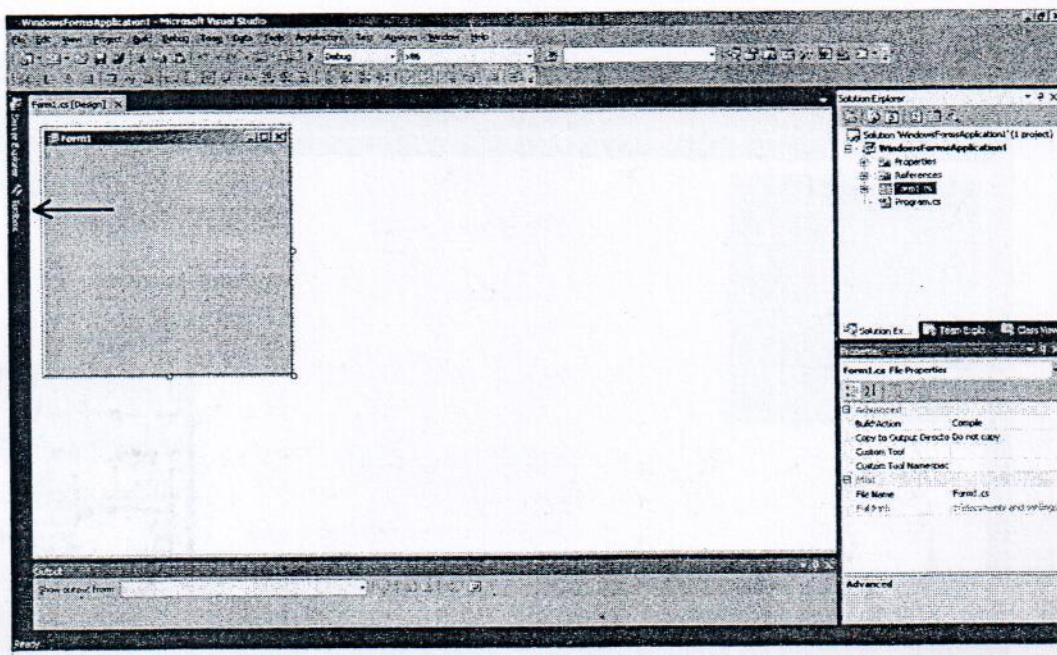
برای این منظور برنامه 2010 Visual Studio Start Page را اجرا کنید. در لبه Start Page (همان پنجره آغازین)،... را انتخاب کرده و در پنجره باز شده، زبان برنامه نویسی را روی C# قرار دهید و نوع Windows را انتخاب کنید و از بین option موجود، Windows Forms Application را انتخاب کنید، در آخر نیز نام پروژه و مکان ذخیره‌سازی آن را مشخص کنید. مراحل ذکر شده، در شکل پایین مشخص است.



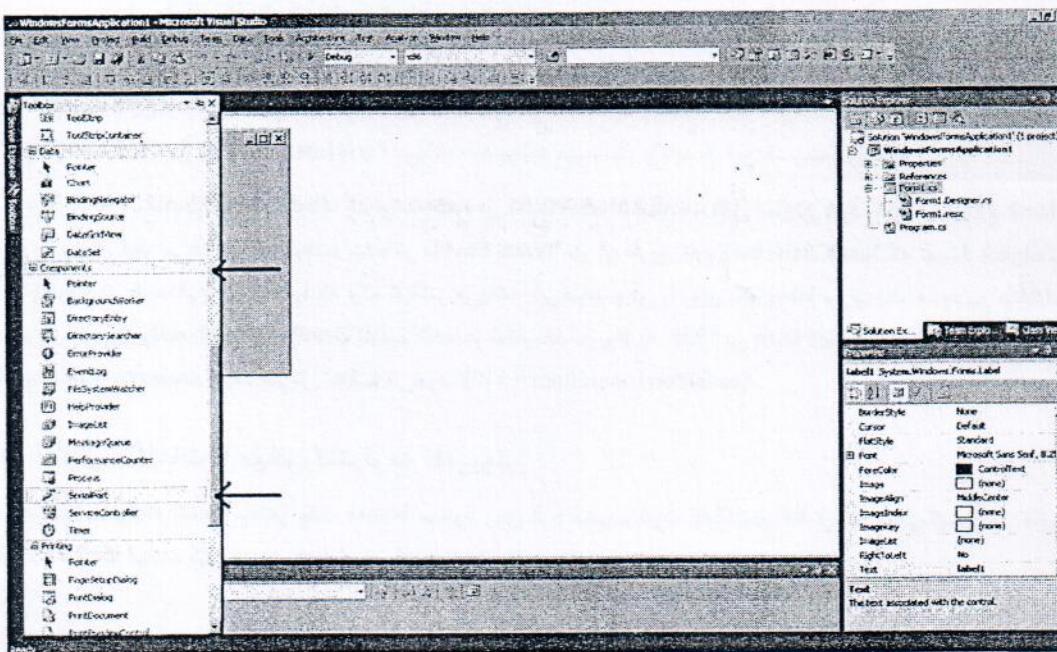
شکل ۱-۱۸- ایجاد پروژه انتقال سریال داده بین کامپیوتر و میکروکنترلر

پس از زدن کلید OK پروژه، ایجاد می‌شود و محیطی مانند شکل ۱-۱۸ مشاهده می‌شود. در سمت راست این محیط، یک Solution Explorer مشاهده می‌گردد که یک لیست از فایل‌های تولید شده توسط Visual Studio در طول ایجاد پروژه است. Form1.cs را انتخاب کرده و Form1 را مشاهده کنید. این همان فرمی است که قرار است با اجرای برنامه، ظاهر شود. به این فرم که در واقع یک Windows Forms است می‌توانید component‌های دلخواه را اضافه نمایید. با توجه به فلش موجود در شکل پایین،

نشانگر موس را روی لبه Toolbox قرار داده تا یک لیست از ابزارهای موجود، به نمایش درآیند. در شکل بعد، این لیست را مشاهده می‌کنید.

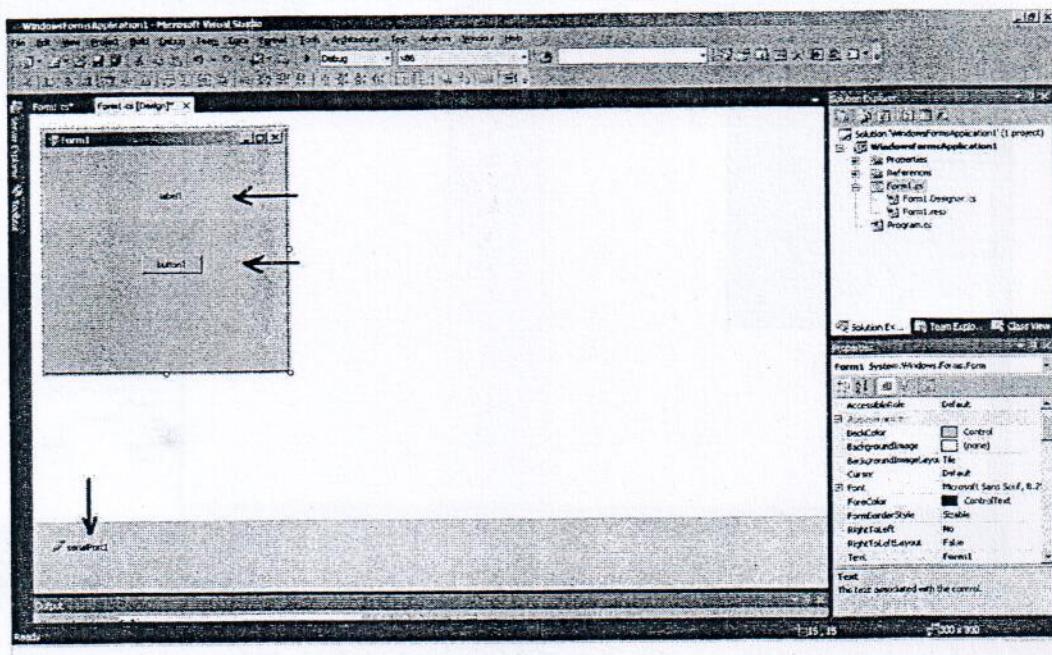


شکل ۲-۱۸- ایجاد فرم برای اضافه کردن کامپوننت‌های مورد نیاز



شکل ۳-۱۸- مشاهده لیست ابزارها

از قسمت Components یک Label و یک Button و از قسمت SerialPort یک drag را کرده و در drop Form1 نمایند.



شکل ۴-۱۸- انتخاب یک serial port component از قسمت serial port و قرار دادن آن در فرم

حال روی فایل Form1.cs در Solution Explorer کلیک راست کنید و View Code را انتخاب نمایید تا یک پنجره جدید ظاهر شود. پنجره جدید باز شده با نام Form1.cs می‌باشد که برنامه مورد نظر را باید در این قسمت بنویسیم.

در صورت استفاده از CodeVision، کد نوشته شده در ReadChar() اطلاعات قابل نمایش خود را از طریق تابع putchar() روی پورت سریال قرار می‌دهد و کد نوشته شده در Visual Studio نیز از طریق تابع ReadChar(char) که توسط object که توسط پورت سریال فراخوانی می‌شود، این اطلاعات را که یک کاراکتر می‌باشد می‌خواند و آن را روی یک label می‌نویسد. همچنین BaudRate پورت سریال در این برنامه باید با BaudRate تنظیم شده در تنظیمات مربوط به Usart در CodeVision یکی باشد که این کار نیز توسط property BaudRate پورت سریال تنظیم می‌شود. (serialPort1.BaudRate = 110)

۱-۱۸- انتقال اطلاعات از میکروکنترلر به کامپیوتر

در برنامه نوشته شده زیر، هر وقت button موجود روی فرم فشرده شود، کاراکتری که توسط میکروکنترلر به کامپیوتر فرستاده شده است خوانده شده و روی label روی فرم نمایش داده می‌شود.

```
#using ..

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            this.Show();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            serialPort1.Open();
            if (serialPort1.IsOpen)
            {
                char a;
                serialPort1.BaudRate = 110;
                a = (char)serialPort1.ReadChar();
                label1.Text = a.ToString();
            }
            else label1.Text = "Serial Port is Closed";
            serialPort1.Close();
        }
    }
}
```

شکل ۵-۱۸- برنامه انتقال اطلاعات از میکروکنترلر به کامپیوتر

۱۸-۲- انتقال اطلاعات از کامپیوتر به میکروکنترلر

در برنامه نوشته شده زیر، هر وقت button موجود روی فرم فشرده شود، کاراکتری توسط کامپیوتر به میکروکنترلر فرستاده می‌شود و میکروکنترلر توسط متده `getchar()`، کاراکتر ارسالی را دریافت می‌کند.

```
#using ..

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

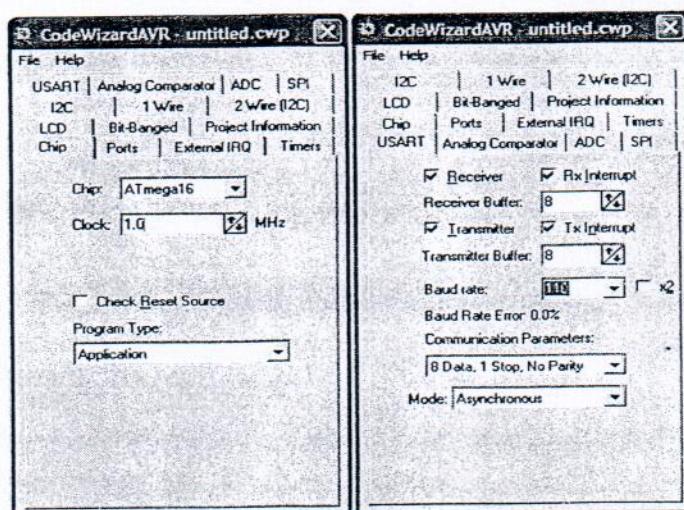
        private void Form1_Load(object sender, EventArgs e)
        {
            this.Show();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            serialPort1.Open();
            if (serialPort1.IsOpen)
            {
                char[] buffer = new char[1];
                buffer[0] = 'a';
                serialPort1.BaudRate = 110;
                serialPort1.Write(buffer, 0, 1);
            }
            else label1.Text = "Serial Port is Closed";
            serialPort1.Close();
        }
    }
}
```

شکل ۶-۱۸- برنامه انتقال اطلاعات از کامپیوتر به میکروکنترلر

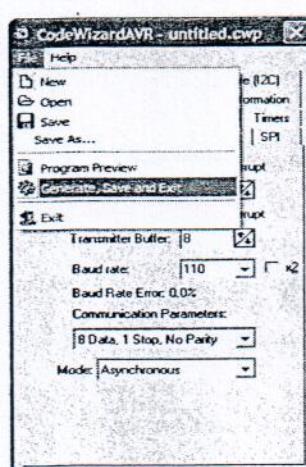
۱۹- پیوست ۳ (ارتباط با کامپیوتر از طریق برنامه Hyper Terminal)

در این پیوست به چگونگی برقراری ارتباط مابین میکروکنترلر و کامپیوتراز طریق برنامه Hyper Terminal میپردازیم. برنامه Hyper Terminal یکی از برنامه‌های تحت ویندوز است که در نسخه‌های قدیمی ویندوز در Accessories فاردار است. در نسخه‌های جدید ویندوز این برنامه را باید نصب نمود. به منظور استفاده راحتتر برای کار با Hyper Terminal در اینجا برنامه بخش مرتبط با میکروکنترلر را با استفاده از ویزاردهای Codevision تهیه می‌نماییم. ابتدا نرم افزار Code Vision را اجرا کرده و بعد از ایجاد پروژه جدید در ویزارد باز شده و در قسمت USART را انتخاب کرده و مقدار فرکانس پالس ساعت را ۱.۰ MHZ تعیین کنید. سپس در قسمت USART اطلاعات را مانند شکل ۱-۱۹ وارد کنید. با اینکار ارتباط سریال می‌تواند با نرخ بیت ۲۴۰۰ بیت در ثانیه، ۸ بیت داده، یک بیت پایانی و بدون پریتی انجام شود. البته می‌توانید این مشخصات را به دلخواه تغییر دهید.



شکل ۱-۱۹- ویزارد برای تنظیم USART؛ در شکل سمت راست Baude rate برابر با ۲۴۰۰ را انتخاب کنید.

در نهایت مانند شکل ۲-۱۹ کد را Generate کنید.



شکل ۲-۱۹- تولید و ذخیره برنامه

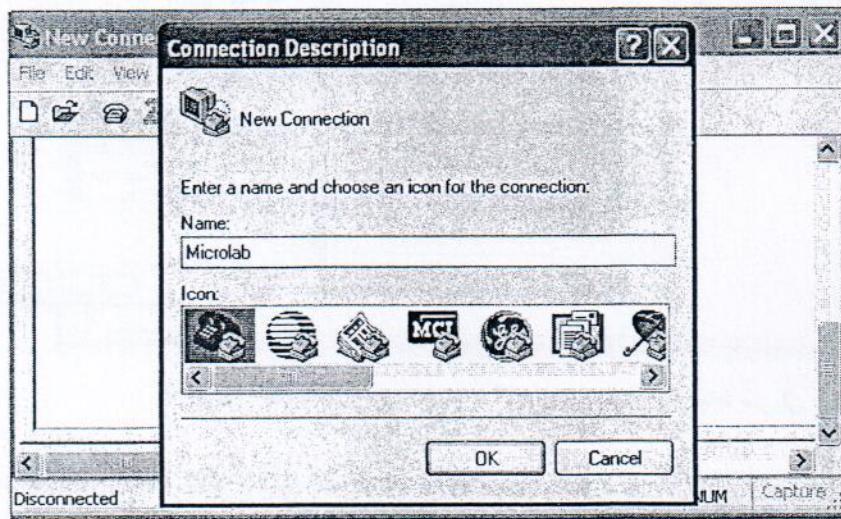
با یک دستور ساده یک کاراکتر را بر روی کامپیوتر و از طریق Hyper Terminal نمایش می‌دهیم. برای این منظور در قسمت برنامه میکروکنترلر دستور ساده زیر را می‌نویسیم:

```
while (1){
    // Place your code here
    putchar ('b');
}
```

این کد پس از برنامه‌ریزی بر روی میکروکنترلر، با آمدن هر پالس ساعت و به طور مداوم کاراکتر 'b' را از طریق Hyper Terminal بر روی مانیتور نشان می‌دهد. کد زیر با نمایش جمله "please enter a key:" بر روی صفحه Hyper Terminal منتظر دریافت داده از کاربر می‌شود، و پس از آنکه کاربر کلیدی را فشرد، آنرا بر روی صفحه نمایش چاپ می‌کند.

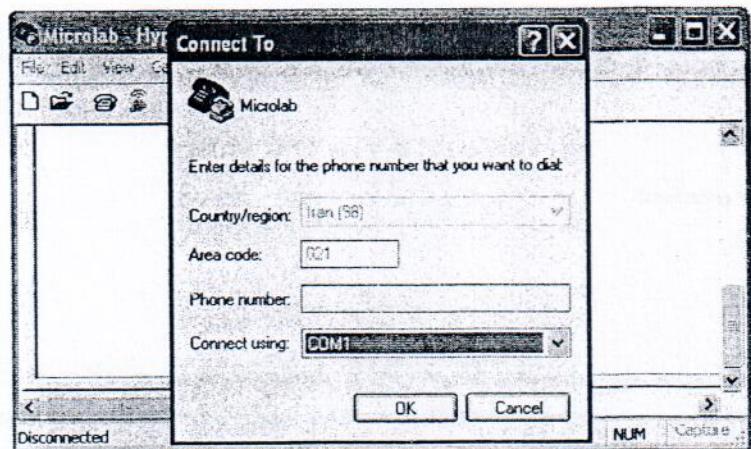
```
while (1){
    // Place your code here
    putsf("please enter a key:");
    x= getchar();
    lcd_putchar (x);
}
```

دسترسی به برنامه Hyper Terminal از طریق منوی All Programs, accessories, accessories, آنگاه منوی Communication امکان‌پذیر است. همانند شکل ۳-۱۹ پس از ورود به محیط Hyper Terminal، از شما خواسته می‌شود تا یک نام به اتصال جدیدی، که قصد برقراری آن را دارید اختصاص دهید.



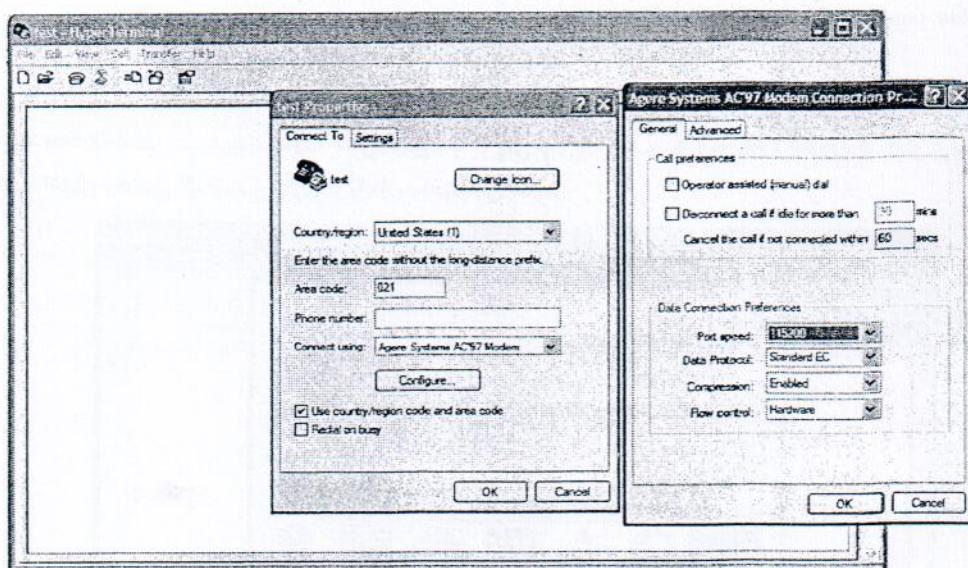
شکل ۳-۱۹-انتخاب نام برای اتصال

شماره پورت سریالی که از طریق آن می‌خواهید با کامپیوتر ارتباط سریال برقرار کنید را همانند شکل ۴-۱۹ مشخص کنید



شکل ۴-۱۹- تعیین پورت سریال مورد نظر از کامپیوتر به عنوان پورت ارتباطی

همانند شکل ۵-۱۹ پس از این مرحله می‌بایست در منوی Files properties و سپس connect To تنظیمات لازم از جمله نرخ بیت مورد نظر (port speed) را مشخص نمایید.



