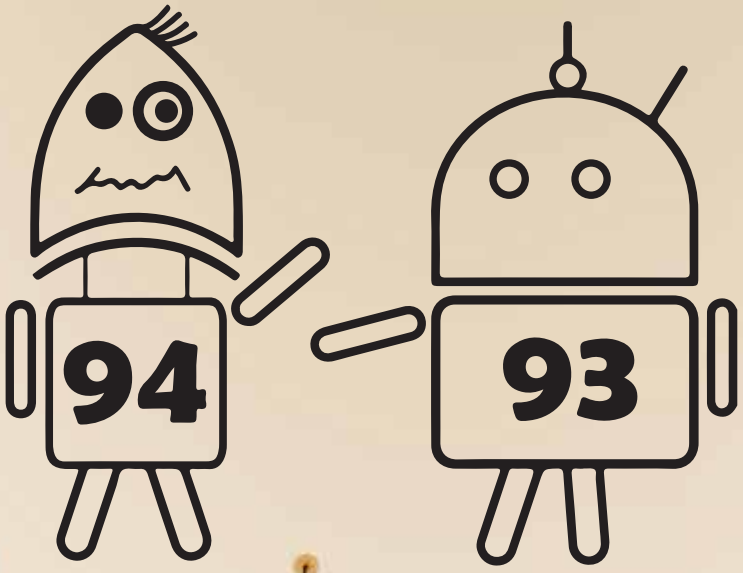
گزارش
نخستین دوره
مسابقات Gatuino



ششم
خرداد
۱۳۹۵

محتوی

خاستگاه

در یکی از روزهای پاییزی سال ۱۳۹۴، در کارگروه مسابقات علمی، از عدم برگزاری برنامه‌ها و مسابقات سخت‌افزاری توسط انجمن علمی انتقاد شده و پیشنهاد شد تا با برگزاری مسابقه‌ای سخت‌افزاری نشاط لازم در دانشجویان دانشکده ایجاد شود. مشکل پیش رو، آشنایی دیرهنگام دانشجویان با مباحث ویژه‌ی سخت‌افزار در درس‌هایی مانند DSD بود. این موضوع می‌توانست در جذب مخاطبان سال نخست و دوم که اتفاقاً برای حضور در مسابقات انگیزه‌ی بیشتری دارند، مشکل ایجاد کند.

مخاطب

با توجه به برگزاری مسابقات FPGA برای دانشجویانی که درس DSD را گذرانده‌اند و آنچه گفته شد. تصمیم بر آن شد تا مخاطب اصلی مسابقه دانشجویان ورودی و سال دوم باشند. بعلاوه با توجه به تفاوت سطح بسیار بین دانشجویان سال سوم و مخاطبان اصلی، برای جهت جلوگیری از ترس دانشجویان سال نخست و سال دوم از شرکت در مسابقه‌ای که مخاطب اصلی آن هستند، تصمیم بر آن شد تا از حضور دانشجویان سال سوم و بالاتر جلوگیری شود.

اهداف

با توجه به برنامه و مخاطبان آن هدف‌های زیر شناسایی شدند:

۱. ایجاد نشاط علمی و فوق برنامه در حوزه‌های سخت‌افزاری در دانشکده
۲. آشنایی ابتدایی دانشجویان جوان دانشکده با حوزه‌های سخت‌افزاری

زمان تقریبی

با توجه به فشردگی برنامه‌های علمی و فوق برنامه در فصل زمستان و این موضوع که دانشجویان سال نخست معمولاً در ترم دوم درس مدار منطقی را خواهند گذراند تصمیم بر آن شد تا مسابقه در فصل بهار برگزار شود.

بررسی اولیه

برای برگزاری مسابقه نیاز بود تا مباحث اصلی و نحوه‌ی برگزاری تعیین گردد. طی بررسی‌های انجام شده در نهایت به این نتیجه رسیدیم که برای جذابیت برنامه، بهتر است مسابقه در چند بخش و به صورت موازی برگزار شده و دانشجویان بتوانند بر حسب علاقه و به صورت پاره‌وقت در تمام قسمت‌ها شرکت کنند. در مشورتی که با دو تن از اساتید دانشکده داشتیم، این بخش‌ها مناسب به نظر رسیدند:

۱. تئوری
۲. ساخت مدار بر روی برد
۳. شبیه‌سازی در برنامه‌هایی مانند Proteus و Quartus
۴. ایجاد مدار با استفاده از زبان‌های توصیف سخت‌افزار
۵. برنامه‌نویسی و کنترل پورت‌ها برای دستگاه‌هایی همچون Raspberry Pi یا Arduino

بررسی دقیق بخش‌های مسابقه

در بررسی‌های گروه برگزاری مسابقه تصمیم گرفته شد تا بخش چهارم (ایجاد مدار با استفاده از زبان‌های توصیف سخت‌افزار) حذف شود. دلیل حذف این قسمت آشنایی برخی دانشجویان سال دوم و عدم آشنایی اغلب دانشجویان سال نخست و سال دوم با این قسمت بود. به عبارت دیگر در صورت ارائه سوال برای این قسمت سربار یادگیری هرچند اجمالی زبان‌های توصیف سخت‌افزار برای دانشجویان نا آشنا با این موضوع، رقابت در مسابقه را تحت تأثیر قرار می‌داد. بعلاوه مسابقات FPGA به اندازه کافی به این بخش رسیدگی می‌کند.

در بررسی‌های بعدی سه بخش نخست ادغام شده و قرار شد دانشجویان مسائل را پس از حل با استفاده از ابزارهای شبیه‌سازی پیاده‌سازی کنند و در نهایت با انتقال مدار ایجاد شده بر روی بردهای FPGA، هم با این بردها آشنا شده و هم به صورت فیزیکی از صحت مدار خود مطلع شوند.

بعلاوه با توجه به توانایی‌های دستگاه‌های Arduino تصمیم بر آن شد تا بخش آخر برنامه‌نویسی و کنترل پورت‌ها برای دستگاه‌های Arduino باشد. این بخش می‌تواند به نوعی بخش ساخت مدار بر روی بورد را هم در بر بگیرد.

