دستور کار شماره یک

بخش یک: بررسی روش‌های مختلف پیاده‌سازی ال‌ای‌دی چشمک زن در آردوینو

دانشجوها: مهسا عربی و محمد نصری

استاد: جناب محمد لالی

تاریخ: پائیز ۱۴۰۱

چکیده:

ابتدا به این می‌پردازیم که کریستال چیست و چه کاری در آردوینو انجام می‌دهد. و کریستال تعبیه شده در آردوینو چه جنسی دارد. و به این می‌پردازیم که برای پیاده‌سازی تابع چشمک زن روش‌های مختلفی وجود دارد که ما در این گزارش آن‌ها را بررسی می‌کنیم.ما مجموعه‌ای از آزمایش‌ها را با موضوعیت آردوینو و چراغ چشمک‌زن انجام می‌دهیم. ابتدا مفاهیم Blocking Function و Non-Blocking Function را توضیح دهیم. پیاده سازی‌ها و پروژه‌های عملی به ترتیب در پارت‌های یک تا چهار قرار می‌گیرند. و در بخش اخر هم به مفهوم thread در آردوینو می‌پردازیم و سعی می‌کنیم که تابع چشمک زن را به این وسیله پیاده سازی کنیم.

مقدمه و معرفی:

در این آزمایش‌ها ما به روش‌های مختلف ایجاد چرا چشمک زن به کمک دو آردوینو مختلف پرداختیم و هم به صورت نرم افزاری و برنامه نویسی روی آردوینو کار کردیم و هم با استفاده از آردوینو یک سخت افزار طراحی کردیم. این مجموعه آزمایش در پنج بخش مختلف انجام شده است. در بخش اول صرفا به توضیح یک سری مفاهیم مانند کریستال و نحوه‌های مختل استفاده از آن(تابع‌های مختلفی که از کریستال استفاده می‌کنند) پرداختیم. در بخش دوم با استفاده یکی از توابعی که در بخش اول توضیح دادیم قطعه کدی می‌نویسیم که چراغ خود آردوینو را هر نیم ثانیه خاموش و روشن کند. برنامه مربوط به بخش دوم در فولدر part2 موجود است و تنها کافی است که آن را روی آردوینو uno بفرستید تا ال‌ای‌دی آن شروع به چشمک زدن بکند. اما در بخش سوم در محیط پروتئوس یک آردوینو UNO داریم که تعدادی ال‌ای‌دی به آن متصل هستند که هر نیم ثانیه با هم خاموش و روشن می‌شوند. و در این بخش از تابع millis() استفاده کرده‌ایم. فایل‌های مربوط به این بخش در فولدر part3 موجود هستند. در بخش چهارم با استفاده از یک کلید و یک آردوینو ال‌ای‌دی‌هایی که در بخش قبل با هم دیگه خاموش و روشن می‌شدند را این بار به صورت پشت سر هم خاموش و روشن کردیم. به این صورت که هر کدام نیم ثانیه بعد از دیگری روشن می‌شوند و ال‌ای‌دی قبلی خاموش می‌شود. البته در این بخش از تابع delay() برای ایجاد وقفه استفاده کردیم. فایل‌های مربوط به این بخش در فولدر part4 موجود هستند و می‌توانید در این فولدر به آن‌ها دسترسی داشته باشید.

در بخش پنج هم به سراغ مفهوم رشته (thread) در آردوینو رفتیم و روش های متفاوت برنامه ریزی تسک ها را به تفصیل توضیح دادیم و علاوه برآن کدهای پیاده سازی را هم در فولدر part5 اضافه کردیم.

روش‌ها و تجهیزات مورد استفاده

در بخش‌های مختلف از روش‌ها و تجهیزات متفاوتی استفاده کردیم که هر کدام را به تفکیک در پایین می‌بینید.

در بخش اول صرفا کار تحقیقاتی انجام شده است و نتیجه کار را در بخش نتایج می‌توانید بخوانید. منابع را هم در بخش منابع آورده‌ایم . در صورتی که مایل به مطالعه بیشتر بودید می‌توانید به آن مراجعه کنید.

بخش دوم

در این بخش باید یک آردوینو UNO در اختیار داشته باشید. این تنها چیزی است که نیاز دارید. می‌توانید کد مربوط به این بخش را که در فولدر part2 موجود است روی آردوینو بفرستید و ال‌ای‌دی روی خود آردوینو شروع به چشمک زدن می‌کند.