شکل ۵-۱۹- انجام سایر تنظیمات لازم برای برقراری ارتباط سریال

پورت سریال کامپیوتر را از طریق ارتباط RS232C به کانکتور DB9 و کابل متصل به این کانکتور را به تراشه MAX232 و نهایتاً میکروکنترلر متصل نمایید. با فشردن دکمه در Hyper Terminal ارتباط مابین میکروکنترلر و کامپیوتر برقرار می‌گردد.

۲۰- پیوست ۴: کد ارتباط به LCD در حالت ۴ بیتی

کد زیر برای نمایش کاراکترها توسط یک LCD کاراکتری در حالت ۴ بیتی قابل استفاده است.

```
;*****
;File:          m8_LCD_4bit.asm
;Title:         ATmega8 driver for LCD in 4-bit mode (HD44780)
;Assembler:     AVR assembler/AVR Studio
;Version:       1.0
;Created:      April 5th, 2004
;Target:        ATmega8
;Christoph Redecker, http://www.avrbeginners.net
;*****  
  
; Some notes on the hardware:  
; ATmega8 (clock frequency doesn't matter, tested with 1 MHz to 8 MHz)  
; PortD.1 -> LCD RS (register select)  
; PortD.2 -> LCD RW (read/write)  
; PortD.3 -> LCD E (Enable)  
; PortD.4 ... PortD.7 -> LCD data.4 ... data.7  
; the other LCD data lines can be left open or tied to ground.  
  
.include "c:\program files\atmel\avr studio\appnotes\m8def.inc"  
  
.equ  LCD_RS      = 1
.equ  LCD_RW      = 2
.equ  LCD_E       = 3  
  
.def  temp       = r16
.def  argument   = r17           ;argument for calling subroutines
.def  return      = r18           ;return value from subroutines  
  
.org 0
rjmp reset  
  
reset:
    ldi    temp, low(RAMEND)
    out   SPL, temp
    ldi    temp, high(RAMEND)
    out   SPH, temp  
  
;LCD after power-up: ("*" means black bar)
;*****|  
;  
    rcall LCD_init  
  
;LCD now:
;|&          | (&: cursor, blinking)
;|  
  
    rcall LCD_wait
    ldi    argument, 'A'           ;write 'A' to the LCD char data RAM
    rcall LCD_putchar  
  
;|&
```

```

;|           |

rcall LCD_wait
ldi argument, 0x80      ;now let the cursor go to line 0, col 0 (address 0)
rcall LCD_command       ;for setting a cursor address, bit 7 of the commands has to be set

;|A          | (cursor and A are at the same position!)
;|           |

rcall LCD_wait
rcall LCD_getchar       ;now read from address 0

;|A&         | (cursor is also incremented after read operations!!!)
;|           |

push return             ;save the return value (the character we just read!)
rcall LCD_delay
pop argument            ;restore the character
rcall LCD_putchar       ;and print it again

;|AA&        | (A has been read from position 0 and has then been written to the next pos.)
;|           |

loop:    rjmp loop

lcd_command8:
    in temp, DDRD      ;used for init (we need some 8-bit commands to switch to 4-bit mode!)
    sbr temp, 0b11110000 ;we need to set the high nibble of DDRD while leaving
    out DDRD, temp      ;the other bits untouched. Using temp for that.
    in temp, PortD      ;set high nibble in temp
    cbr temp, 0b11110000 ;write value to DDRD again
    cbr argument, 0b00001111 ;then get the port value
                           ;and clear the data bits
                           ;then clear the low nibble of the argument
                           ;so that no control line bits are overwritten
                           ;then set the data bits (from the argument) in the Port value
                           ;and write the port value.
    or temp, argument   ;and write the port value.
    out PortD, temp     ;now strobe E
    nop
    nop
    nop
    cbi PortD, LCD_E    ;get DDRD to make the data lines input again
    in temp, DDRD        ;clear data line direction bits
    cbr temp, 0b11110000 ;and write to DDRD

ret

lcd_putchar:
    push argument         ;save the argmuent (it's destroyed in between)
    in temp, DDRD        ;get data direction bits
    sbr temp, 0b11110000 ;set the data lines to output
    out DDRD, temp        ;write value to DDRD
    in temp, PortD        ;then get the data from PortD
    cbr temp, 0b11111110 ;clear ALL LCD lines (data and control!)
    cbr argument, 0b00001111 ;we have to write the high nibble of our argument first
                           ;so mask off the low nibble
                           ;now set the argument bits in the Port value
    or temp, argument

```

```

out    PortD, temp           ;and write the port value
sbi    PortD, LCD_RS        ;now take RS high for LCD char data register access
sbi    PortD, LCD_E         ;strobe Enable

nop
nop
nop
cbi    PortD, LCD_E         ;restore the argument, we need the low nibble now...
pop    argument              ;clear the data bits of our port value
cbr    temp, 0b11110000      ;we want to write the LOW nibble of the argument to
swap   argument              ;the LCD data lines, which are the HIGH port nibble!
cbr    argument, 0b00001111  ;clear unused bits in argument
or     temp, argument        ;and set the required argument bits in the port value
out    PortD, temp           ;write data to port
sbi    PortD, LCD_RS        ;again, set RS
sbi    PortD, LCD_E         ;strobe Enable

nop
nop
nop
cbi    PortD, LCD_E         ;PortD, LCD_RS
cbi    PortD, LCD_RS        ;temp, DDRD
in     temp, DDRD            ;data lines are input again
cbr    temp, 0b11110000
out    DDRD, temp

ret

lcd_command:                   ;same as LCD_putchar, but with RS low!
push   argument
in     temp, DDRD
sbr   temp, 0b11110000
out   DDRD, temp
in     temp, PortD
cbr   temp, 0b11111110
cbr   argument, 0b00001111
or    temp, argument

out    PortD, temp
sbi    PortD, LCD_E
nop
nop
nop
cbi    PortD, LCD_E
pop    argument
cbr    temp, 0b11110000
swap  argument
cbr    argument, 0b00001111
or    temp, argument
out    PortD, temp
sbi    PortD, LCD_E
nop
nop
nop
cbi    PortD, LCD_E
in     temp, DDRD
cbr    temp, 0b11110000
out    DDRD, temp

```

```

ret

LCD_getchar:
    in    temp, DDRD      ;make sure the data lines are inputs
    andi temp, 0b00001111 ;so clear their DDR bits
    out   DDRD, temp
    sbi   PortD, LCD_RS  ;we want to access the char data register, so RS high
    sbi   PortD, LCD_RW  ;we also want to read from the LCD -> RW high
    sbi   PortD, LCD_E   ;while E is high

    nop
    in    temp, PinD     ;we need to fetch the HIGH nibble
    andi temp, 0b11110000 ;mask off the control line data
    mov   return, temp    ;and copy the HIGH nibble to return
    cbi   PortD, LCD_E   ;now take E low again
    nop
    nop
    sbi   PortD, LCD_E   ;same as above, now we're reading the low nibble
    nop
    in    temp, PinD     ;get the data
    andi temp, 0b11110000 ;and again mask off the control line bits
    swap  temp           ;temp HIGH nibble contains data LOW nibble! so swap
    or    return, temp    ;and combine with previously read high nibble
    cbi   PortD, LCD_E   ;take all control lines low again

    ret                 ;the character read from the LCD is now in return

LCD_getaddr:
flag
    in    temp, DDRD      ;works just like LCD_getchar, but with RS low, return.7 is the busy
    andi temp, 0b00001111
    out   DDRD, temp
    cbi   PortD, LCD_RS
    sbi   PortD, LCD_RW
    sbi   PortD, LCD_E

    nop
    in    temp, PinD
    andi temp, 0b11110000
    mov   return, temp
    cbi   PortD, LCD_E

    nop
    nop
    sbi   PortD, LCD_E
    nop
    in    temp, PinD
    andi temp, 0b11110000
    swap  temp
    or    return, temp
    cbi   PortD, LCD_E
    cbi   PortD, LCD_RW

    ret                 ;read address and busy flag until busy flag cleared

LCD_wait:
    rcall LCD_getaddr
    andi return, 0x80
    brne LCD_wait

```

```
ret
```

```
LCD_delay:
```

```
    clr    r2
    LCD_delay_outer:
    clr    r3
    LCD_delay_inner:
    dec    r3
    brne   LCD_delay_inner
    dec    r2
    brne   LCD_delay_outer
```

```
ret
```

```
LCD_init:
```

```
ldi    temp, 0b00001110          ;control lines are output, rest is input
out   DDRD, temp

rcall LCD_delay
ldi   argument, 0x20           ;first, we'll tell the LCD that we want to use it
rcall LCD_command8             ;in 4-bit mode.
                                ;LCD is still in 8-BIT MODE while writing this command!!!

rcall LCD_wait
ldi   argument, 0x28           ;NOW: 2 lines, 5*7 font, 4-BIT MODE!
rcall LCD_command
;

rcall LCD_wait
ldi   argument, 0x0F           ;now proceed as usual: Display on, cursor on, blinking
rcall LCD_command

rcall LCD_wait
ldi   argument, 0x01           ;clear display, cursor -> home
rcall LCD_command

rcall LCD_wait
ldi   argument, 0x06           ;auto-inc cursor
rcall LCD_command
ret
```

دستور کار آزمایشگاه

درس ریزپردازندۀ ۱

دانشکده مهندسی کامپیووتر

دانشگاه صنعتی امیرکبیر

تقدیر و سپاس

بدینویسیله از کلیه کسانی در تهیه این دستورالعمل اینجانب را یاری نموده‌اند یعنی بعضی از مدرسات و دانشجویان آزمایشگاه ریزپردازنده در دانشکده مهندسی کامپیوتر دانشگاه صنعتی امیر کبیر که آزمایشگاه این درس را زیر نظر اینجانب ارائه نموده و یا با هنگام رسانیده‌اند و بخش‌هایی از مطالب این دستورالعمل را تهیه و یا بازبینی نموده‌اند خصوصاً آقایان محمد میرزا آقاتبار، حسین آتشی، خانم حمیده ایزدیار، محمد صالحی، محمدعلی کیوانزاد، حامد رضوانی، فاطمه خادمیان تشکر نموده، برای آنان آرزوی موفقیت در کلیه شرکت‌های زندگی می‌نمایم. جزو حاضر حاوی بخش‌هایی از مطالب مورد نیاز آزمایشگاه ریزپردازنده ۱ می‌باشد و سعی خواهد شد در آینده تدریجاً تکمیل و نوافع آن مرتفع گردد تا بدینویسیله بتواند نیاز دانشجویان و علاقمندان به مبحث ریزپردازنده ۱ را از حیث مطالب علمی و عملی مورد نیاز جهت کسب آشنایی بیشتر و ممارست در این زمینه فراهم آورد. از آنجا که این دستورالعمل مطمئناً دارای نقائص و اشکالاتی است، بدینویسیله از خوانندگان آن تقاضا می‌شود ایرادات آن و نیز پیشنهادات خود را در راستای بهبود هرچه بیشتر آن به اینجانب به یکی از آدرس‌های زیر اطلاع دهند.

تلفن: ۶۴۵۴۲۷۲۲

آدرس الکترونیکی: homayoun@aut.ac.ir

و من الله التوفيق

محمد‌مهدی همایون‌پور

عضو هیات علمی دانشکده مهندسی کامپیوتر

و فناوری اطلاعات دانشگاه صنعتی امیر‌کبیر

۵۲	-۵-۷-۵-انتقال برنامه به میکروکنترلر
۵۳	۶- جلسه چهارم
۵۴	۶-۱- تولید سیگنال بازنشانی
۵۵	۶-۲- تغییر بیت های فیوز
۵۶	۶-۳- تولید سیگنال ساعت (clock)
۵۹	۷- جلسه پنجم
۶۱	۸- جلسه ششم
۶۳	۹- جلسه هفتم
۶۵	۱۰- جلسه هشتم
۶۷	۱۱- جلسه نهم
۶۹	۱۲- جلسه دهم
۷۱	۱۳- جلسه یازدهم
۷۲	۱۴- جلسه دوازدهم
۷۶	۱۵- جلسه سیزدهم
۸۰	۱۶- مراجع
۸۱	۱۷- پیوست ۱: پایه های میکروکنترلر ATMEGA32
۸۲	۱۸- پیوست ۲ (نحوه ارتباط یک برنامه MICROSOFT VISUAL STUDIO با میکروکنترلر)
۸۴	۱۹- انتقال اطلاعات از میکروکنترلر به کامپیوتر
۸۵	۲۰- انتقال اطلاعات از کامپیوتر به میکروکنترلر
۸۶	۲۱- پیوست ۳ (ارتباط با کامپیوتر از طریق برنامه (HYPER TERMINAL
۸۹	۲۲- پیوست ۴: کد ارتباط به LCD در حالت ۴ بیتی
۹۷	۲۳- جلسه پانزدهم (برقرار ارتباط سریال از طریق رابط (TWI

۲- گروه‌بندی دانشجویان، نحوه ارزیابی و نحوه گزارش‌دهی، انجام پروژه پایانی آزمایشگاه

- گروه‌بندی دانشجویان
- بیان چگونگی ارزیابی دانشجویان
- بیان نحوه تهیه پیش‌گزارش و گزارش هر جلسه
- بیان چگونگی انجام پروژه پایانی آزمایشگاه

گروه‌بندی: در اولین جلسه آزمایشگاه گروه‌بندی دانشجویان در گروه‌های ۲ یا حداقل ۳ نفره انجام می‌شود.

ارزیابی: نحوه ارزیابی دانشجویان بر اساس موارد زیر خواهد بود:

- انضباط، رعایت مقررات آزمایشگاه، کار تیمی و حداقل همکاری با استاد آزمایشگاه (۱۰٪)، و سرپرست آزمایشگاه (۵٪).
- کیفیت عملکرد در آزمایشگاه (شامل مطالعه مطلب مورد نیاز قبل از هر آزمایش و تسلط کافی بر مطالب، جستجو و تهیه دیتاشیت قطعات مورد نیاز برای آزمایش، پاسخگویی به سوالات) (۳۵٪)
- تهیه پیش‌گزارش و گزارش کار آزمایش‌ها (۳۰٪)
- پروژه نهایی (۲۰٪)

پیش‌گزارش: پیش‌گزارش به صورت دستنویس تهیه شده و در ابتدای جلسه تحويل استاد آزمایشگاه می‌گردد. پیش‌گزارش باید شامل موارد ذیل باشد.

- صفحه عنوان (شامل عنوان و شماره آزمایش، نام دانشجو، نام استاد آزمایشگاه، تاریخ آزمایش)
- مقدمه و هدف آزمایش
- لیست قطعات بکار رفته در آزمایش
- متن اصلی شامل ارائه شماتیک مدار، توصیف طرز کار مدار، طرز کار قطعات مهم بکار رفته با استفاده از اطلاعات و اشکال برگرفته از دیتاشیت آنها و سایر نکات فنی مهم
- توضیح چگونگی استفاده از ثبات‌های کنترلی و وضعیت

گزارش هر آزمایش می‌باشد بصورت تایپ شده (در محیط Word) و مرتب حداقل تا هفته بعد از آزمایش (غیرقابل تمدید)، به آدرس ایمیل استاد درس آزمایشگاه ارسال یا در سیستم مدیریت دروس دانشکده (moodle) در محل مربوط به آزمایشگاه ریزپردازندۀ بارگذاری گردد. این گزارش شامل موارد زیر می‌باشد:

- برنامه نوشته شده و کامنت‌گذاری بخش‌های مهم آن
- بررسی مشکلات پیش‌آمده در طول آزمایش و نحوه حل و برطرف کردن آنها
- ارائه نظرات و ایده‌های پیشنهادی برای بهبود عمکرد و کارایی مدار آزمایش
- مراجع استفاده شده شامل سایت‌های مفید، کتاب و جزو
- پیوست‌ها

۳- جلسه اول

هدف از آزمایش:

- ساخت مدار تغذیه ثبیت شده
- طراحی شماتیک و مدار چاپی^۱ به کمک نرم افزار آنتیوم

قطعات مورد نیاز:

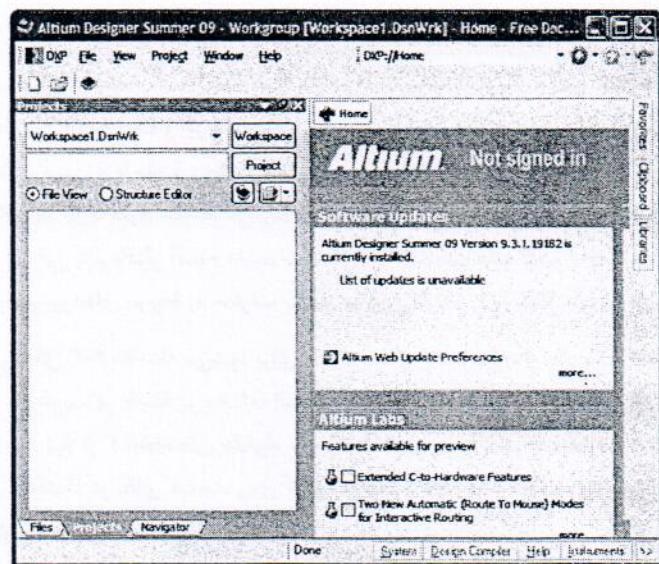
- ترانزیستور BC107: ۱ عدد
- تراشه گیت Nand: ۱ عدد
- رگولاتور ۷۸۰۵: ۱ عدد
- خازن ۴۷۰ میکروفاراد ۱۶ ولتی: ۱ عدد
- خازن ۴۷ یا ۱۰۰ میکروفاراد ۱۰ ولتی: ۱ عدد
- مقاومت ۲۰۰ مگا اهم: ۱ عدد
- مقاومت ۱۰۰ کیلواهم: ۱ عدد
- مقاومت ۱۳۳۰ اهم: ۱ عدد
- مقاومت ۱۸ کیلواهم: ۱ عدد
- ۱ LED

۱-۳- تغذیه ثبیت شده

از آنجا که در ساخت چراغ چشمکزن در این جلسه آزمایشگاه، نیاز به تغذیه ثبیت شده می‌باشد و از طوفی در آزمایش‌های جلسات بعدی با استفاده از میکروکنترلر ATmega32، نیاز به این نوع تغذیه خواهیم داشت، لذا در ادامه نحوه ساخت یک ولتاژ تغذیه ۵ ولتی ثبیت شده ارائه خواهد شد.

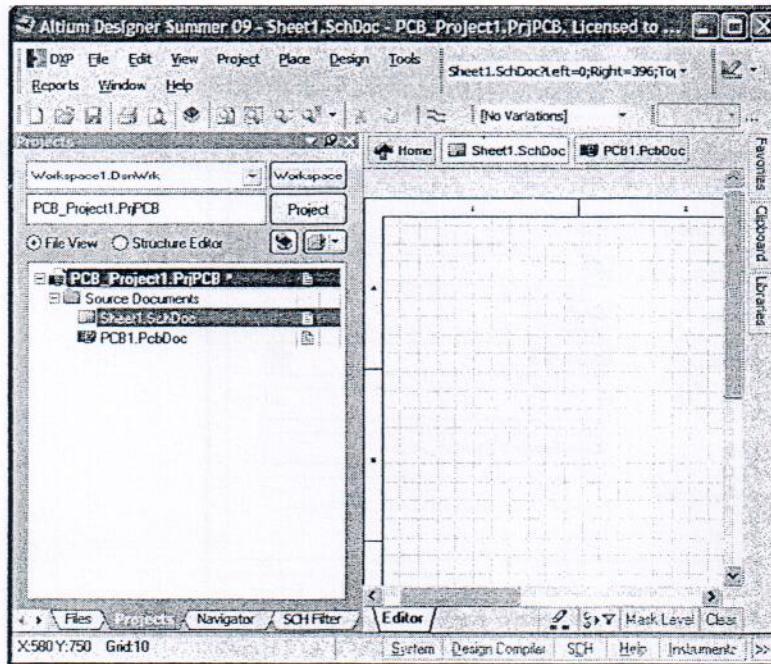
برای تامین ولتاژ تغذیه ثبیت شده و حفاظت قطعات دیجیتال در مقابل ولتاژ‌های بالا، می‌توان از یک رگولاتور یا ثبیت‌کننده ولتاژ استفاده نمود. وظیفه ثبیت‌کننده ولتاژ، دریافت سیگنال با ولتاژ ثبیت نشده در ورودی، ثبیت آن به مقدار ولتاژ موردنظر و تحويل سیگنال با ولتاژ ثبیت شده در خروجی آن است. ولتاژ تغذیه ثبیت شده مورد نیاز در آزمایش‌های این آزمایشگاه از جمله برای تغذیه گیت‌های Nand و تغذیه میکروکنترلر ATmega32، ۵ ولت می‌باشد. لذا برای بدست آوردن ولتاژ تغذیه ثبیت شده ۵ ولتی از تراشه 7805 استفاده می‌نمائیم. همانطور که در شکل ۱-۳ مشاهده می‌شود، ثبیت‌کننده 7805 ۷۸۰۵ دارای سه پایه می‌باشد. پایه اول از سمت چپ، پایه ورودی می‌باشد که سیگنال ورودی به این پایه وارد می‌شود، پایه دوم، به زمین متصل می‌شود و پایه‌ی سوم، خروجی رگولاتور است که سیگنال با ولتاژ ثابت ۵ ولت را فراهم می‌کند. شکل ۱-۳ مدار مورد نیاز برای تبدیل یک ولتاژ یکسو شده به یک ولتاژ ثبیت شده را ارائه می‌نماید. خوب است که ولتاژ سیگنال ورودی به رگولاتور در حدود بیشتر از ۲ ولت از سیگنال خروجی آن بیشتر باشد. چنانچه این اختلاف ولتاژ زیاد شود، در زمانیکه جریان قابل توجهی از رگولاتور کشیده می‌شود، رگولاتور گرم می‌شود که گرمای حاصل را تاحدی می‌توان توسط خنک‌کننده‌های آلومینیمی که به بدنه رگولاتور پیچ می‌شوند، کاهش داد. شکل زیر برای حالتی رسم شده است که شماره ثبیتساز از مقابل دیده شود.

