

کوییز سوم

سیستمهای نهفته و بیدرنگ

**استاد درس**

دکتر حسینی منزه

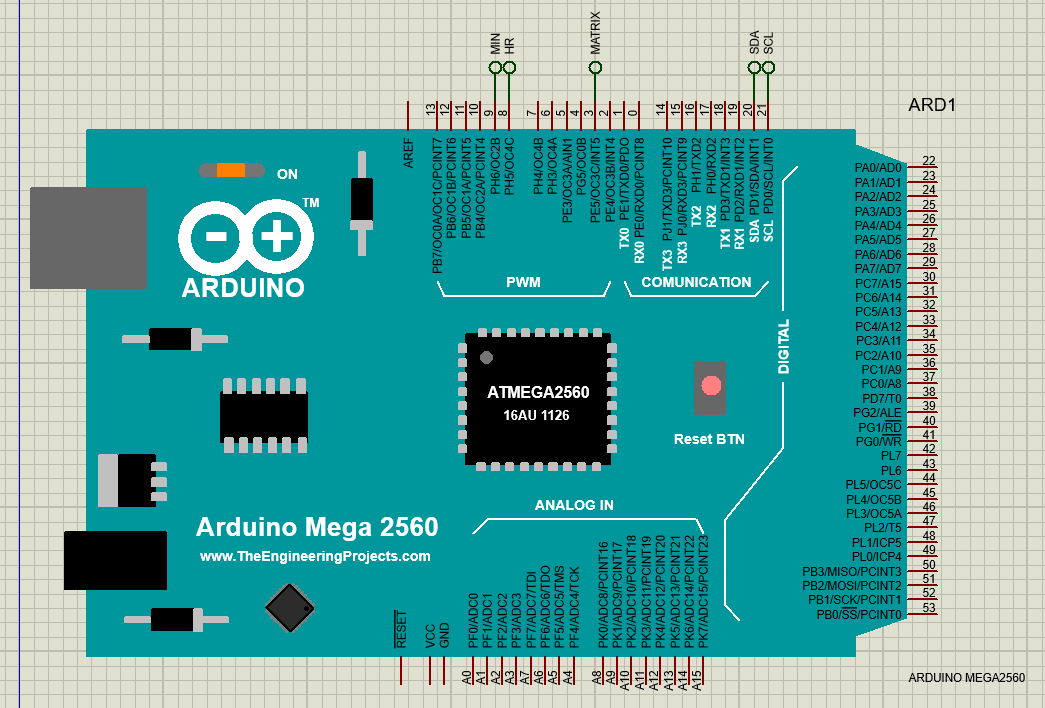
**دانشجو**

امیرحسین احمدی

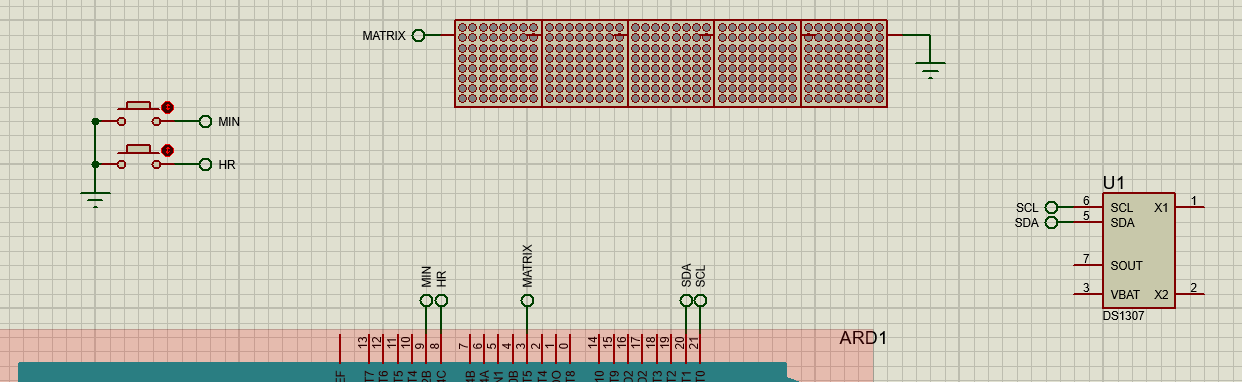
97522292

بهار 1401

برای حل این سوال از یک پردازنده Arduino Mega 2560 کمک میگیریم که تصویر آن در نرم افزار Proteus را در زیر میبینید (برای اضافه کردن آن به Proteus نیاز به اضافه کردن تعدادی Library جداگانه بود.)