بخش سوم

در بخش سوم پروژه در محیط پروتئوس است. بنابراین نیاز به برنامه پروتئوس دارید.

در محیط پروتئوس هم از تجهیزات مختلفی استفاده شده است که به شرح زیر هستند:

برد آردوینو UNO.

هشت عدد ال‌ای‌دی زرد که به پایه‌های آردوینو متصل شده‌اند.

هشت عدد مقاومت هفتاد و سه اهم که ال‌ای‌دی‌ها توسط آن‌ها به زمین متصل شده‌اند.

یک عدد کلید فشاری که توسط مقاومت پول-آپ به برد آردوینو متصل شده است.

برد آردوینو را هم برنامه نویسی کرده‌ایم. کل فایل پروژه را می‌توانید در فولدر part3 ببینید.

بخش چهارم

بخش چهارم هم در محیط پروتئوس است. بنابراین به برنامه پروتئوس نیاز دارید.

در محیط پروتئوس هم از تجهیزات مختلفی استفاده شده است که به شرح زیر هستند:

یک Arduino Atmega25600 استفاده شده و فایل hex. را روی آن اجرا و از ۸ LED و یک push button و مقاومت‌هایی برای کنترل جریان به LED ها وصل شده است.

برای اجرای آن میتوان از محیط شبیه ساز پروتئوس استفاده کرد یا اینکه برنامه را روی برد Arduino Atmega25600 ریخت و در محیط واقعی نتیجه را مشاهده کرد.

بخش پنجم

در بخش پنجم بیشتر بحث تحقیقاتی وجود دارد و برای آشنایی با زمانبندی تسک‌ها و کار با thread ها در آردوئینو مطرح است و برای نوشتن برنامه ها میتوان از کتابخانه هایی استفاده کرد که معرفی شده و روی برد آردوئینو نتیجه عملی را دید.