PCB (Printed Board Circuit)^۱



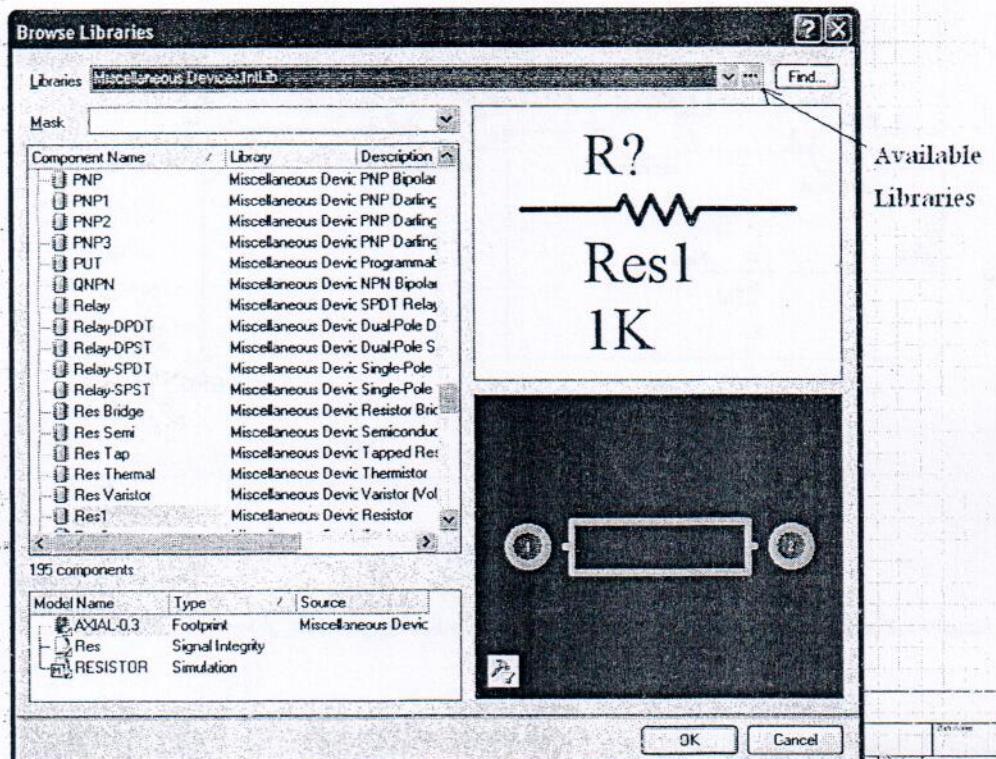
شکل ۲-۳. محیط نرم افزار آلتیوم

برای شروع کار از منوی File در بالای صفحه، یک PCB Project جدید را بسازید و به این ترتیب یک پروژه جدید به وجود می‌آوریم. سپس مشابه عمل بالا ابتدا یک فایل Schematic و سپس یک فایل PCB به این پروژه اضافه می‌کنیم. پس از ایجاد پروژه و اضافه کردن فایل‌های مذکور به پروژه، محیط نشان داده شده در شکل ۲-۳ به صورت شکل ۳-۳ دیده می‌شود.



شکل ۳-۳. اضافه کردن فایل Schematic و PCB به پروژه

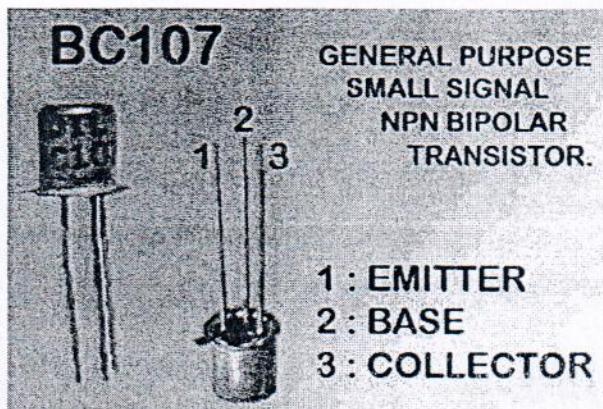
با فشردن شکل ۵-۳ Browse Library ظاهر شده و با استفاده از Available libraries می‌توان کتابخانه‌های مختلف را به پروژه اضافه کرد.



شکل ۵-۳-منوی Browse Libraries

بدين صورت می‌توان تمام قطعات مورد نیاز را به شماتیک اضافه کرد. پس از قرار دادن مقاومت‌ها، به مقداردهی و نامگذاری آن‌ها می‌پردازیم. برای این منظور روی اسم هر مقاومت (R?)، دوبار کلیک کرده و بدين ترتیب صفحه‌ای نمایش داده می‌شود که می‌توان در فیلد Value آن نام دلخواه را وارد کرد. چنانچه روی مقدار مقاومت (1k) دوبار کلیک کنیم، می‌توان در صفحه باز شده مقدار دلخواه را با ذکر واحد آن وارد کرد. البته روش دیگری نیز وجود دارد و آن به این ترتیب است که روی خود قطعه دوبار کلیک کنیم. در این حالت لیست کامل تری باز شده و در قسمت Designator نام عنصر و در بخش Value مقدار آن را وارد کرده و نهایتاً با فشردن دکمه Enter از صفحه کلید، همه تغییرات به یکباره اعمال خواهد شد. دقیقاً همین مراحل را برای انتقال خازن‌های مورد نیاز و ترانزیستور انجام می‌دهیم. این عناصر هم در همان کتابخانه مذکور قرار دارند و این بار به جای عبارت Res1 عبارت Cap و NPN را در صفحه کتابخانه تایپ می‌کنیم.

برای ترانزیستور مورد استفاده نیز به سبب نزدیکی پایه‌های آن، باید Footprint آن به شکل دیگری تبدیل شود. در اینجا ما از شکل HDR1X3 از کتابخانه IntLib استفاده نمودیم. همچنین ویژگی‌های ترانزیستور مورد استفاده در این مدار در شکل ۸-۳ آمده است.



شکل ۸-۳. پایه‌های ترانزیستور مورد استفاده در مدار

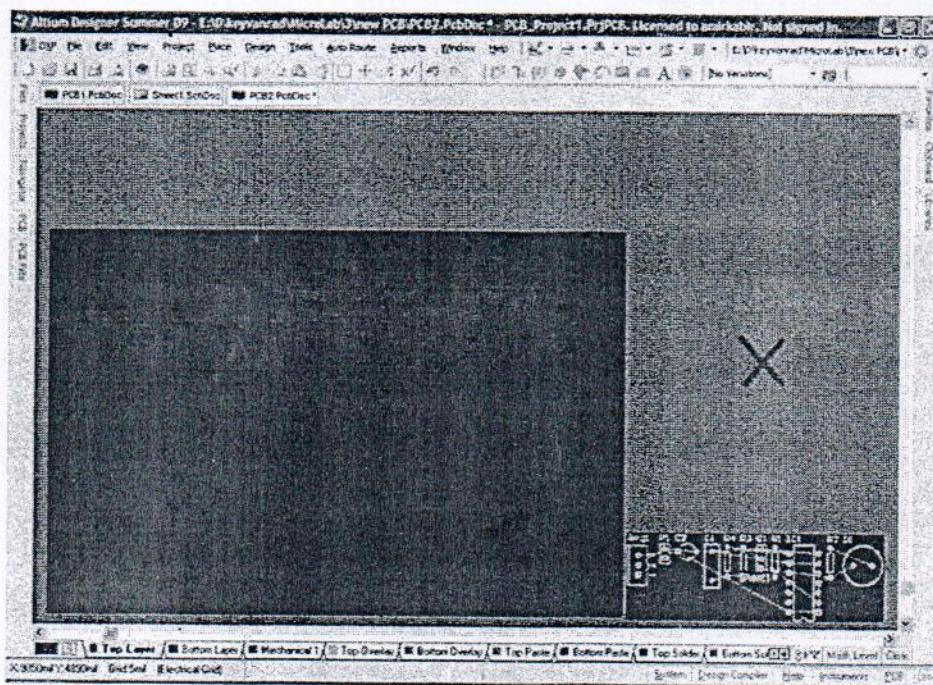
همان طور که در نقشه مدار دیده می‌شود، به سه گیت NAND احتیاج داریم. برای این منظور از کتابخانه Motorola Logic Gate.IntLib سه قطعه MC74HC00AN انتخاب می‌کنیم. با توجه به این که در IC‌های موجود در بازار ۴ گیت NAND جاسازی شده است، در این مرحله باید اقدام ظرفی انجام دهیم. برای این منظور ابتدا نام هر دو قطعه را یکسان انتخاب می‌کنیم، سپس روی یکی، از دو قطعه دوبار کلیک کرده و در منوی باز شده به کمک فلش‌های موجود، گیت NAND مذکور را به عنوان قسمت دوم از مجموعه (Part 2/4) انتخاب می‌کنیم. البته این کار توسط خود نرم‌افزار نیز به صورت اتوماتیک، انجام می‌شود. در ادامه در قسمت Designator نام قطعه را انتخاب کرده و دکمه Enter را فشار می‌دهیم. به این ترتیب در مرحله مسیردهی، نرم‌افزار هر سه گیت NAND را از مجموعه یک IC انتخاب خواهد کرد. برای رگولاتور استفاده شده در مدار نیز می‌توان از کتابخانه Motorola Power Mgt Voltage Regulator.IntLib را استفاده نمود.

برای اتصال منبع تغذیه و زمین به بورد به یک قطعه هدر (Header) دوتایی احتیاج داریم. این قطعه را از کتابخانه Miscellaneous Connectors.IntLib تحت عنوان Header2 انتخاب کرده و وارد صفحه شماتیک کرده و نام دلخواهی برای آن در نظر می‌گیریم. در مرحله پایانی ورود عناصر، باید تغذیه و اتصال زمین را وارد صفحه شماتیک کنیم. این موارد را می‌توان از منوی عناصر بالای صفحه، انتخاب کرد.

پس از چینش اجزاء نوبت به سیم‌کشی بین آن‌ها می‌رسد. برای این منظور دو مکانیزم وجود دارد. اول این که از منوی Place و یا از لیست اشکال بالای صفحه، گزینه Wire را انتخاب کرده و اتصالات لازم را به وجود آوریم. وقتی که بین دو قطعه سیم می‌کشیم، مکان مناسب برای اتمام سیم جایی است که علامت ضربدر قرمز در آنجا ظاهر می‌شود. در شرایطی که مدار بسیار متراکم و یا فواصل سیم‌کشی طولانی باشد، می‌توان از مکانیزم Net استفاده کرد. برای انجام این کار روی نقاطی که می‌خواهیم به هم متصل شوند، دو NetLabel قرار داده و نام مشابهی را برای آن‌ها برمی‌گزینیم. به عنوان مثال در مدار شماتیک شکل ۶-۳ برای اتصال مقاومت R4 به مقاومت R1، از دو Net استفاده کرده‌ایم. Net را می‌توان از آیکن موجود در بالای صفحه انتخاب نمود.

خروج احتمالی از محدوده بورد بر حذر می‌دارد. برای تعیین این محدوده، باید از لایه‌های پایین صفحه Keep Out Layer را انتخاب کرده و سپس دور تادور قطعه را سیم کشی کرد.

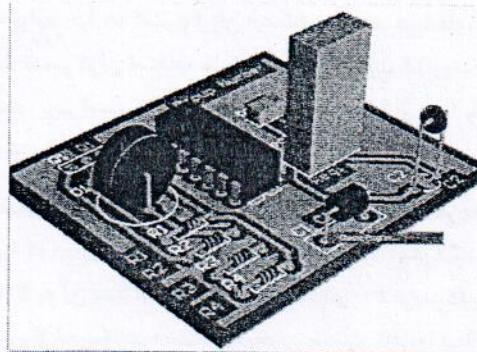
با انجام مراحل فوق مقدمات عملیات مسیر دهی توسط نرم‌افزار انجام شده است. در این مرحله از منوی Auto Route گزینه All را انتخاب می‌کنیم. در صفحه‌ای که باز می‌شود، به صورت پیش‌فرض استراتژی مسیردهی Default 2 Layer Board است. بهتر است، در صورت فشرده نبودن مدار و نبود محدودیت ابعاد برای آن، حالت یک لایه را انتخاب کرد تا هزینه ساخت بورد کمتر شود. اگر یک لایه انتخاب شود، باید آن را لایه زیرین در نظر بگیریم تا قطعات را بر روی بورد گذاشته و از زیر لحیم کنیم. در این استراتژی بخشی از سیم‌ها از بالا و بخش دیگر از قسمت زیرین بورد عبور خواهد نمود. حال دکمه Edit Rules را فشار می‌دهیم. در بخش Routing از گزینه Width پنهانی حداقل، حداً کثر و ایده‌آل سیم‌ها را انتخاب می‌کنیم. در مدار مورد بررسی ما هر سه مقدار با هم مساوی و برابر ۰.۶ mm (معادل 23.622 mil) انتخاب شدند. یک mil برابر ۰.۰۰۱ اینچ می‌باشد و هر اینچ معادل ۲.۵۴ سانتی‌متر است. پارامتر مهم دیگر Minimum Clearance در بخش Electrical است که فاصله بین خطوط و مسیرها را مشخص می‌کند که در این مدار برابر با ۱ mm (معادل 39.37 mil) انتخاب شده است. این مقدار را می‌توانید در زیرمنوی Clearance در بخش Constraints، که زیرمجموعه منوی Electrical است تغییر دهید. پس از تعیین پارامترها، دکمه All Route را فشار می‌دهیم و عملیات مسیردهی به سرعت صورت پذیرفته و گزارش سیم‌نم از نحوه عملکرد آن، قابل مشاهده خواهد بود. در صورتی عملیات routing با موقوفیت انجام شده است که در انتهای گزارش سیستم، تعداد عدم اتصال (failed to connect) و اتصال کوتاه (connections) برابر صفر باشد. همچنین در بخش Routing Layers در گزینه Routing می‌توان لایه‌های مورد استفاده در پروژه را تعیین نمود. در این مرحله عملای بخش عمده کار به پایان رسیده است. اما یک سری کارهای تکمیلی باقی مانده که در ادامه آن‌ها را فهرست‌وار مرور می‌کنیم.



شکل ۹-۳- صفحه PCB مدار چشمکزن

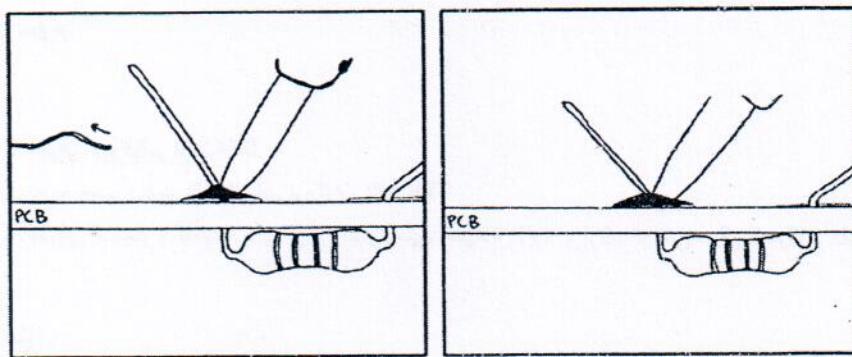
در PCB نیز با Hold کردن قطعه و زدن دکمه Space می‌توان قطعه را چرخاند. میزان چرخش به ازای هر بار زدن دکمه Tools→Component Placement→Auto Placer از قسمت Tools در Preferences می‌توان از نرم‌افزار برای قرار دادن اتوماتیک قطعات، استفاده نمود. نکته قابل توجه دیگر، امکان مشاهده مدار به صورت سه بعدی است. از طریق Tools→Legacy Tools→Legacy 3D View می‌توان قطعات، مورد استفاده و سیم‌کشی انجام شده را مشاهده نمود.

در شکل ۱۱-۲ نمایش سه بعدی مدار طراحی شده مشاهده می‌شود.



شکل ۱۲-۳- مشاهده سه بعدی PCB طراحی شده

در صورتی که در طراحی مدار متوجه مشکلی شده باشیم، ابتدا صفحه‌های محافظت نویز را حذف کرده و سپس از منوی Tools و ردیف Un-Route گزینه All را انتخاب می‌کنیم. به این ترتیب می‌توان تغییرات دلخواه را اعمال کرد. ضمناً چنانچه بخواهیم از طریق فایل شماتیک تغییراتی ایجاد کنیم، می‌توان پس از انجام تغییرات و انجام عملیات Update در صفحات شماتیک و PCB کار را ادامه داد. در پایان کار باید فایل PCB تکمیل شده را با دو فرمت PCB 4.0 Binary File و PCB 3.0 Binary File در فولدری کار ذخیره کرده و برای تولید بورد به مراکز ساخت بورد مدار چاپی تحویل می‌دهیم. قبل از ارسال فایل PCB برای ساخت بورد، کنترل می‌کنیم که PCB طراحی شده شامل تمامی اتصالات بوده و با مدار موجود بر روی شماتیک بطور کامل مطابقت داشته باشد. GND و VCC مربوط به تراشه‌ها نیز خوب است کنترل شوند. مرحله بعد قرار دادن قطعات بر روی بورد مدار چاپی و لحیم‌کاری آنها و نهایتاً تست، رفع ایراد و راهاندازی مدار چشمکزن است. در ادامه با لوازم لحیم‌کاری و نحوه انجام صحیح لحیم‌کاری و نیز با روش تست اتصالات آشنا خواهیم شد.

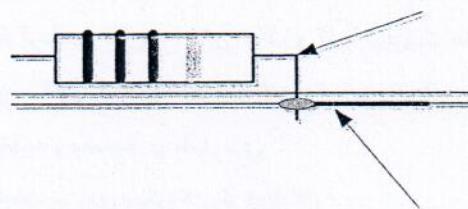


شکل ۱-۴- روش لحیم کاری

بر عکس عمل لحیم کاری، عمل لحیم زدایی در موقعی چون تعمیر مدارات الکترونیکی به منظور تعویض قطعه آسیب دیده و یا تعویض یک قطعه در زمان طراحی لازم می شود. لحیم زدایی توسط مکنده لحیم یا توسط فتیله لحیم انجام می شود. مکنده لحیم وسیله‌ای است که پس از گرم کردن لحیم توسط هویه، با مکش خود، لحیم را از محل لحیم کاری جدا می کند. فتیله لحیم نیز عملاً یک مس باقته شده است که پس از آنکه محل لحیم را به اندازه کافی گرم کردیم، اتصال فتیله لحیم به لحیم ذوب شده موجب جذب لحیم به فتیله و برداشته شدن لحیم از سطح لحیم می شود.

