دانشکده مهندسی برق و کامپیوتر

دستور کار آزمایشگاه ریزپردازنده

نسخه ۰,۲

تهیه کننده‌گان:

محمدرضا رحیمی

محمد اذانگو

تابستان ۱۳۹۴


بسم الله الرحمن الرحيم

فهرست مطالب

عنوان **صفحه**

قوانین و مقررات

۱) هر گروه (متشکل از دو دانشجو) موظف است هر هفته پیش گزارش همان آزمایش و گزارش‌کار آزمایش قبلی را به صورت کتبی ارائه دهد. در صورت آماده نبودن پیش گزارش، نمره‌ای به دانشجویان تعلق نگرفته و غیبت برای آن‌ها منظور خواهد شد.

۲) دانشجویان حتما باید در گزارش خود، کدها را با کمک نرم‌افزارهای شبیه‌ساز مثل Proteus شبیه سازی کرده و تحلیل نتایج شبیه سازی را با در نظر گرفتن حالات مختلف مورد بررسی قرار دهند.

۳) آمادگی دانشجویان در اجرای کدهای مشابه و مرور سرفصل‌های مرتبط درس ضروری می‌باشد و جزو ارزیابی کلاسی به حساب می‌آید.

۴) دانشجویان باید پس از انجام آزمایش سیستم خود را مرتب و صندلی‌ها را در جای خود قرار دهند.

۵) دانشجویان می‌توانند با هماهنگی قبلی تنها در یک جلسه آزمایشگاه غیبت داشته باشند و بیش از یک جلسه غیبت موجب صفر شدن نمره آن خواهد شد.

۶) قبل از شروع به سیم‌بندی بورد حتما از خاموش بودن آن اطمینان حاصل کنید.

۷) پس از پایان سیم‌بندی و قبل از روشن کردن بورد حتما از صحت سیم‌بندی با مشورت مسئول آزمایشگاه مطمئن شوید.

۸) هنگام جابجا کردن سیم‌ها به جهت جلوگیری از قطع سیم‌ها، بدنه فیش و نه سیم را گرفته و از کانکتور جدا و یا به آن وصل نمایید.

۹) از جداکردن کابل پروگرامر جدا خودداری کنید.

۱۰) همه پورت‌های USB کامپیوتر های آزمایشگاه غیر فعال هستند و اتصال حافظه‌های جانبی ممنوع می‌باشد به این ترتیب هر گروه پس از اتمام آزمایش باید اصلاحات کد خود را به صورت کتبی جهت تکمیل گزارش‌کار یادداشت نماید و یا از آن عکس بگیرد.

۱۱) همه برنامه‌ها و کدهای نوشته شده توسط دانشجویان با خاموش شدن سیستم به طور خودکار پاک می شوند.

۱۲) استفاده از لپ تاپ در آزمایشگاه مجاز است.

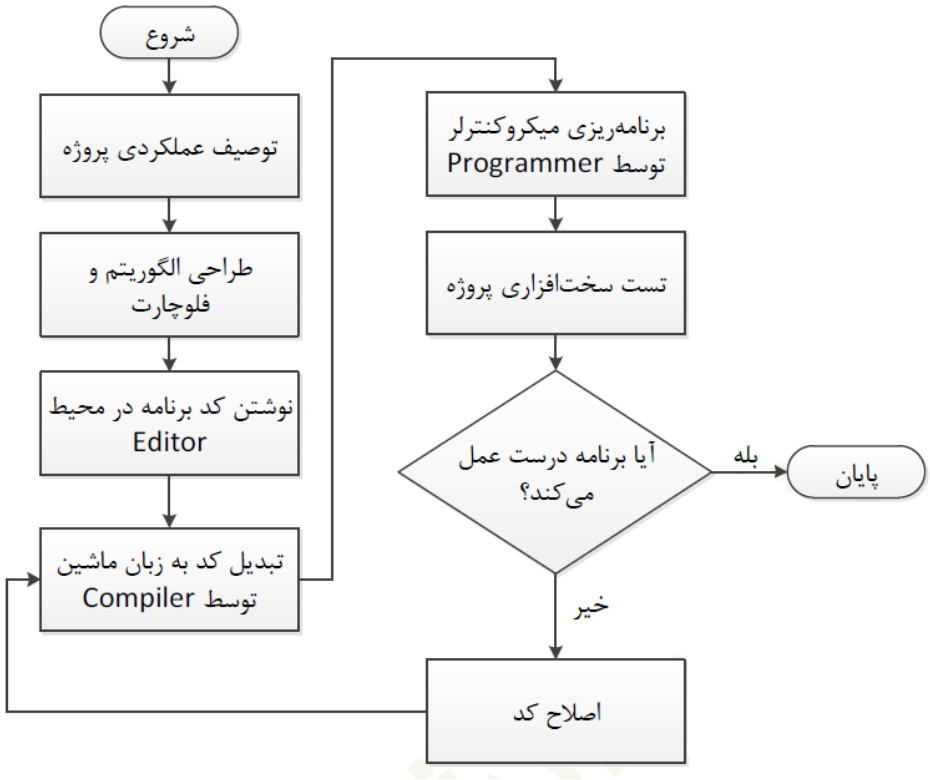
توجه:

با توجه به بروز رسانی تجهیزات آزمایشگاه، دستور کار پیش‌رو هم زمان با ترم اول سال تحصیلی ۹۵-۱۳۹۴ تکمیل خواهد شد، به همین جهت لازم است جهت دسترسی به آخرین نسخه دستور کار به صورت همیشگی به سایت درس به آدرس زیر مراجعه نمایید. پیش بینی می‌شود دستورکار پیش رو سه بار حداقل در طول ترم جاری به روز رسانی شود و نسخه نهایی دستورکار برای ترم آتی آماده شود.

http://saba.kntu.ac.ir/eecd/People/Azangoo/Microprocessor_Lab.html

روش حل مسئله:

لازم است برای پیاده‌سازی پروژه‌های خود در درس ریزپردازنده، نقشه راهی مانند فلوچارت شکل زیر را طی نمایید.



مراحل انجام یک پروژه ریزپردازنده

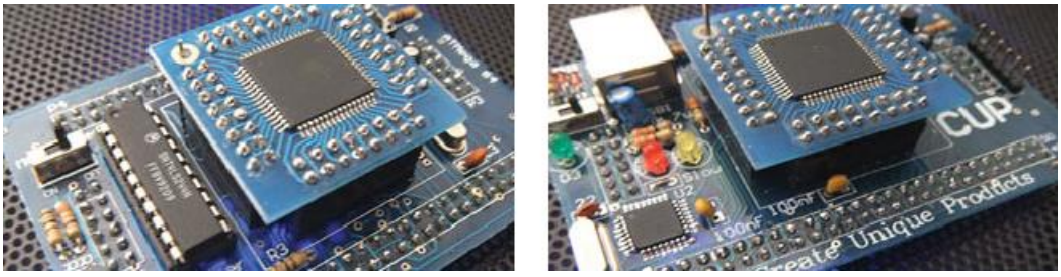
فصل ۱– آشنایی با سخت‌افزار

در ادامه به معرفی بردهای جدید مورد استفاده در آزمایشگاه ریزپردازنده دانشگاه خواجه نصیرالدین طوسی خواهیم پرداخت.

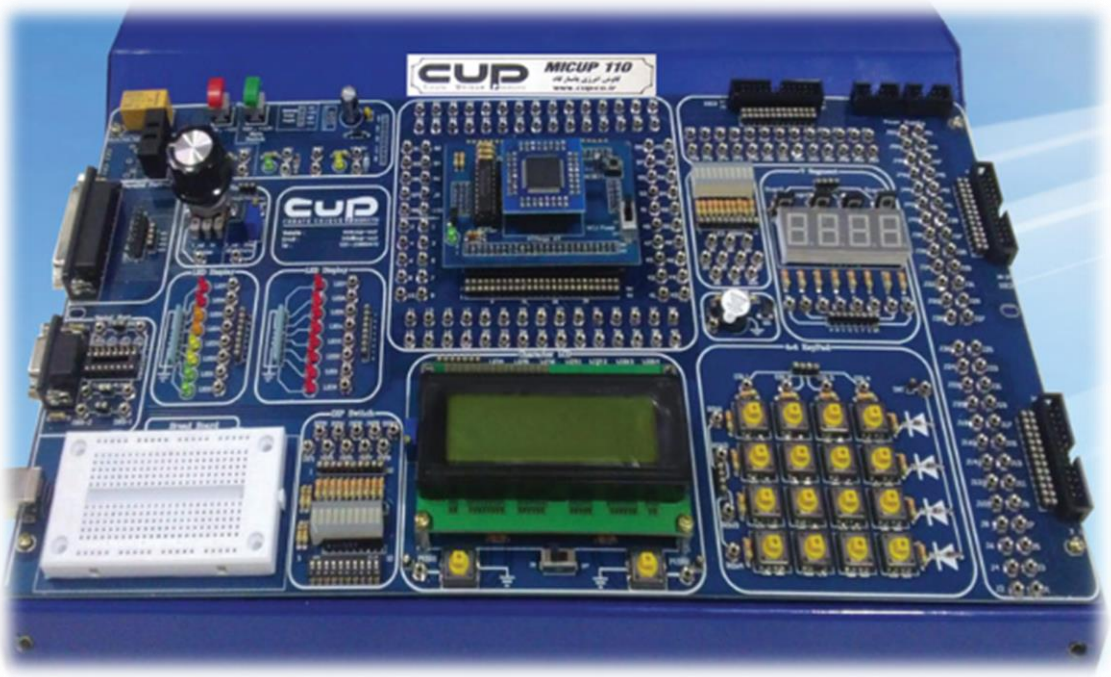
۱–۱– آشنایی با بورد های موجود در آزمایشگاه

برای کار با میکروکنترلر همچنین دستگاه‌های جانبی آن باید از بوردهای موجود در آزمایشگاه کمک گرفت این بوردها به سه قمست کلی تقسیم می‌شوند.

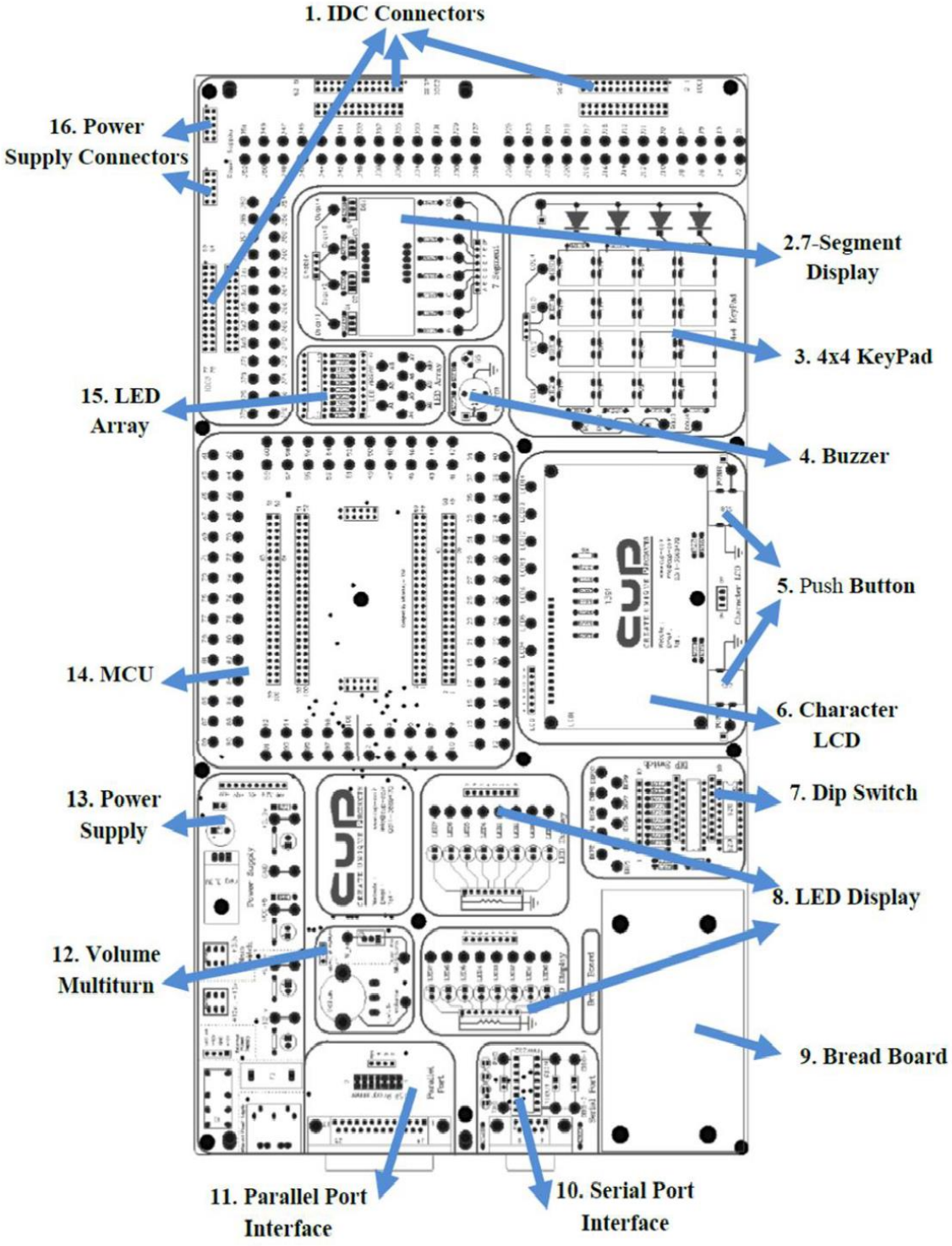
بورد اصلی که هسته میکروکنترلر بر روی آن قرار دارد و بعضی واحد های جانبی ضروری مثل صفحه کلید ماتریسی و نمایشگر هفت قسمتی و LED و ... بر روی آن قرار دارد که آنها را در شکل زیر می‌بینید.