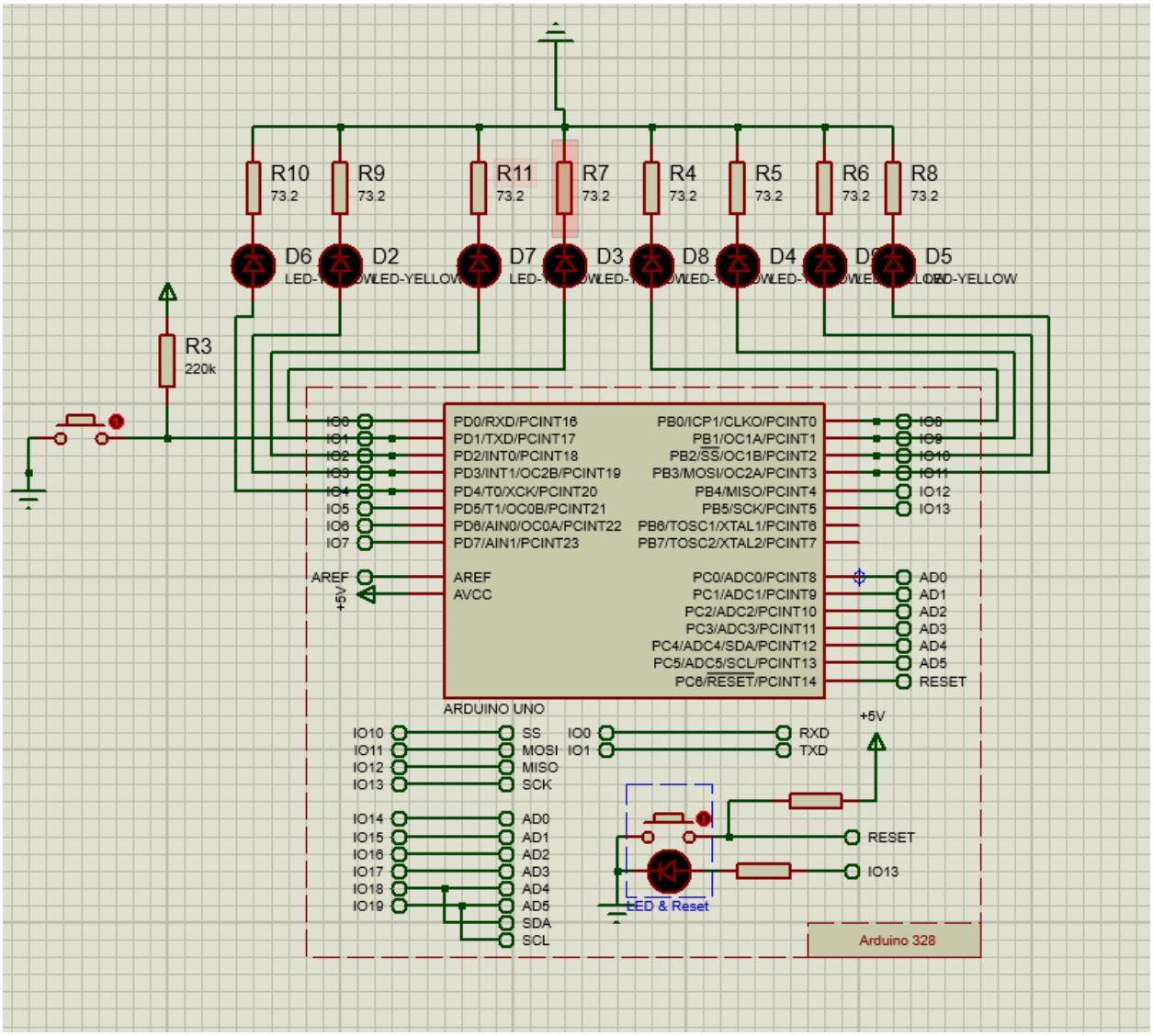
روش آزمایش

بخش دوم:

در این بخش ما بعد از نوشتن قطعه کدمان که در فولدر part2 موجود است آن را به آردوینو ارسال کردیم و بدون مشکل شروع به کار کرد. این قطعه کد را می‌توانید در فولدر مذکور و با نام part2.ino ببینید. برای تکرار آزمایش کافی است که این فایل را روی آردوینو UNO آپلود کنید.

بخش سوم:

در این بخش در قسمت تجهیزات همه تجهیزات مورد استفاده را ذکر کردم و آنها را به طریقی که در عکس پایین می‌بینید به هم متصل کردیم.



تصویر ۱: تصویر سخت افزار طراحی شده در بخش سوم

بعد از این که این تجهیزات را به همین صورت متصل کردیم در قسمت سورس کد برنامه کدی را نوشتیم که در فولدر مربوط به این بخش تحت اسم code موجود است. شما هم می‌توانید آن را در قسمت کد کپی کنید و از آن استفاده کنید.

بخش چهارم

هدف اجرای یک برنامه خاموش و روشن شدن به ترتیب هشت LED در صورت فعال شدن پرت دیجیتال شماره صفر و اجرای این برنامه در شبیه ساز آردئینو در پروتئوس بوده است.

این برنامه را در Arduino IDE نوشته و فایل hex. آن با نام code.ino و همینطور کد آن در ضمیمه موجود است.

این برنامه را روی شبیه ساز Arduino ATMega2560 در محیط پروتئوس اجرا و فایل آن و همینطور اسکرین ریکورد از نتیجه آن در ضمیمه موجود است.

بخش پنجم

برنامه blinking LED را با استفاده از threads به دو روش از روش های که در توضیحات مربوط به بخش ۵ آورده شده است نوشته شده و در ضمیمه موجود است.

توضیحات مربوط به فایل blinking-led-schedueler : سه عدد LED را به پین‌های شماره ۸ ، ۹ و ۱۳ متصل می‌کنیم.سپس در loop شماره ۱ ، LED متصل به پین شماره۸ را با تاخیر ۱ ثانیه به صورت چشمک زن در می‌آوریم و همچنین در loop شماره ۲ ، LED متصل به پین شماره ۹ را با تاخیر ۱۰۰ میلی‌ثانیه به صورت چشمک زن در می‌آوریم و همچنین در loop شماره ۳ ، به وسیله ارتباطط سریال و دریافت دو کاراکتر ۰ و ۱ ، LED متصل به پین شماره ۳ را خاموش روشن می‌کنیم.در صورتی که کدهای زیر را بر روی آردوینو DUE آپلود کنید مشاهده خواهید کرد که delay های موجود در loop ها بر روی هم تاثیری نمی‌گزارند و هر loop به صورت مستقل پردازش می‌شود.