در نهایت گزینه‌ای مبنی بر حل چالش‌های سخت‌افزاری (مطابق با تحقیقات تئوری) با استفاده از زبان‌های برنامه‌نویسی مطرح شد. به عنوان مثال توزیع taskها بر روی پردازنده‌های با توان‌ها و مصرف‌های متفاوت و پیش‌بینی miss بعدی در دسترسی به cache (برای پیش واکشی) مطرح شدند.

با این اوصاف بنا شد تا مسابقه در سه بخش موازی به این شرح برگزار شود:

۱. شبیه‌سازی برنامه‌نویسی چالش‌های سخت‌افزاری
۲. طراحی و پیاده‌سازی مدارهای منطقی با استفاده از نرم‌افزارهای شبیه‌سازی
۳. طراحی و پیاده‌سازی با استفاده از بوردهای Arduino

سوالات هر بخش در انتهای گزارش قرار گرفته‌است.

برنامه مسابقه

با توجه به اهداف و بخش‌های مسابقه برنامه به این شکل تعیین شد:

۱. (۳۰ دقیقه) پذیرش دانشجویان و گروه‌بندی در صورت عدم وجود گروه.
با توجه به چندقسمتی بودن مسابقه و حالت کارگاهی داشتن گروه‌ها باید ترجیحا کامل باشند.

۲. (۲۰+۲۰+۲۰ دقیقه) افتتاحیه - معرفی هر بخش و ارائه آموزش های لازم.
   I. معرفی مدارهای Arduino و آنچه لازم است انجام دهند.
   II. معرفی برنامه شبیه‌سازی و شیوه کار با آن.
   III. معرفی چالش (مسأله) سخت‌افزاری که باید آن را با برنامه‌نویسی حل کنند.

۳. (باقی زمان مسابقه) برگزاری مسابقه در سه بخش به صورت موازی به طوری که دانشجویان بتوانند در صورت انجام صحیح فعالیت در مدت زمان معقول (مثلا ۴۰ دقیقه) از سوالات امتیاز دریافت کرده و به سراغ سوالات دیگر یا قسمت‌های دیگر بروند.
   I. در قسمت Arduino اطلاعات بوردها و قطعات به صورت مکتوب در اختیار دانشجویان قرار بگیرد و در هنگام پیاده‌سازی برگزار کنندگان راهنمایی‌های لازم را انجام داده و در عیب یابی نیز کمک داشته باشند.
   II. در قسمت طراحی و پیاده‌سازی مدار، تیم‌های باید سوالات را طراحی و در محیط شبیه‌ساز پیاده‌سازی کنند. در نهایت شبیه‌سازی خود را در موارد خواسته شده بر روی بورد FPGA برده و تست کنند. با توجه به واسط کاربری سخت برنامه‌های شبیه‌سازی و تنظیمات خاص برای پروگرم کردن بوردهای FPGA لازم است برگزار کنندگان راهنمایی‌های لازم را بارها انجام دهند.

III. در این بخش دانشجویان باید کدهای خواسته شده را ایجاد کرده و برگزار کنندگان کد آنها را مورد سنجش قرار داده و بر اساس آن امتیاز دهند. بعلاوه خوب است برگزارکنندگان در رفع مشکلات دستوری کدها و ابهامات صورت سوال کمک نمایند تا تمرکز بر روی حل سوال باشد.

۴. (۶۰ دقیقه) اختتامیه

I. سخنرانی در مورد سخت‌افزار و آینده پیش رو (با توجه به این که مخاطبان برنامه دانشجویان سال نخست و سال دوم هستند این قسمت از اهمیت ویژه‌ای برخوردار است)

II. تقدیر از تیم‌های شرکت کننده و اهداء جوایز – با توجه به ماهیت مسابقه تصمیم بر آن شد تا جایزه هر یک از سه تیم برتر، یک بورد سخت‌افزاری (مانند Raspberry Pi و Beagleboard) باشد تا بتوانند فعالیت گروه خود را بر روی آن ادامه دهند. بعلاوه برخی اساتید با توجه به فعالیت دانشجویان در این مسابقه نمره امتیازی در نظر گرفتند.

توصیه‌هایی برای آیندگان

- توصیه می‌شود تغییرات مسابقه در راستای اهداف باشد. برای مثال گسترش مسابقه در حدی که دانشجویان سال سوم و بالاتر یا دانشجویان دیگر رشته‌ها را در بر گیرد توجیهی ندارد.
- توصیه می‌شود بررسی هدف «آشنایی ابتدایی دانشجویان جوان دانشکده با حوزه‌های سخت‌افزاری» در حد متعادل دنبال شود چرا که سخت، پیچیده، یا طولانی شدن مسابقه باعث کاهش مخاطبان و کاهش تأثیر مورد نظر بر روی دانشکده خواهد بود.
- توصیه می‌شود سوالات و فعالیت‌ها مخصوص دانشجویان سال نخست و دوم طراحی شود و علاوه بر بررسی دقیق نوع-فعالیت‌ها، در مورد قرار دادن آن‌ها در مسابقه با اساتید محترم مشورت شود.
- توصیه می‌شود تغییرات مسابقه باعث قطع ارتباط مستقیم و مستمر تیم برگزاری با دانشجویان در هنگام مسابقه نشود. بازخوردهای بسیار مثبتی از ارتباط مستقیم تیم برگزاری و آموزش‌ها و راهنمایی‌های رو در رو دریافت شده‌است.
- توصیه می‌شود در کنار و همراستا با اهداف اصلی هدف آشنایی دانشجویان با ابزارها و دستگاه‌های جدید (مانند Arduino و Raspberry Pi و Quartus و FPGA) نیز مد نظر قرار گرفته‌شود.