۱-۴- تست اتصالات

پس از دریافت بورد و لحیم کاری قطعات، برای اطمینان از لحیم کاری صحیح، می توان با انجام تست اهمی (یا تست بوق) از اتصال کامل قطعات به بورد مدار چاپی بعد از لحیم کاری مطمئن گردید. برای این منظور همانند شکل ۲-۴، یک سر مولتی‌متر را بر روی پایه قطعه یا پایه تراشه و سر دیگر آن را بر روی مدار متصل به پایه بر روی بورد قرار می دهیم. چنانچه مقاومت صفر بود و یا بوق اتصال شنیده شد، می توان از صحت لحیم کاری مطمئن گردید. خوب است که در صورت امکان قطعات را قبل از نصب و لحیم کاری تست و از صحت کار کرد آنها مطمئن شد.



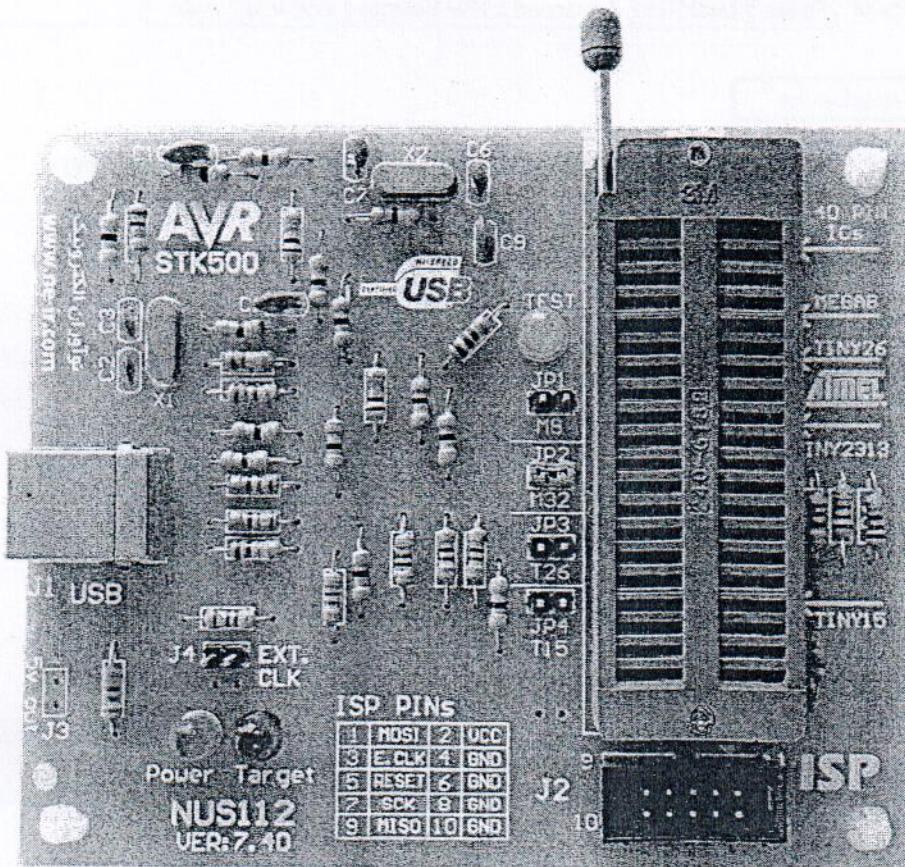
شکل ۲-۴- کنترل صحت اتصال پس از لحیم کاری

• سازگاری با تمامی ویندوزهای ۳۲ بیتی و ۶۴ بیتی (WIN10, WIN8.1, WIN8, WIN7, WIN VISTA, ...)

(WIN XP, WIN 2000)

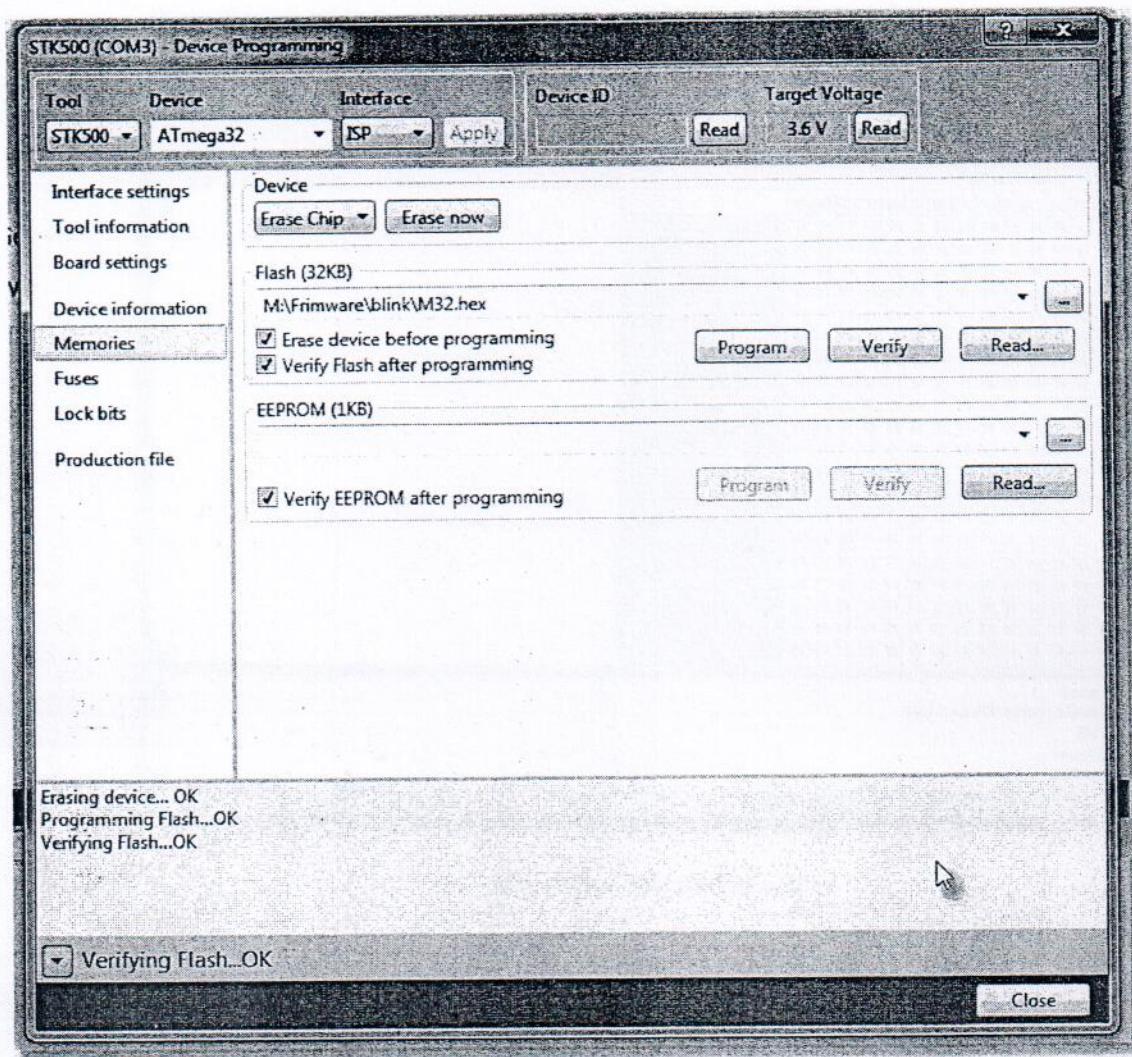
• عدم نیاز به منبع تغذیه جداگانه (تغذیه از طریق پورت USB)

• به همراه کابل رابط USB

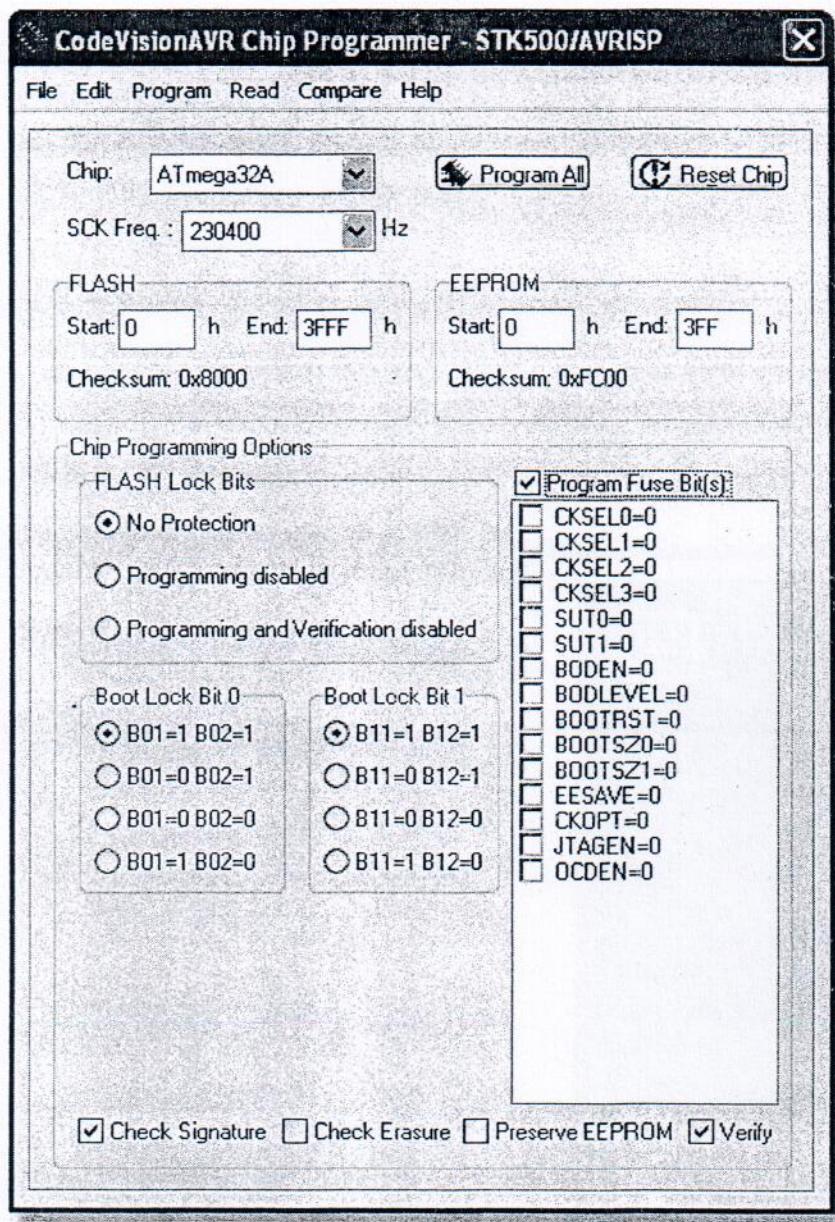


شکل ۱-۵- پروگرامر STK500 ساخت شرکت نوآوران الکترونیک

این پروگرامر توانایی کار در نرمافزارهای AVR Studio4 و AVR Studio6، بسکام (Bascom AVR) و کدویژن (Codevision AVR) را در همه نسخه‌ها، دارا می‌باشد. این توانایی به شما این امکان را می‌دهد که بعد از نوشتن برنامه و کامپایل آن، بدون هیچ وقفه‌ای و بدون اینکه بخواهد نرمافزار دیگری را باز نمایید، بالاصله دکمه Program را زده و نتیجه را در میکروکنترلر خود مشاهده کنید. این کار سرعت کار شما را بسیار افزایش می‌دهد و شما دغدغه کار با یک نرمافزار ناآشنای دیگر را نخواهید داشت. شما می‌توانید از نرمافزارهای AVR Studio4 و AVR Studio6، بسکام (Bascom AVR) و کدویژن (Codevision AVR) در زمان دسترسی به برد پروگرامر STK500 را به ترتیب در شکل ۲-۵ الی شکل ۵-۵ مشاهده می‌نمایید.



شكل ٣-٥ - نرم افزار ٦ AVR Studio

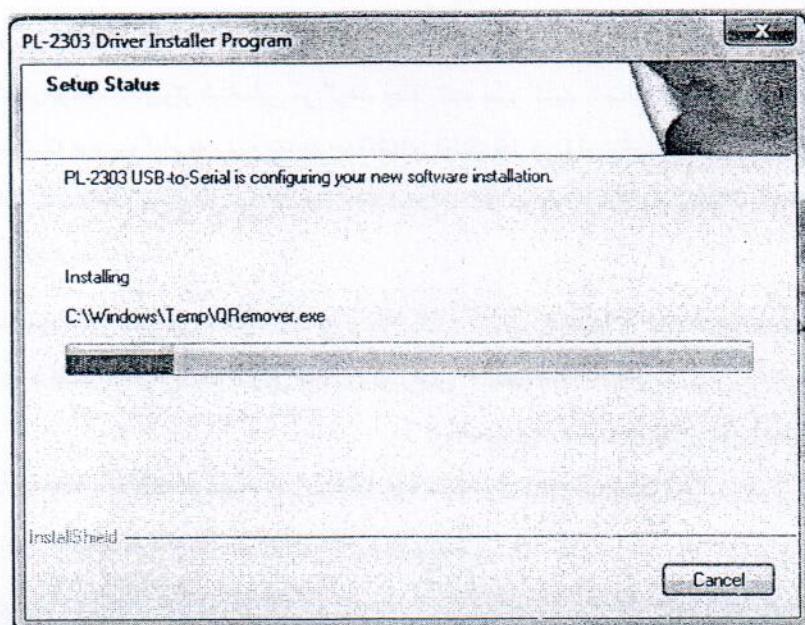


شکل ۵-۵ - نرم افزار CodeVision AVR

۲-۵- نحوه نصب درایور USB پروگرامر STK500

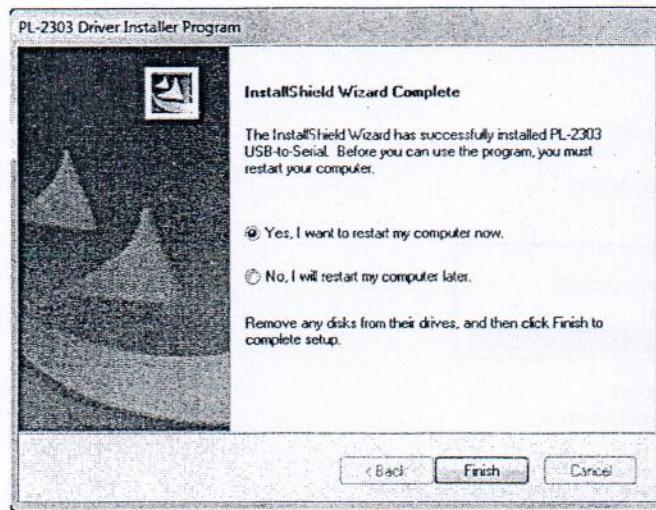
مراحل نصب و کار با پروگرامر STK500 :

۱ - نصب درایور پروگرامر (فقط بار اول)



شکل ۵-۷- روند نصب درایور USB پروگرام

روی دکمه Finish کلیک و سپس کامپیوتر را Restart نمایید و به مرحله بعد بروید.



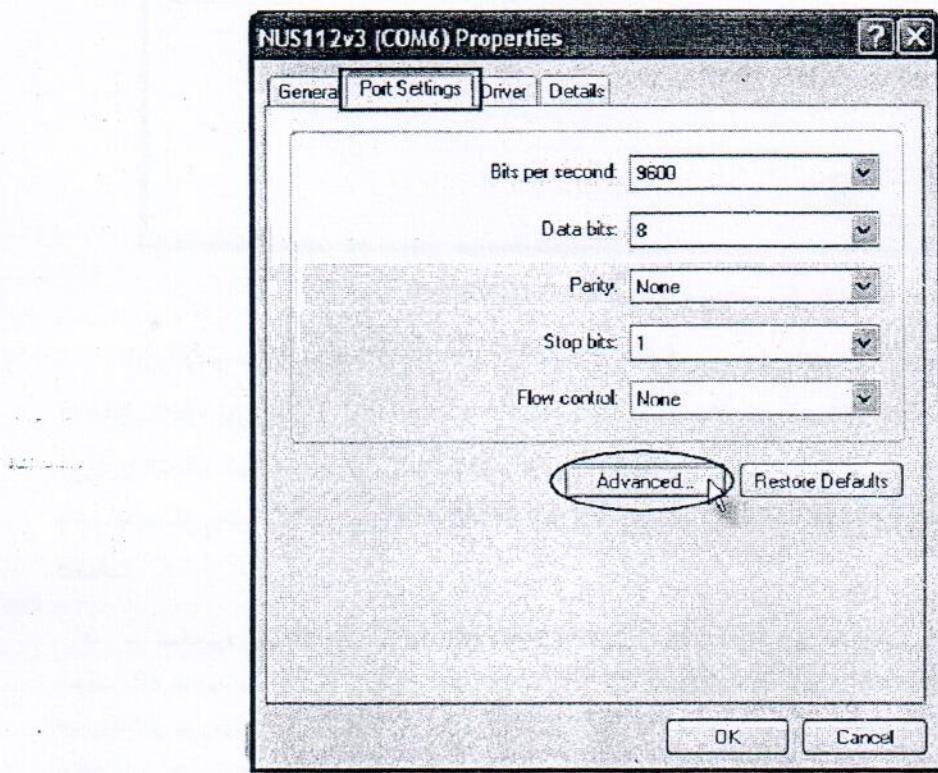
شکل ۵-۸- خاتمه نصب درایور USB پروگرام

در صورتی که در حین کار به مشکل برخوردید یا به پیغام خطایی برخوردید به لینک زیر مراجعه کنید:

<http://www.ne-ir.com/>

امکان دارد شماره پورت سریال (مجازی) شما از عدد ۹ تجاوز نماید. به علت اینکه این موضوع در نرم افزارهای پروگرام پیش بینی نشده است شما باید با اعمال تغییراتی شماره آن را دلخواه انتخاب کنید.

در بخش (Prolific USB-to-Serial Comm Port (COM X) Ports (COM&LPT) بر روی نام Properties کلیک راست نموده و گزینه Port Settings را انتخاب نمایید. از منوی باز شده در بخش Advanced گزینه Port Settings را انتخاب نمایید.



شکل ۱۰-۵ - تغییر شماره پورت سریال

در این منو در بخش Com Port Number شماره پورت سریال (مجازی) خود را بین عدد ۴ تا ۹ به طور دلخواه انتخاب نموده و بر روی کلید OK فشار دهید و پیغام ظاهر شده را تائید نمایید و پنجه را بر روی Properties را بیندید. دستگاه پروگرام را از USB جدا نموده و مجدداً وصل نمایید. حال شماره پورت سریال (مجازی) شما به عدد دلخواه شما تغییر کرده است.

اگر در هنگام پروگرام کردن برنامه پروگرامر بیغام خطای را نشان داد، موارد زیر را دوباره بررسی نمایید:

- ۱- مطمئن شوید که درایور پروگرامر نصب شده باشد و سختافزار پروگرامر برای کامپیوتر شناسایی شده باشد
- (از طریق Device Manager)
- ۲- چک کنید که کابل USB بطور صحیح وصل شده باشد و یا کابل آن را تعویض نمایید.
- ۳- مطمئن شوید که آی‌سی میکروکنترلر بطور صحیح بر روی سوکت ZIF قرار گرفته باشد.
- ۴- مطمئن شوید کابل ISP پروگرامر به برد وصل شده باشد.
- ۵- آیا آی‌سی میکروکنترلر شما سالم است؟
- ۶- آیا فیوزبیت‌های میکروکنترلر را خود، تغییر نداده‌اید؟

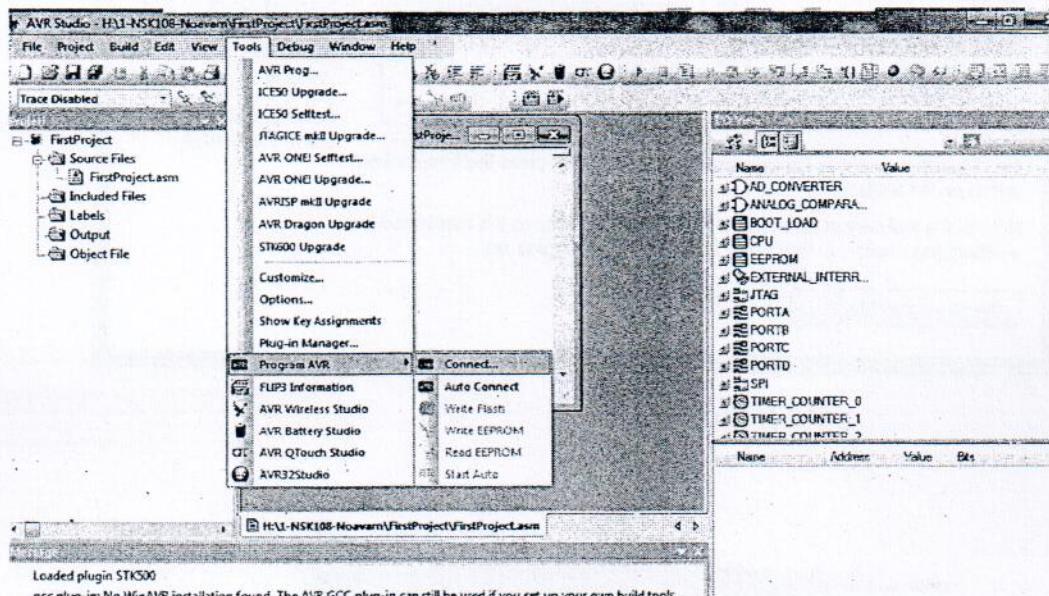
۴-۴- نحوه نصب نرمافزار پروگرامر و تنظیم نوع پروگرامر در آن

این پروگرامر توانایی کار با نرمافزارهای Bascom AVR و AVR Studio و Code Vision و AVR Studio را دارد. به این نرمافزارها می‌توانید از طریق CD محصول پروگرامر یا اینترنت دسترسی یافته و نصب نمایید. بعد از انجام مراحل فوق دستگاه آماده استفاده می‌باشد. در بخش‌های زیر استفاده از بعضی از نرمافزار پرکاربرد فوق که پروگرامر STK500 را نیز پشتیبانی می‌کنند معرفی و نحوه پروگرام کردن میکروکنترلر در زمان بیان هر یک از این نرمافزارها بیان شده است.