نتایج و توضیح در مورد آن‌ها

بخش یک[1]:

کریستال

ابتدا باید تعریفی از کریستال و وظیفه آن در آردوینو داشته باشیم. کریستال یک قطعه است که برای ما کلاک ایجاد می‌کند. کریستال در بردهای مختلفی می‌تواند وجود داشته باشد و در آردوینو هم وجود دارد. در صورتی که کلاک نداشته باشیم آردوینو امکان انجام دادن هیچ کاری ندارد. اکثر کریستال‌ها از جنس کوارتز هستند اما در در آردوینو UNO جنس کریستال از سرامیک است. این باعث می‌شود که هزینه ساخت آردوینو کاهش پیدا کند.

تابع ()delay

این تابع یک آرگومان صحیح به عنوان ورودی می‌گیرد. این عدد نشان دهنده این است که مدت زمان تاخیر ایجاد شده چند میلی ثانیه است. و در زمانی که برنامه به این خط می‌رسد به اندازه مشخص شده در این خط توقف می‌کند و هیچ کاری انجام نمی‌دهد. و بعد از این که مدت زمان مشخص سپری شد به خط بعدی می‌رود. برای مثال قطعه کد زیر را بررسی می‌کنیم.

```
1:  void loop() {
2: digitalWrite(led1,1);
```

```
3:  delay(500);
4:  digitalWrite(led1,0);
5:  delay(500);
}
```

در این قطعه کد که مربوط به بخش تکرار شونده برنامه ما است وقتی برنامه به خط سوم می‌رسد به اندازه پانصد میلی ثانیه یا نیم ثانیه متوقف می‌شود و سپس به خط بعدی می‌رود. در واقع به اندازه نیم ثانیه برنامه ما متوقف یا بلاک می‌شود. به همین دلیل می‌گوییم که delay() یک Blocking Function است.

تابع millis()

این تابع مدت زمان سپری شده از شروع کار برنامه تا زمانی که فراخوانی شده است را به میلی ثانیه نشان می‌دهد. و می‌توانیم اینطوری از آن استفاده کنیم که میزان زمان سپری شده را در دو یا چند جا و زمان مختلف پیدا کنیم و در صورتی که اختلاف آن‌ها برابر با مقدار مشخصی شد کاری را انجام بدهد. در قطعه کد زیر یک نمونه کد ساده از این تابع را برای چراغ چشمک زن می‌بینیم.

```
unsigned long previousMillis = 0;

void loop() {
 unsigned long currentMillis = millis();

  if (currentMillis - previousMillis >= ۵۰۰) {
    // save the last time you blinked the LED
    previousMillis = currentMillis;

    // if the LED is off turn it on and vice-versa:
    if (ledState == LOW) {
      ledState = HIGH;
    } else {
      ledState = LOW;
    }

    // set the LED with the ledState of the variable:
    digitalWrite(۱۳, ledState);
  }
}
```

این قطعه کد باعث می‌شود چراغ روی برد آردوینو هر نیم میلی ثانیه خاموش و روشن شود.

این تابع non-blocking است و برای اجرا شدن برنامه را متوقف نمی‌کند. بنابراین نسبت به تابع ()delay به صورت بهینه از سخت افزار ما استفاده می‌کند.

تابع ()micros

این تابع مانند تابع ()millis مدت زمان سپری شده از زمان شروع برنامه را نشان می‌دهد اما با واحد میکرو ثانیه. و نحوه استفاده از آن هم مانند تابع ()millis است.

بخش دوم:

در بخش دوم ال‌ای‌دی خود آردوینو چشمک زن شد و هر نیم ثانیه خاموش و روشن می‌شد. ما این کار را با استفاده از تابع ()millis انجام دادیم و این کار باعث می‌شد که سخت افزار ما نسبت به حالتی که از تابع ()delay استفاده کردیم درگیری کمتری داشته باشد.

بخش سوم:

در این بخش هر هشت ال‌ای‌دی متصل به آردوینو به صورت هماهنگ هر نیم ثانیه چشمک می‌زدند. یعنی همگی با هم روشن می‌شدند و نیم ثانیه بعد خاموش می‌شدند. و با توجه به این که در این بخش از تابع ()millis استاده کردیم می‌توانیم بگوییم که سخت افزار ما مانند زمانی که از تابع ()delay استفاده می‌کردیم بلاک نخواهد شد.

