**عنوان پروژه:** چراغ مطالعه با رعایت آسایش چشمی (ECRL)  
**اعضای تیم:** محمدمهدی مرادی گلین قشلاقی، مرتضی نظری سومرین  
**مؤسسه:** دانشگاه تبریز  
**استاد راهنما:** دکتر محمدزاده  
**تاریخ تحویل:** ۱۴۰۴/۰۳/۰۴ (۲۰۲۵/۰۵/۲۵)

**مراحل طی‌شده از ابتدای پروژه تاکنون:**

* در آغاز، ایده‌ی اولیه‌ی ما ساخت چراغی بود که بتواند خود را با نور محیط تطبیق دهد.
* در ادامه، این ایده بهبود یافت تا بجای تطبیق خروجی‌ها با نور محیط، کمبود نور محیط را جبران نماید. پس از جست‌وجو در منابع اینترنتی، نسبت ترکیب نور زرد به نور سفید را به‌صورت ۰.۶ به ۰.۴ تعیین کردیم. در این مرحله مشخصات اولیه پروژه شکل گرفت که در فایل specification.cpp در دایرکتوری اصلی پروژه قرار دارد.  
  ایده‌ی اصلی ما در اینجا شکل جدی‌تری به خود گرفت. در این مرحله قصد داشتیم از ماژول ESP32 و یک فتوسل برای خواندن نور محیط استفاده کنیم (با علم به اینکه فتوسل قابلیت تشخیص RGB ندارد. اما بعدا نظرمان به آردوینو تغییر پیدا کرد.
* مرحله‌ی بعدی که پس از تعطیلات فروردین ‌ماه آغاز شد، مربوط به پیاده‌سازی فیزیکی پروژه بود. در این زمان، تمام مؤلفه‌های سخت‌افزاری و نرم‌افزاری آماده‌ی مونتاژ شده بودند. در این نسخه از پروژه، از فتوسل استفاده کردیم و گزارش اول ما نیز این مرحله را پوشش می‌دهد. هرچند نتایج به‌دست‌آمده با اهداف اصلی ما فاصله داشت، اما توانستیم درک عمیقی از نحوه‌ی کار با آردوینو به‌دست آوریم و همچنین قابلیت تنظیم شدت نور LEDها را متناسب با نور محیط (خوانده شده توسط فتوسل) پیاده‌سازی کنیم. طبق گزارش اول و دوم، در این مرحله با مشکلات نرم‌افزاری مواجه شدیم، چرا که فتوسل یک سنسور غیرخطی است و مدیریت آن کمی دشوار است. از آنچایی دشوار است که تعیین بازه های تقسیم بندی نور و threshold خطا کاری دشوار است.
* پس از آن، تا حد امکان تلاش کردیم تا مشکلات نرم‌افزاری را رفع کنیم. توانستیم نسخه‌ای قابل‌قبول از نرم‌افزار را پیاده‌سازی کنیم که البته مخصوص سخت‌افزار سطح اولیه یعنی فتوسل بود. این امر قابل قبول بود، چرا که نرم‌افزار را به‌گونه‌ای توسعه دادیم که قابلیت گسترش و توسعه در آینده را داشته باشد؛ به‌عبارتی، پس از جایگزینی سنسور رنگی TCS34725، می‌توانستیم ویژگی‌های نرم‌افزاری اختصاصی آن را بدون تغییر اساسی در ساختار نرم‌افزار موجود اضافه کنیم. این یک پیشرفت حرفه‌ ای برای ما محسوب میشد.
* در مرحله ی سوم، از ترکیب سنسور TCS34725 و فتوسل استفاده کردیم. نرم‌افزار این مرحله شامل قطعات کد متعددی برای کنترل، هم‌ترازی و تثبیت عملکرد دو سنسور به طور همزمان بود. اما در ادامه مسیر توسعه، تصمیم گرفتیم به طراحی مبتنی بر یک سنسور واحد روی آوریم. چرا که مشاهده کردیم تأثیر سنسور TCS34725 در خروجی‌ها چشمگیرتر است. بنابراین، بدون تردید، تنها از این سنسور استفاده کردیم و فتوسل حذف شد. این یک نقطه‌ی عطف در پروژه ما بود.
* پس از این گذار از فتوسل به سنسور TCS34725، لازم بود تغییرات عمده‌ ای در نرم ‌افزار ایجاد کنیم. در این مرحله با مشکلات متعددی مواجه شدیم، به‌طوری‌که شش نسخه‌ی مختلف از نرم ‌افزار را روی آردوینو پیاده ‌سازی و آزمایش کردیم، تا اینکه در نهایت در نسخه‌ ی نهم به بهترین نتیجه دست یافتیم. در ادامه، توضیح مختصری از عملکرد این نسخه ارائه می‌شود:  
  نور محیط به سه سطح تقسیم شده است: بسیار تاریک، مناسب (راحت)، و بسیار روشن. در دو وضعیت بسیار تاریک و بسیار روشن، رفتار سیستم یکسان است: تا زمانی که تعادل نور برقرار نشده باشد، شدت نور خروجی کاهش می‌یابد و در صورت تداوم عدم تعادل، سیستم به ‌طور همزمان هر دو نوع نور (زرد و سفید) را خاموش می‌کند.  
  در وضعیت مناسب (راحت)، خروجی PWM افزایش می‌یابد، در حالی‌که نسبت کلی نورها (۰.۶ زرد به ۰.۴ سفید) حفظ می‌شود.
* پس از نهایی ‌سازی نرم‌ افزار، نوبت به پیاده‌ سازی مؤلفه‌ های خروجی رسید. در تمام مراحل پیشین، از LED به‌عنوان منبع نور خروجی استفاده می‌کردیم. اما در این مرحله تصمیم گرفتیم از لامپ ‌های رشته ‌ای ۱۲ ولت قابل ‌دیمر استفاده کنیم؛ زیرا این لامپ‌ ها شدت نور بالایی دارند و نسبت به لامپ‌ های ۲۲۰ ولت قابل ‌دیمر، ساده ‌تر قابل ‌کنترل هستند. در این مسیر با چالش ‌های متعددی روبه ‌رو شدیم، چرا که آشنایی کافی با قطعات و مدارات مربوطه (مانند TLP250، MOSFET نوع N مدل IRFZ) آشنایی نداشتیم، و همچنین باید ملاحظات مربوط به فرکانس‌ های بالا و جریان ‌های ناخواسته ‌ی ناشی از لامپ ‌های رشته ‌ای در هنگام خاموش شدن در فرکانس بالا را در نظر می ‌گرفتیم. مدار مربوطه طراحی، قطعات مورد نیاز تهیه، و طبق طرح مونتاژ شدند. خوشبختانه مدار در همان تلاش اول به‌ درستی عمل کرد. این مرحله بسیار پر استرس بود، چرا که احتمال آسیب جدی یا حتی سوختن کامل ماژول آردوینو وجود داشت، اما در عمل، مدار عملکردی دقیق و موفق داشت.

