تمرینات دستور کار اول آزمایشگاه ریزپردازنده

تمرین ۱ : اجرای برنامه LED چشمک زن در محیط پروتئوس

هدف اجرای یک برنامه چمک زن در محیط پروتئوس روی یک برد شبیه سازی آردوئینو بوده است.

در ابتدا باید برنامه را نوشته و در IDE آردوئینو اجرا کرد تا فایل hex. ساخته شود.

سپس در پروتئوس یک Arduino Atmega را اضافه کرده و به پرت ۱۳ آن یک مقاومت و یک LED وصل میکنیم و سپس برنامه hex. را به برد اردوینو داده و اجرا میکنیم.

پس از اجرا دیده میشود که LED پس از نیم ثانیه روشن و خاموش میشود.

فایل hex. و همینطور پروژه پروتئوس در ضمیمه در فایل blink قرار دارد.

نتیجه اجرای برنامه در پروتئوس بصورت گیف در فایل ضمیمه blink وجود دارد.

تمرین ۲ : مدار reset در آردوئینو

هر cpu وقتی روشن میشود نیاز دارد که ریست شود و برنامه خانه صفر حافظه را اجرا کند و آردوئینو هم از این قاعده مستثنی نیست و برای همین به مداری نیاز دارد که اتوماتیک cpu را ریست کند که این مهم با استفاده از یک خازن و یک مقاومت قابل انجام است. ریست (Reset) در واقع به نوعی ، یک وقفه است. و می تواند توسط هر یک از منابع تحریک کننده آن رخ دهد.

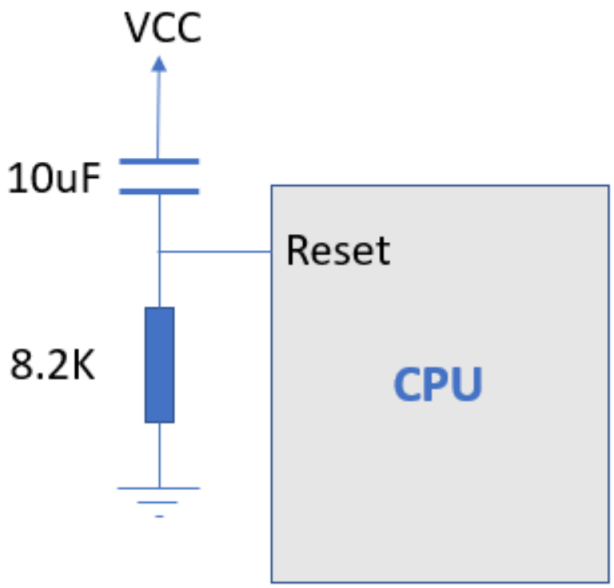
اگر میکروکنترلر ریست نشود، یک ریسک حساب میشود زیرا فرض کنید که میکروکنترلر با مقدارهای رندوم در رجیسترها شروع به کار کند، در اینصورت تمام کارها بعد از آن اشتباه میشود پس حتما باید از ریست مطمئن شویم.

برای ریست در میکروکنترلرها از روشهایی میتوان استفاده کرد بعضی از آنها را می توان برنامه ریزی کرد و بعضی دیگر را می توان با ایجاد شرایط خاص بیرونی در مدار الکترونیکی بوجود آورد.

یک مدار ریست اتوماتیک مدرن از یک تایمر استفاده می کند. این تایمر آنبورد تنها زمانی فعال می شود که منبع تغذیه به آستانه ولتاژ از پیش تعیین شده ای برسد که فراتر از آن سیگنال ریست فعال می شود و از راه اندازی صحیح سیستم اطمینان حاصل می کند.

با در نظر گرفتن انقضای تایمر پس از یک دوره زمانی خاص و در نتیجه غیرفعال شدن خروجی برق مدار ریست و منجر به فعال شدن دستگاه با خاموش شدن سیگنال ریست می شود. اکنون دستگاه می تواند شروع به کار کند. سیگنال بازنشانی برق از اجرای هر نرم افزاری توسط CPU جلوگیری می کند تا زمانی که حداقل آستانه ولتاژ برق برآورده شود و ساعت پایدار باشد.