۴-۵- معرفی AVR Studio 4

- به منظور برنامه‌نویسی اسمبلی برای میکروکنترلرهای خانواده AVR می‌توان از اسمبلرهای مختلفی استفاده نمود. یکی از این اسمبلرها AVR Studio نام دارد. این نرمافزار برای نوشتن و دیباگ کردن برنامه اسمبلی برای میکروکنترلرهای خانواده AVR بکار می‌رود. محیط نرمافزار AVR Studio 4 در شکل زیر نشان داده شده است.

برای برنامه‌ریزی میکروکنترلرهای AVR در محیط نرم‌افزار 4 AVR Studio، باید کابل رابط را به درگاه USB کامپیوتر خود وصل کنید و سر دیگر کابل را به مدار پروگرامر (کانکتور J1) وصل کنید. در این صورت LED سبز زنگ باید روشن شود. همانطور که گفته شد میکروکنترلر را به دو روش قرار دادن روی سوکت ZIF برد پروگرامر و یا به روش ISP می‌توان پروگرام نمود. از منوی Tools در برنامه 4 AVR Studio، گزینه Connect و سپس گزینه Program AVR را انتخاب نمایید.



شکل ۱۳-۵ - فرآیند اتصال به پروگرامر در نرم‌افزار 4 Visual Studio

در این منو شماره پورت سریال (مجازی) که در طی مراحل نصب درایور مشاهده نموده‌اید را انتخاب نموده و سپس بر روی گزینه Connect کلیک کنید. امکان دارد شماره پورت سریال (مجازی) شما از عدد ۹ تجاوز نماید که چون این موضوع در نرم‌افزارهای پروگرامر پیش‌بینی نشده است، شما باید با اعمال تغییراتی شماره آن را به نحو مناسب تغییر دهید تا بین ۳ تا ۹ باشد.

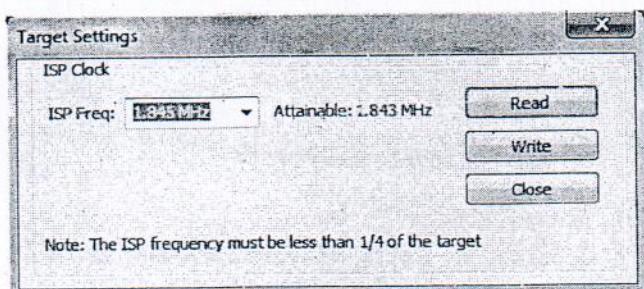
۱-۵-۵- تنظیمات بخش‌های پروگرامر نرم‌افزار AVR Studio

تنظیم بخش (سرعت برنامه‌ریزی) Programming Mode and Target Settings

در صورتی که میکروکنترلر را تازه خریداری کرده‌اید، تنظیمات فیوزبیت ساعت این میکروکنترلر به روش استفاده از مدار RC داخلی کالیبره شده با فرکانس 1MHz می‌باشد. در بخش Programming Mode Target Settings در شکل قبلی گزینه ISP را انتخاب نمائید.

تنظیمات گزینه Settings

به کمک این منو می‌توانید سرعت فرکانس ISP را به دلخواه تغییر دهید. بدین ترتیب که عدد مورد نظر را انتخاب نموده و بر روی دکمه write کلیک نمائید. فرکانس ISP همواره باید کمتر از یک چهارم فرکانس کاری میکروکنترلر باشد.



شکل ۱۶-۵- تنظیم ساعت ISP

Main منوی

به کمک این منو امکان انتخاب میکروکنترلر، پاک کردن حافظه برنامه، خواندن اطلاعات میکروکنترلر میکروکنترلر فراهم می‌باشد.

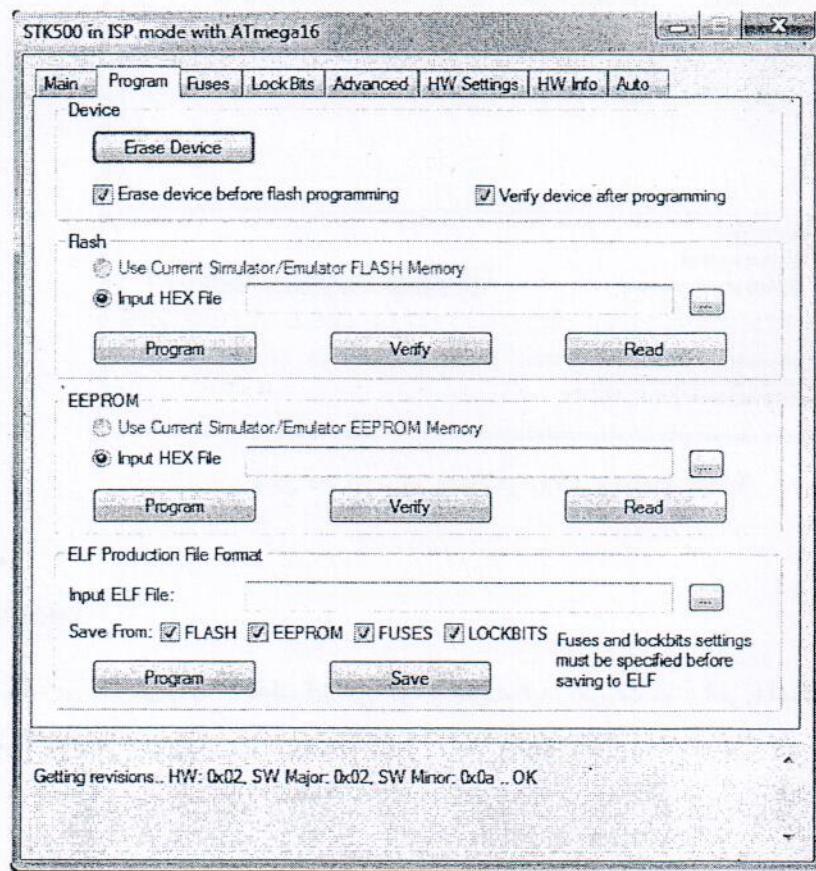
: این گزینه فایل هگز مورد نظر شما که قبلاً با گزینه Load Flash انتخاب کرده‌اید را بر روی حافظه میکروکنترلر انتقال می‌دهد.

: این گزینه یک بار اطلاعات ریخته شده بر وری حافظه Flash را می‌خواند و با اطلاعات فایل هگز انتخاب شده مقایسه می‌کند، به عبارت دیگر چک می‌کند که تمامی اطلاعات ریخته شده بر وری حافظه Flash به طور صحیح منتقل شده باشند.

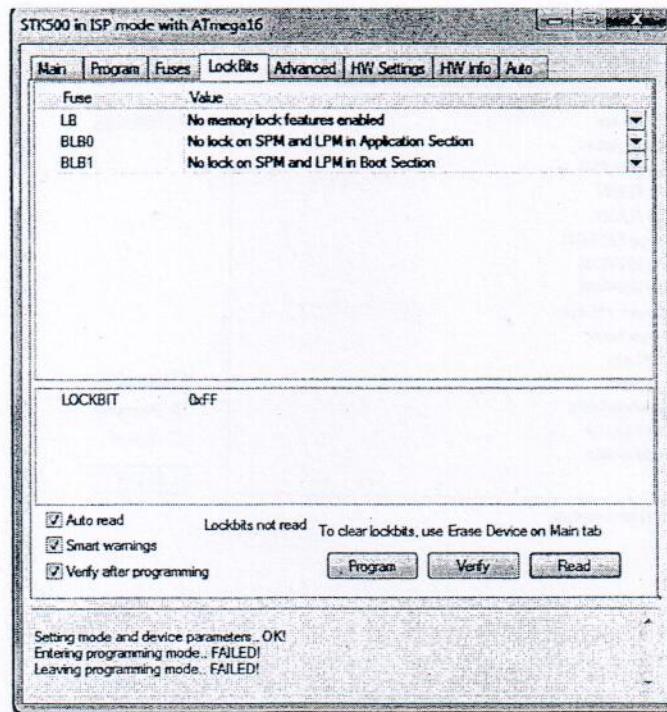
: این گزینه فایل هگز موردنظر شما را روی حافظه EEPROM میکروکنترلر انتقال می‌دهد.

: این گزینه اطلاعات داخل حافظه EEPROM میکروکنترلر را می‌خواند.

: این گزینه یک بار اطلاعات ریخته شده بر وری حافظه EEPROM را می‌خواند و با اطلاعات فایل هگز انتخاب شده مقایسه می‌کند به عبارت دیگر چک می‌کند که تمامی اطلاعات ریخته شده بر وری حافظه EEPROM به طور صحیح منتقل شده باشند.



شکل ۱۸-۵ - منوی Program در نرم‌افزار 4 Visual Studio



شکل ۲۰-۵ - منوی Lock Bits در نرم افزار ۴ Visual Studio

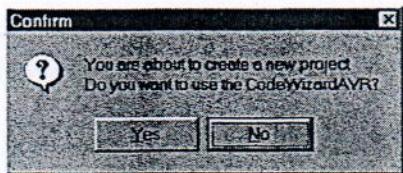
منوی Auto

به کمک این منو می توانید یک یا چند مورد از عملیاتی که در منوهای قبلی قابل انجام بودند را انتخاب و آنها را به ترتیب اجرا نمود. هر کدام از مراحلی را که می خواهید طی یک عملیات برنامه ریزی انجام دهید را تیک زده و سپس بر روی گزینه Start کلیک نمائید.

- تنظیمات دسترسی به برنامه خارجی
- شناسایی منبع بازنشانی تراشه
- مقداردهی اولیه در گاههای ورودی/خروجی
- مقداردهی اولیه وقفهای خارجی
- مقداردهی اولیه زمان سنج/شمارندها
- مقداردهی اولیه زمان سنج نگهبان
- مقداردهی اولیه USART و ارتباط سریال بافر شده مبتنی بر وقفه
- مقداردهی اولیه مبدل آنالوگ به دیجیتال
- مقداردهی اولیه مقایسه کننده آنالوگ
- مقداردهی اولیه واسط SPI
- مقداردهی اولیه گذرگاه I2C، سنسور دمای LM75، ترمومتر DS1621
- تراشه‌های ساعت واقعی DS1307 و DS1302، PCF8583 و PCF8563
- مقداردهی اولیه گذرگاه ۱ سیمه و سنسورهای دمای DS1820/DS18S20
- مقداردهی اولیه نمایش دهندهای LCD

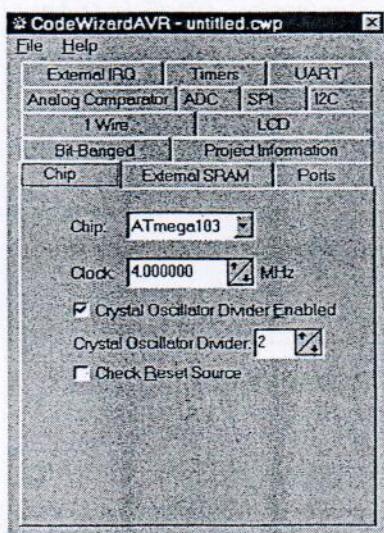
بعضی از قابلیت‌هایی که توسط محیط توسعه مجتمع CodeVisionAVR IDE قابل دسترسی هستند، عبارتند از:

- می‌توان یک فایل برنامه (سورس) جدید را ایجاد، ذخیره و پرینت نمود.
- می‌توان فایل برنامه (سورس) را از نو باز نمود، تغییر نام داد و یا ادیت نمود.
- می‌توان یک پروژه جدید را ایجاد نمود. در این زمان از کاربر سوال می‌شود که آیا مایل است از امکان ویزارد استفاده نماید یا خیر.
- پروژه را ذخیره نمود یا پروژه قبلی را باز نمود.
- می‌توان فایل سورسی را به یک پروژه اضافه یا کم کرد.
- می‌توان option پروژه را تعیین نمود (شامل نوع میکروکنترلر و فرکانس ساعت آن، مدل حافظه، فرمت فایل خروجی، بهینه‌سازی از جهت سرعت و حجم کد، حجم پشته، حجم RAM داخلی و خارجی، نوع برنامه Boot Loader یا Application)، تعیین مسیر فایل‌های کتابخانه و include
- پاک کردن حافظه تراشه و امکان چک کردن خالی بودن حافظه EEPROM و FLASH
- برنامه‌ریزی حافظه FLASH تراشه مورد نظر و چک کردن صحت انتقال برنامه به آن
- برنامه‌ریزی حافظه EEPROM تراشه مورد نظر و چک کردن صحت انتقال برنامه به آن
- برنامه‌ریزی بیت‌های فیوز و قفل
- اجرای برنامه منتقل شده به تراشه
- امکان دیباگ کردن برنامه و تعیین مسیر آن
- امکان ادیت کردن بافر داده EEPROM و FLASH
- دارای یک ترمینال برای برقراری ارتباط سریال
- دارا بودن امکان پیکربندی اسپلر
- امکان تنظیم پارامترهای محیط IDE



شکل ۲۴-۵-ایجاد پروژه جدید

پس از تائید ایجاد پروژه جدید، پنجره‌ی CodeWizardAVR که در شکل زیر آمده است، نمایش داده می‌شود.



شکل ۲۵-۵-پنجره‌ی CodeWizardAVR

Chip ۱-۲-۶-۵

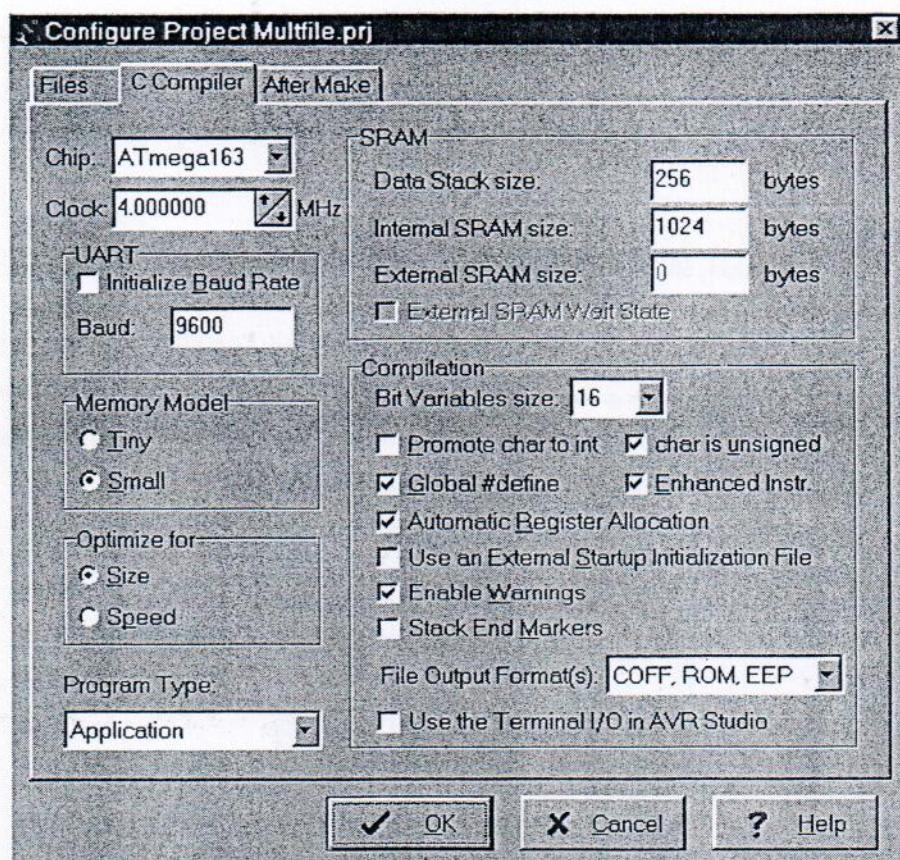
در قسمت Chip، که در شکل بالا نیز نمایش داده شده است، شما می‌توانید گزینه‌های تراشه مورد استفاده را تنظیم کنید. نوع میکروکنترلر، با استفاده از Chip list box مشخص می‌شود. فرکانس پالس ساعت تراشه بر حسب MHZ با استفاده از Clock box مشخص می‌گردد. برای تراشه‌های AVR که دارای تقسیم کننده‌های crystal oscillator می‌باشند، گزینه اضافه晶振分频器 مشاهده می‌شود. این گزینه، این امکان را به شما می‌دهد تا تقسیم کننده‌های crystal oscillator را فعال و یا غیر فعال کنید. چنانچه crystal oscillator فعال شود، می‌توانید با استفاده از Crystal Oscillator Divider Enabled نسبت تقسیم را مشخص کنید.

Port ۲-۲-۶-۵

برای تنظیمات پورت‌های ورودی و خروجی، باید گزینه Port را انتخاب نمود. در این صورت جعبه دیالوگی مانند شکل ۲۶-۵ مشاهده خواهد شد. برای مشخص کردن پورت‌های مورد استفاده در برنامه، می‌توانید پورت مورد نظر را انتخاب کرده و سپس ورودی/خروجی بودن هریک از بیت‌های آنرا در قسمت Data Direction تعیین نمایید. با کلیک کردن بر روی Pull up/Output value، می‌توانید تنظیمات زیر را انجام دهید:

۳-۶-۵- تنظیم گزینه های کامپایلر C

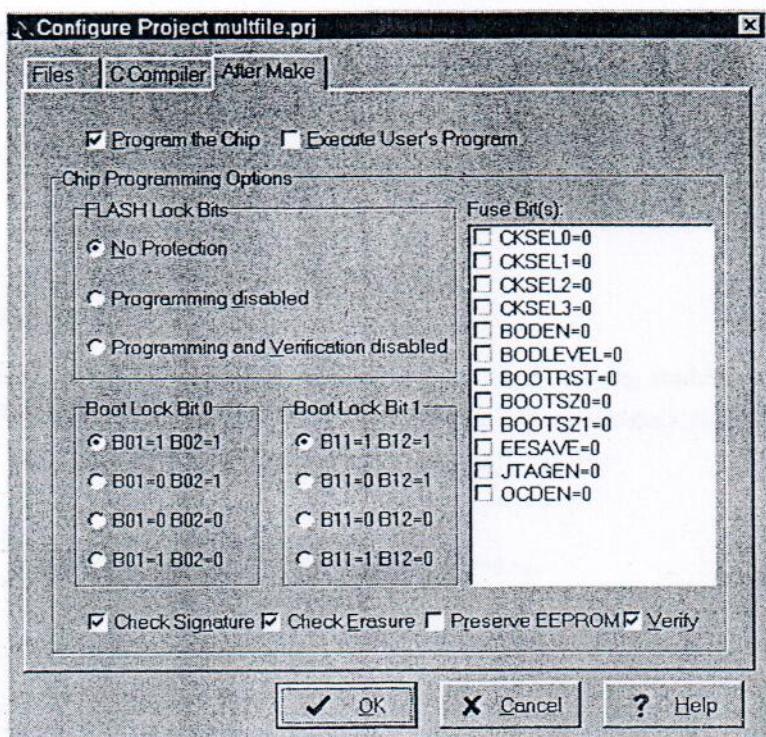
برای تنظیم گزینه های کامپایلر C برای پروژه ای که در حال حاضر باز می باشد، باید از منوی Project | Configure استفاده کنید و گزینه Compiler C را انتخاب کنید (به شکل ۲۸-۵ توجه کنید).



