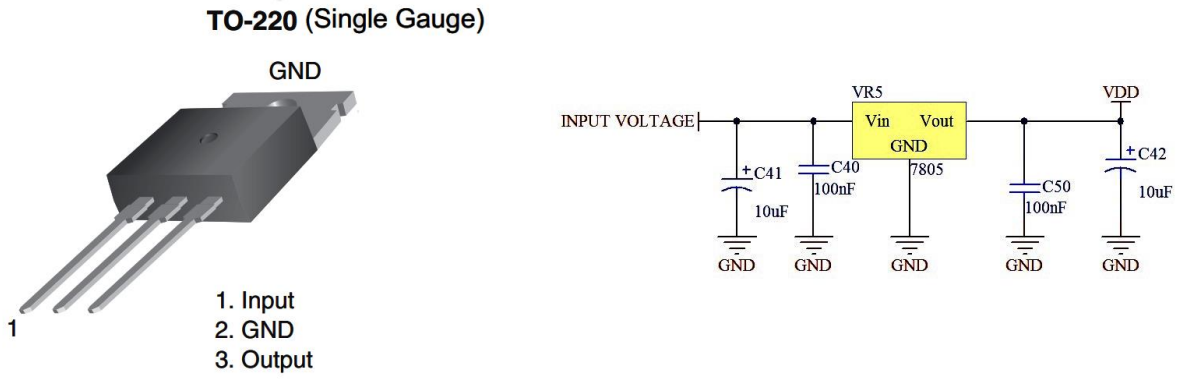
راه‌اندازی میکروکنترلر

پیش از اجرای هر پروژه‌ای روی میکروکنترلر لازم است که در ابتدا میکروکنترلر به درستی راه‌اندازی شود. در ادامه و پیش از شرح پروژه به ارائه‌ی برخی از مهم‌ترین ملزومات در راه‌اندازی میکروکنترلر پرداخته می‌شود که در پیاده‌سازی‌های عملی حتماً باید در نظر گرفته شوند.

مدار تغذیه:

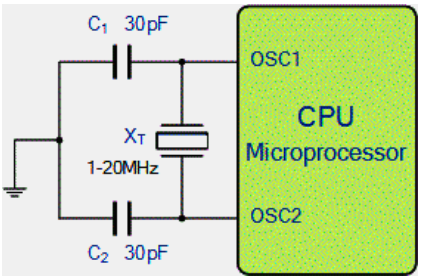
برای راه اندازی میکروکنترلر ابتدا باید ولتاژ تغذیه مناسب برای آن فراهم شود. به این منظور، به طور معمول از یک منبع تغذیه DC و یک IC رگولاتور استفاده می‌شود. به طور معمول، رگولاتور مورد استفاده، IC رگولاتور خطی به شماره 7805 می‌باشد. شکل ۱ نمای ظاهری و مدار مورد استفاده را نمایش می‌دهد:



شکل ۱: نمای ظاهری رگولاتور و مدار مورد استفاده برای تغذیه.

مدار اسیلاتور:

هر پردازنده‌ای برای اینکه بتواند به درستی کار کند باید سیگنال ساعت آن از طریقی فراهم شود. یک میکروکنترلر به طور معمول دارای چند منبع کلاک است که می‌توان توسط پیکربندی‌های سخت‌افزاری منبع کلاک مورد نظر را انتخاب نموده و استفاده کرد. یکی از انواع اسیلاتورهای ساده و دقیق مورد استفاده در مدارهای مبتنی بر میکروکنترلر، اسیلاتور کریستالی است. مدار مورد استفاده برای اسیلاتور کریستالی در شکل ۲ نمایش داده شده است.



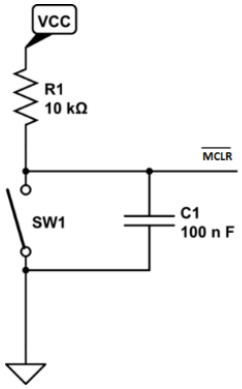
شکل ۲: مدار مورد استفاده برای اسیلاتور کریستالی.