برای بررسی دقیق تر به گزارش های 1 تا 6 مراجعه شود.

**بررسی دقیق تر نرم افزار پروژه:**

### ۱. حالت) Fallback بازگشت اضطراری( چیست؟ چه مسئولیتی دارد؟

حالت **Fallback** زمانی فعال می‌شود که سیستم نتواند هیچ نوری را تشخیص دهد، یعنی شرایط **تاریکی مطلق** یا **عدم دریافت داده معتبر از سنسور** برقرار باشد. این حالت زمانی تشخیص داده می‌شود که:

currentYellowRatio == -1

**وظیفه حالت Fallback :**

اطمینان از اینکه حتی در صورت خرابی سنسور یا تاریکی کامل، سیستم بتواند نوری پایه فراهم کند.

* جلوگیری از شرایط ناراحت‌کننده یا خطرناک (مثلاً مطالعه در تاریکی مطلق).
* استفاده از مقادیر پیش‌فرض برای روشن‌سازی LEDها:

newYellowLED = default\_k \* targetYellowRatio;

newWhiteLED = default\_k \* (1.0 - targetYellowRatio);

**۲. برنامه از مقدار YOI/WOI = -1 چه برداشتی دارد؟**

این مقدار زمانی ایجاد می‌شود که:

float currentYellowRatio = (total > 0) ? YOI / total : -1;

اگر total برابر صفر باشد (یعنی هیچ نوری از LEDها یا محیط ثبت نشده)، برنامه نسبت زرد را به **عدد -1** تنظیم می‌کند.

**برداشت برنامه:**

* فعال‌سازی **حالت (Fallback) مود ۱**.
* نشان‌دهنده شرایطی مانند:
  + سنسور از کار افتاده یا قطع شده.
  + تاریکی کامل در محیط.