شکل ۲۸-۵- تنظیم گزینه های کامپایلر C

تراسه مورد نظر را در قسمت Chip انتخاب کنید. فرکانس پالس ساعت پردازنده نیز باید مشخص شود که در Delay Dallas Semiconductor DS1820/DS1822 Temperature Sensors Function، Function مقداردهی اولیه UARTها مورد استفاده قرار می گیرد. اگر برنامه از ارتباط سریال استفاده می کند، می بایست گزینه Initialize I Wire Protocol Functions، Function را انتخاب نموده و Baud را نیز تعیین کنید. کامپایلر بطور خودکار کد start-up را برای مقداردهی اولیه UART Baud Rate، تولید می کند. مدل حافظه مورد نیاز را در قسمت Memory Model مشخص نمایید. مدل حافظه به دو صورت Tiny و Small می باشد. برنامه کامپایل شده، را برای داشتن حجم کم، سرعت اجرای بالا با استفاده از گزینه های Optimize for|Size، Optimize for|Speed می توان بهبود بخشید. برای برنامه هایی که خود برنامه ریز (Self Programming) هستند، نوع برنامه به دو صورت می تواند انتخاب شود:

- Application •
- Boot Loader •



شکل ۳۰-۵- انتقال برنامه‌ی به تراشه‌ی AVR

با انتخاب گزینه Program the Chip، پس از کامپایل و اسambil کردن، برنامه بطور خودکار با استفاده از نرمافزار Built_in برنامه‌ریزی، به تراشه AVR منتقل خواهد شد. هر یک از موارد ذیل یا همه با هم از طریق گزینه Program the Chip قابل اجرا خواهند بود.

۱. پاک کردن حافظه‌های تراشه
۲. بررسی کردن خالی بودن حافظه‌های EEPROM و FLASH
۳. برنامه‌ریزی حافظه FLASH و تائید محتوای آن
۴. برنامه‌ریزی حافظه EEPROM و تائید محتوای آن
۵. برنامه‌ریزی بیت‌های فیوز و قفل

شما می‌توانید تراشه‌ای را که می‌خواهید برنامه‌ریزی کنید از قسمت Chip انتخاب کنید. برای محافظت از کپی کردن برنامه‌تان، باید گزینه‌های مرتبط را با استفاده از FLASH Lock Bits انتخاب کنید.

در هنگام پروگرام کردن با استفاده از نرم افزار CodeVision، باید به این نکته دقت داشت که با زدن گزینه All، بیت‌های فیوز در حالتی که مقادیر درستی برای آن‌ها تنظیم نشده است، در میکروکنترلر مقداردهی نشوند. اینکار می‌تواند موجب قرار گرفتن میکروکنترلر در وضعیت ناخواسته گردد. به عنوان مثال تنظیم بیت‌های فیوز برای حالتی که میکروکنترلر از ساعت کریستالی استفاده نماید، موجب می‌شود که میکروکنترلر دیگر با ساعت داخلی خود کار نکند و تا اتصال کریستال و خازن‌های مربوطه به مکیروکنترلر امکان استفاده از میکروکنترلر فراهم نخواهد بود. بنابراین بهتر است همواره برای پروگرام کردن از گزینه Program -> Erase Chip استفاده کنید.

- زیرمجموعه‌ی پلاگین مورد نظر (eClipse Plugins for AVR/...) را بوسیله‌ی کلیک کردن روی علامت مثلث‌شکل کنار نام پلاگین باز کنید و دو زیرمجموعه‌ی موجود در دسته‌ی Other را انتخاب کنید و دکمه‌ی Next را در پایین پنجره کلیک کنید.
- سپس در پنجره‌ی باز شده (Install) گزینه‌ی I accept the terms in the license agreement را انتخاب کرده و دکمه‌ی Next را در پایین پنجره کلیک کنید.
- اکنون در پنجره‌ی Install باید در قسمت Features to install دو عنوان AVR Eclipse Plugin... قابل مشاهده باشد. در این صورت دکمه‌ی Finish را در پایین صفحه کلیک کنید.
- سپس در پنجره‌ی باز شده (Verification) با کلیک کردن دکمه‌ی All ، نصب پلاگین را تایید کنید.
- پس از اتمام نصب پلاگین، محیط Wascana برای اضافه کردن پلاگین جدید به امکانات قابل استفاده برای کاربر، باید restart شود. پس با کلیک کردن دکمه‌ی Yes در جواب پیغام نرمافزار، فرآیند نصب را تکمیل کنید. بعد از آغاز دوباره‌ی محیط Wascana ، امکان ایجاد پروژه‌ی AVR به محیط اضافه شده است.

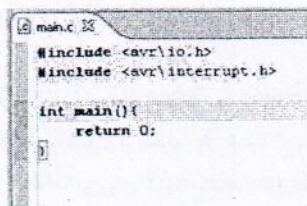
۳-۲-۵- انجام یک پروژه‌ی ساده با WinAVR

پس از نصب plugins مربوطه از منوی File قسمت C -> New را انتخاب می‌کنیم. در پنجره‌ی باز شده نام پروژه را در Project Name وارد کرده و تنظیمات زیر را اعمال می‌نماییم:

Project Type: AVR Cross Target Application

Toolchain: AVR-GCC Toolchain

در صفحه‌ی AVR Target Properties نوع تراشه AVR و سرعت ساعت مورد نظر را انتخاب و سپس دگمه Finish را کلیک می‌کنیم تا پروژه ساخته شود. حال، روی پروژه ساخته شده راست کلیک می‌کنیم. از منوی باز شده از گزینه‌ی هتو، روی گزینه‌ی Source File کلیک می‌کنیم و نام فایلی که ایجاد می‌شود را main.c می‌گذاریم. برای شروع، کد پایه زیر را می‌نویسیم:



```
main.c
#include <avr/io.h>
#include <avr/interrupt.h>

int main(){
    return 0;
}
```

شکل ۳-۱۵- برنامه main.c

کتابخانه‌ی io.h دارای ثبات‌های ورودی و خروجی برای AVR و Interrupt.h برای دسترسی به وقفه‌ها می‌باشد. کتابخانه Interrupt.h در این پروژه استفاده نمی‌شود و صرفا برای معرفی آورده شده است. باید توجه داشت که در اینجا برخلاف CodeVision ویزاردی برای مقداردهی به ثبات‌های AVR وجود ندارد، لذا باید تسلط کافی بر استفاده از روی ثبات‌های کنترلی و وضعیت داشته باشیم. کد لازم برای چشمک زدن یک LED که به پایه PA0 میکروکنترلر متصل است در ذیل آورده شده است.تابع delay() از کتابخانه‌ی delay.h برای ایجاد تأخیر زمانی استفاده شده است.

۶- جلسه چهارم

هدف از آزمایش:

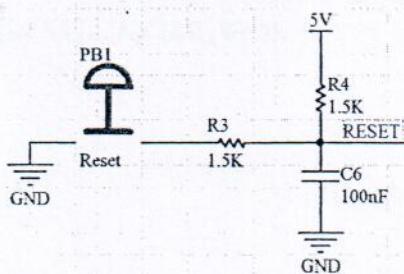
- نصب میکروکنترلر بر روی برد برقی و برقراری اتصالات به صورت دستی^۱
- تولید سیگنال بازن Shanی
- چگونگی برنامه ریزی بیت های فیوز
- تولید سیگنال ساعت (clock)

قطعات مورد استفاده:

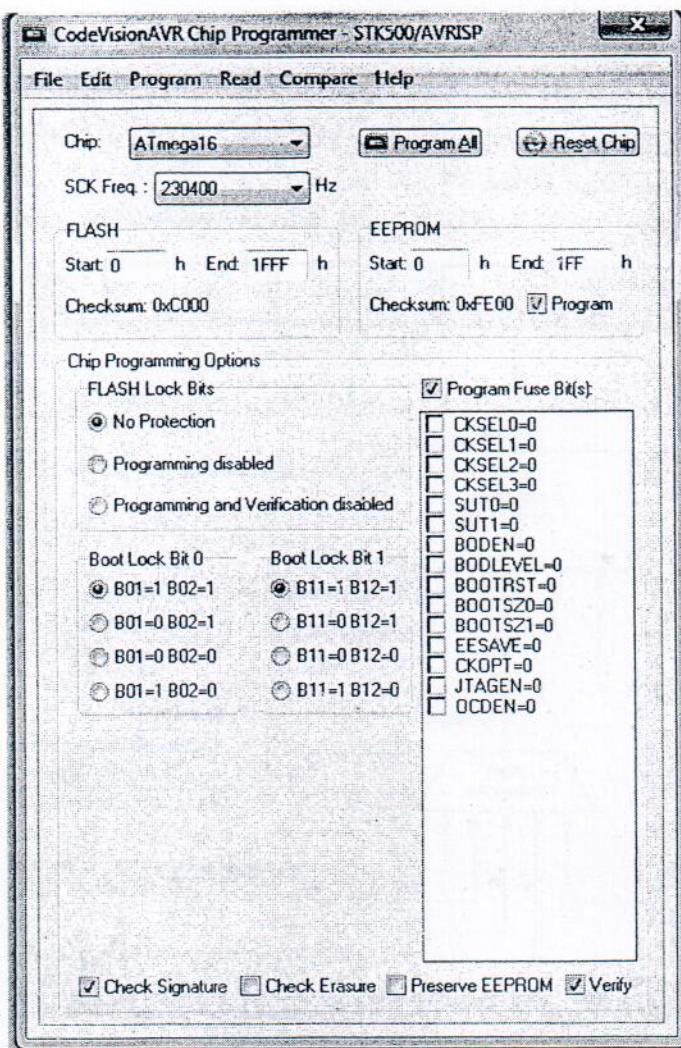
- برد برقی: ۱ عدد
- میکروکنترلر ATmega32 یا ATmega16: ۱ عدد
- مقاومت ۱.۵K: ۲ عدد
- خازن ۱۰۰nf: ۱ عدد و خازن ۲۲ یا ۲۷ پیکوفاراد: ۲ عدد
- کلید فشاری: ۱ عدد
- کریستال کوارتز: ۱ عدد

۶-۱- تولید سیگنال بازن Shanی

برای بهبود عملکرد سیستم مبتنی بر میکروکنترلر، لازم است که مداری نیز برای بازن Shanی کردن آن ظراحتی و استفاده نمائیم. می دانیم که میکروکنترلر دارای پایه ای بنام ریست یا بازن Shanی (Reset) است. این پایه در میکروکنترلر ATmega32 فعال روی سطح پائین بوده و وظیفه آن بردن میکروکنترلر به وضعیت بازن Shanی است. این بازن Shanی بلطفاً پس از وصل تغذیه میکروکنترلر و یا بصورت دستی قابل انجام می باشد. برای این منظور، مدار بازن Shanی زیر پیشنهاد می شود. پایه RESET میکروکنترلر را به مدار بازن Shanی ارائه شده در شکل ۶-۱ متصل می کنیم. با فشردن کلید PB1 عمل بازن Shanی بصورت دستی انجام می شود. این مدار همچنین قادر است در زمان وصل نمودن تغذیه میکروکنترلر، میکروکنترلر را بازن Shanی نماید.



^۱ در بعضی از جلسات اولیه آزمایشگاه، اتصال بین پایه های میکروکنترلر و قطعات جانبی بر روی برد برقی و توسط دانشجو انجام می شود. اینکار موجب آشنایی بیشتر دانشجویان با پایه های میکروکنترلر و اتصال قطعات جانبی به آنها می گردد. در آزمایشات بعدی برای بالا بردن سرعت انجام آزمایش ها از برد آموزشی استفاده خواهد شد.

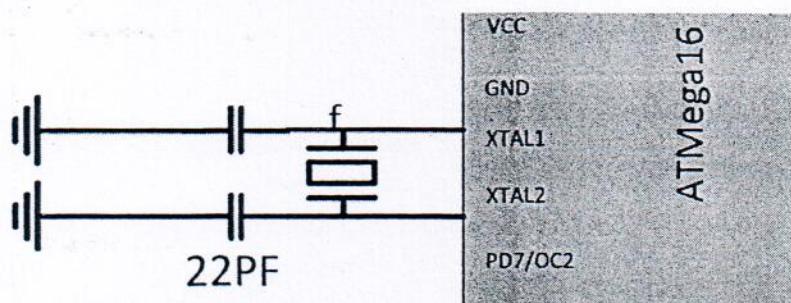


شکل ۳-۶- تغییر مقادیر بیت‌های فیوز

بیت‌های فیوزی را که می‌خواهیم برنامه‌ریزی (0 شوند) را تیک می‌زنیم و با انتخاب گزینه Program Fuse bit مورد نظر را برنامه‌ریزی می‌نمائیم. در هنگام پروگرام کردن با استفاده از نرم افزار CodeVision، باید به این نکته دقت داشت که با زدن گزینه Program All، بیت‌های فیوز در حالتی که مقادیر درستی برای آن‌ها تنظیم نشده است، در میکروکنترلر مقداردهی نشوند. اینکار می‌تواند موجب قرار گرفتن میکروکنترلر در وضعیت ناخواسته گردد. به عنوان مثال تنظیم بیت‌های فیوز برای حالتی که میکروکنترلر از ساعت کریستالی استفاده نماید، موجب می‌شود که میکروکنترلر دیگر با ساعت داخلی خود کار نکند و تا اتصال کریستال و خازن‌های مربوطه به مکیروکنترلر امکان استفاده از میکروکنترلر فراهم نخواهد بود. بنابر این بهتر است همواره برای پروگرام کردن از گزینه Program -> Erase Chip و سپس گزینه Program -> Flash استفاده گردد تا بدین ترتیب فقط برنامه در حافظه فلاش قرار گیرد. در صورت نیاز به تغییر بیت‌های فیوز حتماً در ابتدا مقادیر فعلی بیت‌های فیوز را از قسمت Read -> Fuse Bits خوانده و سپس بیت‌های فیوز مورد نظر را تغییر داده و نهایتاً از قسمت Program آنها را برنامه‌ریزی نمائید.

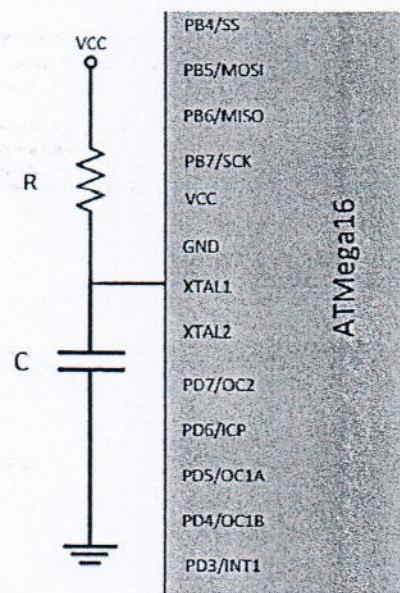
• اسیلاتور کریستالی فرکانس پائین (32.768 کیلوهرتز)

الف- با استفاده از دیتاشیت، میکروکنترلر ATmega32 و برنامه‌ریزی بیت‌های فیوز میکروکنترلر، سیگنال ساعت را به روش استفاده از کریستال تولید نمایید. شکل ۵-۶ محل قرار گرفتن کریستال با فرکانس f (بسته به کریستال موجود در آزمایشگاه که حداقل می‌تواند ۱۶ مگاهرتز باشد) و خازن‌های 22PF مورده نیاز برای تولید ساعت به این روش را نشان می‌دهد.



شکل ۵-۶- مدار تولید ساعت به روش استفاده از کریستال

ب- با استفاده از دیتاشیت میکروکنترلر ATmega32 و برنامه‌ریزی بیت‌های فیوز میکروکنترلر، سیگنال ساعت را به کمک روش اسیلاتور RC خارجی تولید نمایید (شکل ۶-۶). تغییرات فرکانس ساعت با تغییر مقادیر مقاومت و خازن را با استفاده از یک فرکانس‌متر یا اسیلوسکوپ مشاهده نمایید. فرکانس ساعت بر روی کدام پایه میکروکنترلر قابل مشاهده است؟



شکل ۶-۶- مدار تولید ساعت به روش استفاده از مدار RC خارجی

۷- جلسه پنجم

هدف از آزمایش:

- کار با درگاهها
- سرکشی (polling) یک پایه از یک درگاه
- کار با رله و بازر
- تولید تأخیر نرمافزاری
- کار با زمانسنج نگهبان

قطعات مورد نیاز:

- برد برد: ۱ عدد
- میکروکنترلر ATmega16 یا ATmega32: ۱ عدد
- مقاومت ۵۶۰ اهم: ۲ عدد
- مقاومت ۲.۲ کیلواهم: ۴ عدد
- مقاومت ۱.۵K: ۱ عدد (برای مدار ریست)
- خازن 100nf: ۱ عدد (برای مدار ریست)
- کلید فشاری: ۲ عدد
- LED: ۲ عدد
- کلید فشاری ۲ عدد
- ترانزیستور 2N3804: ۲ عدد
- رله ۵ ولت: ۱ عدد

میخواهیم به اهداف این آزمایش در قالب ساخت یک مدار برای روشن و خاموش کردن یک LED دست یابیم. با توجه به شکل ۱-۷ اقدامات زیر را انجام دهید. در این آزمایشات تولید ساعت میکروکنترلر به روش استفاده از مدار RC داخلی و با فرکانس ۱MHz باشد. مدار ریست را نیز مشابه مدار شکل ۱-۶ در همه آزمایشاتی که با برداشت کار میکنید بیندید.

الف- کلید SW1 را به پایه PD2 وصل کنید. این پایه را به عنوان ورودی برنامه‌ریزی کنید و آنرا بالاکش کنید. یک عدد LED و یک عدد مقاومت را مطابق شکل زیر به پایه PC1 وصل کنید. این پایه را در وضعیت خروجی قرار دهید. برنامه‌ای به زبان اسمبلی بنویسید که با فشردن کلید SW1 دیود LED1 روشن شود. اطلاع از وضعیت پایه PD2 متصل به SW1 به روش سرکشی انجام می‌شود.

ب- یک بازر را با واسطه یک ترانزیستور همانند شکل زیر به پایه PC7 متصل کنید. مشابه بند قبل برنامه‌ای بنویسید که با فشار کلید SW1 بازر روشن و با برداشتن دست از روی کلید، بازر خاموش شود. برای روشن شدن بازر پایه PC7 باید high شود.

ج- مشابه بند قبل کاری کنید که با هر بار فشردن کلید SW1 رله Relay1 وصل و LED2 روشن و با فشار مجدد کلید SW1 رله قطع و LED2 خاموش شود. رله با واسطه یک ترانزیستور به پایه PC0 وصل است. برای روشن شدن رله میبایست پایه PC0 در

۸- جلسه ششم

هدف از آزمایش:

- تهیه برنامه وقفه بازنشانی
- کار با وقفه‌های خارجی
- کار با درگاهها
- کار با نمایش‌دهنده 7-Seg

قطعات مورد نیاز:

- میکروکنترلر ATmega32 یا ATmega16 ۱ عدد
- مقاومت ۵۶۰ اهم: ۲ عدد
- مقاومت ۲،۲ کیلواهم: ۴ عدد
- کلید فشاری: ۲ عدد
- LED: ۲ عدد
- کلید فشاری: ۲ عدد
- ترانزیستور NPN: ۲ عدد
- ترانزیستور PNP: ۱ عدد
- نمایش‌دهنده 7-Seg: ۱ عدد
-

می‌خواهیم به اهداف این آزمایش، در قالب ساخت یک مدار برای روشن و خاموش کردن یک LED با استفاده از وقفه دست باییم. اقدامات زیر را انجام دهید. در این آزمایشات تولید ساعت میکروکنترلر به روش استفاده از مدار RC داخلی و با فرکانس ۱MHz باشد.

الف- روال وقفه بازنشانی را نوشه و در آن مقداردهی ثبات اشاره‌گر پشته را انجام دهید. حالات مختلفی که در آنها بتوان بردار وقفه و نیز روال وقفه را در بخش رامانداز (boot) یا بخش کاربردی (application) حافظه فلاش قرار داد را امتحان نمایید.

ب- پایه PD2 که کلید SW1 به آن متصل است، ورودی وقفه خارجی INT0 نیز می‌باشد. می‌خواهیم با یک بار فشردن کلید SW1 دیود نوری LED1 در شکل ۱-۸ روشن و با فشردن همین کلید برای بار دوم، LED1 خاموش گردد. روشن و خاموش کردن LED در داخل روتین وقفه خارجی INT0 انجام گردد. حالات مختلف ممکن حساسیت وقفه به لبه و حساسیت به سطح را برای ورودی وقفه امتحان کنید. برنامه مربوطه را شامل تنظیمات اولیه ثبات‌های کنترلی مورد استفاده، برنامه روتین وقفه و سایر اقدامات لازم ارائه نمایید.