در مدارات عملی، کریستال کوارتز و خازن‌های مربوط به آن باید تا حد امکان نزدیک پایه‌های میکروکنترلر قرار گیرند و در صورت بلند بودن پایه‌ی خازن‌ها، این پایه‌ها کوتاه شوند.

برای آشنایی بیشتر با منابع کلاک میکروکنترلرهای PIC و نحوه‌ی پیکربندی به بخش Oscillator Configurations از Datasheet میکروکنترلر مراجعه کنید.

مدار Reset:

برای عملکرد صحیح میکروکنترلر پایه‌ی Reset خارجی میکروکنترلر باید به ولتاژ مثبت تغذیه متصل گردد. برای این کار می‌توان از یک مقاومت Pull up استفاده کرد. شکل ۳ یک مدار مناسب را برای پایه‌ی Reset نمایش می‌دهد که شامل مقاومت Pull up ، سوئیچ Reset و خازن Debounce می‌باشد.



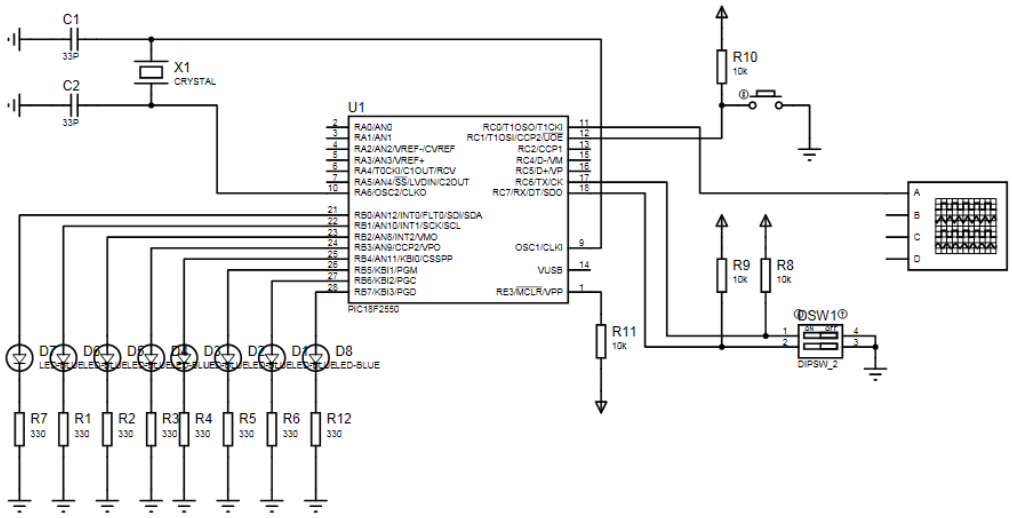
شکل ۳: مدار مناسب برای پایه‌ی Reset.

شرح پروژه:

در این پروژه با استفاده از پایه‌های ورودی و خروجی میکروکنترلر PIC18F2550 کارهای مختلفی از جمله فلاشر LED با آرایش‌های متفاوت و مدولاسیون عرض پالش انجام خواهد شد.

- **سخت افزار**

مداری مطابق شکل ۴ ببندید. این مدار شامل یک میکروکنترلر PIC18F255 و تعدادی قطعه شامل Dip Switch، LED و ... می‌باشد. مدار تغذیه، Clock و Reset مطابق با توضیحات ابتدایی فایل به مدار اصلی اضافه شده است.



شکل ۴: شماتیک مدار مورد نیاز برای پروژه.

- **نرم افزار**

برنامه ای بنویسید که با توجه به وضعیت Dip Switch متصل شده عملیات زیر را انجام دهد:

وضعیت 00:

در این حالت میکروکنترلر LEDهای متصل به پورت را به ترتیب و با فاصله زمانی 200 ms به صورت 1 of N روشن و خاموش کند.

وضعیت 01:

در این حالت میکروکنترلر LEDهای متصل به پورت را به ترتیب و با فاصله زمانی 200 ms به صورت آبشاری روشن و خاموش کند.