**۳. هیسترزیس (Hysteresis) چیست و چرا استفاده شده است؟**

**هیسترزیس** یک ناحیه بافر (حاشیه) اطراف محدوده راحتی نور است که با مقدار زیر تعریف می‌شود:

const float hysteresis = 2.0;

**هدف استفاده از هیسترزیس:**

* جلوگیری از تغییرات سریع بین حالات مختلف برنامه در اثر نویز یا نوسانات ناچیز نور محیط.
* تثبیت عملکرد سیستم و جلوگیری از چشمک‌زدن یا تغییرات مکرر.
* مشابه مفهومی به نام "Debouncing" در الکترونیک دیجیتال.

مثال استفاده:

isTooDark = ambientTotal < (L\_min - hysteresis);

isTooBright = ambientTotal > (L\_max + hysteresis);

**۴ .هر کدام از حالات Mod 1-4 چه کاری انجام می‌دهند؟**

**حالت ۱: Fallback**

if (currentYellowRatio == -1)

* در صورت عدم دریافت نور از سنسور.
* استفاده از مقادیر پیش‌فرض برای LEDها.
* فعال‌سازی حالت اضطراری برای تامین حداقل نور.

**حالت ۲ و ۳: نور بیش از حد یا کمبود نور**

else if (isTooDark || isTooBright)

* **نور کم:** کاهش تدریجی نور.
* **نور زیاد:** کاهش تدریجی نور. (رفتار دو حالت یکی در نظر گرفته شده)
* کاهش شدت LEDها با **ضریب کاهش 0.9**. (یعنی نور مرحله n، 0.9 ضریبی از مرحله قبلی خود است)
* حفظ نسبت نور زرد به سفید در محدوده مطلوب (0.58 تا 0.62).

**حالت ۴: در محدوده آسایش**

else

* تنظیم دقیق نسبت نور زرد و سفید بدون تغییر محسوس در شدت کل نور.
* اگر نسبت زرد کم باشد : افزایش تدریجیLED زرد.
* اگر نسبت زرد زیاد باشد : افزایش تدریجی LED سفید.
* میزان افزایش های تدریجی 0.1 حالات قبلی هستند.
* تنظیم بر اساس خطای موجود با نسبت هدف (0.6).

**۵. فرمول‌های ریاضی استفاده شده در کد و هدف آن‌ها:**

**الف. برآورد نور محیط:**

ambientWhite = min(r, min(g, b));

ambientYellow = (r + g) / 2.0;

* استخراج نور سفید و زرد از داده‌های RGB.
* تبدیل داده‌های خام به اطلاعات قابل استفاده برای کنترل نور.
* <https://www.analog.com/en/resources/technical-articles/map-colors-of-a-cie-plot-and-color-temperature-using-an-rgb-color-sensor.html?utm_source=chatgpt.com>
* <https://www.sciencedirect.com/science/article/abs/pii/S0039914022000406?utm_source=chatgpt.com>

**ب. محاسبه شدت خروجی نور:**

YOI = prevYellowLED + ambientYellow;

WOI = prevWhiteLED + ambientWhite;

total = YOI + WOI;

currentYellowRatio = (total > 0) ? YOI / total : -1;

* برآورد سهم نور تولیدشده توسط LED و محیط.
* کمک به تنظیم رنگ و دمای نور.
* اگر مقدار -1 شود، یعنی خطایی رخ داده و ورودی ها درست نبوده اند، یا سنسور درست کار نکرده، به طور کلی خطایی بوده که صلاح به ادامه کار نیست و برنامه اقدام به جبران این خطا میکند:
* if (currentYellowRatio == -1) {
* Serial.println("Mode: Darkness Fallback");
* newYellowLED = default\_k \* targetYellowRatio;
* newWhiteLED  = default\_k \* (1.0 - targetYellowRatio);
* inFallbackMode = true;
* }
* با در نظر گرفتن خطاهای ممکن، سیستم در حالت عادی(سالم بودن قطعات و شرایط مناسب) فقط به یک علت میتواند چنین اتفاقی بیافتد و آن تاریکی مطلق است. در نتیجه ما یک استانداری در نظر گرفته ایم که در این حالت مقادیر خروجی را به حالت دیفالت درآورد.

**پ. ضریب مقیاس برای کنترل روشنایی:**

newYellowLED = round(prevYellowLED \* scalingFactor);

* کاهش تدریجی نور برای اجتناب از تغییرات ناگهانی.