تراشه های قابل نصب بر روی بورد اصلی



بورد اصلی (آبی رنگ) ریزپردازنده



شماتیک و اجزای بورد اصلی (آبی رنگ)

این بورد با هدف آشنایی کاربر با مفاهیم اساسی در زمینه میکروکنترلر طراحی شده و به منظور افزایش کارایی و بازده آموزشی شش ویژگی مهم در این بورد لحاظ شده است:

۱- طراحی ماژولار و ترسیم شماتیک مدار
۲- قابلیت کار با انواع پردازنده (همچون AVR، PIC، FPGA و ...) بدون تعویض بورد اصلی
۳- قابلیت اتصال همزمان دو بورد جانبی استاندارد
۴- آزاد بودن اتصالات سخت افزاری و در دسترس بودن کلیه پایه‌ها در بورد اصلی
۵- استفاده از کانکتورهای استاندارد
۶- امکان تعریف آزمایش‌های متعدد با توجه به نیازهای آزمایشگاه

از دیگر امکانات برد اصلی می‌توان به موارد زیر اشاره کرد
۱- وجود Pin Headerهای متناظر با کانکتورهای استاندارد برای ارتباط با بردبورد و ماژول‌های دیگر
۲- در دسترس بودن ولتاژهای ۳/۳+ و ۱۲+ و ۱۲- و ۵+ ولت
۳- رعایت انواع حفاظت‌ها در برابر خرابی و نویز در ماژول‌های مختلف
۴- ...

اجزای بورد ارتباطی اصلی شامل موارد زیر می‌باشد:

- IDC Connector:

این کانکتورها جهت اتصال بورد اصلی به بوردهای جانبی استفاده می‌شوند. در کنار هر کانکتور IDC به ازای هر پایه، یک کانکتور 2mm و همچنین یک پین هدر متناظر وجود دارد که با استفاده از آن می‌توان اتصال دلخواه را با ماژول MCU برقرار نمود. بنابر این می‌توان یک پایه خروجی یا ورودی در بورد جانبی را از این طریق به پایه‌های میکروکنترلر متصل کرد. در بورد اصلی سه عدد IDC بیست و شش عددی تعبیه شده است که پین هدرها و کانکتورهای متناظر با پایه‌های آن به طور نظیر به نظیر و پیوسته از ۱ تا ۷۸ شماره-گذاری شده‌اند.

- 7-Segment Display:

این نمایشگر چهار رقمی برای نمایش اعداد و برخی حروف لاتین مورد استفاده قرار می‌گیرد. در این ماژول یک پایه Enable به نام های Digit 1...4 از نوع پین هدر و کانکتور 2mm وجود دارد، اما هشت پایه داده برای هر چهار 7-segment مشترک هستند که یک کردن هر کدام از پایه‌های داده موجب روشن شدن Segment متناظر با آن خواهد شد. بنابر این برای نمایش

یک عدد خاص بر روی هر 7-Segment باید داده مناسب را بر روی پایه‌های داده (پین‌هدر و یا کانکتور 2mm با نام‌های A..DP) ارسال کرده و Enable آن را یک کرد. چناچه این کار به صورت متناوب و با فرکانس مناسب انجام شود می‌توان به صورت همزمان اعداد ۴ رقمی دلخواه را بر روی چهار 7-Segment رویت کرد.

- 4x4 KeyPad:

این صفحه کلید ماتریسی برای اعمال فرمان دلخواه به میکروکنترلر استفاده می‌شود. سخت افزار این صفحه کلید به شکلی طراحی شده است که میکرو کنترلر بتواند آن را هم به روش Polling و هم به روش وقفه بخواند. در این ماژول چهار کانکتور 2mm و پین هدر متناظر برای سطرها، چهار کانکتور 2mm و پین هدر متناظر برای ستون‌ها و یک کانکتور 2mm و پین هدر متناظر برای وقفه تعبیه شده است.

- Buzzer:

از این المان برای تولید صدای بوق اسفاده می‌شود و با اعمال یک موج مربعی با فرکانس دلخواه به کانکتور 2mm و یا پین هدر متناظر، می‌توان صدای متناظر با آن فرکانس را ایجاد کرد.

- Push Button:

از این کلیدها برای اعمال فرمان دلخواه به میکرو کنترلر استفاده می‌شود. فشار دادن این کلیدها یک سیگنال صفر در کانکتور 2mm و پین هدر متناظر ایجاد می‌کند و به محض رها کردن کلید، خروجی به سطر یک باز می‌گردد. قرار گرفتن این کلیدها در زیر LCD این امکان را به کاربر می‌دهد تا در ساخت منوها بر روی LCD، عملکرد این کلیدها را تعریف نماید.

- Character LCD:

این نمایشگر LCD که برای نمایش عبارت‌های دلخواه استفاده می‌شود، از نوع کاراکتری بوده و ظرفیت نمایش آن ۴ خط ۲۰ کاراکتری است. پایه‌های کنترلی RS و RW و E و پایه‌های داده D4 تا D7 از طریق هفت کانکتور 2mm و پین هدر متناظر در دسترس هستند. همچنین، در کنار LCD یک کلید ON/OFF تعبیه شده است که در زمان استفاده از LCD باید در وضعیت ON قرار داشته باشد. برای جلوگیری از آسیب LCD در هنگامی که از LCD استفاده نمی‌شود آن را خاموش نمایید.

- Dip Switch:

از این کلیدها برای اعمال مقادیر دیجیتال ورودی ثابت، تا سقف ۱۰ بیت به میکرو کنترلر استفاده می‌شود. با قرار دادن هر یک از کلیدها در وضعیت ON خروجی متناظر با آن (کانکتور 2mm و پین هدر) یک می‌شود و چنانچه کلیدی در وضعیت OFF قرار داشته باشد، خروجی متناظر با آن (کانکتور 2mm و پین هدر) صفر خواهد بود.

- LED Display

در این بورد دو سری نمایشگر LED هشت تایی تعبیه شده است که از این LEDها برای کنترل وضعیت پورت‌های میکروکنترلر و یا خروجی ماژول‌های دیجیتال (مثلا خروجی وقفه ماژول Keypad) می‌توان استفاده کرد. چنانچه سیگنال اعمال شده به کانکتور 2mm و یا پین هدر متناظر با هر یک از LEDها در سطح یک باشد، LED متناظر با آن روشن شده و اگر در سطح صفر باشد، LED متناظر با آن خاموش خواهد شد.

- Bread Board:

چنانچه نیاز باشد از یک المان خاص مانند یک IC که در بورد وجود ندارد استفاده شود، این این بردبورد می‌توان استفاده کرد.

- Serial Port Interface:

برای برقراری ارتباط سریال میان میکروکنترلر و رایانه می‌توان از این ماژول واسط استفاده کرد. این ماژول شامل کانکتور DB9 و مدار تبدیل سطح ولتاژ با استفاده از MAX232 است. این ماژول به شیوه‌ای طراحی شده است که پایه‌های Rx و Tx قبل و بعد از تبدیل سطح ولتاژ از طریق کانکتور 2mm و پین هدر متناظر، در دسترس هستند.

- Parallel Port Interface:

برای برقراری ارتباط موازی میان میکروکنترلر و رایانه می‌توان از این ماژول واسط استفاده کرد. از این ماژول برای پروگرام کردن میکروکنترلر نیز استفاده می‌شود. به همین دلیل در این ماژول یک کانکتور ISP برای پروگرام کردن میکروکنترلر در خارج از بورد، تعبیه شده است. علاوه بر این، با استفاده از این کانکتور ISP می توان از یک پروگرامر خارج از بورد که مجهز به خروجی ISP باشد برای پروگرام کردن میکروکنترل روی برد استفاده کرد.

- Volume Multiturn:

برای ایجاد ولتاژ آنالوگ متغیر می‌توان از Volume و یا Multiturn موجود در این ماژول استفاده کرد. این دو المان نوعی پتانسیومتر هستند و تفاوت آن‌ها این است که Multiturn دقت بالاتری داشته و برای رسیدن از مقدار حداقل به مقدار حداکثر آن، چندین دور قابل

چرخش است، این در حالی است که Volume دقت کمتری داشته و محدوده چرخش آن 270 درجه است. باید توجه داشت که ولتاژ مرجع Volume ثابت و برابر ۵+ ولت است و لذا خروجی کانکتور 2mm و پین هدر متناظر آن ولتاژ متغیری در محدوده صفر الی ۵+ ولت خواهد بود. اما ولتاژ مرجع Multiturn متصل نبوده و باید از ماژول تغذیه، یکی از ولتاژهای ۳/۳+ یا ۵+ یا ۱۲+ یا ۱۲- را به پایه V-ref آن متصل کرده و ولتاژ مرجع آن را تامین کرد.

- Power supply:

این ماژول وظیفه تامین ولتاژهای تغذیه سیستم را بر عهده دارد. ولتاژ تغذیه مورد نیاز المان-ها و ماژول‌های مختلف از پیش متصل شده و نیازی به اتصال تغذیه در ماژول‌ها نیست، اما چنانچه در کاربردهای خاص نیاز به ولتاژ ۳/۳+ و یا ۵+ و یا ۱۲+ و یا ۱۲- باشد، این ولتاژها از طریق کانکتور 2mm و پین هدر متناظر آن در دسترس هستند. توجه شود که با فشار دادن کلید Main Switch ولتاژ ۵+ و ۳/۳+ در مدار برقرار خواهد شد و در این حالت اگر کلید ۱۲+ ۱۲- نیز فشار داده شود، این دو ولتاژ نیز در مدار برقرار خواهند شد. توجه شود که Main switch کلید اصلی تغذیه بوده و در صورت غیرفعال بودن آن کلید ۱۲+ ۱۲- نیز تغذیه نخواهد داشت و بی‌اثر خواهد بود.

- MCU:

این بخش محل نصب بورد MCU[۱] یا همان کنترل کننده مرکزی است. این بخش به شیوه‌ای طراحی شده که ماژولار بوده و هر نوع پردازنده‌ای با حداکثر ۱۰۰ پایه بر روی آن قابل نصب خواهد بود. آزمایشگاه میکروی خواجه نصیر هم اکنون دارای چهار بورد پردازنده ATMega 64 & 2560 و میکروهای PIC و FPGAهای Spartan برای نصب در این بخش می‌باشد. پس از اتصال بورد MCU بر روی پین‌هدرهای مربوطه، پایه‌های تراشه از طریق ۱۰۰ کانکتور 2mm و پایه‌های پین‌هدر متناظر، به صورت نظیر به نظیر در دسترس خواهند بود. توجه شود که ترتیب کانکتورها و پین‌هدرهای متناظر بر روی این ماژول عینا متناظر با پایه‌های تراشه است. برای مثال پین‌هدر و کانکتور شماره ۱۴ مستقیما به پایه ۱۴ تراشه موجود بر روی برد MCU (برای مثال ATMega64) متصل خواهد بود. بنابراین در صورتی که لازم باشد اتصالی به پایه ۱۴ میکرو کنترل برقرار شود، کافیست از پین هدر و یا کانکتور شماره ۱۴ استفاده شود. لازم

[۱] Multipoint control unit

به ذکر است که به دلیل حفاظت در برابر اتصالات اشتباه، پایه‌های کریستال خارجی و تغذیه میکروکنترلر به پین‌هدر و کانکتور متناظر وصل نشده‌اند.