تشکر

در نهایت جا دارد از دکتر اجلالی برای حمایت، راهنمایی و مشورت در تمام مراحل، دکتر گودرزی برای مشورت در مورد مسابقه و بخش‌های آن، دکتر حسابی و دکتر سربازی‌آزاد برای حمایت معنوی و معرفی مسابقه به دانشجویان، دکتر اسدی برای تأمین بوردها و قطعات مورد نیاز برای برگزاری مسابقه، دکتر جهانگیر و دکتر صامتی برای تأمین جوایز مسابقه، و دکتر بیات برای معرفی مسابقه و سخنرانی اختتامیه در مورد سخت‌افزار و آینده آن تشکر نماییم.

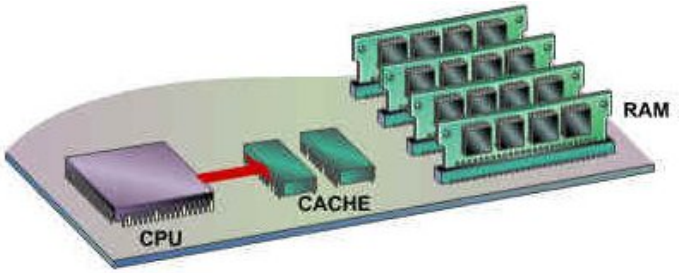
تیم برگزاری (ترتیب حروف الفبا)

- زهرا اسماعیل نژاد
- علی انصاری
- محمد بخشعلی‌پور
- ایمان جامی مقدم
- آیناز حاجی‌مرادلو
- فاطمه سادات حق‌پناه
- محمد رازقی
- یوسف سلیمانی
- طاها شاهرودی
- مهتا شفیعی ثابت
- امین کلانتر
- محمد کهزادی
- عرفان لقمانی
- امیرعلی معین‌فر
- مهدیه موحد راد
- سارا مهدی زاده شهری
- سید شراره میرزرگر
- آرمین وکیل

سوال شبیه‌سازی پیش‌واکشی

MISS DUINO

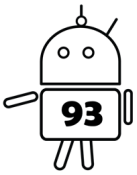
در سیستم‌های کامپیوتری برای بهره‌گیری هم‌زمان از حجم بالا و تأخیر کم، از چندین سطح حافظه استفاده می‌شود. در این سیستم‌ها معمولاً حافظه‌ی نزدیک‌تر به پردازنده (که Cache نامیده می‌شود) دارای حجم کم و سرعت بالا و حافظه‌ی دورتر از پردازنده (که RAM نامیده می‌شود) دارای حجم زیاد و سرعت پایین می‌باشد.



هر گاه که پردازنده درخواستی به حافظه داشته باشد، ابتدا آدرس مورد نظر را در Cache جستجو می‌کند، چنانچه آدرس مورد نظر در Cache موجود باشد، سریعاً به آن دسترسی پیدا می‌کند، در غیر این صورت می‌بایست به RAM مراجعه کرده و تأخیر بسیار زیادی را تحمل کند. چنانچه آدرس A که توسط پردازنده درخواست شده است، در Cache موجود نباشد، اصطلاحاً گفته می‌شود که آدرس A دچار فقدان[1] شده است.
یکی از راه‌های افزایش سرعت اجرای برنامه‌ها این است که فقدان‌های برنامه، پیش از وقوع، پیش‌بینی شده و به درون Cache آورده شوند. به این عمل پیش‌واکشی[2] گفته می‌شود. به عنوان مثال چنانچه آدرس‌های A و $A+d$ و $A+2\times d$ به ترتیب دچار فقدان شوند، می‌توان پیش‌بینی کرد که لحظاتی بعد آدرس $A+3\times d$ نیز توسط پردازنده درخواست شده و دچار فقدان خواهد شد. در این وضعیت برای افزایش سرعت اجرای برنامه بهتر است که آدرس $A+3\times d$ را پیش‌واکشی کرده و به درون Cache بیاوریم. در این سوال شما می‌بایست برنامه‌ای بنویسید که در هر لحظه آدرس یک فقدان را دریافت کند و آدرس فقدان بعدی را (بر اساس هر اطلاعاتی که از گذشته نگه می‌دارید) پیش‌بینی کند. در هر گام، چنانچه نتوانید فقدان بعدی را پیش‌بینی کنید (به طور عمده در اوایل اجرای برنامه)، می‌بایست 1- را به عنوان پیش‌بینی فقدان بعدی چاپ کنید.