وضعیت 10:

در این حالت میکروکنترلر یک پالس PWM با فرکانس 100 Hz را روی یکی از پایه‌های میکروکنترلر ایجاد می‌کند. برنامه باید به گونه‌ای باشد که با هر بار فشار دادن میکرو سوئیچ، Duty Cycle پالس ۱۰٪ افزایش

پیدا کند و با رسیدن به مقدار ۱۰۰٪ دوباره از ابتدا شروع کند. پین مورد نظر را به اسیلوسکوپ متصل نموده و پالس خروجی را نمایش دهید.

برای تنظیمات میکروکنترلر از خطوط زیر در ابتدای کد خود استفاده کنید:

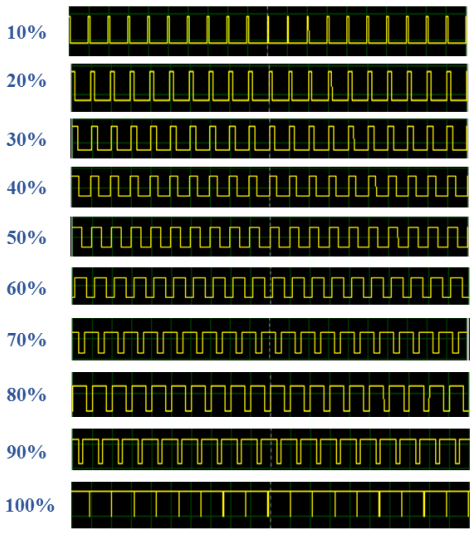
```
#include <p18f2550.inc>
LIST    P=18F2550
CONFIG  WDT=OFF          ; WATCHDOG TIMER OFF
CONFIG  FOSC = XT_XT     ; CONFIGURE CLOCK SOURCE TO EXTERNAL CRYSTAL
CONFIG  LVP = OFF        ; LOW VOLTAGE PROGRAMMING DISABLED
CONFIG  BOR = OFF        ; Brown Out Reset OFF
CONFIG  VREGEN = OFF     ; USB voltage regulator OFF
CONFIG  PBADEN = OFF     ; PORTB<4:0> pins are configured as digital I/O on Reset.
CONFIG  MCLRE = ON       ; MCLR pin enabled; RE3 input pin disabled.
```

- **تولید موج PWM**:

مدولاسیون عرض پالس (PWM[1])، تکنیکی است که به صورت گسترده در کنترل موتورهای جریان مستقیم (DC[2]) مورد استفاده قرار می‌گیرد. به طور کلی سرعت موتور DC به سه عامل بستگی دارد: ۱) بار موتور، ۲) ولتاژ و ۳) جریان. برای یک بار ثابت می‌توان با استفاده از روشی به نام مدولاسیون عرض پالس در موتور DC به یک سرعت ثابت رسید. در این روش با تغییر دادن عرض پالسی که به موتور داده می‌شود، می‌توان مقدار توانی را که به موتور داده می‌شود کم یا زیاد کرد که در نتیجه سرعت موتور کاهش یا افزایش می‌یابد. توجه کنید که در این روش دامنه‌ی موجی که به موتور داده می‌شود ثابت است و این Duty Cycle موج مربعی داده شده است که تغییر می‌کند.