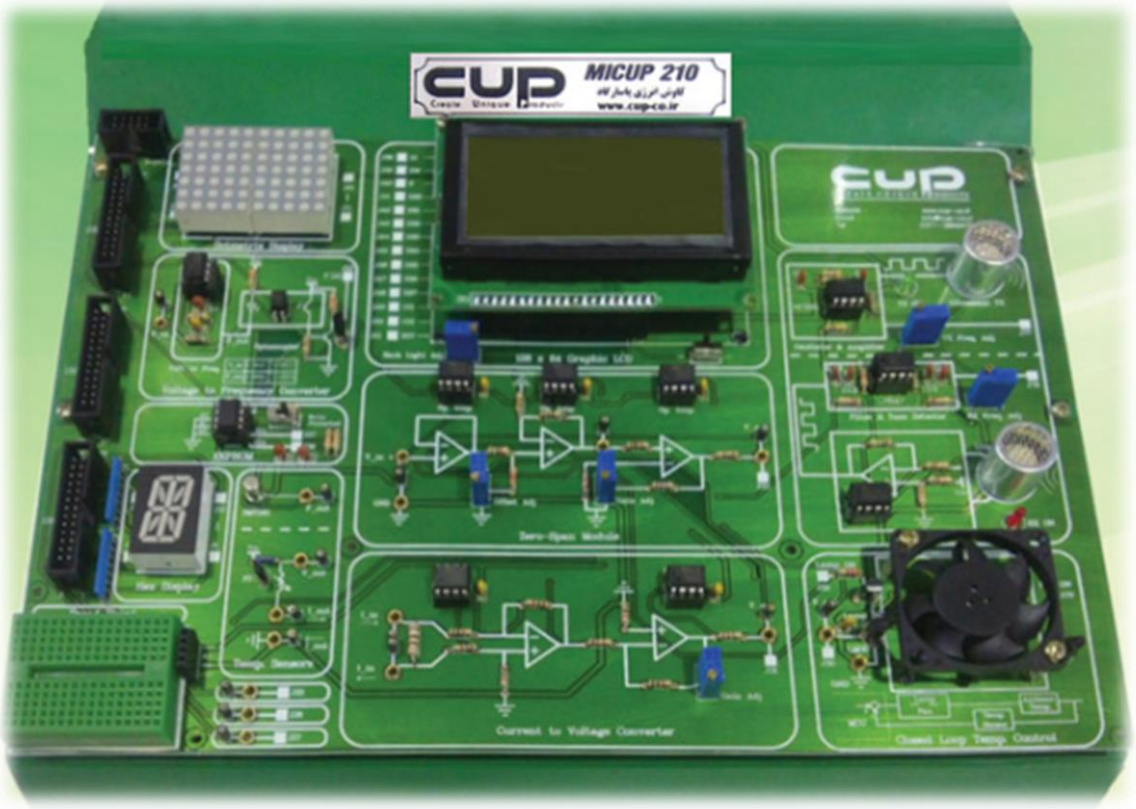
- LED Array:

عملکرد این ماژول مشابه LED Display است. پیکربندی این نمایشگر به شکلی است که بری استفاده به عنوان نمایشگر وضعیت نیز مناسب است. از نمایشگر وضعیت برای کاربردهایی نظیر نمایش ظرفیت پر شده یک مخزن، یا درصد پیشرفت یک فرایند استفاده می‌شود، در این حالت روشن بودن هر LED (از مجموع ۱۰ عدد LED) معادل ۱۰ درصد خواهد بود.

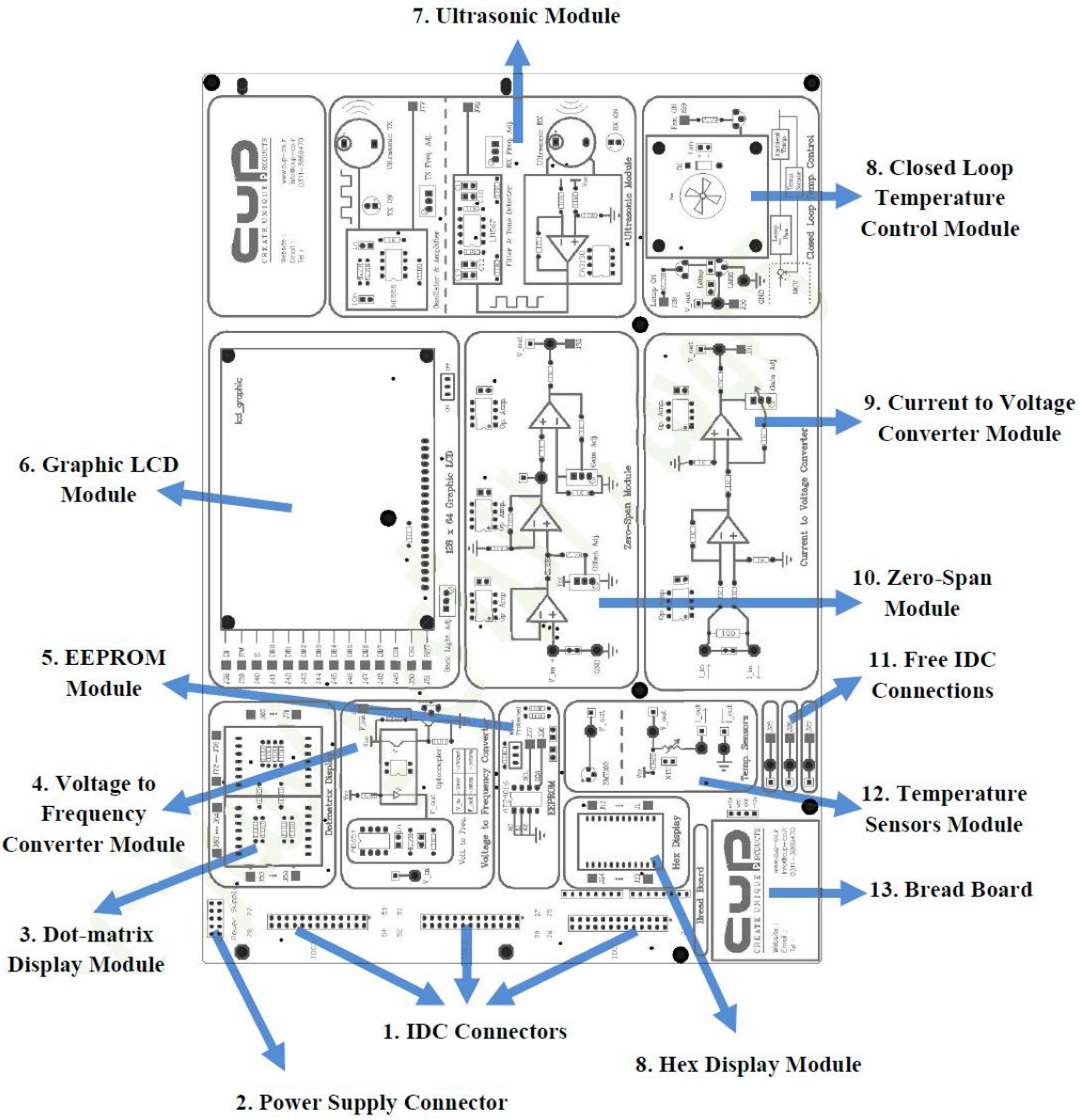
- Power Supply Connector:

از این کانکتورها برای تامین ولتاژ تغذیه بوردهای جانبی استفاده می‌شوند. بنابراین برای کارکردن با بوردهای جانبی لازم است کانکتور IDC تغذیه (۱۰تایی) و کانکتور IDC داده (۲۶تایی)، میان بورد اصلی و بورد جانبی متصل شود. در بورد اصلی دو عدد کانکتور IDC تغذیه تعبیه شده است، بنابراین به طور همزمان می‌توان ولتاژ تغذیه بورد جانبی را تامین نمود.

بورد سنسور و نمایشگر با هدف آشنایی کاربر با نحوه اتصال انواع سنسورها و نمایشگرها به میکروکنترلر طراحی شده و به منظور افزایش کارایی و بازده آموزشی طراحی آن به صورت منعطف و ماژولار صورت گرفته است. این بورد را می‌توان در کنار بورد اصلی استفاده و از طریق پردازنده مرکزی المان‌های آن را کنترل و مانیتور کرد.



بورد سنسور و نمایشگر (سبز رنگ)



شماتیک و اجزای اصلی بورد سنسور و نمایشگر (سبز رنگ)

بورد پروتکل‌های ارتباطی با هدف آشنایی کاربر با نحوه عملکرد پروتکل‌ها مختلف باسیم و بیسیم و نحوه تبادل اطلاعات آن‌ها با میکروکنترلر طراحی شده و به منظور افزایش کارایی و بازده آموزشی طراحی آن به صورت منعطف و ماژولار صورت گرفته است. این بورد را می‌توان در کنار بورد اصلی استفاده و از طریق پردازنده مرکزی المان‌های آن را کنترل و مانیتور کرد.

بورد پروتکل‌های ارتباطی (سفیدرنگ رنگ)

۱-۲- آشنایی با میکروکنترولر AVR

میکروکنترلر چیست؟

پیشرفت روزافزون تکنولوژی باعث ورود دستگاه‌های جدیدی به بازار شده است که زندگی امروزی را هر لحظه متحول می‌کنند بخش عمده ای از این پیشرفت ها مرهون وجود میکروکنترلرها و میکروپروسسورها است. میکروکنترلرها، قطعات الکترونیک قابل برنامه‌ریزی هستند. در حالت کلی، این تراشه‌ها دارای پردازشگر مرکزی، حافظه داخلی و امکانات ارتباطی برای کنترل دستگاه‌های دیگر هستند که استفاده از آنها باعث افزایش کارایی و کاهش حجم مدارهای الکترونیک خواهد شد. همچنین، ارتقا و به‌روزرسانی سیستم‌های مبتنی بر میکروکنترلرها بسیار راحت است و نیازی به تعویض قطعه اصلی سیستم، یعنی میکروکنترلر نبوده و تنها کافی است بار دیگر برنامه ریزی شوند. امروزه قیمت پایین و دسترسی آسان به میکروکنترلرها موجب افزایش روزافزون محبوبیت و کاربرد آنها در صنایع مختلف شده است.

تفاوت میکروپروسسور، میکروکامپیوتر و میکروکنترلر

شاید تاکنون به وفور با عبارت‌های میکروپروسسور، میکروکامپیوتر و میکروکنترلر برخورد کرده باشید در اغلب اوقات این عبارت‌ها را به صورت مترادف و جایگزین یکدیگر در نظر میگیرند؛ اما در حقیقت این سه واژه تفاوت‌های اساسی با هم دارند و نمی‌توان آنها را معادل یکدیگر قلمداد کرد.

یک میکروپروسسور در واقع یک واحد پردازشگر مرکزی است. این تراشه‌ها در گذشته با استفاده از مدارهای مجتمع با مقیاس متوسط و بزرگ طراحی می‌شدند. شرکت اینتل در سال ۱۷۹۱ برای نخستین بار پیاده سازی قطعات CPU، ALU ، رجیسترها و مدار کنترل باس را در تراشه ۴۰۰۴ انجام داد و به این ترتیب اولین میکروپروسسور ساخته شد. گاهی یک میکروپروسسور، مدارها و قطعات ورودی خروجی جانبی و حافظه ها در کنار یکدیگر قرار میگیرند تا کامپیوتری کوچک را به منظور تحلیل اطلاعات و کاربردهای کنترلی شکل دهند به چنین سیستم هایی واژه میکروکامپیوتر اطلاق می‌گردد حال چنانچه قطعات سازنده یک میکروکامپیوتر در یک تراشه سیلیکون در کنار یکدیگر قرار گیرند، این تراشه میکروکنترلر نامیده می‌شود.

مزایای میکروکنترلر در مقایسه با میکروکامپیوتر

۱- هزینه خرید یک میکروکنترلر در مقایسه با هزینه خرید یک میکروکامپیوتر با اجزای مشابه، ارزان تر است
۲- کاهش تعداد اجزای سیستم که به ساده‌تر شدن عملیات تولید و مونتاژ منجر می‌شود
۳- کاهش تعداد اجزای سیستم قابلیت اطمینان را افزایش می‌دهد

انواع میکروکنترلرها

شرکت Texas Instruments در سال 1794 اولین میکروکنترلر تجاری را با نام TMS1000 به بازار عرضه کرد و پس از آن سایر شرکت‌های بزرگ الکترونیک به عرضه میکروکنترلرها پرداختند از مهمترین خانواده های شناخته شده میکروکنترلر میتوانیم به 8048 و 8051 محصول Intel ، 6811 محصول Motorala ، Z8 محصول Zilog، PICمحصول Microchip ، H8 محصول Hitachi و AVR محصول Atmel اشاره کنیم. اگرچه در اساس هسته اصلی همه خانوادهای میکروکنترلر یکسان است؛ ویژگی‌های ظاهری، امکانات جانبی، سرعت کاری و بسته‌بندی آنها با هم متفاوتند. میکروکنترلر خانواده 8051 محصول Intel ، سالیانی دراز به دلیل بهره‌منهدی از پورت های ورودی و خروجی فراوان و سرعت نسبتاً مناسب، از جایگاه ویژه‌ای در بین میکروکنترلرها برخوردار بود. میکروکنترلر 8051 دارای نمونه‌های سازگار متفاوتی است که از آن میان می‌توان به 89C51 محصول Atmel و(DS5000 با قابلیت نگهداری زمان و دارای حافظه) NVRAM محصول Dallas اشاره کرد. تراشه های 89C51 و 89S51 به دلیل فراوانی و قیمت پایین به شدت مورد توجه قرار گرفتند در این میان، تراشه 89S51 به دلیل برنامریزی آسان تر، توجه برنامنویسان را بیشتر جلب کرد.