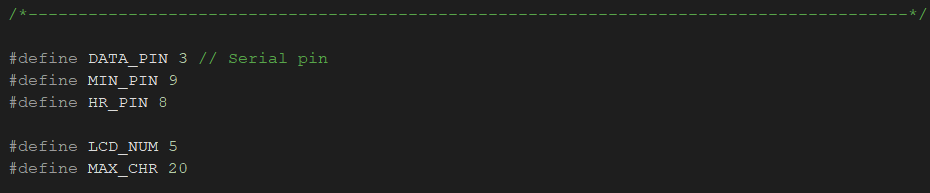


برای پیاده سازی این سوال، نیاز به یک کلاک خارجی برای گرفتن زمان داشتیم که از قطعه DS1307 برای این کار استفاده کردیم. همچنین برای پیاده سازی تابلوی روان 5 قطعه WS2812X8X8 که به صورت سطری به هم متصل کرده ایم استفاده کردیم که هر کدام شامل 8 \* 8 قطعه LED میباشند. برای تنظیم ساعت و دقیقه نیز از دو دکمه استفاده کردیم که زمین وصل شده اند.

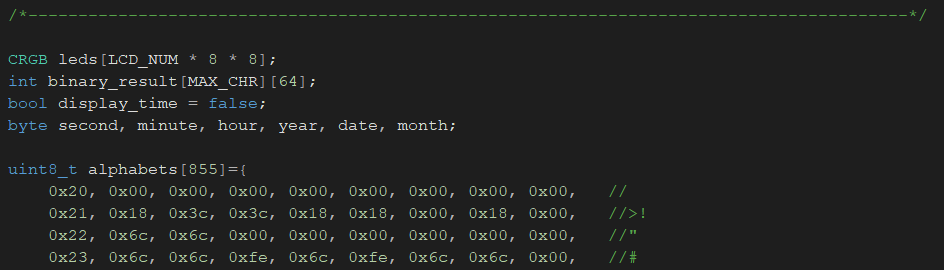


همانگونه که میبینید، دو دکمه را به دو Pin 8 و 9 ، LED ها را به Pin 3 و کلاک را به پین های مربوطه وصل کرده ایم.

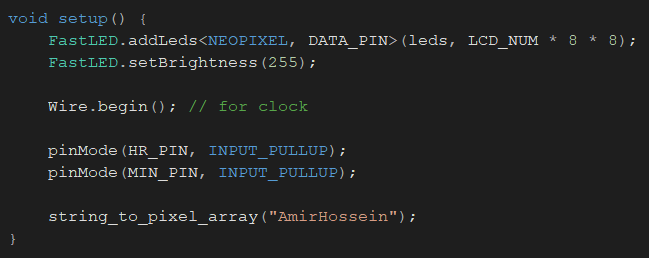
در کد، ابتدا هر کدام از Pin های گفته شده را تعریف میکنیم تا در ادامه نیاز به نوشتن اعداد نباشد. همچنین تعداد LCD های استفاده شده را تعریف میکنیم تا در ادامه بتوانیم در محاسبات استفاده کنیم. MAX\_CHR نیز حداکثر مقدار حروفی است که میتوانیم نشان دهیم (به صورت متحرک) که در ادامه از آن استفاده خواهیم کرد.