در اجرای ریست، فاکتورهای آستانه زمان و ولتاژ مشخصه یک مدار هستند. دوره زمانی ریست با استفاده از شارژ یک خازن که به صورت سری با یک مقاومت قرار می گیرد، تعیین می شود. هنگامی که برق اعمال می شود، جریان از خازن عبور می کند و ولتاژ خازن به آرامی افزایش می یابد. در ابتدا، ولتاژ کمتر از ولتاژ آستانه پین ورودی تنظیم مجدد است و تمام عناصر موجود در CPU در حالت تنظیم مجدد قرار دارند. و سپس، ولتاژ بالاتر از آن ولتاژ آستانه است، پین تنظیم مجدد یک "۱" می گیرد و سیستم اولیه می شود. مقادیر مقاومت و خازن قدرت تأخیر تنظیم مجدد را تعیین می کند.



روشهای دیگر :

۱) مقدار ولتاژ آستانه میکروکنترلر

هنگامی که ولتاژ تغذیه متصل به پایه VCC میکرو کنترلر کمتر از مقدار ولتاژ آستانه شود. میکرو کنترلر resetمی شود.

۲) قرار دادن کلید ریست خروجی متصل به پایه ریست میکروکنترلر

اگر یک پالس با سطح صفر منطقی به مدت طولانی تر از سیکل پالس ساعت میکرو کنترلر ، به پایه خارجی Resetاعمال شود ، میکرو کنترلر ریست می شود .

۳) استفاده از تایمر نگهبان در میکرو کنترلر

اگر تایمر نگهبان (Watchdog Timer) فعال شود.(این تایمر با تنظیمات برنامه ای فعال می شود.) میکروکنترلر ریست می شود. این تایمر وظیفه عملکرد صحیح برنامه میکرو کنترلر را برعهده دارد.

تمرین ۳ : مقاومت pull-up

در حالتی که در یک مدار کلید وصل نباشد مدار مه به زمین و نه به منبع تغزیه وصل است در این حالت معلوم نیست که مدار چه مقداری دارد ۰ یا ۱ به حالت اصطلاحا float می‌گویند، در این حالت اگر از یک مقاومت pull-up یا pull-down استفاده کنیم، میتوانیم مقدار مدار را در حالت قطع بودن کلید مشخص کنیم.

در واقع مقاومت pull-up را بین منبع تغزیه و یک پین از مدار منطقی گذاشته میشود تا مقدار مدار وقتی کلید باز است،کنترل شود.

اندازه این مقاومت مهم است و باید در حدی بزرگ باشد که وقتی کلید وصل میشود به اندازه لازم ولتاژ از منبع تغذیه به سمت مدار و سپس زمین جریان پیدا کند، اما همزمان باید در حدی باشد که وقتی کلید قطع است جریان مناسبی را رد کند و نباید خیلی بزرگ باشد، چون از طرفی هر چقدر مقاومت pull-up بزرگ باشد پین دیرتر به ولتاژ پاسخ میدهد.

معمولا باید سعی شود که مقاومت pull-up از ۰,۱ مقاومت ورودی پایه میکروکنترلر بزرگتر نباشد.

مقدار مقاومت pull-up طبق قانون اهم، از تقسیم ولتاژ منبع تغذیه بر جریانی که می خواهیم از تغذیه کشیده شود (هنگام اتصال کلید) به دست می آید.

$R = Vcc / I$

تمرین ۴ : ذخیره رشته رندوم در EEPROM

هدف ذخیره یک رشته رندوم ۲۰۰ کاراکتری در EEPROM است.

در برنامه یک تابع برای نوشتن یک کاراکتر در EEPROM نوشته شده است و برای ساخت رشته رندوم ابتدا تمام کاراکترهای مجاز را در یک لیست ذخیره و سپس به تعداد که نیاز داریم با استفاده از تابع رندوم از لیست برداشته و به رشته اضافه میکنیم و در نهایت رشته تولید شده را با استفاده از تابع در EEPROM ذخیره میکنیم.

فایل برنامه در ضمیمه وجود دارد.

منابع :

https://www.theengineeringprojects.com/2017/01/simple-arduino-led-example-proteus.html

/https://emadars.ir/how-to-connect-the-reset-key-to-the-microcontroller

/https://hardwarebee.com/introduction-to-power-on-reset-circuit

https://ww1.microchip.com/downloads/en/Appnotes/doc4284.pdf

https://learn.sparkfun.com/tutorials/pull-up-resistors/all

/https://roboticsbackend.com/arduino-write-string-in-eeprom