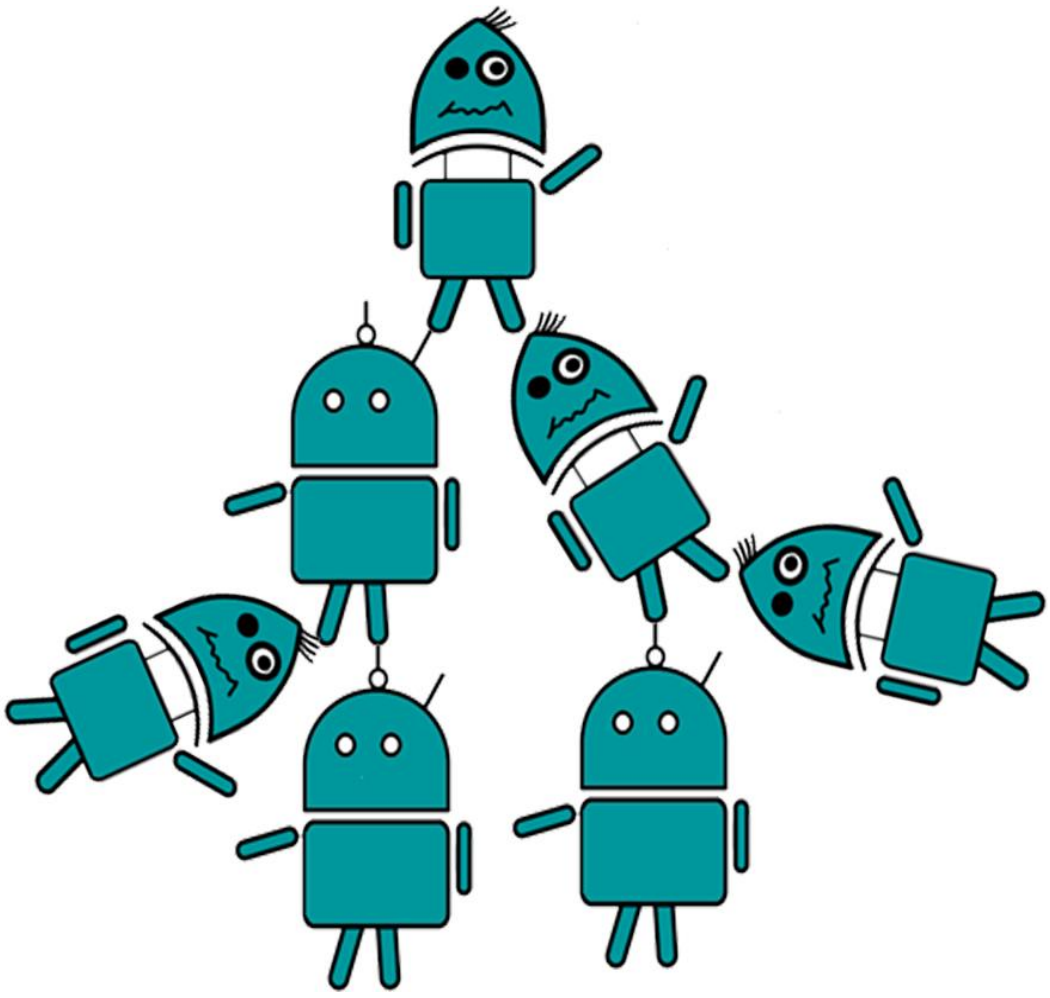
گام	ورودی نمونه (اعداد ۶۴ بیتی دسیمال)	خروجی احتمالی برنامه‌ی شما (اعداد ۶۴ بیتی دسیمال)
۱	123453	-1
۲	123454	-1
۳	123455	123456
۴	123456	123457

ملاک امتیازدهی در این سوال تعداد پیش‌بینی‌های درست می‌باشد. چنانچه دو برنامه تعداد پیش‌بینی درست یکسانی داشته باشند، برنامه‌ای که تعداد پیش‌بینی اشتباه کم‌تری داشته باشد، امتیاز بالاتری دریافت خواهد کرد.



[1] Miss
[2] Prefetching

سوالات طراحی و پیاده‌سازی مدارها



فهرست:

دست گرمی

نندوینو و خوردوینو می‌خواهد برای بررسی خویشتن و یادگیری pin assignment دو بیت A و B را از کلیدهای FPGA دریافت کرده و خروجی NAND و XOR آن را روی دو LED مجزا نمایش دهند.

به کمک برگزار کنندگان این کار را انجام دهید :)

اعداد اول

خوردوینو که یکی از بی‌حافظه ترین گیت‌های دانشکده است، در حفظ کردن اعداد اول دچار مشکل شده‌است. شما و نندوینو قرار است تا با استفاده از یک مدار با تعداد کمی گیت دو ورودی AND و NAND و OR و NOR و XOR و XNOR و NOT به او کمک کنید. پس دست به کار شده و مداری طراحی کنید که یک عدد ۴ بیتی دریافت کرده و خروجی ۱ دهد اگر عدد مربوطه اول باشد.

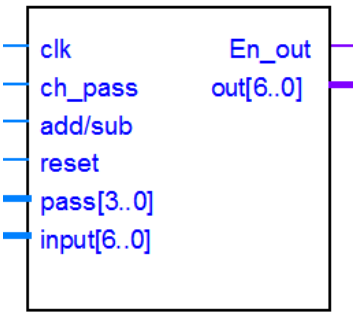
(حداکثر امتیاز به گروهی تعلق می‌گیرد که مدار با کمترین تعداد گیت را طراحی کند – پس امتیاز شما ممکن است تغییر کند.)

(تست مدار توسط waveform یا بر روی FPGA انجام شود)

مقدمه: کیف پول دیجیتالی

پول رایج در کشور Gatuino سکه‌های سیلیکونی است و آن‌ها در خریدهای خود از سکه استفاده می‌کنند. اما از آن‌جایی که حمل و نقل سکه برای مردم سخت است، مسئولین کشور به فکر ایجاد پول الکترونیکی افتاده اند. آن‌ها می‌دانند که انتقال پول الکترونیک، نگهداری و جا به جایی آن خیلی راحت تر از سکه است. با توجه به اینکه Xorduino تنها کسی در کشور است که با مدارات الکترونیکی آشنایی دارد از او خواسته‌اند که یک کیف پول الکترونیکی بسازد. اما متأسفانه او کارکردن با مدارات منطقی و دیجیتالی را به خوبی نمی‌داند. او از شما خواسته که در ساخت این کیف پول به او کمک کنید.

نمای شماتیک کیف پول به صورت زیر خواهد بود:



برای طراحی باید به نکات زیر توجه کنید:

- کیف پول طراحی شده یک مدار ترتیبی است که با استفاده از ورودی clk پالس ساعت مورد نیاز آن داده می‌شود.
- برای امنیت کیف پول، یک رمز ۴ بیتی برای آن در نظر گرفته شده است که به صورت پیش فرض مقدار 1010 است و با فعال کردن ورودی ch_pass می‌توان مقدار ورودی ۴ بیتی pass را به عنوان رمز جدید تعیین کرد.
- مقدار پول ذخیره شده در کیف پول می‌تواند از صفر تا ۱۲۷ تغییر کند(نمی تواند منفی باشد). این مقدار به صورت پیش‌فرض صفر است و در هر لحظه از طریق خروجی ۷ بیتی out قابل مشاهده است.
- نحوه کار با کیف پول به این صورت است که در صورت وارد کردن صحیح رمز ۴ بیتی، خروجی En_out یک شده و می‌توان یکی از عملیات‌های اضافه کردن پول یا برداشت پول را انجام داد. در صورت اشتباه وارد کردن رمز، خروجی En_out صفر شده و نمی‌توان هیچ عملیاتی روی مقدار پول ذخیره شده انجام داد.
- عملیات اضافه کردن پول به این صورت است که باید مقدار پول مورد نظر را در ورودی ۸ بیتی input قرار داده و ورودی add/sub مقدار صفر باشد تا در لبه‌ی بعدی کلاک، این مقدار با پول موجود جمع شده و حاصل جمع در کیف پول ذخیره شود.