ج- پایه PD3 که کلید SW2 به آن متصل است، ورودی وقفه خارجی INT1 نیز می‌باشد. می‌خواهیم با یک بار فشردن کلید INT1، LED1 مدت ۲ ثانیه روشن و سپس خاموش گردد. به منظور بررسی امکان داشتن وقفه تودرتو، در هنگام ورود به وقفه INT0، بیت وقفه سراسری I را ۱ نمایید و امکان پذیرش وقفه در حین اجرای روتین وقفه INT0 را فراهم سازید. بدین ترتیب چنانچه

۹- جلسه هفتم

هدف از آزمایش:

- کار با وقفه خارجی
- راهاندازی یک کیبورد ماتریسی
- نمایش کاراکترها توسط نمایش دهنده ۷ قطعه‌ای

قطعات مورد نیاز:

- میکروکنترلر ATmega16
- کیبورد
- نمایش دهنده ۷ قطعه‌ای
- ترانزیستور PNP
- مقاومت
- دیود
- کلید فشاری
- LED

در این آزمایش هدف طراحی یک کیبورد ماتریسی و نمایش مقادیر دریافتی از کیبورد توسط نمایش دهنده ۷ قطعه‌ای است. سختافزار شکل ۹-۱ شامل یک کلید، یک LED، یک کیبورد ماتریسی و یک کیبورد دارای تعداد سطر است که به پایه‌های PC4 الی PC7 متصل هستند. ستون‌های کیبورد با واسطه تعدادی مقاومت به پایه‌های PC0 الی PC3 متصل هستند. همین ستون‌ها از طرف دیگر با واسطه تعدادی مقاومت و دیود به پایه INT0 میکروکنترلر متصل هستند.

پایه‌های PC0 الی PC3 که به ستون‌های کیبورد وصل هستند در حالت ورودی و بصورت بالاکش و سطرهای کیبورد (پایه‌های PC4 الی PC7) را در حالت خروجی قرار داده‌ایم. ستون‌ها از طریق ۴ عدد دیود بصورت "و سیمی" (wired and) به هم مرتبط شده و با فشرده شدن هر یک از کلیدها، یکی از ستون‌ها در سطح منطقی ۰ قرار می‌گیرد. بدین ترتیب دیود مربوط به آن ستون در بایاس مستقیم قرار گرفته و اتصال آند دیودها که به ورودی وقفه خارجی INT0 متصل هستند در سطح منطقی ۰ قرار گرفته و می‌تواند باعث بروز وقفه گردد. بهتر است که پایه INT0 را در وضعیت ورودی بالاکش قرار دهیم. بدین ترتیب با فشردن یک کلید این پایه از وضعیت منطقی ۱ به وضعیت ۰ می‌رود. برای این منظور می‌توانید حساسیت پایه INT0 را در حالت حساسیت به لبه پایه رونده قرار دهید.

از آنجا که هر میکرو به طور پیش‌فرض به گونه‌ای تنظیم شده که بیت فیوز مربوط به JTAGEN در آن یک شده است، لذا پایه‌هایی از پورت C در وضعیت استفاده برای منظور JTAG قرار دارند و نمی‌توانند به عنوان پایه درگاه بصورت عادی استفاده شوند. لذا برای استفاده معمولی از این پورت باید ابتدا فیوز بیت JTAGEN را غیر فعال نماییم (حالت فیوز بیت برنامه‌ریزی نشده). در صورت استفاده از CodeVision، به منظور آزادسازی بیت‌های پورت C و استفاده از آنها به عنوان پایه‌های درگاه برای استفاده‌های عادی، بعد از خواندن بیت‌های فیوز (در قسمت Read->Fuse Bits)، مقدار بیت فیوز JTAGEN را غیر فعال نموده و سپس مقادیر بیت‌های فیوز را در میکروکنترلر مقداردهی نمائید (در قسمت Program->Fuse Bits).

اقدامات زیر را انجام دهید:

۱۰- جلسه هشتم

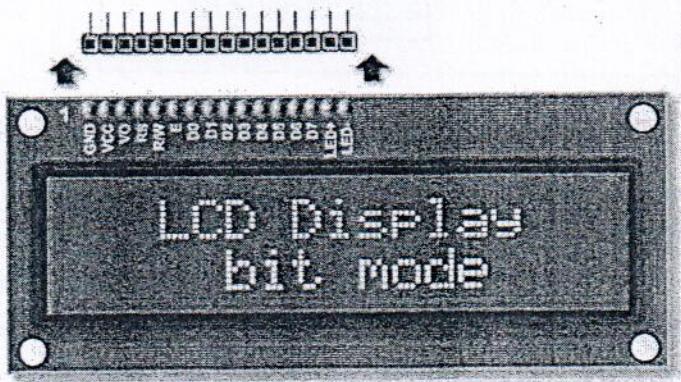
هدف از آزمایش:

- اتصال LCD به میکروکنترلر و نمایش اطلاعات دریافتی از کیبورد توسط آن

قطعات مورد نیاز:

- میکروکنترلر ATmega16
- کیبورد ۱۶ کلیدی
- LCD نمایش دهنده
- مقاومت
- پتانسیو متر
- دیود
- کلید فشاری

می خواهیم یک نمایش دهنده کاراکتری را به میکروکنترلر متصل و عبارت "Hello World" را بر روی آن نمایش دهیم. از یک LCD به شماره A LM02L، TS1620A و یا BONA2002 استفاده نمائید. شما باید از یک LCD و مشخصات و اسمای پایه های آن در شکل زیر ملاحظه می گردید.



شکل ۱-۱۰- نمایش دهنده کاراکتری (LCD) و مشخصات پایه های آن

LCD را همانند شکل ۲-۱۰ به درگاه A میکروکنترلر متصل کنید. برای تنظیم کنترast LED یک پتانسیو متر با مقدار 5K اهم بین پایه VO و GND مطابق شکل قرار دهید. کد اسsemblی ارتباط با LCD کاراکتری از طریق اینترنت قابل دسترسی است. یکی از این برنامه ها کد اسsemblی LCD_4bit.asm است که از طریق سایت های مشخص شده در مراجع [۱] و [۲] قابل دسترسی است. این مراجع در انتهای این بخش معرفی شده اند. یکی از توابع این برنامه تابع lcd_putchar است که کاراکتر موجود در یکی از ثبات های رجیستر فایل را بر روی LCD نمایش می دهد. سایر توابع مهم موردنیاز در این کد lcd_putchar، lcd_init، LCD_wait، LCD_getaddr و LCD_getchar است. لذا در صورت استفاده از این کد، می بایست درگاه D را در این کد به درگاه A تغییر نام دهید.

11- جلسه نهم

هدف از آزمایش:

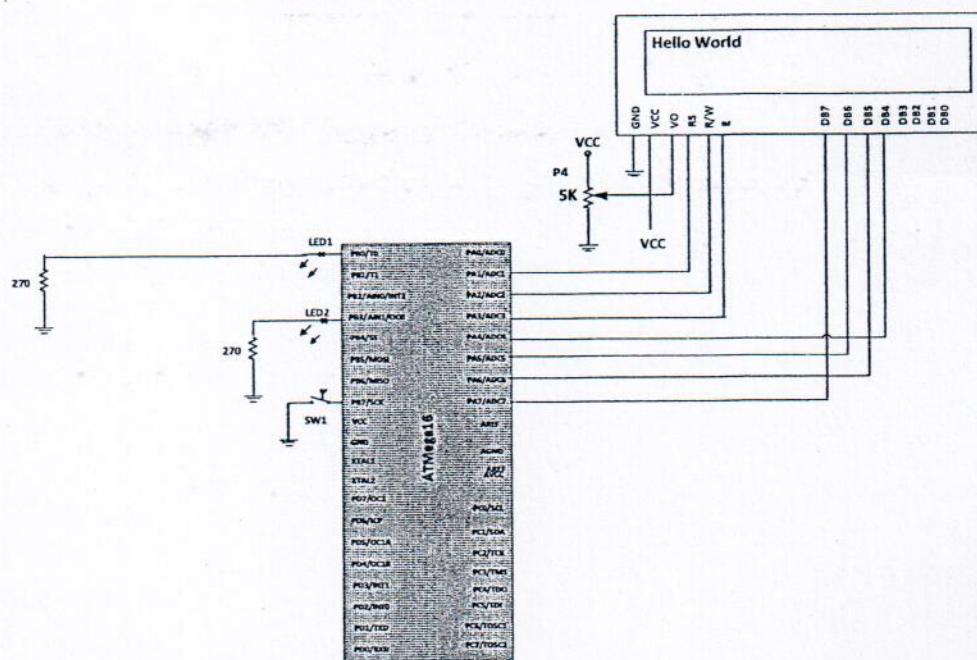
- کار با زمان‌سنج/شمارنده • میکروکنترلر در حالت عملکرد عادی
- کار با زمان‌سنج/شمارنده • میکروکنترلر در حالت CTC

قطعات مورد نیاز:

- میکروکنترلر ATmega16
- مقاومت
- LED
- کلید فشاری

الف- می‌خواهیم در مدار شکل ۱-۱۱ دیود نوری LED1 بطور متناوب روشن و خاموش شود و هر بار روشن شدن ۱ ثانیه بطول بیانجامد (با سیکل وظیفه ۰.۵۰٪). برای این منظور ساعت میکروکنترلر را روی ۱MHZ تنظیم کنید. اگر از زمان‌سنج • استفاده کرده و پیش‌ تقسیم کننده را روی تقسیم بر ۱۰۲۴ تنظیم کرده باشید، با ۴ بار مشاهده TOV0، زمان یک ثانیه سپری شده است. چرا؟ تنظیمات ثبات‌های کنترلی زمان‌سنج/شمارنده • و برنامه کار سیستم را بنویسید. از زمان‌سنج/شمارنده • در حالت عملکرد عادی استفاده کنید.

ب- می‌خواهیم که در مدار شکل ۱-۱۱ دیود نوری LED2 با فرکانس ۲ هرتز روشن و خاموش شود. برای این منظور ساعت میکروکنترلر را روی ۱MHZ تنظیم کنید. از زمان‌سنج/شمارنده • در حالت پاک کردن زمان‌سنج در برابر مقایسه (مود CTC) استفاده کرده و با استفاده از وقفه ناشی از مقایسه و مطابقت (compare-match)، دیود نوری LED2 را با فرکانس ۲ هرتز روشن و خاموش کنید. اسمی پایه‌های میکروکنترلر از جمله پایه OC0 که به پایه PB3 متصل است در شکل ۱-۱۱ ارائه شده است.



شکل ۱-۱۱- اتصال LED‌ها به درگاه B میکروکنترلر

۱۲- جلسه دهم

هدف از آزمایش:

- کار با زمان سنج/شمارنده ۰ و ۱ میکروکنترلر
- شمارش پالس های موجود بر روی پایه T1
- اندازه گیری فرکانس

قطعات مورد نیاز:

- میکروکنترلر
- نمایش دهنده LCD
- فانکشن ژنراتور
- پتانسیومتر
- مقاآمت های لازم

در مدار شکل ۱-۱۲ می خواهیم یک موج مربعی را از طریق پایه T1 (پایه PB1) متعلق به زمان سنج/شمارنده ۱ دریافت و تعداد پالس های ظاهر شده بر روی این پایه را شمارش کنیم. اگر تعداد یانس های یک موج مربعی را در ۱ ثانیه شمارش کنیم فرکانس آن موج مربعی بدست خواهد آمد. از زمان سنج/شمارنده ۰ برای تولید زمان ۱ ثانیه استفاده می کنیم. با راه اندازی زمان سنج/شمارنده ۰، شمارش پالس های دریافتی روی ورودی T1 را آغاز می کنیم و هنگامی که زمان ۱ ثانیه سپری شد، مقدار موجود در ثبات زمان سنج/شمارنده ۱ و ۱ پس از تبدیل به یک عدد ددهدی بر روی نمایش دهنده LCD نمایش می دهیم. برای سهولت فرض کنید که فرکانس موج ورودی حداکثر ۲۵۵ هرتز می باشد. با اتصال یک عدد فانکشن ژنراتور و تولید یک موج مربعی با دامنه تغییرات ۰ و ۵ ولت و اتصال آن به پایه T1، مدار فوق را امتحان نمائید.

اقدامات زیر را انجام دهید:

- الف- سخت افزار برد آموزشی را مطابق شکل ۱-۱۲ برای انجام این آزمایش آماده کنید.
- ب- تنظیمات ثبات های کنترلی زمان سنج/شمارنده های شماره ۰ و ۱ را انجام دهید.
- ج- برنامه کار سیستم را بنویسید و سیستم را راه اندازی و روند آزمایش و مشاهدات خود را گزارش نمایید.

۱۳- جلسه یازدهم

هدف از آزمایش:

- کار با زمان سنج/شمارنده • در حالت PWM سریع
- کار با زمان سنج/شمارنده • در حالت PWM با فاز صحیح
- تنظیم شدت روشنایی LED یا موتور DC ساده توسط موج PWM

قطعات مورد نیاز:

- میکروکنترلر
- ترانزیستور PNP 1N4148
- دیود
- مقاومت‌های لازم
- کیبورد ماتریسی با ۱۶ کلید
- LED: ۱ عدد
- موتور DC

مدار شکل ۱-۱۳ را درنظر بگیرید. می‌خواهیم روشنایی دیود نوری LED1 را با تغییر چرخه وظیفه موج PWM که از طریق پایه OC0 متصل به پایه PB3 تولید می‌شود تنظیم نمائیم می‌خواهیم برنامه کار این مدار را به گونه‌ای بنویسید که متناسب با رقمی که توسط کیبورد وارد می‌نمایید، میزان روشنایی LED بین حداقل مقدار (بازه رقم ۰) و حداکثر مقدار (بازه رقم ۹) تغییر نماید. خواسته فوق را در دو حالت زیر به انجام رسانید: با زیاد شدن سیکل وظیفه موج ایجاد شده، آیا شدت نور LED زیاد می‌شود یا کم؟ چرا؟

الف- تنظیمات ثبات‌های کنترلی زمان سنج/شمارنده • و برنامه کار سیستم را برای حالتی که از زمان سنج/شمارنده • در مود PWM سریع استفاده شده باشد انجام دهید.

ب- تنظیمات ثبات‌های کنترلی زمان سنج/شمارنده • و برنامه کار سیستم را برای حالتی که از زمان سنج/شمارنده • در مود PWM با فاز صحیح استفاده شده باشد انجام دهید.

ج- در صورت در اختیار داشتن یک موتور الکتریکی DC کوچک آنرا به جای دیود نوری بین کلکتور ترانزیستور و پایه زمین قرار داده و دور موتور را به کمک موج PWM در دو حالت فوق کنترل کنید. برای این منظور مقاومت ۲۷۰ اهم را تا حد مناسب کرده یا حذف کنید.

۱۴- جلسه دوازدهم

هدف از آزمایش:

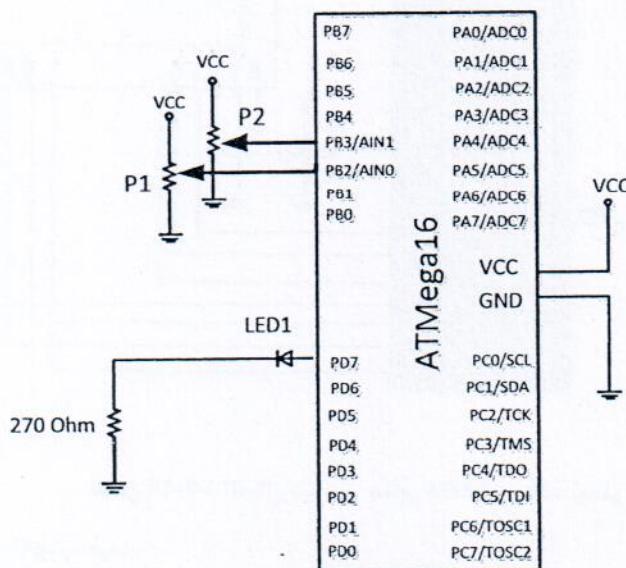
- کار با مقایسه کننده آنالوگ میکروکنترلر
- کار با مبدل آنالوگ به دیجیتال میکروکنترلر
- اندازه‌گیری دما و نمایش بر روی LCD

قطعات مورد نیاز:

- میکروکنترلر
- سنسور دمای LM35
- نمایش دهنده LCD
- پتانسیومتر
- LED

کار با مقایسه کننده آنالوگ:

در این آزمایش ابتدا می‌خواهیم با نحوه کار مقایسه کننده آنالوگ آشنا شویم. برای این منظور مدار شکل ۱-۱۴ را بر روی یک عدد برد بیندید. در اینجا از دو عدد پتانسیومتر ۵ یا ۱۰ کیلواهم استفاده شده است. سر بالای هر پتانسیومتر به VCC سریائین به زمین و سر وسط به پایه AIN0 یا AIN1 متصل شده است.

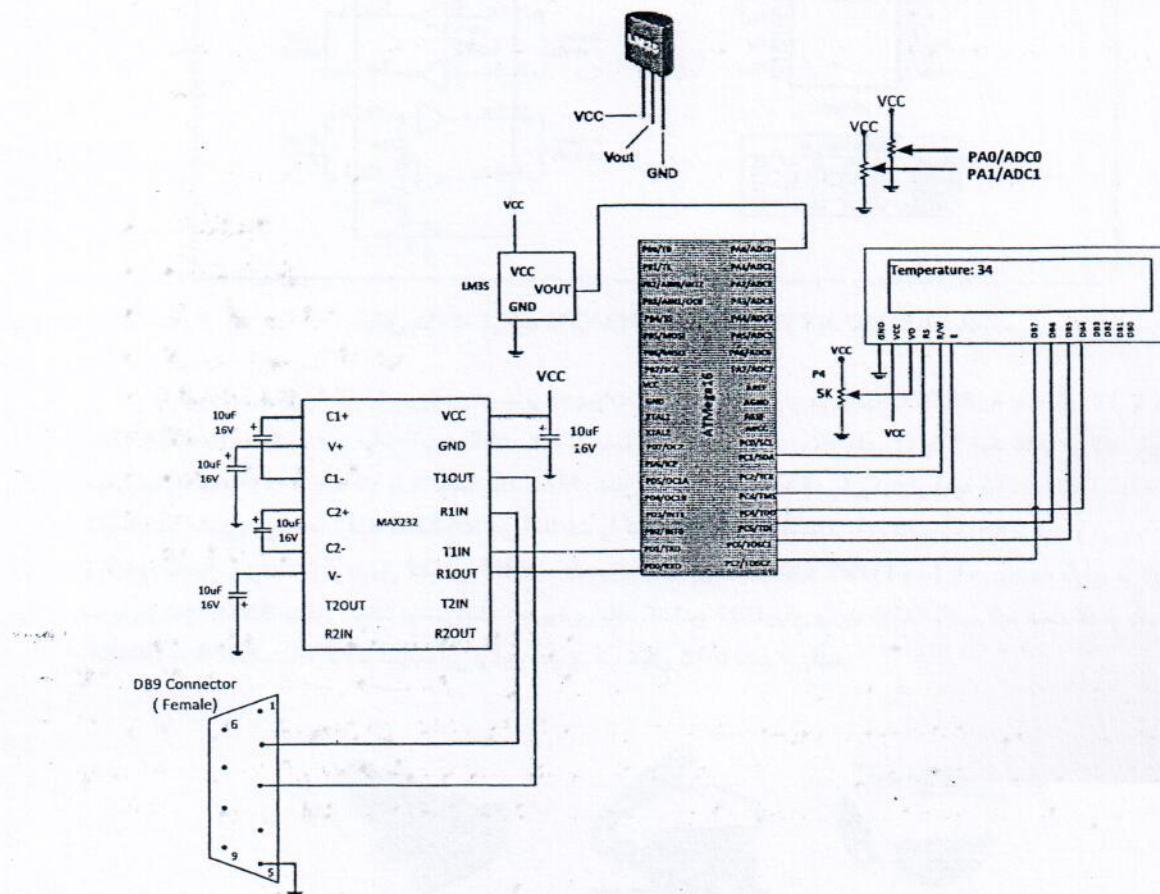


شکل ۱-۱۴ - اتصال پتانسیومرهای P1 و P2 به ورودی‌های مقایسه کننده آنالوگ

از ساعت کالیبره شده داخلی با فرکانس ۱MHz استفاده نمایید. مطابق شکل ۱-۱۴ سر وسط یک پتانسیومتر P1 را به پایه AIN0 (سر مثبت مقایسه کننده آنالوگ) و سر وسط پتانسیومتر P2 را به پایه AIN1 (سر منفی مقایسه کننده آنالوگ) متصل

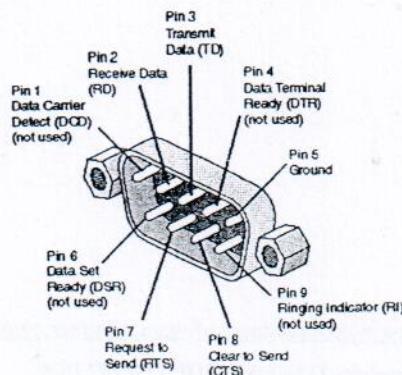
ج- در یک آزمایش دیگر سنسور دما را از پایه PA0 جدا کرده و سرهای وسط پتانسیومترهای P1 و P2 را مطابق شکل ۲۴-۵ به ورودی‌های ADC0 (PA0) و ADC1 (PA1) میکروکنترلر متصل نمائید و اختلاف ولتاژ بین آنها را توسط مبدل آنالوگ به دیجیتال اندازه‌گیری و در صورت مثبت یا منفی بودن این اختلاف ولتاژ، LED متصل به پایه PD7 را روشن و خاموش نمائید. با ثابت نگه داشتن پیج تنظیم پتانسیومتر P1 و چرخانیدن پیج تنظیم پتانسیومتر P2 می‌توانید اختلاف ولتاژها را منفی و مثبت نمائید و مقادیر اختلاف ولتاژ را بر روی LCD نمایش دهید. نظمیمات ثبات‌های کنترلی مورد نیاز و نیز برنامه کار سیستم را بنویسید.