بخش چهارم:

در این بخش هشت LED به آردوئینو متصل و در صورتی که پوش باتن که پرت دیجیتال صفر متصل است را فعال کنیم LED ها به ترتیب با فاصله ۵۰ میلی ثانیه شروع به روشن شدن میکنند و هر وقت LED بعدی روشن شود LED قبل خاموش میشود

هدف اجرای یک برنامه خاموش و روشن شدن به ترتیب هشت LED در صورت فعال شدن پرت دیجیتال شماره صفر و اجرای این برنامه در شبیه ساز آردئینو در پروتئوس بوده است.

بخش پنجم

در آردوئینو مفهوم threads امکان زمانبندی اجرای task ها را موثر میکند.

یک thread به معنای یک دستور پردازشی است. دستوری که فقط یک هدف دارد و تنها یک کار انجام می دهد. پس از thread بحث multi threading به میان میاید که به این معنا است که همزمان چند دستور با کارها و اهداف مستقل از هم در پردازشگر، پردازش و اجرا می شود.

تنها پردازنده هایی از برنامه نویسی چند نخی پشتیبانی می کنند که دارای چند هسته درونی باشند. در برد آردوینو UNO یا NANO این امکان وجود ندارد چراکه پردازنده آن میکروکنترلر ATmega328 می باشد که تنها می تواند در هر بار یک دستور را اجرا کند. اما در میان آردوئینوها تنها Arduino DUE که دارای پردازشگر ARM است، این امکان را دارد.

کتابخانه‌ای به نام Scheduler مختص آردوینو DUE وجود دارد که به وسیله آن می‌توان چند تابع را به صورت همزمان در برنامه اجرا کرد. به عبارت دیگر برنامه شما می‌تواند دارای چند loop باشد که همگی بدون ایجاد وقفه در دیگری در حال پردازش می‌باشند. در واقع توسط این کتابخانه CPU به چند بخش تقسیم می‌شود و هر بخش وظیفه اجرای یک loop را بر عهده می‌گیرد.

اما در آردوئینوهای دیگر که امکان مالتی تردینگ وجود ندارد نیز با روشهایی میتوان تسکها را برنامه ریزی و از مفهوم thread استفاده کرد. به این صورت که هر ترد را بصورت یک تابع تعریف کرده و با استفاده از تابع ()millis تردها را زمانبندی کرد.

در این زمینه کتابخانه های مختلفی نوشته شده است که میتوان به ThreadHandler اشاره کرد که در آن میتوان یک کلاس MyThread نوشت و از Thread ارث بری کرد و ترد خود را به عنوان یک شی از این کلاس مقداردهی کرد.

کتابخانه مفید دیگر در این زمینه Seeed_Arduino_FreeRTOS است که امکان زمانبندی تسک ها را در آردوئینو به ما میدهد.

راه دیگر استفاده از Zerynth است که در اینصورت برنامه باید به زبان پایتون نوشته شود، فایل py. برنامه LED چشمک زن نوشته شده با استفاده از این روش در ضمیمه موجود است ، برای مطالعه بیشتر در این زمینه لینک در منابع آخر گزارش موجود است[7].

نتیجه گیری

ما در این مجموعه آزمایش روی تابع‌های مختلف برای ایجاد چراغ چشمک زن کار کردیم. و نتایج را هم در بخش مربوط به نحوه انجام آزمایش نوشتیم.

منابع:

[1] https://randomnerdtutorials.com/why-you-shouldnt-always-use-the-arduino-delay-function/
[2] https://github.com/adamb314/ThreadHandler/
[3] https://www.youtube.com/watch?v=oeP_NiajWME/
[4] https://wiki.seeedstudio.com/Software-FreeRTOS/
[5] https://digispark.ir/multi-thread-programming-arduino-due/
[6] https://www.hackster.io/luigifcerfeda/multiple-blinking-leds-at-different-rates-with-threads-using-viper-4aacd2/
[7] https://create.arduino.cc/projecthub/adamb314/how-to-run-57-hard-real-time-threads-on-an-arduino-uno-b8e742/