- عملیات برداشت پول نیز به این صورت است که باید مقدار پول مورد نظر را در ورودی ۸ بیتی input قرار داده و ورودی add/sub مقدار یک باشد تا در لبه‌ی بعدی کلاک، این مقدار از پول موجود کم شده و حاصل در کیف پول ذخیره شود.
- با یک کردن ورودی reset باید رمز کیف پول و مقدار پول ذخیره شده به حالت پیش‌فرض برگردانده شود.

باید توجه کنید که:

- در طراحی باید از امکانات شماتیک نرم افزار کوارتوس(quartus) استفاده کنید.
- فقط می‌توانید از گیت‌های پایه and, or, not, xor, xnor و D flip-flop استفاده کنید.
- می‌توانید از آی‌سی جمع‌کننده 7483 در طراحی خود استفاده کنید. اما در صورت پیاده‌سازی شماتیک جمع‌کننده، امتیاز بیشتری کسب خواهید کرد.

کیف پول دیجیتالی - فاز اول: تشخیص درستی رمز ورودی

ورودی: رشته ۴ بیتی به عنوان رمز وارد شده

خروجی: En_out تک بیت یک نشان‌دهنده صحت و تک بیت صفر نشان‌دهنده نادرستی رمز ورودی است.

رمز اصلی کیف پول را ۱۰۱۰ در نظر بگیرید.

کیف پول دیجیتالی - فاز دوم: تغییر موجودی کیف پول

ورودی: رشته ۷ بیتی به عنوان ورودی – تک بیت add/sub

خروجی: موجودی کیف پول در ۷ بیت

کیف پول دیجیتالی - فاز سوم: تغییر رمز کیف پول

ورودی: رشته ۴ بیتی به عنوان رمز جدید – تک بیت ch_pass

مسابقه سه نفره

در یک مسابقه‌ی سه نفره، خوردوینو و نندوینو و اوردوینو قرار است با یکدیگر مسابقه دهند. مجری یک سوال را مطرح کرده و آنها با فشردن یک دکمه نوبت خود را دریافت می‌کنند. مجری از ما خواسته است تا مداری طراحی کنیم که علاوه بر یک کلاک سریع به عنوان ورودی سه دکمه ورودی بگیرد و شماره (عددی بین ۱ تا ۳) اولین نفری که دکمه را فشار داد به عنوان نوبت بر روی 7 Segment نمایش دهد.

برای تبدیل عدد دودویی به 7 Segment می‌توانید از 7447 IC که در کوارتوس موجود است استفاده نمایید.

استفاده از تمام گیت‌های پایه و Flip Flopها مجاز است.

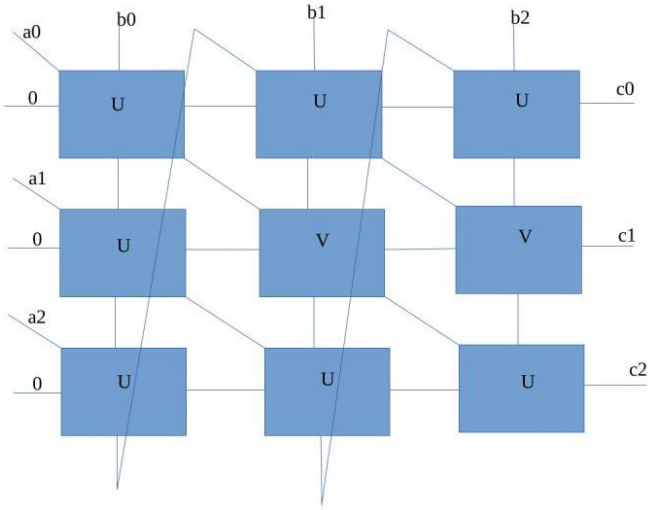
در نهایت طرح خود را بر روی FPGA بریزید.

ضرب چند جمله‌ای‌ها

فرض کنید A(x) و B(x) دو سه جمله‌ای (مثلا به فرم $a_0 + a_1x + a_2x^2$) باشند که ضرایب آن‌ها را به ترتیب با a0, a1, a2, b0, b1, b2 باشد. میخواهیم حاصلضرب این دو سه جمله‌ای را محاسبه کرده و آن را به صورت یک چند جمله‌ای با نام C(x) نمایش دهیم که ضرایب آن نیز c0, c1, c2 میباشد. با آنجام عملیات ضرب و ساده‌سازی ریاضی میتوان به رابطه‌ی ماتریسی زیر رسید:

$$\begin{matrix} c0 & a0 & a2 & a1 & b0 \\ (c1) = (a1 & a0+a2 & a2+a1)^* & (b1) \\ c2 & a2 & a1 & a0+a2 & b2 \end{matrix}$$

حال میخواهیم دو ماژول(سلول) با نام‌های V و U را پیاده‌سازی کرده تا با معماری زیر بتوانیم ضرایب چندجمله‌ای C(x) را در ۳ کلاک به دست اوریم. معماری‌ای که قرار است از این دو سلول استفاده کند به شکل زیر است:



در این بخش قرار است که ابتدا شما با کمک گیت‌های and, xor و فلیپ فلاپ و گیت‌های پایه‌ای دیگری که میشناسید سلول‌های V و U را در نرم‌افزار کوارتوس پیاده‌سازی کرده و سپس با نمونه(instance) گرفتن از هر کدام و کنار هم قرار دادن ان‌ها مطابق شکل بالا معماری لازم برای به دست آوردن ضرایب را پیاده‌سازی کنید. توجه کنید که سلول‌های مورد نظر از کمترین تعداد گیت استفاده کنید(به عنوان راهنمایی نیز به این مسأله دقت کنید که در در ماژول از ۳ خروجی، ۲ خروجی که به صورت سری نیستند به استفاده از فلیپ فلاپ دقیقاً قبل از خروج نیازمندند.)