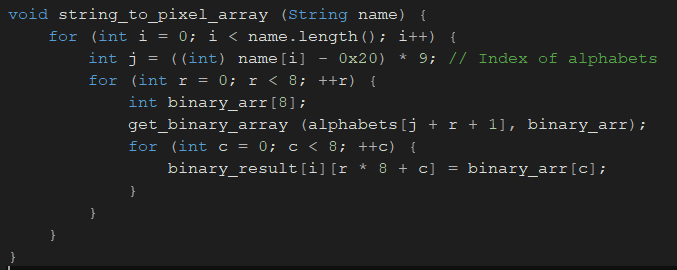
در ادامه متغییر های Global که در ادامه نیاز داریم را تعریف میکنیم. ابتدا یک آرایه به طول تعداد LED های موجود در تابلوی روان است که از نوع CRGB است. CRGB مربوط به FastLED است که میتوان در آن رنگ هر LED را تعیین کرد. یک آرایه عددی به اسم binary\_result داریم که مقادیر 0 و 1 میگیرد و نشان میدهد که LED خاموش است یا خیر. برای راحتی کار این آرایه متفاوت از leds در نظر گرفتیم. یک متغییر Boolean داریم که نشان میدهد که در حال نمایش زمان هستیم یا خیر. برای تمام مقادیر ثانیه، دقیقه، ساعت، سال، تاریخ و ماه یک متغییر گرفتیم زیرا در هنگام ارتباط به Clock خارجی و Update کردن آن باید کل این مقادیر را رد و بدل کنیم. در آخر یک آرایه داریم که به ازای هر کاراکتر قابل نمایش در کد ما 9 متغییر را نگه میدارد که اولین متغییر برابر با مقدار اسکی آن کاراکتر است و 8 متغییر بعدی اعدادی هستند که اگر به حالت Binary در نظر بگیریم، وضعیت روشن و خاموش بودن LED ها را نشان میدهند اگر بخواهیم کاراکتر مورد نظر را در یک 8 در 8 نمایش دهیم.



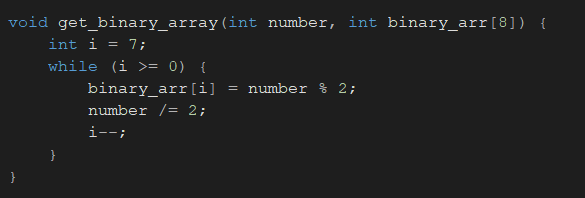
در تابع setup ابتدا FastLED را Initialize میکنیم و که در آن Pin خروجی، آرایه خروجی و سایز آرایه را به FastLED میدهیم و میزان روشنایی Pixel ها را نیز تعیین میکنیم. سپس Wire را که برای ارتباط با Clock خارجی به آن نیاز داریم begin میکنیم. دو Pin مربوط به تنظیم ساعت و دقیقه را به عنوان ورودی تنظیم میکنیم و تابع string\_to\_pixel\_array را با اسم خودم صدا میزنم تا با تبدیل آن به آرایه binary آماده نمایش روی LCD ها شود.



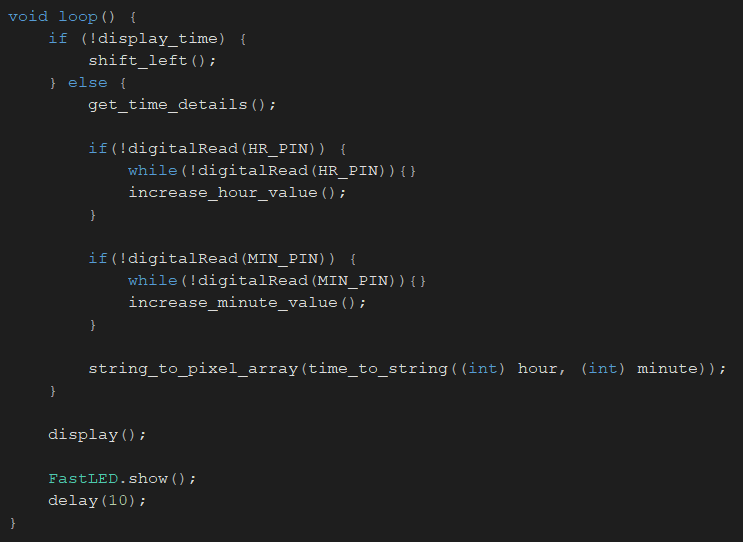
تابع string\_to\_pixel\_array که پیش تر به آن اشاره شد به این صورت عمل میکند که به ازای هر کاراکتر در String داده شده به آن Index آن در آرایه alphabets که توضیح داده شد را پیدا میکند و به ترتیب تمام اعداد آن را با استفاده از get\_binary\_array به آرایه Binary تبدیل کرده و درون binary\_result که توضیح داده شد میریزد.