امروزه میکروکنترلرهای خانواده AVR و PIC به سرعت در حال تکامل هستند وچشم انداز روشنی از آینده میکروکنترلرها را ترسیم میکنند. این گروه از میکروکنترلرها قابلیت و توانایی‌های بینظیری را در خود جای داده‌اند و محبوبیت و کارایی آن‌ها روزبه‌روز در حال افزایش است.

در این مجموعه تمرکز ما بر روی میکروکنترلرهای خانواده AVR و به طور خاص تراشهه ATmega64 می‌باشد.

آشنایی با خانواده‌های مختلف میکروکنترلرAVR

شرکت ATMEL با ایجاد تحول در معماری ریزپردازنده‌ها جهت کاهش کد، باعث پیدایش نسل جدیدی از پردازشگرها به نام AVR شد که علاوه بر کاهش و بهینه‌سازی مقدار کدها، به طور واقع هر عملیات را تنها در یک

سیکل پالس ساعت توسط معماری RISC انجام میدهد و از 32 رجیستر همه منظوره استفاده می‌کند این مقوله باعث شده است که AVR ها 4 تا 12 بار سریعتر از پردازشگرهای معمولی باشتد.
تکنولوژی حافظه کم مصرف و غیرفرار شرکت ATMEL برای برنامه ریزی AVR مورد استفاده قرارگرفته است. درنتیجه حافظه‌های EEPROM و FLASH در داخل مدار، قابل برنامه ریزی (ISP) هستند.
AVR به عنوان پردازشگرهای RISC با دستوراتی طراحی شده‌اند که باعث می شود حجم کد تولید شده کم و سرعت بالاتری بدست آید.
در همه محصولات AVR ، مجموعه دستورالعمل‌ها و معماری یکسان هستند؛ بنابراین زمانیکه حجم کدهای دستورالعمل شما که قرار است بر روی پردازشگر بارگذاری شود به دلایلی افزای یابد، یعنی بیشتر از ظرفیت پردازشگری که شما در نظر گرفته‌اید شود، می‌توانید از همان کدها استفاده کرده و بدون هیچ تغییری آن کدها را در یک پردازشگری با ظرفیت بالاتر بارگذاری کنید.
در AVR ها پالس ساعت اسیلاتور با پالس ساعت داخلی سیستم یکسان هستند و هیچ تقسیم‌کننده‌ای وجود دارد که اختلاف فازی برای پالس ساعت ایجاد کند در داخل بعضی پردازشگرها پالس ساعت ورودی از اسیلاتور به سیستم را با نسبت 1:4 یا 1:12 تقسیم می کنند که خود باعث کاهش سرعت می شود بنابراین AVR ها، 4 تا12 بار سریعتر و مصرف آنها نیز نسبت به ریزپردازنده‌های کنونی کمتر است، چرا که در پردازشگرهای AVR از تکنولوژی CMOSاستفاده شده است.
توان مصرفی پائین این ریزپردازنده‌ها برای استفاده بهینه از باطری طراحی شده است که موجب می گردد در ابزار و وسایل سیار معمول و رایج باشند . این پردازشگرها از شش روش گوناگون برای کاهش توان مصرفی، جهت انجام عملیات بهره میبرند و تغذیه آنها تا ۱٫۸ ولت پائین میآید که باعث طولانی‌تر شدن عمر باطری می‌شود.
در جدول زیر، پسوندهای مربوط به ولتاژ کاری میکروکنترلر AVR و مفاهیم آن آمده است:

پسوند مربوط به ولتاژ کاری	محدوده ولتاژ تغذیه	فرکانس قابل قبول کریستال
میکروکنترلر AVR بدون پسوند	4.5 تا 5 ولت	0 تا 16 مگاهرتز
میکروکنترلر AVR با پسوند L	2.7 تا 5.5 ولت	0 تا 8 مگاهرتز
میکروکنترلر AVR با پسوند V	1.8 تا 5.5 ولت	0 تا 4 مگاهرتز

بررسی ویژگیهای میکروکنترلر ATmega64

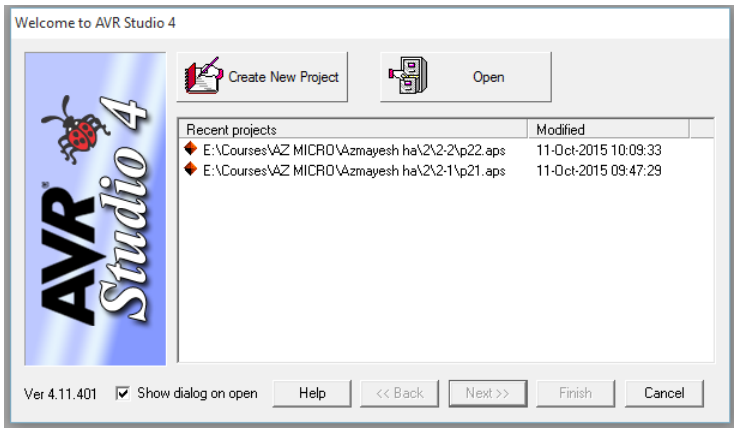
- اسیلاتور RC داخلی کالیبره شده
- منابع وقفه داخلی و خارجی
- ۳۲ رجیستر همه منظوره
- ۶ مد خواب(sleep)
- دارا بودن ۱۳۰ دستور قوی(اجرای اکثر آنها در یک سیکل)
- دارای ۶۴ کیلوبایت حافظه فلش قابل برنامه ریزی
- مجهز بهBoot Loader
- دارای ۲ کیلوبایت حافظه EEPROM داخلی
- دارای ۴ کیلو بایت حافظه SRAM داخلی

- قفل نرم افزاری
- ارتباطJTAG
- دارای دو تایمر / کانتر ۸ بیتی با مقسم فرکانسی مجزا و مد مقایسه
- دو تایمر / کانتز ۱۶ بیتی با مقسم فرکانسی مجزا، مد مقایسه و مدCapture
- دارای RTC با اسیلاتور مجزا
- شش کانال PWM با رزولوشن قابل برنامه‌ریزی
- هشت کانال ADC ، دارای ۱۰ بیت
- ارتباط سریال دو سیمه
- دارای دو ارتباط سریالUSART
- دارایSPI
- دارای Watchdog با اسیلاتور مجزا
- دارای ۵۳ خط ورودی / خروجی قابل برنامه ریزی
- دارای ۴۰ پایه در نوعTQFT
- ولتاژ عملیاتی:
  - ۲,۷ تا ۵,۵ ولت برای ATmega64L
  - ۴,۵ تا ۵,۵ ولت برای ATmega64
- فرکانس کاری:
  - ۰ تا ۸ مگاهرتز برایATmega64L
  - ۰ تا ۱۶ مگاهرتز برایATmega64

فصل ۲– برنامه نویسی اسمبلی برای AVR

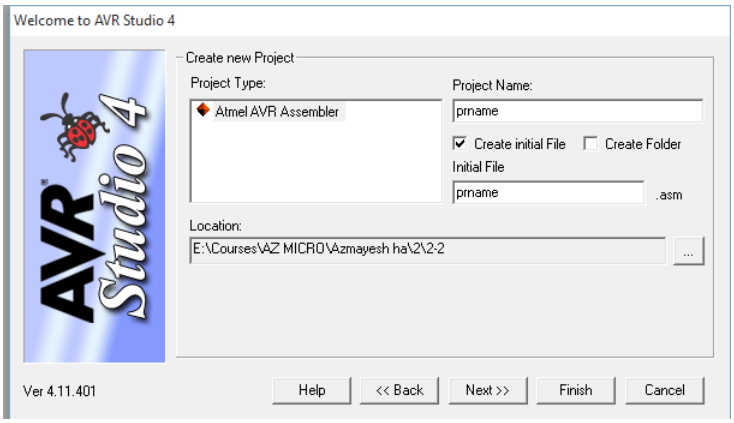
۲–۱–شروع کار با نرم‌افزار AVR Studio

برای شروع کار با این نرمافزار در پنجره باز شده در صفحه آغازین که در شکل ۲-۱ آمده است. بر روی Creat New Project کلیک می‌کنیم.



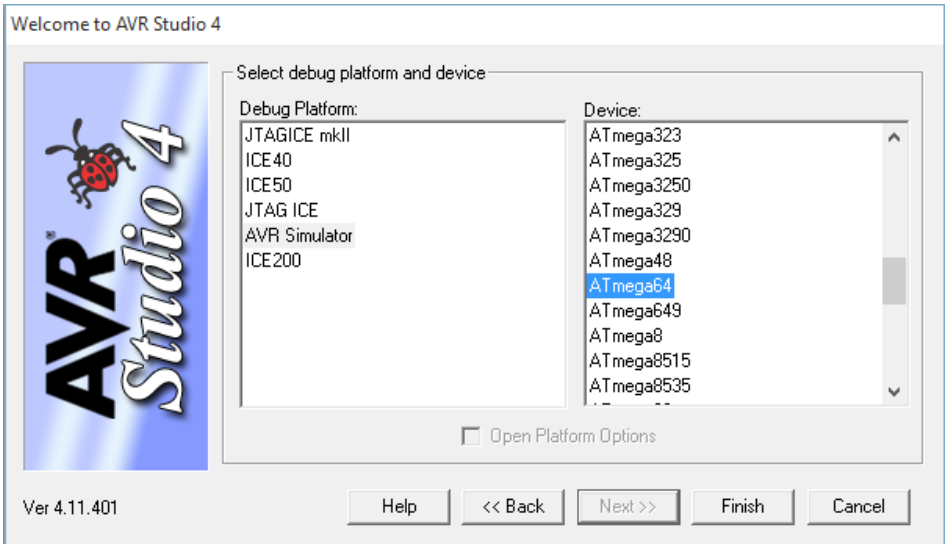
شکل ۲-۱: پنجره صفحه شروع AVR Studio

سپس پنجره انتخاب نام پروژه که در شکل ۲-۲ آمده است نام مناسب و محل ذخیره سازی را مشخص می‌کنید.



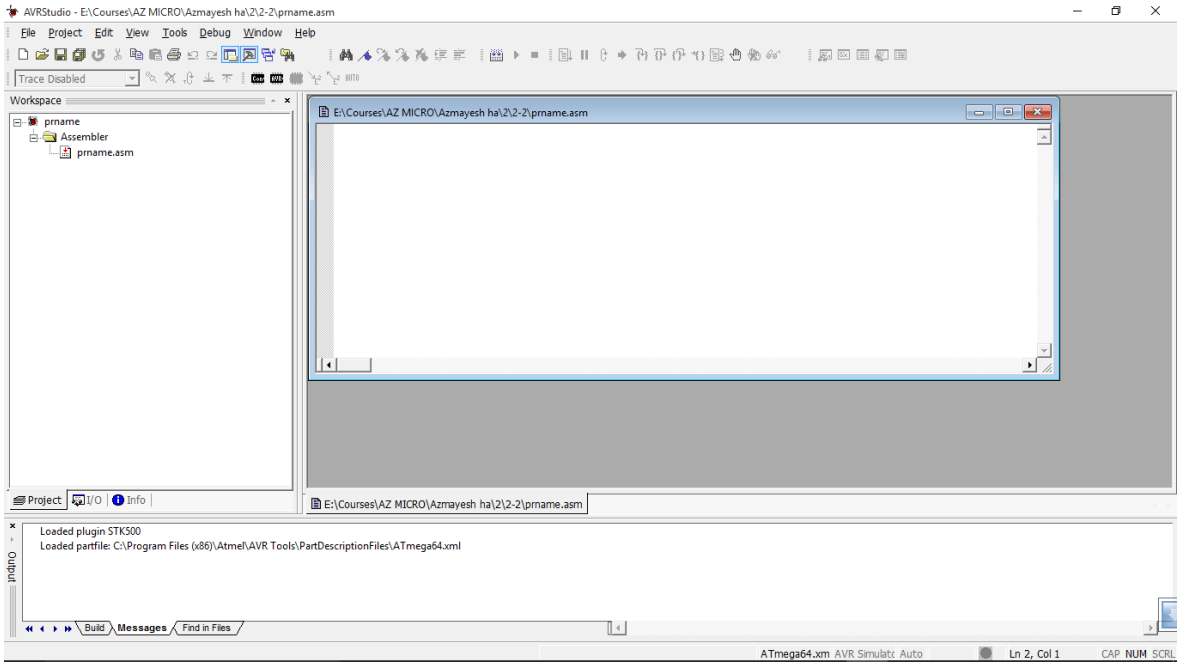
شکل ۲-۲: پنجره انتخاب نام پروژه

در قسمت بعد باید Debuger را معرفی کرد. پنل سمت چپ Debug platform گزینه AVR Simulator و در پنل سمت راست میکرو ATmega64 را انتخاب کنید و سپس روی Finish کلیک کنید.