Call of DES. Modern FPGAography

به تازگی ماموران مخفی در دستگاه دشمن، موفق شدند شواهدی به دست آورند که اسناد دشمن با الگوریتم DES رمز می‌شوند. بنابراین برای افشای اسناد دشمن، از Xorduino و Nanduino خواسته شده است که الگوریتمی برای رمز کردن و رمزگشایی به روش DES ارایه دهند. Xorduino به دنبال پیاده سازی نرم افزاری این الگوریتم است اما Nanduino معتقد است که پیاده سازی سخت افزاری روی FPGA بهتر و سریع تر جواب می‌دهد. آیا می‌توانید به این دو نفر کمک کنید تا بهترین و سریع ترین راه ممکن برای افشای اسناد دشمن را به ماموران مخفی ارایه دهند؟

گام اول: در این مرحله شما باید یک رشته ورودی را از مسوول مسابقه گرفته و خروجی مورد نظر به او ارایه دهید. رشته ورودی یک عدد ۸ بایتی است همراه با یک عملیات که رمز کردن یا رمزگشایی از رشته مورد نظر است. خروجی نیز رشته ۸ بیتی حاصل از اعمال عملیات بر روی رشته ی ورودی است.

گام دوم: در این گام شما باید به مقایسه ی زمان مورد نیاز برای رمز کردن رشته ای به طول ۲۷ میلیون کاراکتر بر روی PC و FPGA بپردازید. کلاک FPGA به روی ۲۷ مگاهرتز تنظیم شده است و در هر کلاک یک رشته ۶۴ بیتی را رمز می‌کند. خروجی مورد نظر، تخمین شما از زمان مورد نیاز برای آماده شدن خروجی در FPGA و PC و نیز نسبت سرعت عملکرد این دو است. آیا حق با Xorduino است یا با Nanduino؟

گام سوم: با توجه به مشاهدات خود در قسمت قبلی، دلیل عملکرد ضعیف تر PC را می‌توانید توضیح دهید؟ چرا یک پردازنده چند گیگاهرتزی اینتل بروی PC با پیاده سازی زبان سریعی چون C عملکرد ضعیف تری در مقایسه با FPGA با کلاک ۲۷ مگاهرتز میدهد؟ آیا می توانید زمان سربار کار با فایل در پیاده سازی نرم افزاری را محاسبه کنید؟ سربار کار سیستم عامل چطور؟ چه عواملی دیگری به نظر شما می‌تواند روی سرعت خروجی پیاده سازی نرم افزاری تاثیر گذار بوده باشد؟

گام چهارم: در آزمایش قسمت ۲، FPGA توسط تولید کننده‌ی کلاک ۲۷مگاهرتزی به کار گرفته شده بود. FPGA می‌تواند روی فرکانس کاری ۵۰ مگاهرتزی نیز کار کند. در اینصورت سرعت FPGA چند برابر PC می شود؟ در صورت صرف نظر از محدودیت کلاک بورد FPGA، به نظر شما چه عواملی حداکثر سرعت به کارگیری بورد را محدود می‌کند؟

گام پنجم: با دیدی که از پیاده سازی سخت افزاری روی FPGA و نیز پیاده سازی نرم افزاری بروی PC به دست آوردید، در هر دو سطح چه راهکارهایی برای افزایش سرعت یا گرفتن خروجی بیشتر در زمان یکسان پیشنهاد می‌دهید؟

راهنمای کار با بورد FPGA

در این سوال، FPGA به صورت برنامه ریزی شده به شما تحویل داده می شود. لذا اکیدا از خاموش کردن آن خود داری کنید. در صورت خاموش کردن بورد، برنامه ریزی مجددا آن با اعمال کسر امتیاز صورت خواهد گرفت.

برای کار با بورد، به نحوه عملکرد PIN های مختلف روی بورد توجه کنید:

SW0 – SW7 (in): از این PIN ها برای مقداردهی رشته ورودی استفاده می شود. رشته ی ورودی DES، ۶۴ بیتی است. لذا برای وارد کردن کامل یک رشته، باید با استفاده از این ورودی و نیز ورودی location نسبت به مقداردهی رشته ورودی اقدام کنید.

SW17-SW15 (location): این ۳ بیت، محل قرار گیری رشته‌ی 8 بیتی فوق را مشخص می‌کند. به عنوان مثال اگر مقدار SW17:SW15 = 3 باشد . SW7:SW0 = 00100110 باشد به این معنا است که قرار است در بایت سوم رشته ی ورودی مقدار ۰۰۱۰۰۱۱۰ ریخته شود.

SW14 (clock): کلاک است. ریخته شدن ورودی ۸ بیتی در location و نیز نمایش رشته‌ی خروجی در چراغ های LED با اعمال لبه ی بالای کلاک صورت می گیرد.

SW10 (reset): با یک کردن این PIN، بورد برای اعمال گام دوم آماده میشود. پیش از اعمال SW10 برای انجام گام دوم، reser را یکبار ۱ کرده و دوباره ۰ کنید، سپس SW10 را یک کنید.

SW12: اگر یک باید بورد در حالت خواندن رشته ی ورودی است. یعنی مقدار موجود در in با توجه به مقدار location در لبه کلاک در رشته ی ۶۴ بیتی ورودی قرار خواهد گرفت. اگر ۰ باشد، مقدار بایت location رشته خروجی در لبه ی بالای کلاک در LEDهای LEDR0 تا LEDR7 به نمایش در می‌آید.

SW11 (DES mode): این ورودی مشخص می کند که ماژول در حالت رمزکننده است یا رمزگشا. این PIN در حالت ۱، رمزگشا و در حالت ۰، رمز کننده است.