**ت. حفظ نسبت مطلوب زرد به کل:**

while ((ratio < 0.58 || ratio > 0.62) && ...)

* تضمین هماهنگی رنگ نور با نسبت هدف.
* کنترل دمای رنگ برای آسایش چشمی.

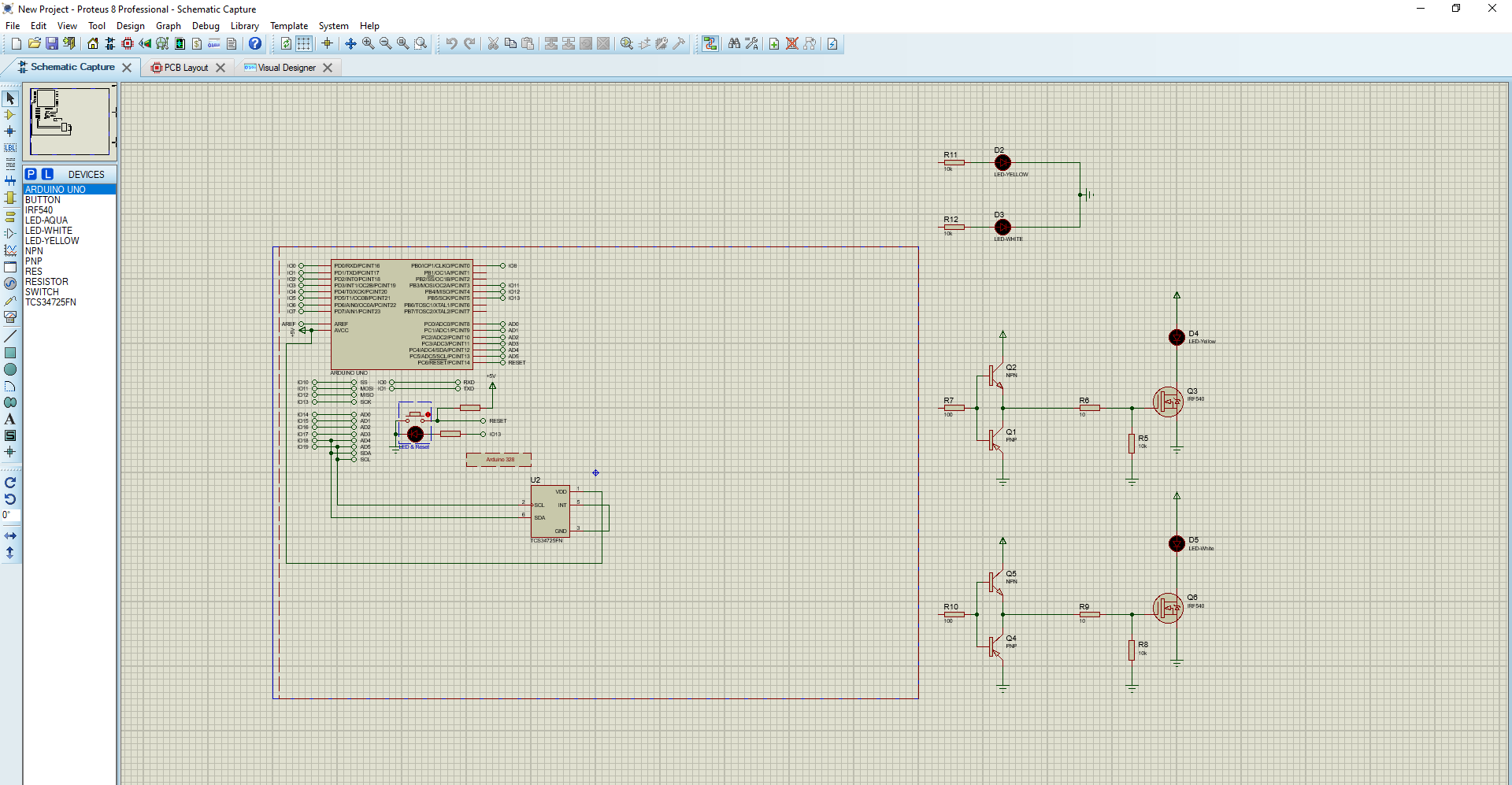
**ث. تنظیم دقیق در حالت راحتی:**

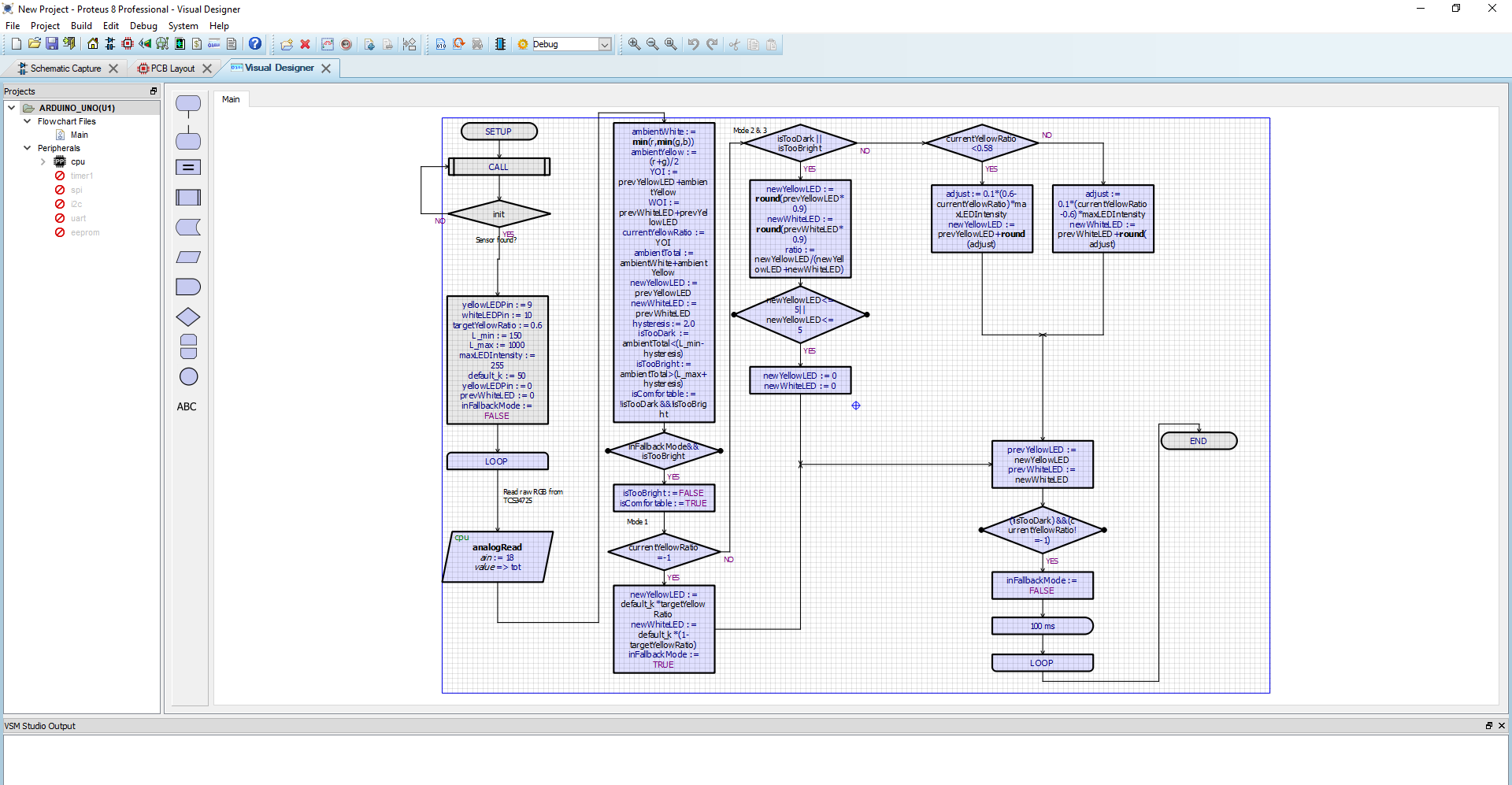
float adjust = 0.1 \* (difference\_from\_target) \* maxLEDIntensity;

* تنظیم تدریجی با ضریب کنترل برای جلوگیری از پرش و نوسان.
* دو حالت difference from target چنین هستند:
* 0.6 – currentYellowRatio
* currentYellowRatio - 0.6

**شبیه سازی پروژه:**

در این قسمت دو نوع خروجی مشاهده میشود. خروجی نوع اول (که ساده تر است) مربوط به حالت اولیه خروجی هاست (LED ها) و خروجی دوم مربوط به مدار لامپ رشته ای هاست. در هردو حالت پین 9 به لامپ زرد و پین 10 به لامپ سفید متصل میشود.

****

**بلوک دیاگرام پروژه:**