شکل ۲-۳: پنجره انتخاب Debuger

سپس پنجره editor برنامه مثل زیر باز می‌شود که امکان نوشتن برنامه در آن وجود دارد.



شکل ۲-۴: پنجره editor پروژه برای نوشتن برنامه به زبان اسمبلی

پس از نوشتن برنامه باید آن را Build کنید که با زدن کلید F7 و یا منوی project قسمت Build این کار صورت می گیرد و فایل hex. برنامه ساخته می شود. پس از اطمینان از وصل پروگرمر به کامپیوتر از منو Tools/Program AVR/Auto connect را می زنیم که در صورت متصل بودن صحیح پنجره زیر باز می‌شود. در سربرگ Main نوع Device را روی ATmega64a قرار می دهیم و در سربرگ Poragram در قسمت مشخص شده آدرس فایل Hex. را وارد می کنیم. حال با کلیک کردن رو Program حافظه Flash میکرو با برنامه مورد نظر پر می‌شود و میکرو شروع به کار می کند.

۲–۲– موارد لازم برای یک برنامه اسمبلی در AVR

برنامه زیر حداقل کد لازم برای یک برنامه اسمبلی را نشان می دهد. البته در این برنامه قرار است فقط همه پین های درگاه C در سطح یک منطقی قرار گیرند.

```
.include "m64def.inc"     // library contain registers and addresses
      .def temp=r16
     .org 0x00
      rjmp Main
     .org 0x60            // after all registers of micro
;----------------------Main Function---------------------;
 Main:
      ldi temp,low(ramend)
      out spl,temp
      ldi temp, high(ramend)
      out sph,temp      //Stack Pointer ini
      ldi temp,0xff
      out ddrc,temp     //Portc → Output
      out portc,temp    //Portc =11111111
finish:                 //infinite loop
      rjmp finish
```

۲–۳– آزمایش اول: روشن کردن LED

هدف از انجام این آزمایش آشنایی با برنامه نویسی اسمبلی وآشنایی با پورت‌های میکرو و چگونگی نحوه ساختن تاخیر و آشنایی با دستور Shift و Rotate است. ۸ LED وجود دارد که پس از اعمال ۸ بار شیفت باید دوباره اولین LED روشن شود. توجه داشته باشید که ثبات‌هایی که برای ارسال داده به درگاه در میکرو قرار دارند PORTX و برای دریافت از درگاه PINX نام دارند که X می تواند نام هر یک از درگاه‌ها باشد. ورودی و یا خروجی بودن هر پین درگاه نیز با ثباتDDRX همان درگاه قابل تنظیم است نوشتن ۱ در این ثبات به معنای خروجی شدن پین متناظر آن در درگاه است.

۲–۳–۱– پیش‌گزارش

۱) تفاوت ATMEGA64 و ATMEGA64A چیست؟

۲) ATMEGA64 چند پورت دارد و این پورت ها چه وضعیت هایی را می توانند داشته باشند؟

۳) با مراجعه به برگه اطلاعاتی(DataSheet) میکرو تحقیق کنید حداکثر جریانی که یک پورت می‌تواند بدهد در حالت کشیدن(Sink) و اعمال کردن(Source) چقدر است؟

۲–۳–۲– مراحل آزمایش

۱) برنامه ای بنویسید که همه پین‌های درگاه(Port) C را روشن کند.

۲) سپس بورد را روشن کنید و میکرو را با استفاده از نرم‌افزار AVR Studio برنامه‌ریزی کنید.

۳) بورد را خاموش کنید و درگاه C را به قسمت LED ها متصل کنید. این درگاه بین پایه‌های ۳۵ تا ۴۲ قرار دارد با استفاده از سیم‌هایی با رنگ مناسب آنها را به LED های بورد اصلی متصل کنید.

۴) برنامه ای بنویسید که یکی از ۸ LED را روشن کرده و با فاصله زمانی ۱ ثانیه شیفت دهد و پس از پایان شیفت این روند ادامه پیدا کند.

۲-۴- آزمایش دوم: Timer/Counter

یکی از مهمترین واحدهایی که هر پردازنده برای اندازه‌گیری واحدهای زمانی خود در اختیار دارد Timer/Counter است. با شمارنده (Counter) قبلا آشنا هستیم هر شمارنده با اعمال یک لبه یا سطح پالس ساعت بسته به حالتی که طراحی می‌شد تغییر وضعیت می‌داد در ساده‌ترین حالت یک کانتر ۸ بیتی می توانست^{۲۸} وضعیت داشته باشد و پس از این مقدار پالس ساعت دوباره حالت ها تکرار می شدند. تایمر نیز در ساده‌ترین حالت با overflow شدن شمارنده ساخته می‌شود طوری که با تنظیم دوره تناوب پالس ساعت می‌توان مدت زمان خاصی را اندازه گرفت. به طور مثال در شمارنده بالا بافرض دوره تناوب 1us برای پالس ساعت تایمر هر ۲۵۶ میکرو ثانیه یکبار می تواند وقفه ایجاد کند. البته در این میکرو مدهای مختلف دیگری نیز برای تایمر وجود دارد. مثلا تایمر می‌تواند به جای سرریز شدن در مقدار نهایی در مقداری خاص که توسط ثبات OCRX مشخص شده است وقفه ایجاد کند به این ترتیب آزادی عمل و دقت بیشتری در ایجاد زمان‌ها برای طراح به وجود می آید. در مد دیگر تایمر می تواند مولد سیگنال PWM باشد این سیگنال که داده را در میزان روشن و یا خاموش بودن خود نگه داشته است برای بسیاری کاربرد ها نظیر کنترل دور موتورهای DC و منابع تغذیه سوییچینگ و یا اعمال به بازرها دارند. یکی دیگر از مدهای کاری تایمر مد تسخیر (Capture) است به این ترتیب که تایمر در مد نرمال خود کار می کند و اگر به یکی از ورودی های ICPn یک لبه که قابل تنظیم است اعمال شود مقدار تایمر در ثبات ICRX تایمر ذخیره می‌شود و در واقع رخداد ها برچسب زمانی میخورند و می توان فاصله‌ی آنها را اندازه گرفت. نوع دیگری از تایمر برای جلوگیری از Hang کردن میکرو تعبیه شده که به Watchdog معروف است. روال کار آن نیز به این ترتیب است که اگر پس از مدت زمان قابل تنظیمی اگر مقدار آن توسط برنامه‌نویس ریست نشود میکرو را ریست می کند و برنامه از آدرس صفر حافظه دوباره شروع می‌شود.

۲-۴-۱- پیش‌گزارش

۱) تفاوت PWM سریع و PWM تصحیح فاز چیست؟

۲) چگونه از Timer/Counter در مد شمارنده استفاده کنیم تا بتوانیم وقایع خارجی مثل عبور و مرور از داخل یک گیت را بشماریم و یا چرخش یک موتور را اندازه بگیریم؟

۲-۴-۲- مراحل آزمایش

۱) برنامه ای بنویسید که همه پین‌های درگاه C را با فاصله زمانی ۱ ثانیه روشن و خاموش کند. برای ایجاد این فاصله زمانی از تایمر صفر انتخاب کنید.

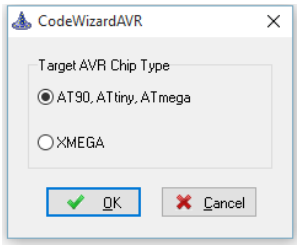
۲) پس از برنامه‌ریزی میکرو بورد را خاموش کنید و LED های برد را به پایه‌های ۳۵ تا ۴۲ متصل کنید. و پس از بازدید اتصالات بورد را روشن کنید.

۳) برنامه ای بنویسید که با استفاده از تایمر صفر در مد Fast PWM حجم صدای بازر را کنترل کند. برنامه طوری نوشته شود که پس از هر ۱ ثانیه عرض پالس‌ها از ۱۰ درصد کاهش یافته و این روال تکرار شود. برای این کار باید پایه ۱۴را به واحد بازر وصل کنید.

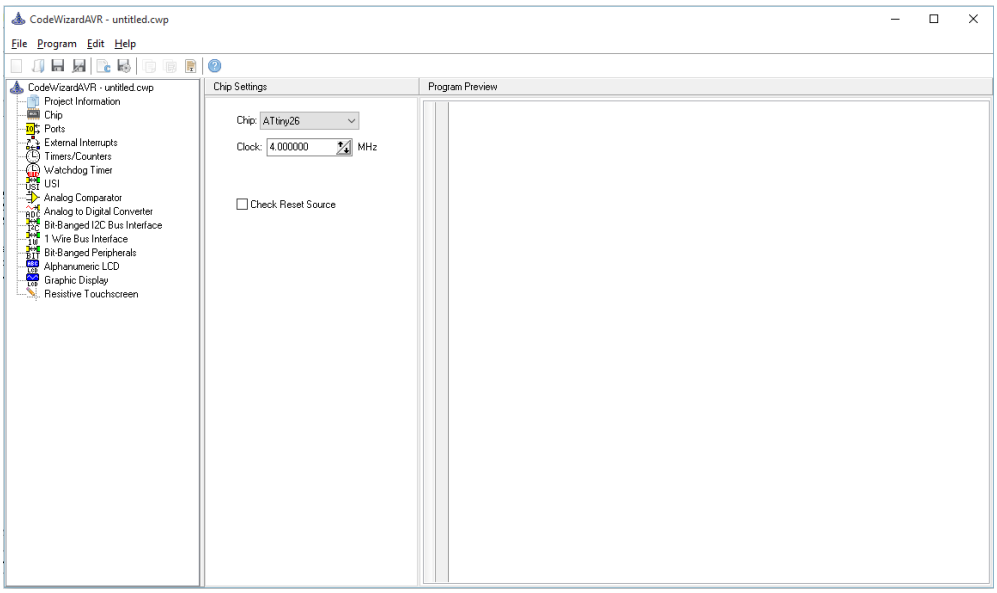
فصل ۳– برنامه‌نویسی C برای AVR

۳–۱– شروع کار با نرم‌افزار Codevision

امتیاز اصلی این نرم‌افزار داشتن ابزار Wizard آن می‌باشد طوری که دست برنامه‌نویس برای مقداردهی اولیه سخت‌افزارهای داخلی میکرو و تنظیم آنها باز می‌شود. این کار ضریب خطا را تا حد زیادی کاهش می‌دهد. برای درست کردن یک پروژه در ابتدا روی CodeWizard کلیک کنید. البته می توان پروژه را بدون استفاده از این ابزار هم ساخت ولی این روش ساده‌تر و دقیق‌تر است. پس از کلیک، پنجره شکل زیر باز می‌شود.



با انتخاب سری MEGA پنجره Wizard باز می‌شود. این پنجره پر از سربرگ‌های مختلف برای سخت‌افزارهای مختلف میکرو است که در اولین سربرگ امکان انتخاب chip و سرعت آن وجود دارد.



پس از اتمام تنظیمات لازم و ذخیره کردن هر سه فایل مورد نیاز پنجره editor باز شده و حاوی اطلاعاتی است که در قسمت قبل انتخاب شده بود.

۳–۲– موارد لازم برای برنامه C در AVR

برنامه زیر حداقل کد لازم برای راه‌اندازی یک میکرو را نشان می دهد. البته در این برنامه قرار است تمام پین های درگاه C در سطح یک منطقی قرار گیرند.

```
#include <mega64a.h>
#define Output PORTC
#define OutDir DDRC
//-------------------Main Function----------------//
void main(void)
{
  OutDir =  0xff;
  Output = 0xff;
  while(1)              // infinite loop
  {
  }
}
```

۳-۳- آزمایش سوم: نمایشگر هفت-قطعه‌ای

نمایشگرهای هفت-قطعه‌ای[۱] از هفت LED تشکیل شده‌اند که به ترتیب با a-g نام‌گذاری می‌شوند این عناصر در رنگ‌ها و اندازه‌های مختلف در بازار موجود هستند و در دو صورت کاتد مشترک و آند مشترک بسته بندی می‌شوند. گاهی تعدادی از این نمایشگر ها را با هم بسته بندی می‌کنند و برای هر عدد یک پایه کاتد یا اند بسته به نوع نمایشگر بیرون می‌دهند برای کار با چنین قطعاتی نمایشگر ها باید به صورت مداوم جاروب شوند سرعت جاروب باید بیشتر از پسماند چشم باشد تا چشم عملیات جاروب را احساس نکند و آن را پیوسته ببیند (همان تکنینکی که تلویزیون‌ها نیز از آن استفاده می‌کنند).