SW10 (change_mode): با یک کردن این PIN، بورد به حالت مورد نیاز برای گام دوم می رود و ورودی کلاک به جای SW14 از مولد کلاک ۲۷مگاهرتزی خود FPGA اعمال خواهد شد.LEDR0 پس از اتمام عملیات، روشن خواهد شد.

LEDR0 – LEDR7: خروجی الگوریتم DES توسط مقدار ورودی location در این LEDها قرار می‌گیرد.

LEDG0: در گام دوم بعد از اینکه FPGA تمام رشته به طول ۲۷میلیون کاراکتر را رمز کرد، این LED را روشن می کند.

راهنمای کار با نرم افزار C:

ابتدا فایل اجرایی برنامه در پوشه DES در Desktop قرار داده شده است.

Cmd را باز کنید.

به مسیر پوشه DES بروید.

با اعمال دستور زیر می توانید فایل sample.txt را رمز کرده و خروجی اش را در sample.enc بریزید.

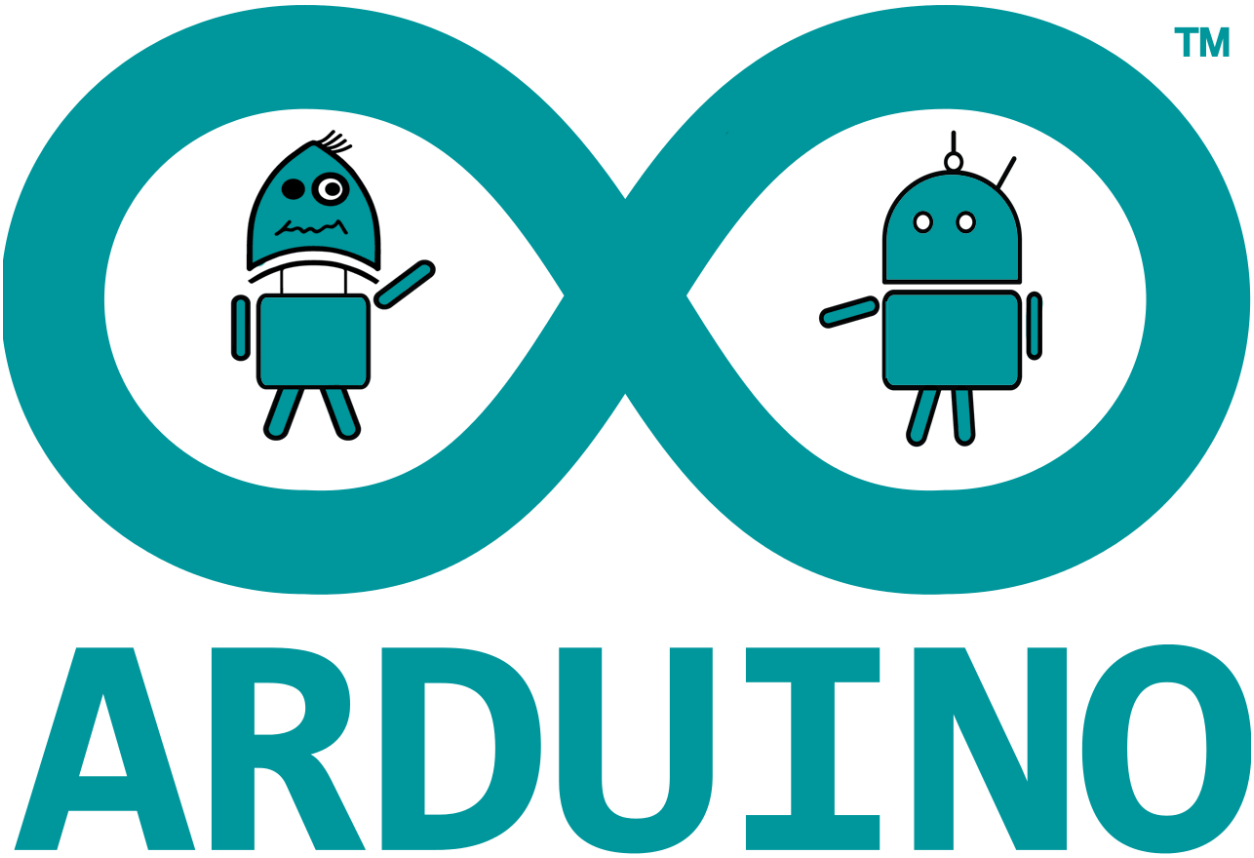
```
run_des.o -e keyfile.key sample.txt sample.enc
```

با اعمال دستور زیر می‌توانید فایل رمز شده را رمزگشایی کنید.

```
run_des.o –d keyfile.key sample.enc sample_decrypted.txt
```

در صورت بروز هرگونه ابهام در مورد سوالات یا پیاده سازی از راهنمایان کمک بجویید.

سوالات طراحی و پیاده سازی مدار آردوینو



فهرست

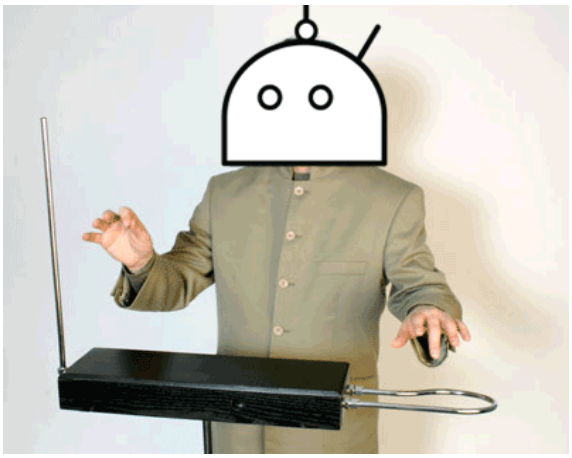
مدارها:

سنسورها:

مراجع:

ترمین[1](*)

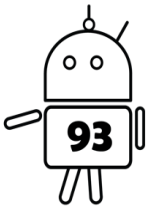
ترمین نوعی ابزار موسیقی است که در سال ۱۹۱۹ توسط لئون ترمین فیزیکدان روسی اختراع شد. این ساز بدون لمس شدن نواخته میشود! صدای ترمین توسط دو نوسان ساز به وجود می آید که با هم به ارتعاش در می آیند. یکی از نوسان سازها در فرکانسی با طیفی بالاتر از حد شنوایی انسان عمل میکند و فرکانسهای نوسان ساز دیگر با ورود دست به میدان مغناطیسی تغییر میکند. ضرباهنگ فرکانس که تفاوت میان فرکانسهای دو نوسان ساز است، صدایی است که میشنویم. در این قسمت شما به نندوینو و خوردوینو که تصمیم به یادگیری این ساز گرفته اند کمک میکنید تا یک ترمین ساده را به کمک فتوسل پیاده سازی کنند.