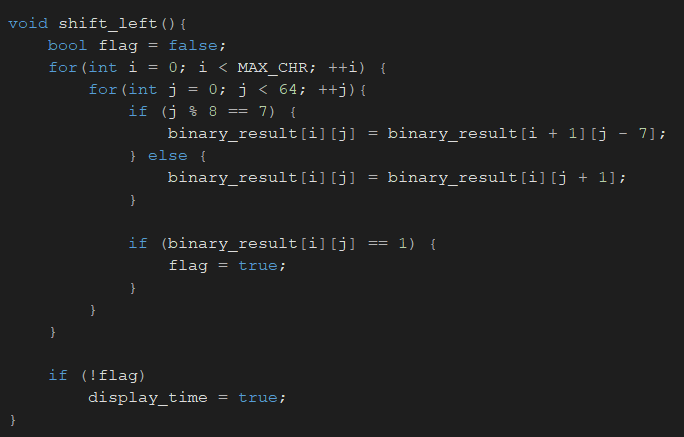
در تابع get\_binary\_array که پیشتر اشاره شد با تقسیم بر 2 کردن مداوم عدد آرایه Binary را از آن استخراج میکند.



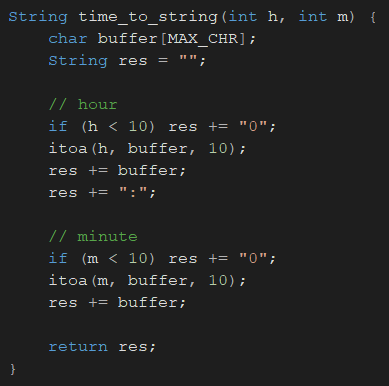
در تابع loop به این صورت عمل میکنیم که در صورتی که هنوز نمایش اسم تمام نشده (display\_time برابر با false است) تابع shift\_left را صدا میزنیم که Pixel های موجود در binary\_result را یک واحد شیفت میدهد تا اسم به صورت متحرک باشد. در غیر این صورت ابتدا با صدا زدن get\_time\_details اطلاعات زمانی را از Clock میگیرد، سپس در صورت فشرده شدن دکمه های تنظیم زمان هر کدام از ساعت و دقیقه را تنظیم میکنیم و در نهایت با تبدیل کردن دقیقه و ساعت به String و فرستادن آن به string\_to\_pixel\_array آن ها را در خروجی قرار میدهیم. در نهایت تابع display که آرایه binary\_result را درون leds میریزد صدا میزنیم و با دستور FastLED.show تابع leds را خروجی میدهیم و در آخر مقداری Delay میدهیم تا نمایش Pixel ها را بتوانیم حس کنیم.



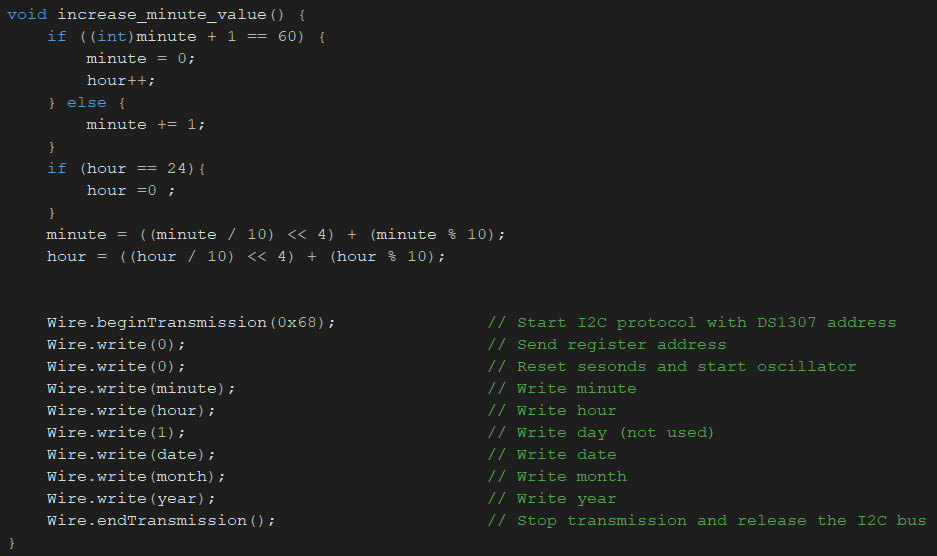
در تابع shift\_left به ازای هر led آن را برابر با led سمت چپی باید قرار دهیم که برای خانه های هر کاراکتر در binary\_result به خانه ی بعدی مراجعه میکنیم ولی برای خانه های آخر مقدار خانه های اول کاراکتر بعدی را بر میداریم. همچنین چک میکنیم که اگر تمام Pixel ها خاموش بودند یعنی نمایش اسم به پایان رسیده و display\_time را True میکنیم.