۳-۳-۱- پیش‌گزارش

۱) نقشه شماتیک نمایشگر هفت-قطعه‌ای را رسم کنید.(هر دو نوع)

۲) می‌خواهیم با چهار عدد نمایشگر هفت-قطعه‌ای یک ساعت دیجیتال بسازیم چگونه آنها را کنار هم بگذاریم که بین ساعت شمار و دقیقه شمار دو نقطه قرار گیرد؟توضیح دهید.

۳) حداقل تعداد دفعات جاروب کردن نمایشگرها برای اینکه مقدار مورد نظر پیوسته دیده شوند چقدر است؟

۳-۳-۲- مراحل آزمایش

۱) برنامه‌ی یک شمارنده معکوس از ۹ به ۰ را بنویسید طوری که فاصله زمانی هر شمارش برابر یک ثانیه باشد برای اینکار از تایمر ۱ در مد CTC استفاده کنید. همچنین از درگاه C برای راه‌اندازی نمایشگر هفت-قطعه‌ای استفاده کنید برای اینکار باید پایه های ۳۵ تا ۴۱ میکرو را به ترتیب به پایه های A-G نمایشگر وصل کنید. فقط یکی از نمایشگر ها را فعال کنید این کار با وصل پایه مربوطه فعالساز به VCC امکان‌پذیر است. فرکانس کار میکرو را روی ۸ مگاهرتز تنظیم نمایید. حتما برای نمایش اعداد نمایشگر در متن برنامه تابعی بنویسید.

[۱] 7 segment

۲) برنامه‌ی قسمت اول را طوری اصلاح کنید که همه نمایشگرها روشن شوند اما شمارشی صورت نگیرد و هر یک عددی مختلف را نمایش دهند. برای اینکار باید کنترل پایه های فعالساز را نیز به میکرو سپرد و با سرعتی معلوم به ترتیب نمایشگر ها را جاروب کرد.

۳) برنامه قسمت اول و دوم را ترکیب کنید طوری که عدد ۱۱۱۹ نمایش داده شود سپس مثل بند اول با استفاده از تایمر ۱ یک شمارنده معکوس با فاصله زمانی ۱ ثانیه بسازید طوری که با رسیدن به ۱۱۱۰ دوباره شمارش را از ۱۱۱۹ شروع کند.

۳-۴- آزمایش چهارم: صفحه‌کلید ماتریسی

در آزمایش‌های قبل با میکروسوئیچ کار کردیم اما با زیاد شدن کلید ها در ورودی پین های زیادی از میکرو درگیر می‌شود برای اجتناب از این وضعیت کلید ها را به حالت ماتریسی باید بست به این ترتیب با استفاده از یک پورت می توان به جای ۸ کلید ۱۶ کلید را خواند به همین ترتیب می توان تعداد کلید ها را با افزایش تعداد پورت ها افزایش داد خواندن کلید به این ترتیب انجام میشود که سطر به سطر کلید ها را فعال می کنیم(یا صفر یا یک منطقی) و سپس ستون به ستون آنها را می خوانیم یا برعکس. مثلا اگر سطر یک را فعال کردیم ستون یک بطور مثال مربوط به شماره ۷ است. علاوه بر این می خواهیم لرزش‌های ناخواسته (bunce) کلید را نیز با استفاده از برنامه‌نویسی حذف کنیم. این لرزش‌ها بسته به جنس کلید ها کم و زیاد می شوند اگر آنها را حذف نکنیم با یک بار زدن کلید میکرو آن را چند بار تعبیر می کند.

۳-۴-۱- پیش‌گزارش

۱) میزان زمانی که باید منتظر لرزش کلید بمانیم حداکثر چقدر است؟

۲) حداکثر زمانی که صفحه کلید باید پویش شود چقدر است؟

۳) اگر بخواهیم فقط زمانی که کلیدی فشرده شود تابع پویش صفحه کلید اجرا شود باید چگونه کد بنویسیم و احیانا مدار را چه تغییری دهیم؟ توضیح دهید.

۴) آیا ایده ای دارید که به جای استفاده از ۸ پین میکرو برای ۱۶ کلید فقط ۲ پین میکرو اشغال شود؟ به ازای این صرفه جویی چه بهایی پرداخته اید؟

۳-۴-۲- مراحل آزمایش

۱) صفحه کلید را به درگاه C که در پایه‌های ۳۵ تا ۴۲ قرار دارد وصل کنید طوری که ستون‌ها به نیبل بالایی و سطرها به نیبل پایینی متصل شوند. نمایشگر هفت-قطعه‌ای را به درگاه A وصل کنید برای این کار پایه‌های ۵۱ تا ۴۵ را به پایه‌های a-g متصل کنید.

۲) برنامه ای بنویسید که در صورت زدن کلید سطر اول ستون اول PORTD.0 را روشن و در صورت زدن کلید سطر اول ستون دوم همان PORTD.0 را خاموش کند.

۳) برنامه ای بنویسید که ۱۰ رقم را بر روی نمایشگر هفت-قطعه‌ای نمایش دهد بقیه کلیدها را نیز به شکل دلخواه a-f را نمایش دهد.

۴) به برنامه بند قبل امکان لرزش‌گیری (Debunce) کلید را نیز اضافه کنید.

۳–۵– آزمایش پنجم: راه‌اندازی LCD الفبایی (Alphametric)

نمایشگرهای LCD مدت‌هاست به عنوان یک نمایشگر کم حجم با توان مصرفی پایین به بازار ارائه شده‌اند. واز نظر شکل توان مصرفی و زاویه موثر دید طبقه بندی می‌شوند. به طور کلی در نوع تک‌رنگ آنها به دو نوعه کارکتری و گرافیکی تقسیم می‌شوند. در این آزمایش می خواهیم یک نمایشگر ۲۰x۲ کاراکتری را راه‌اندازی کنیم. LCD برخلاف عناصر استفاده شده در آزمایش های قبل یک عنصر فعال است و در داخل آن یک میکرو و مقدار ی حافظه قرار دارد برای ارتباط با میکرو داخل LCD باید به برگه اطلاعاتی آن مراجعه کرد که قسمتی از آن در ضمیمه ب آمده است. همچنین باید بدانیم که بعد از روشن شدن، LCD باید مقداردهی اولیه شود مثلا دستورهایی وجود دارد که موجب پاک کردن LCD و انتقال مکان نما به اول سطر و یا وجود یا عدم وجود مکان نما را تعیین می کند و نوع دیگری از دستورها داده هستند و باید روی LCD نمایش داده شوند این کار به کمک پایه RS انجام می‌شود. در آخر برای اینکه میکرو داخلLCD دستور ها را قبول کند باید به پایه E یک لبه پایین رونده اعمال کرد.

۳–۵–۱– پیش‌گزارش

۱) با LCD کارکتری فقط می‌توان کاراکترهای انگلیسی را نمایش داد برای نمایش کاراکترهای فارسی چه ایده ای دارید؟

۲) زمان لازم برای اینکه یک کاراکتر بر روی صفحه نمایش نوشته شود چقدر است؟

۳) اگر بخواهیم یک سیستم کم مصرف قابل حمل تولید کنیم چه نوع نمایشگری را پیشنهاد می‌دهید. با فرض اینکه از ATMega64 با فرکانس ۴ مگاهرتز استفاده کرده‌ایم. راجع به انواع نمایشگرها و مصرف توان آنها تحقیق کنید.

۳–۵–۲– مراحل آزمایش

۱) پایه های درگاه C را به LCD برای این کار بایستی پایه های ۳۵ تا ۴۲ را به پایه های مربوطه نظیر خود در LCD وصل کنید در صورت استفاده از Code Wizard اتصال هر پین به پایه های متناظر در توضیحات کد اضافه می شوند.

۲) برنامه ای بنویسید که نام خودتان را روی LCD نمایش دهد.

۳) برنامه ای بنویسید که نام خودتان و هم گروهیتان را در دو خط جداگانه نمایش دهد.

۴) برنامه ای بنویسید که هر کلیدی که بر روی صفحه کلید ماتریسی فشرده می‌شود یک کاراکتر دلخواه بر روی LCD نمایش داده شود.

۳–۶– آزمایش ششم: راه‌اندازی ADC

مبدل های آنالوگ به دیجیتال و دیجیتال به آنالوگ از جمله مهمترین ابزارهای در دست طراحان سیستم هستند که امکان ارتباط با دنیا واقعی را برای میکرو فراهم می کند اندازه‌گیری هر مقدار آنالوگ مثل ولتاژ و جریان که مربوط به سنسورها هستند بر عهده‌ی مبدل آنالوگ به دیجیتال (ADC) است. کاری که مبدل دیجیتال به آنالوگ برعکس آن را انجام می‌دهد با داشتن چنین ابزارهایی می توانیم یک سیستم کنترلی را طراحی کنیم. به فرض مثال می توانیم یک سنسور دما را با ADC بخوانیم و با با توجه به نیاز کاربر و دستور به Heater یا Cooler دمای محیط را اندازه بگیریم حتی با تنظیم ضرائب کنترلی در میکرو می توانیم نسبت به فضای تحت کنترل بهترین دما را ارائه دهیم و بقیه کاربرد هایی از این دست. هر ADC برای کار کردن نیاز به یک ولتاژ مرجع دارد که بتواند مقدار آنالوگ را با درصدی نسبت به آن مقدار دهی کند، همچنین از جمله پارامترهای یک ADC می توان نرخ نمونه برداری و دقت را نام برد از نظر معماری ساخت ADC نیز با توجه به پهنای باند و توان مصرفی و حداکثر دقت قابل تحویل طبقه‌بندی می‌شوند. در ATMEGA64 یک ADC با دقت (Resulotion) 10bit وجود دارد به این معنا که این مبدل قادر است حد فاصل صفر ولت تا ولتاژ مرجع خود را به $۲^{۱۰}$ یا ۱۰۲۴ قسمت تقسیم کند و تغییرات ولتاژ کمتر از این گام را حس نمی‌کند که با فرض ولتاژ مرجع ۵ ولت این حساسیت کمی کمتر از ۵ میلی ولت می شود. همچنین نرخ نمونه برداری این مبدل کمتر از [Msample/sec]1 می باشد بسته به مد کاری که مبدل در آن قرار دارد و در برگه اطلاعاتی میکرو آمده است تعداد ۱۲ تا ۲۵ پالس با معکوس همین فرکانس نمونه برداری زمان لازم است تا مبدل اطلاعات آنالوگ را به دیجیتال تبدیل کند که با فرض حداکثر سرعت می توان حدود ۲۰ تا ۳۰ میکرو ثانیه را تخمین زد البته فرکانس نمونه‌برداری را می توان در مبدل تعیین کرد. ولتاژ مرجع نیز از پایه های AVCC و AREF و ولتاژ ۲/۵۴ ولت داخلی قابل تعیین است.

۳-۶-۱- پیش‌گزارش

۱) بازار ایران را برای یک یا چند ADC که دقت ۸ بیت با نرخ نمونه‌برداری چند نانو ثانیه و یک ADC که دقت ۱۲ بیت با نرخ نمونه برداری چند میکرو ثانیه داشته باشد بگردید و نام قطعه و کارخانه سازنده آن و محل فروش آن را در پیش‌گزارش خود بیاورید.

۲) تحقیق کنید چگونه میکروکنترلر فقط با یک ADC می تواند ۸ ورودی را بخواند و به ازای این مزیت چه بهایی پرداخته است.

۳) برای کاهش نویز در ADC تکنیک هایی وجود دارد که در برگه اطلاعاتی تعدادی از آنها آمده است آنها را نام برده توضیح دهید.

۳-۶-۲- مراحل آزمایش

۱) ولتاژ مرجع ADC را به پین AREF میکرو اعمال کنید برای این کار پایه شماره ۶۲ میکرو را به ولتاژ ۵ ولت وصل کنید.

۲) سر وسط پتانسومتر موجود در برد اصلی را به پین ۶۱ که کانال صفر مبدل است وصل کنید. همچنین پین شماره ۲۵ را به یک LED وصل کنید

۳) برنامه‌ای بنویسید که مقدار آنالوگ را از کانال صفر بخواند اگر مقدار آن بیش از ۵۱۲ یا نصف ولتاژ مرجع بود PORTD.0 را یک کرده و در غیر این صورت همان پین را صفر کند.