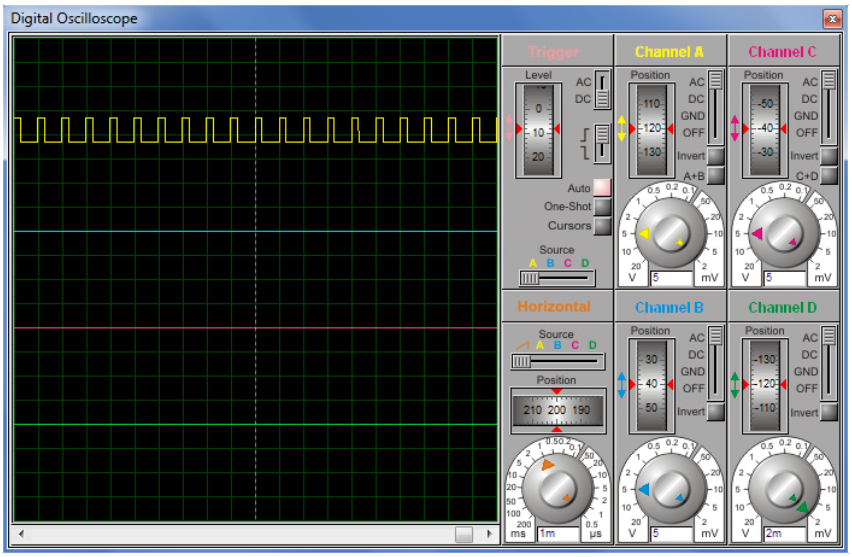
در شکل ۵ موج‌های مربعی با دوره‌ی تناوب یکسان وDuty Cycle های متفاوت نمایش داده شده است. در این شکل دوره‌ی تناوب و دامنه‌ی همه‌ی موج‌های مربعی ایجاد شده یکسان است و تنها چیزی که در این موج‌ها متفاوت است Duty Cycle آن‌ها است. اگر یک موج مربعی تولید شده و به یک موتور DC متصل به یک بار ثابت اعمال شود، با افزایش Duty Cycle موج، سرعت موتور DC افزایش می‌یابد.

[1] Pulse Width Modulation
[2] Direct Current



شکل ۵: موج‌های مربعی با دوره‌ی تناوب یکسان و Duty cycleهای متفاوت.

در بخشی از این پروژه قرار است شما یک موج PWM بر روی یکی از پایه‌های خروجی میکروکنترلر تولید کنید که Duty Cycle آن از طریق یکی از پایه‌های میکروکنترلر قابل کنترل است. میکروکنترلر در پایه‌ی خروجی خود یک موج PWM تولید می‌کند که برای نمایش این موج از یک اسیلوسکوپ در نرم‌افزار پروتئوس استفاده می‌شود. با هر بار فشرده شدن کلید باید Duty Cycle موج PWM تولید شده ۱۰٪ افزایش یابد. با رسیدن Duty Cycle به مقدار ۱۰۰٪ اگر مجدداً کلید ورودی فشرده شود باید Duty Cycle موج تولید شده به مقدار ۱۰٪ بازگردد. **دقت کنید که دامنه و دوره‌ی تناوب موج باید در همه‌ی حالت‌ها ثابت باقی بماند.** در شکل ۶ نمایش موج تولید شده بر روی اسیلوسکوپ نرم‌افزار پروتئوس نشان داده شده است.



شکل ۶: ابزار اسیلوسکوپ در نرم‌افزار پروتئوس.

توضیحات:

۱- کد نوشته شده برای پروژه به زبان اسمبلی را به همراه فایل‌های شبیه‌سازی Proteus و همچنین یک گزارش کامل و **تایپ‌شده** از پروژه، در یک فایل zip تا پایان مهلت تعیین شده ارسال کنید. حتماً نام فایل ارسالی خود را با فرمت «Name_StudentNumber» انتخاب کنید.

۲- توجه داشته باشید که بخش اصلی نمره به نتیجه‌ی به دست آمده تعلق خواهد گرفت که نمود آن دریافت نتیجه‌ی صحیح در شبیه‌سازی با Proteus است.

۳- سعی کنید اتصالات مدار به پایه‌های میکروکنترلر را مطابق با شماتیک شکل ۴ انجام دهید. این کار باعث می‌شود، در صورتی که به هر دلیلی شبیه‌سازی Proteus شما با مشکل مواجه شود، پروژه‌ی شما قابل ارزیابی باشد و نمره‌ی آن را از دست ندهید.

۴- همچنین، به احتمال بسیار بالا امکان تمدید مدت زمان پروژه وجود نخواهد داشت. پس سعی کنید با برنامه‌ریزی درست دچار مشکل در تحویل آن نشوید.