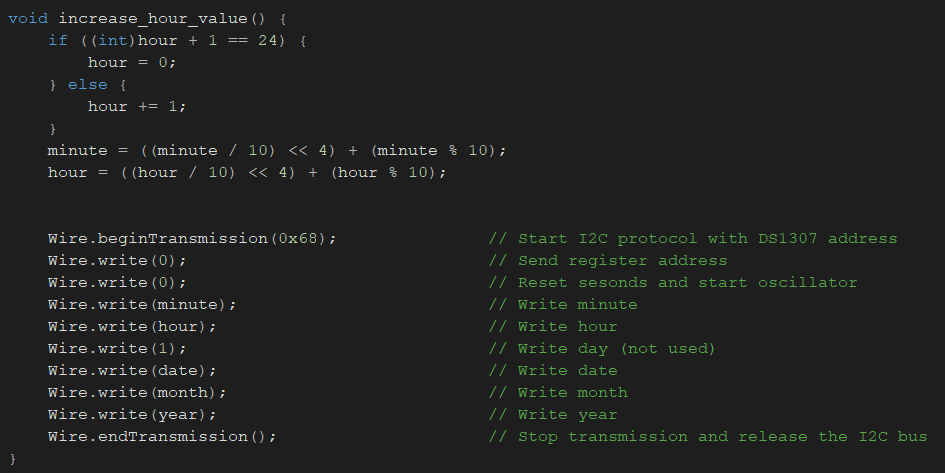
در تابع time\_to\_string با گرفتن ساعت و دقیقه و با استفاده از تابع itoa برای تبدیل string به int ساعت و دقیقه را به فرمت HH:MM در میاوریم. (هر کدام کم تر از 10 بود یک صفر پشت آن میگذاریم.)



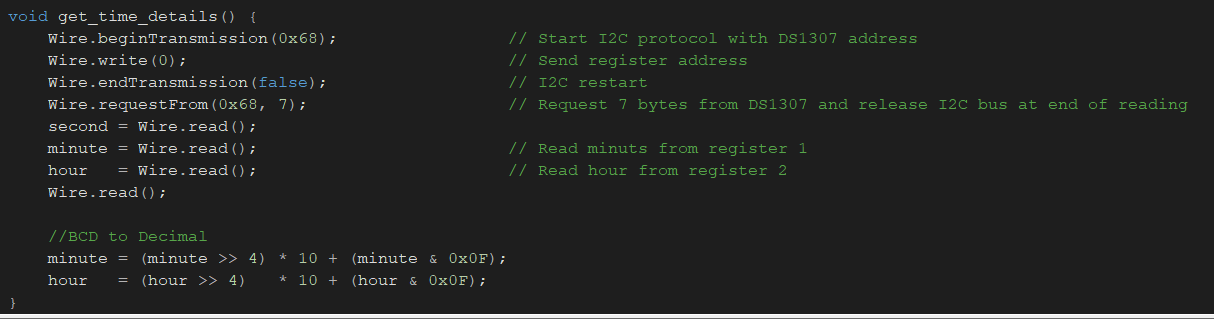
در تابع increase\_minute\_value مقدار دقیقه را یکی بالا برده و در صورت 60 شدن صفر کرده و ساعت را بالا میبریم و در صورت 24 شدن ساعت آن را نیز صفر میکنیم. مقدار آن را به BCD تبدیل کرده و با استفاده از wire به Clock ارسال میکنیم.



در تابع increase\_hour\_value نیز مانند بالا عمل کرده فقط با این تفاوت که مقدار ساعت را یکی زیاد میکنیم.



در تابع get\_time\_details مقادیر زمانی را با استفاده از wire خوانده و دو متغییر دقیقه و ساعت که برای ما اهمیت دارند را از BCD به Decimal تبدیل میکنیم تا بتوانیم از آن ها استفاده کنیم.



در نهایت در تابع display مقادیر binary\_result را از اول تا جایی که میتوان درون leds میریزیم تا بتوانیم خروجی دهیم. اگر LED روشن باشد آن را قرمز و اگر خاموش باشد آن را مشکی میکنیم.