۴) برنامه‌ی بند قبل را طوری اصلاح کنید که مقدارADC را به ده قسمت تقسیم کند و روی نمایشگر هفت-قطعه ای نمایش دهد.نمایشگر را به درگاه C متصل کنید.(پایه های ۳۵ تا ۴۱ به ترتیب به پایه‌های a-g)

۵) برنامه بند ۴ را طوری اصلاح کنید که مقدار به دست آمده را روی LCD کاراکتری نمایش دهد.(مقداری بین ۰تا ۱۰۰ درصد)

۳–۷– آزمایش هفتم: راه‌اندازی UART/USART

استانداردهای ارتباطی زیادی وجود دارد همه آنها برای افزایش امنیت ارتباط و به طور کلی امکان ایجاد ارتباط تبیین می‌شود. یکی از ساده ترین و قدیمی ترین این استاندارد ها USART است. استاندارد UART عملا استانداردی است که در این آزمایش آن را خواهیم بست. روال کار به این ترتیب است که هیچ سیمی به عنوان کلاک برای همزمان سازی بین ورودی و خروجی وجود ندارد باری همین حرف S از آن برداشته شده است USART استانداری است که کیبرد و ماوس های قدیمی که با کانکتور PS2 به کامپیوتر متصل می شدند از آنها استفاده می کردند. اما UART فقط از دوسیم تشکیل شده است این ارتباط دو طرفه است(Full Duplex) به این معنا که هر دو طرف به شکل همزمان می‌توانند برای یکدیگر داده بفرستند. چون سیم کلاکی وجود ندارد ناچار باید زمان‌بندی بین فرستنده و گیرنده از قبل هماهنگ شود. ارتباط با یک سطح پایین رونده به مدت زمانی که برابر عکس نرخ ارسال اطلاعات (Baud Rate) است شروع می‌شود و به آن بیت Start می گوییم بعد از آن ۸ بیت داده مورد نظر با همان فاصله‌های زمانی قرار دارند در صورت نیاز به تشخیص خطا در گیرنده از یک بیت Parity که می توان زوج یا فرد باشد، بعد از ۸ بیت داده استفاده کرد و در نهایت برای پایان از یک یا دو بیت stop استفاده می‌شود این زمانبندی و نرخ ارسال اطلاعات باید در فرستنده و گیرنده تنظیم شوند.

۳–۷–۱– پیش‌گزارش

۱) دیاگرام زمانی برای ارسال داده UART را رسم کنید.

۲) همه امکانات و مدهایی که واحد USART این میکرو می تواند در آنها کار کند را نام ببرید.

۳) اگر یک میکرو واحد USART نداشت (مثل ATtiny13) و بخواهیم با همین استاندارد با آن در ارتباط باشیم ایده‌ای به صورت نرم افزاری دارید؟توضیح دهید.

۳-۷-۲- مراحل آزمایش

۱) باید پایه‌های RXD و TXD را به پایه‌های TXD و RXD واسط USB در بورد جانبی (سفید رنگ) وصل کنید برای این کار پایه شماره ۲۸ میکرو را به پایه RXD و پایه ۲۷ را به پایه TXD واسط وصل کنید.

۲) برنامه‌ای بنویسید که نام خودتان را برای کامپیوتر ارسال کند و از طریق نرم‌افزار Docklight داده ها را مشاهده کنید.

۳) برنامه ای بنویسید که کد های ارسال شده توسط نرم‌افزار را روی LCD کارکتری نمایش دهد.

۳–۸– آزمایش هشتم: راه‌اندازی GLCD

همانطور که در آزمایش ۵ گفته شد چند نوع نمایشگر وجود دارد، که نمایشگر گرافیکی تک‌رنگ یکی از آن انواع است. این ماژول نیز مانند نمایشگر کاراکتری بسته به ابعادش دارای یک یا تعدادی پردازنده است. در این نمونه ابعاد نمایشگر ۱۲۸× ۶۴ پیکسل می باشد. هر یک از ۱۲۸ ستون به هشت صفحه[۱] تقسیم می‌شوند. که هر یک از آن صفحه‌ها به ۸ بیت هستند. به این ترتیب در هر ستون دسترسی بایتی می‌توان داشت. در موارد که صفحه نیاز به تازه‌سازی[۲] دارد نیاز است که به میزان لازم از فضای RAM میکرو به متغییرهای هر پیکسل اختصاص داده شود. همچنین نمایشگر دارای پایه‌های کنترلی و داده است که ترتیب آنها در ضمیمه ج آمده است. نیمی از صفحه را یک پردازنده و نیمی دیگر را پردازنده دیگر کنترل می‌کند. برای انتخاب نیم صفحه های ۶۴×۶۴ از پایه CS1 و CS2 استفاده می‌شود. مانند نمایشگر کاراکتری از یک پتانسومتر برای تنظیم کنتراست صفحه استفاده می‌شود. پایه های E و RS و R/W نیز وظایف مشابه نمایشگر کاراکتری دارند. در این نمایشگر از هر هشت پایه داده برای مقداردهی باید استفاده شود به این ترتیب برخلاف نمایشگر کاراکتری بیش از یک پورت برای راه‌اندازی آن نیاز است.

۳–۸–۱– پیش‌گزارش

۱) ساختار پایه های نمایشگر گرافیکی ۱۲۸×۶۴ و نمایشگر رنگی سلفون N95 را در دو جدول مجزا بنویسید.

۲) حداقل زمان لازم برای نوشته شدن یک پیکسل بر روی نمایشگر گرافیکی حدودا چقدر است؟

۳) اگر بخواهیم پروژه‌ای برای تایپ حروف فارسی بر روی نمایشگر تعریف کنیم. حداقل چند کاراکتر و در کدام حافظه میکرو باید بسازیم؟ چرا؟

---

[۱] page

[۲] Refresh

۳-۸-۲- مراحل آزمایش

۱) از روشن بودن کلید on/off روی بورد میکرو و روی بورد نمایشگر اطمینان حاصل کنید.

۲) ابتدا بورد سبز رنگ را کنار بورد اصلی قرار داده و کابل تخت [۱] تغذیه بورد را از قسمت بالایی آن بورد اصلی متصل کنید. همچنین پایه ۲۷تا ۵۲ هر دو بورد را به هم وصل کنید. این کار با متصل کردن کانکتورهای IDC امکان‌پذیر است.

۳) لیست پایه‌های نمایشگر در سمت چپ آن نوشته شده است. پورت های C را به پورت داده(D0 تا D7) و پورت A را به پورت کنترل نمایشگر اختصاص دهید. برای راحتی با همین ترتیبی که پایه‌های کنترلی در بورد وجود دارند از PA.0 شروع به انتساب کنید.

۴) با کمک نرمافزار GLCD Tools نام خود و هم‌گروهی خود را به کد تبدیل کرده و سپس بر روی نمایشگر نمایش دهید.

۵) برنامه ای بنویسید که با فشردن یک کلید در PA.7 یک واحد به عدد نمایشگر اضافه کند و در صورت رسیدن به ۹ شمارش از ۰ آغاز شود.

[۱] Flat cable

ضمیمه أ – مجموعه دستورات اسمبلی برای AVR

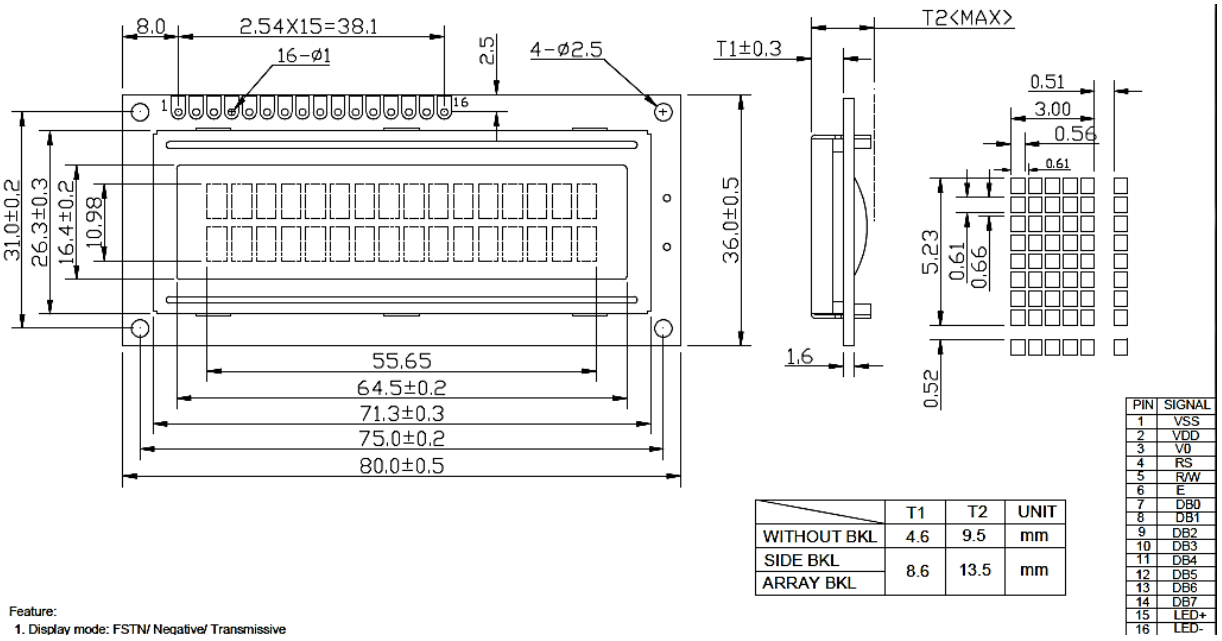
Mnemonics	Operands	Description	Operation	Flags	#Clocks
ARITHMETIC AND LOGIC INSTRUCTIONS					
ADD	Rd, Rr	Add two Registers	Rd ← Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry two Registers	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl,K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two Registers	Rd ← Rd - Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	Rd ← Rd - K	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry two Registers	Rd ← Rd - Rr - C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	Rd ← Rd - K - C	Z,C,N,V,H	1
SBIW	Rdl,K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	Rd ← Rd • Rr	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	Rd ← Rd • K	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	Rd ← Rd v K	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	Rd ← Rd ⊕ Rr	Z,N,V	1
COM	Rd	One's Complement	Rd ← 0xFF – Rd	Z,C,N,V	1
NEG	Rd	Two's Complement	Rd ← 0x00 – Rd	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd • (0xFF - K)	Z,N,V	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V	1
DEC	Rd	Decrement	Rd ← Rd – 1	Z,N,V	1
TST	Rd	Test for Zero or Minus	Rd ← Rd • Rd	Z,N,V	1
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	Z,N,V	1
SER	Rd	Set Register	Rd ← 0xFF	None	1
MUL	Rd, Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z,C	2
MULS	Rd, Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 ¯ (Rd x Rr) << 1	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 ¯ (Rd x Rr) << 1	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	R1:R0 ¯ (Rd x Rr) << 1	Z,C	2

BRANCH INSTRUCTIONS					
RJMP	k	Relative Jump	PC ← PC + k + 1	None	2
IJMP		Indirect Jump to (Z)	PC ← Z	None	2
JMP	k	Direct Jump	PC ← k	None	3
RCALL	k	Relative Subroutine Call	PC ← PC + k + 1	None	3
ICALL		Indirect Call to (Z)	PC ← Z	None	3
CALL	k	Direct Subroutine Call	PC ← k	None	4
RET		Subroutine Return	PC ← STACK	None	4
RETI		Interrupt Return	PC ← STACK	I	4
CPSE	Rd,Rr	Compare, Skip if Equal	if (Rd = Rr) PC ← PC + 2 or 3	None	1/2/3
CP	Rd,Rr	Compare	Rd – Rr	Z, N,V,C,H	1
CPC	Rd,Rr	Compare with Carry	Rd – Rr – C	Z, N,V,C,H	1
CPI	Rd,K	Compare Register with Immediate	Rd – K	Z, N,V,C,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3
SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC ← PC + 2 or 3	None	1/2/3
BRBS	s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC←PC+k + 1	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC←PC+k + 1	None	1/2
BREQ	k	Branch if Equal	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if Lower	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if Minus	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if Plus	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V= 0) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if (N ⊕ V= 1) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T Flag Set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1	None	1/2

BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1/2
DATA TRANSFER INSTRUCTIONS					
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, - X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, - Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd,Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2
ST	- X, Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	- Y, Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q,Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q,Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3
SPM		Store Program Memory	(Z) ← R1:R0	None	-
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2
POP	Rd	Pop Register from Stack	Rd ← STACK	None	2
BIT AND BIT-TEST INSTRUCTIONS					
SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
ROL	Rd	Rotate Left Through Carry	Rd(0)←C,Rd(n+1)← Rd(n),C←Rd(7)	Z,C,N,V	1
ROR	Rd	Rotate Right Through Carry	Rd(7)←C,Rd(n)← Rd(n+1),C←Rd(0)	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n=0:6	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	Rd(3:0)←Rd(7:4),Rd(7:4)←Rd(3:0)	None	1
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None	1
SEC		Set Carry	C ← 1	C	1
CLC		Clear Carry	C ← 0	C	1
SEN		Set Negative Flag	N ← 1	N	1
CLN		Clear Negative Flag	N ← 0	N	1
SEZ		Set Zero Flag	Z ← 1	Z	1
CLZ		Clear Zero Flag	Z ← 0	Z	1
SEI		Global Interrupt Enable	I ← 1	I	1
CLI		Global Interrupt Disable	I ← 0	I	1
SES		Set Signed Test Flag	S ← 1	S	1
CLS		Clear Signed Test Flag	S ← 0	S	1
SEV		Set Twos Complement Overflow.	V ← 1	V	1
CLV		Clear Twos Complement Overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set Half Carry Flag in SREG	H ← 1	H	1
CLH		Clear Half Carry Flag in SREG	H ← 0	H	1
MCU CONTROL INSTRUCTIONS					
NOP		No Operation		None	1
SLEEP		Sleep	(see specific descr. for Sleep function)	None	1
WDR		Watchdog Reset	(see specific descr. for WDR/timer)	None	1
BREAK		Break	For On-chip Debug Only	None	N/A

ضمیمه ب – راهنمای کار با LCD کاراکتری

ترتیب پایه های نمایشگر کاراکتری در شکل ۳-۱ آمده است.