سخت افزار شکل ۱-۱۵ را در نظر بگیرید. در این شکل از واسط USART برای ارتباط بین میکروکنترلر و یک کامپیووتر استفاده شده است. میکروکنترلر از طریق دو پایه RXD و TXD یعنی پایه های PD0 و PD1، یک تراشه واسط RS232، یک کانکتور DB9 و یک مبدل RS232 به USB با کامپیووتر ارتباط برقرار می کند. اطلاعات ارسالی از طرف میکروکنترلر برای کامپیووتر می تواند داده های وارد شده از طریق کیبورد، دمای اندازه گیری شده توسط سنسور دما و یا هر دنباله دیگری از داده ها باشد. داده های ارسالی از طرف کامپیووتر می تواند بر روی LCD نمایش داده شود.



شکل ۱-۱۵ - ارتباط سریال از طریق واسط USART

به منظور برقراری ارتباط سریال برای ارسال و دریافت همگام-ناهمگام اطلاعات بین تجهیزات مختلف از تراشه هایی جهت تبدیل اطلاعات از قالب TTL به قالب هایی مانند قالب R232C استفاده می شود. یکی از این تراشه ها تراشه MAX232 می باشد. ساختار تراشه MAX232 در شکل ۲-۱۵ آمده است.



شکل ۴-۱۵ پایه‌های کانکتور ۹ از نوع DB9 Male

به منظور برقراری ارتباط از طریق کامپیوتر با واسط سریال، USART، می‌توانید از کامپوننت‌هایی که برای این منظور بصورت آماده موجود است استفاده نمایید. برای این منظور می‌توانید از کامپوننت USART Serial Connection .NET Component در محیط برنامه‌نویسی .NET استفاده و برنامه‌ای برای ارسال و دریافت سریال از طریق واسط USART بنویسید و بدین ترتیب بین کامپیوتر و میکروکنترلر ارتباط برقرار نمایید. جهت اطلاعات بیشتر به پیوست ۲ مراجعه نمایید. راه دیگر برقراری ارتباط با کامپیوتر با استفاده از ارتباط USART استفاده از برنامه Hyper Terminal یا بخش ترمینال در نرمافزار Codevision است. توضیحات لازم برای استفاده از این برنامه در پیوست ۳ آمده است.

می‌خواهیم ارتباط بین میکروکنترلر و کامپیوتر بصورت ناهمگام و با نرخ بیت ۴۸۰۰ بیت در ثانیه و با فریم‌های داده حاوی یک، بیت شروع، ۷ بیت، داده، توازن، فعال از نوع زوج و ۱ بیت پایانی انجام گردد. فرکانس ساعت میکروکنترلر $f_{osc} = 1 \text{ MHz}$ باشد و با ساعت کالیبره شده داخلی تولید شود. در اینصورت اختلاف بین نرخ بیت تولید شده و نرخ بیت واقعی ۴۸۰۰ بیت در ثانیه در حدود ۲٪ است. به منظور برقراری ارتباط بین میکروکنترلر و کامپیوتر اقدامات زیر را انجام دهید.

الف- ثبات‌های کنترلی مربوط به واسط USART میکروکنترلر را برنامه‌ریزی نمایید.

ب- برنامه لازم برای ارسال اطلاعات تایپ شده از طریق کیبورد کامپیوتر و نمایش آنها بر روی نمایش‌دهنده LCD را بنویسید.

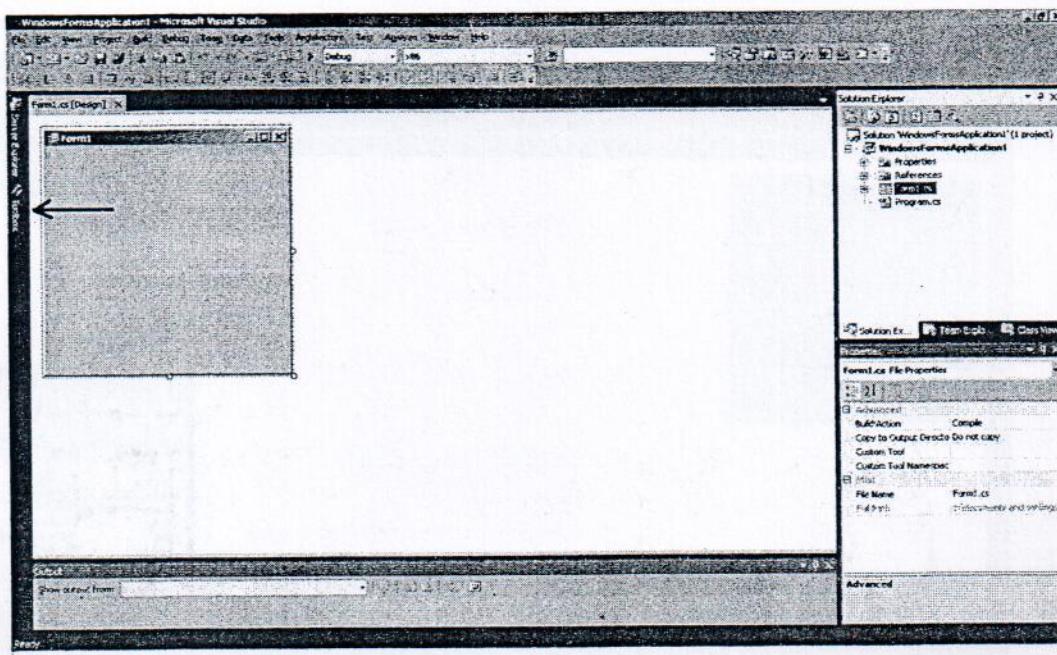
ج- برنامه لازم برای ارسال اطلاعات تایپ شده از طریق کیبورد متصل به میکروکنترلر به کامپیوتر و نمایش آنها بر روی مانیتور کامپیوتر را بنویسید.

۱۷- پیوست ۱: پایه‌های میکروکنترلر ATMega32

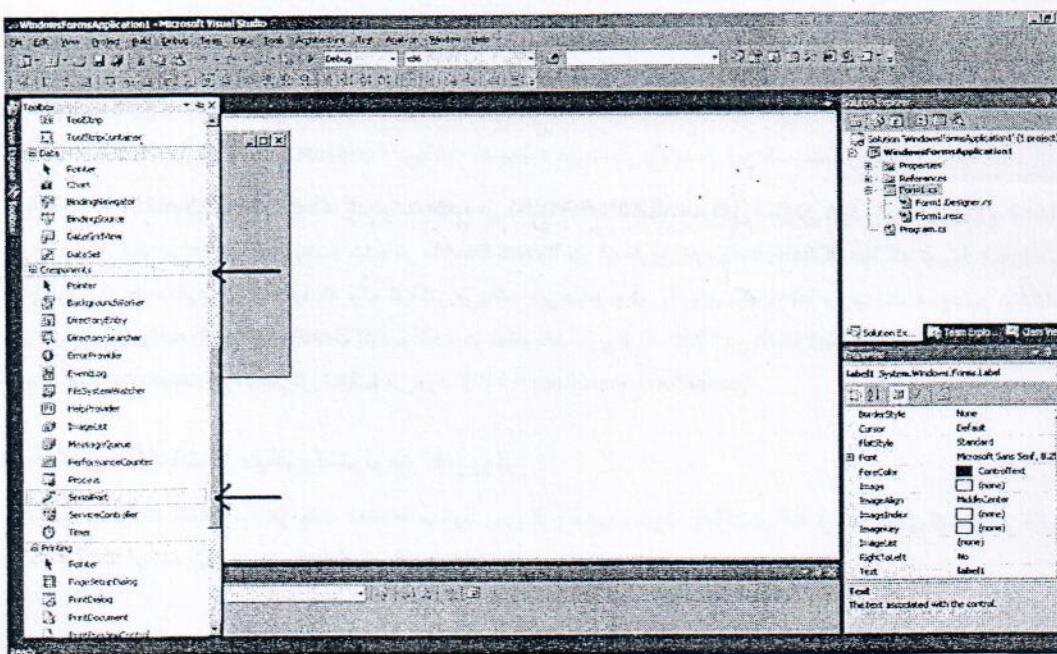
شکل زیر پایه‌های میکروکنترلر ATMega32 را نشان می‌دهد. پایه‌های درگاه‌های A، B، C و D هر کدام به یک رنگ مشخص شده‌اند.

(XCK/T0) PB0	1	40	PA0 (ADC0)
(T1) PB1	2	39	PA1 (ADC1)
(INT2/AIN0) PB2	3	38	PA2 (ADC2)
(OC0/AIN1) PB3	4	37	PA3 (ADC3)
(SS) PB4	5	36	PA4 (ADC4)
(MOSI) PB5	6	35	PA5 (ADC5)
(MISO) PB6	7	34	PA6 (ADC6)
(SCK) PB7	8	33	PA7 (ADC7)
RESET	9	32	AREF
VCC	10	31	GND
GND	11	30	AVCC
XTAL2	12	29	PC7 (TOSC2)
XTAL1	13	28	PC6 (TOSC1)
(RXD) PD0	14	27	PC5 (TDI)
(TXD) PD1	15	26	PC4 (TDO)
(INT0) PD2	16	25	PC3 (TMS)
(INT1) PD3	17	24	PC2 (TCK)
(OC1B) PD4	18	23	PC1 (SDA)
(OC1A) PD5	19	22	PC0 (SCL)
(ICP1) PD6	20	21	PD7 (OC2)

نشانگر موس را روی لبه Toolbox قرار داده تا یک لیست از ابزارهای موجود، به نمایش درآیند. در شکل بعد، این لیست را مشاهده می‌کنید.



شکل ۲-۱۸- ایجاد فرم برای اضافه کردن کامپوننت‌های مورد نیاز



شکل ۳-۱۸- مشاهده لیست ابزارها

```
#using ..

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            this.Show();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            serialPort1.Open();
            if (serialPort1.IsOpen)
            {
                char a;
                serialPort1.BaudRate = 110;
                a = (char)serialPort1.ReadChar();
                label1.Text = a.ToString();
            }
            else label1.Text = "Serial Port is Closed";
            serialPort1.Close();
        }
    }
}
```

شکل ۵-۱۸- برنامه انتقال اطلاعات از میکروکنترلر به کامپیوتر

۱۸-۲- انتقال اطلاعات از کامپیوتر به میکروکنترلر

در برنامه نوشته شده زیر، هر وقت button موجود روی فرم فشرده شود، کاراکتری توسط کامپیوتر به میکروکنترلر فرستاده می‌شود و میکروکنترلر توسط متده `getchar()`، کاراکتر ارسالی را دریافت می‌کند.

```
#using ..

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            this.Show();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            serialPort1.Open();
            if (serialPort1.IsOpen)
            {
                char[] buffer = new char[1];
                buffer[0] = 'a';
                serialPort1.BaudRate = 110;
                serialPort1.Write(buffer, 0, 1);
            }
            else label1.Text = "Serial Port is Closed";
            serialPort1.Close();
        }
    }
}
```

شکل ۶-۱۸- برنامه انتقال اطلاعات از کامپیوتر به میکروکنترلر

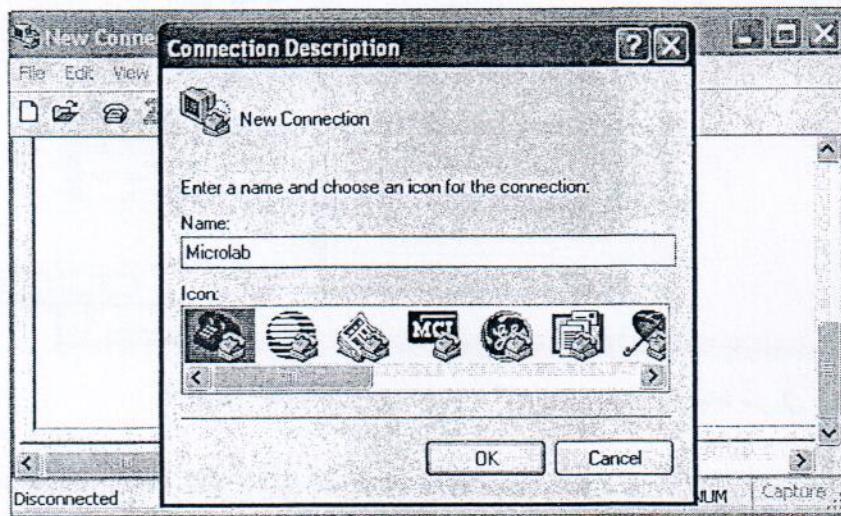
با یک دستور ساده یک کاراکتر را بر روی کامپیوتر و از طریق Hyper Terminal نمایش می‌دهیم. برای این منظور در قسمت برنامه میکروکنترلر دستور ساده زیر را می‌نویسیم:

```
while (1){
    // Place your code here
    putchar ('b');
}
```

این کد پس از برنامه‌ریزی بر روی میکروکنترلر، با آمدن هر پالس ساعت و به طور مداوم کاراکتر 'b' را از طریق Hyper Terminal بر روی مانیتور نشان می‌دهد. کد زیر با نمایش جمله "please enter a key:" بر روی صفحه Hyper Terminal منتظر دریافت داده از کاربر می‌شود، و پس از آنکه کاربر کلیدی را فشرد، آنرا بر روی صفحه نمایش چاپ می‌کند.

```
while (1){
    // Place your code here
    putsf("please enter a key:");
    x= getchar();
    lcd_putchar (x);
}
```

دسترسی به برنامه Hyper Terminal از طریق منوی All Programs, accessories, accessories, آنگاه منوی Communication امکان‌پذیر است. همانند شکل ۳-۱۹ پس از ورود به محیط Hyper Terminal، از شما خواسته می‌شود تا یک نام به اتصال جدیدی، که قصد برقراری آن را دارید اختصاص دهید.



شکل ۳-۱۹-انتخاب نام برای اتصال

شماره پورت سریالی که از طریق آن می‌خواهید با کامپیوتر ارتباط سریال برقرار کنید را همانند شکل ۴-۱۹ مشخص کنید.

۲۰- پیوست ۴: کد ارتباط به LCD در حالت ۴ بیتی

کد زیر برای نمایش کاراکترها توسط یک LCD کاراکتری در حالت ۴ بیتی قابل استفاده است.

```
;*****
;File:          m8_LCD_4bit.asm
;Title:         ATmega8 driver for LCD in 4-bit mode (HD44780)
;Assembler:     AVR assembler/AVR Studio
;Version:       1.0
;Created:      April 5th, 2004
;Target:        ATmega8
;Christoph Redecker, http://www.avrbeginners.net
;*****  
  
; Some notes on the hardware:  
; ATmega8 (clock frequency doesn't matter, tested with 1 MHz to 8 MHz)  
; PortD.1 -> LCD RS (register select)  
; PortD.2 -> LCD RW (read/write)  
; PortD.3 -> LCD E (Enable)  
; PortD.4 ... PortD.7 -> LCD data.4 ... data.7  
; the other LCD data lines can be left open or tied to ground.  
  
.include "c:\program files\atmel\avr studio\appnotes\m8def.inc"  
  
.equ  LCD_RS      = 1
.equ  LCD_RW      = 2
.equ  LCD_E       = 3  
  
.def  temp       = r16
.def  argument   = r17           ;argument for calling subroutines
.def  return      = r18           ;return value from subroutines  
  
.org 0
rjmp reset  
  
reset:
    ldi    temp, low(RAMEND)
    out   SPL, temp
    ldi    temp, high(RAMEND)
    out   SPH, temp  
  
;LCD after power-up: ("*" means black bar)
;*****|  
;  
    rcall LCD_init  
  
;LCD now:
;|&          | (&: cursor, blinking)
;|  
  
    rcall LCD_wait
    ldi    argument, 'A'           ;write 'A' to the LCD char data RAM
    rcall LCD_putchar  
  
;|&
```

```

out    PortD, temp           ;and write the port value
sbi    PortD, LCD_RS        ;now take RS high for LCD char data register access
sbi    PortD, LCD_E         ;strobe Enable

nop
nop
nop
cbi    PortD, LCD_E         ;restore the argument, we need the low nibble now...
pop    argument              ;clear the data bits of our port value
cbr    temp, 0b11110000      ;we want to write the LOW nibble of the argument to
swap   argument              ;the LCD data lines, which are the HIGH port nibble!
cbr    argument, 0b00001111  ;clear unused bits in argument
or     temp, argument        ;and set the required argument bits in the port value
out    PortD, temp           ;write data to port
sbi    PortD, LCD_RS        ;again, set RS
sbi    PortD, LCD_E         ;strobe Enable

nop
nop
nop
cbi    PortD, LCD_E         ;PortD, LCD_RS
cbi    PortD, LCD_RS        ;temp, DDRD
in     temp, DDRD            ;data lines are input again
cbr    temp, 0b11110000
out    DDRD, temp

ret

lcd_command:                   ;same as LCD_putchar, but with RS low!
push   argument
in     temp, DDRD
sbr   temp, 0b11110000
out   DDRD, temp
in     temp, PortD
cbr   temp, 0b11111110
cbr   argument, 0b00001111
or    temp, argument

out    PortD, temp
sbi    PortD, LCD_E
nop
nop
nop
cbi    PortD, LCD_E
pop    argument
cbr    temp, 0b11110000
swap   argument
cbr    argument, 0b00001111
or    temp, argument
out    PortD, temp
sbi    PortD, LCD_E
nop
nop
nop
cbi    PortD, LCD_E
in     temp, DDRD
cbr    temp, 0b11110000
out    DDRD, temp

```

```
ret
```

```
LCD_delay:  
    clr    r2  
    LCD_delay_outer:  
    clr    r3  
    LCD_delay_inner:  
    dec    r3  
    brne   LCD_delay_inner  
    dec    r2  
    brne   LCD_delay_outer  
ret
```

```
LCD_init:  
    ldi    temp, 0b00001110          ;control lines are output, rest is input  
    out    DDRD, temp  
  
    rcall  LCD_delay              ;first, we'll tell the LCD that we want to use it  
    ldi    argument, 0x20          ;in 4-bit mode.  
    rcall  LCD_command8          ;LCD is still in 8-BIT MODE while writing this command!!!  
  
    rcall  LCD_wait               ;NOW: 2 lines, 5*7 font, 4-BIT MODE!  
    ldi    argument, 0x28          ;  
    rcall  LCD_command  
  
    rcall  LCD_wait               ;now proceed as usual: Display on, cursor on, blinking  
    ldi    argument, 0x0F          ;  
    rcall  LCD_command  
  
    rcall  LCD_wait               ;clear display, cursor -> home  
    ldi    argument, 0x01          ;  
    rcall  LCD_command  
  
    rcall  LCD_wait               ;auto-inc cursor  
    ldi    argument, 0x06          ;  
    rcall  LCD_command  
ret
```