	T1	T2	UNIT
WITHOUT BKL	4.6	9.5	mm
SIDE BKL	8.6	13.5	mm
ARRAY BKL			

PIN	SIGNAL
1	VSS
2	VDD
3	V0
4	RS
5	R/W
6	E
7	DB0
8	DB1
9	DB2
10	DB3
11	DB4
12	DB5
13	DB6
14	DB7
15	LED+
16	LED-

شکل ۳-۱: قسمتی از برگه اطلاعاتی LCD کاراکتری۲×۱۶

توضیح وظایف پایه های نمایشگر نیز در جدول ۳-۱ آمده است. البته در آزمایش‌های این دوره از نمایشگر فقط در مد نوشتن استفاده می‌شود. و همیشه می توان آن را به زمین وصل کرد.

جدول ۳-۱: توضیح وظایف پایه‌های نمایشگر کاراکتری

Pin no.	Symbol	External connection	Function
1	Vss	Power supply	Signal ground for LCM
2	V_{DD}		Power supply for logic for LCM
3	V_0		Contrast adjust
4	RS	MPU	Register select signal
5	R/W	MPU	Read/write select signal
6	E	MPU	Operation (data read/write) enable signal
7~10	DB0~DB3	MPU	Four low order bi-directional three-state data bus lines. Used for data transfer between the MPU and the LCM. These four are not used during 4-bit operation.
11~14	DB4~DB7	MPU	Four high order bi-directional three-state data bus lines. Used for data transfer between the MPU
15	LED+	LED BKL power supply	Power supply for BKL
16	LED-		Power supply for BKL

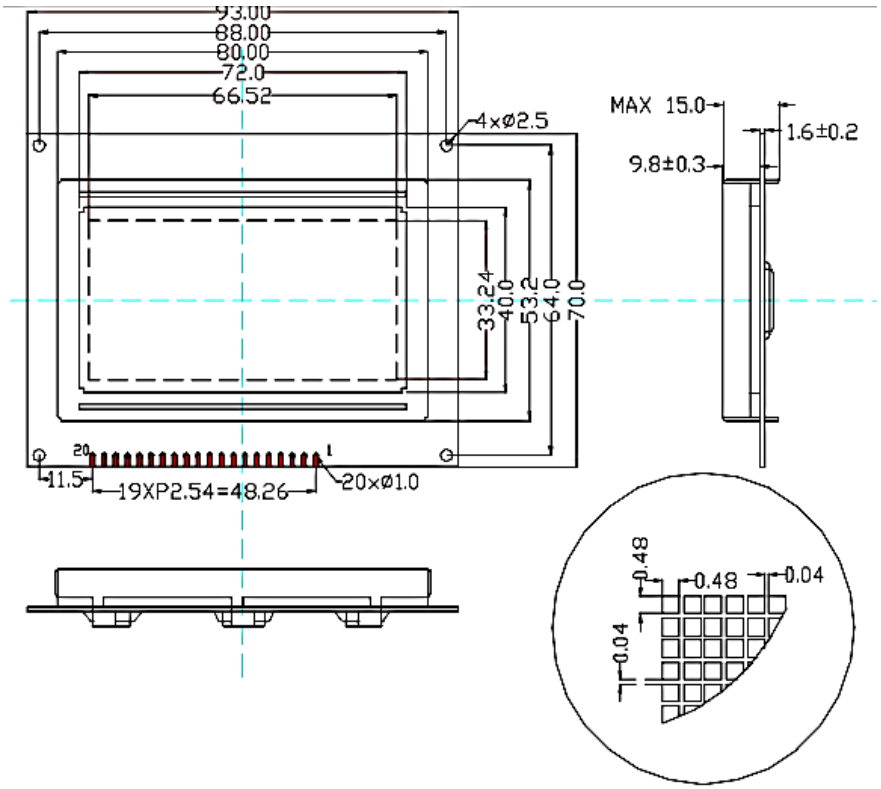
جدول ۳-۲ نیز دستورهای کنترلی نمایشگر کاراکتری را با وضعیت پایه های مربوطه نمایش می دهد.

جدول ۳-۲: دستورهای کنترلی نمایشگر کاراکتری

Instruction	Instruction code										Description	Execution time (fosc= 270 KHZ
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRA and set DDRAM address to "00H" from AC	1.53ms
Return Home	0	0	0	0	0	0	0	0	1	-	Set DDRAM address to "00H" From AC and return cursor to Its original position if shifted. The contents of DDRAM are not changed.	1.53ms
Entry mode Set	0	0	0	0	0	0	0	1	I/D	SH	Assign cursor moving direction And blinking of entire display	39us
Display ON/ OFF control	0	0	0	0	0	0	1	D	C	B	Set display (D), cursor (C), and Blinking of cursor (B) on/off Control bit.	
Cursor or Display shift	0	0	0	0	0	1	S/C	R/L	-	-	Set cursor moving and display Shift control bit, and the Direction, without changing of DDRAM data.	39us
Function set	0	0	0	0	1	DL	N	F	-	-	Set interface data length (DL: 8- Bit/4-bit), numbers of display Line (N: =2-line/1-line) and, Display font type (F: 5x11/5x8)	39us
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address Counter.	39us
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address Counter.	39us
Read busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal Operation or not can be known By reading BF. The contents of Address counter can also be read.	0us
Write data to Address	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	43us
Read data From RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	43us

ضمیمه ج – راهنمای کار با LCD گرافیکی

در شکل ۳-۲ ترکیب پایه‌های GLCD128x64 آمده است. پایه Vo به سر وسط یک پتانسیومتر وصل می‌شود که توسط آن می توان کنتراست نمایشگر را تنظیم کرد. پایه های دیگر پتانسیومتر به VDD و VEE که یک ولتاژ منفی ساخته شده در واحد GLCD است، متصل می شوند و یک ولتاژ منفی قابل تغییر به این ترتیب به پایه V0 می‌رسد.



PIN	1	2	3	4	5	6	7	8	9	10
SIGNAL	Vss	VDD	V0	D/I	R/W	E	DB0	DB1	DB2	DB3
PIN	11	12	13	14	15	16	17	18	19	20
SIGNAL	DB4	DB5	DB6	DB7	CS1	CS2	RES	VEE	A	K

شکل ۳-۲: ترکیب پایه‌های GLCD

در جدول ۳-۳ ترتیب مقداردهی به پایه های GLCD برای انجام عملیات متفاوت در آن را می بینید. البته در آزمایش ها از عملیات خواندن از GLCD استفاده نمیکنیم و به این ترتیب پایه R/W را می توانیم به صورت کلی به زمین متصل کنیم.

جدول ۳-۳: ترتیب مقداردهی به پایه های GLCD

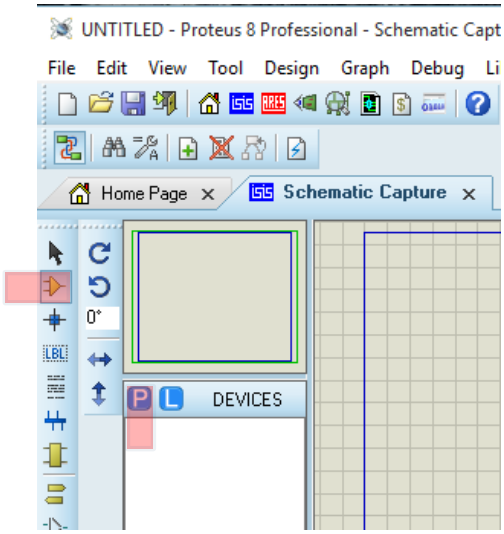
Instruction	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Function
Read Display Date	1	1	Read data								Reads data (DB[7:0]) from display data RAM to the data bus.
Write Display Date	1	0	Write data								Writes data (DB[7:0]) into the DDRAM. After writing instruction, Y address is incriminated by 1 automatically
Status Read	0	1	Busy	0	ON/ OFF	Re-set	0	0	0	0	Reads the internal status BUSY 0: Ready 1: In operation ON/OFF 0: Display ON 1: Display OFF RESET 0: Normal 1: Reset
Set Address (Y address)	0	0	0	1	Y address (0~63)						Sets the Y address at the column address counter
Set Display Start Line	0	0	1	1	Display start line (0~63)						Indicates the Display Data RAM displayed at the top of the screen.
Set Address (X address)	0	0	1	0	1	1	1	Page (0~7)			Sets the X address at the X address register.
Display On/off	0	0	0	0	1	1	1	1	1	0/1	Controls the display ON or OFF. The internal status and the DDRAM data is not affected. 0: OFF, 1: ON

ضمیمه د – شروع کار با نرم‌افزار Proteus

در شکل ۳-۳ پنجره آغازین نرم‌افزار Proteus آمده است با انتخاب ISIS وارد پنجره شماتیک نرم‌افزار می‌شویم.

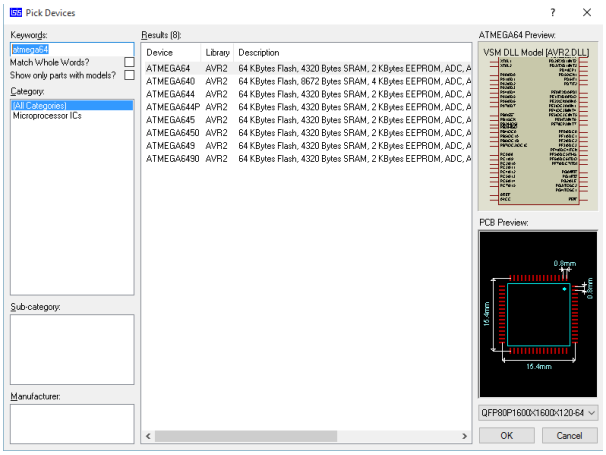
شکل ۳-۳: پنجره آغازین نرم‌افزار Proteus

با انتخاب کتابخانه همانطور که شکل ۳-۴ نمایش می‌دهد. می توان قطعات جدید را اضافه کرد.



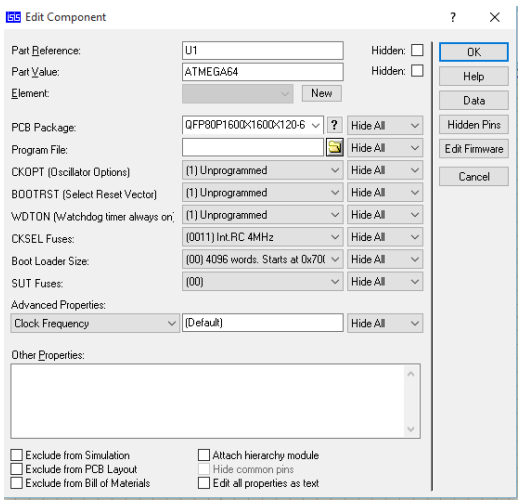
شکل ۳-۴: پنجره کتابخانه شماتیک

در پنجره باز شده همانطور که در شکل ۳-۵ آمده است. نام عنصر مورد نظر را تایپ کنید و سپس آن را انتخاب نمایید.



شکل ۳-۵: پنجره لیست قطعات کتابخانه

پس از انتخاب و اضافه کردن هر عنصر شروع به اتصال آنها کنید. توجه داشته باشید که قطعات مانند میکرو و نمایشگرهای گرافیکی و کاراکتری به ولتاژ تغذیه نیازی ندارند و کافی است پایه‌های داده و کنترلی لازم را به یک‌دیگر متصل کنید. پس از اتمام اتصال ها باید میکرو را برنامه ریزی کنید با دبل کلیک کردن روی میکرو پنجره شکل ۳-۶ باز می‌شود و می توان فایل HEX. ساخته شده را از قسمت Program File درون میکرو بارگذاری کرد. همچنین می توان سرعت عملکرد میکرو و مرجع کلاک آن را از قسمت Cksel Fuses نیز تنظیم نمود. پس از این مراحل کلید Run پایین سمت چپ نرم‌افزار را کلیک کنید. تا مدار شروع به کار کند.



شکل ۳-۶: پنجره تنظیمات میکرو