نندوینو در حال نواختن ترمین

برای این کار سنسور فتوسل و المان پیزو را به آردوینو متصل کنید و با حرکت دست خود بالای فتوسل صداهای مختلف تولید کنید!

راهنمایی:مقادیر خوانده شده از فتوسل بین ۰ تا ۷۰۰ میباشند. برای تولید صدا از فرکانس های بین ۲۰۰ تا ۳۷۰ هرتز استفاده کنید.



[1] Theremin

تشخیص ضربه رمزی(***)

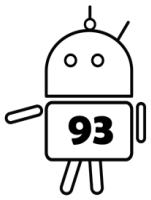
احتمالا فیلم لئون[2] را تماشا کردید. در صحنه ای از فیلم ماتیلدا(همون دختره) و لئون بین خودشان رمزی تعریف می کنند که در صورتیکه به درب به این شکل ضربه وارد کردند فرد دیگر در را باز کند. نندوینو و خوردوینو بعد از تماشای این فیلم جو گیر شده و تصمیم گرفتند مداری بسازند که این کار را انجام دهد. خب حالا شما قرار است به آن ها کمک کنید مداری پیاده کنند که با تشخیص ضربه رمزی درب را باز کند با این تفاوت که نیازی به باز کردن درب نیست و صرفا کافی است یک LED را روشن کنید ☺

جک نیکلسون در حال تست مدار تشخیص ضربه(پشت درب خانه نندوینو و خوردوینو)

میتوانید به شکل پیش فرض یک دنباله ضربه تعریف کنید اما این دنباله میبایست قابل تغییر باشد بدین شکل که ابتدا کاربر با فرستادن حرف d یا p به آردوینو مشخص کند که آیا میخواهد مدار را پروگرام کند یا از آن برای باز کردن درب(روشن کردن LED در اینجا) استفاده کند. برای تشخیص ضربه از المان پیزو استفاده کنید.

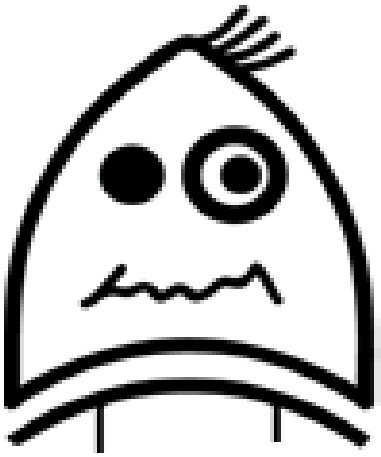
راهنمایی:از توابع مربوط به زمان استفاده کنید .

[2] Leon: The Professional



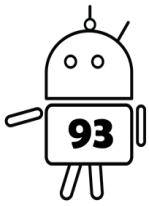
تشخیص حرکت(*)

مسئولیت نگهداری جایزه های مسابقه سخت افزاری gatuino به خوردوینو سپرده شده است اما خوردوینو خسته از فشار میانترم ها و پروژه ها تصمیم گرفته تا از این فرصت استفاده کرده و دور از چشم مسئولین برگزاری بخوابد اما از طرف دیگر اصلا نمیخواهد که آن ها را بین شرکت کننده ها رها کند و کسی بدون بستن یک مدار خفن و باحال جایزه ای بگیرد (!). در این قسمت با استفاده از سنسور PIR و سنسور Piezo element و یا زنگ اخبار[3] مداری طراحی کنید که با نزدیک شدن هریک از شرکت کننده ها به جایزه آلارمی را به صدا در آورد و خوردوینو را قبل از اینکه مسئولین متوجه شوند از خواب بیدار کند . ☺

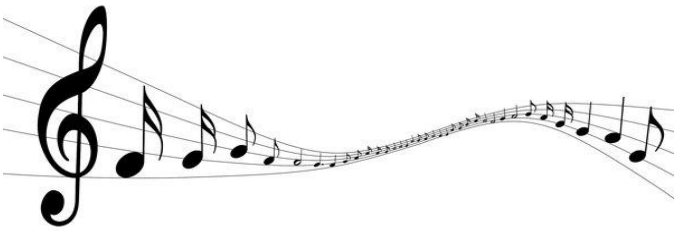


خستگی از قیافه خوردوینو مشخص است

[3] buzzer



پخش کننده ملودی (**)



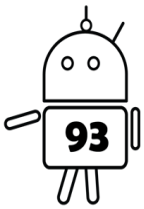
خوردوینو تصمیم به یادگیری پیانو گرفته است اما از آنجا که هزینه آن زیاد است و خوردوینو در حال حاضر بیکار است(آمار بیکاری گیت ها اخیرا رشد قابل توجه ای داشته است)،وی تصمیم گرفت از تکنولوژی ارزان قیمت استفاده کند.

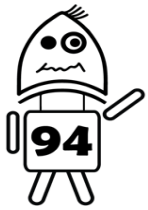
در این قسمت میبایست یک پیانوی ساده به کمک کیبورد خود برای خوردوینو درست کنید. مدار بدین شکل عمل میکند که با گرفتن یکی از حروف A تا L از طریق ارتباط سریال یک ملودی خاص توسط المان پیزو و یا زنگ اخبار[4] پخش میکند.

فرکانس لازم برای تولید نُت های مختلف در ادامه آمده است. به انتخاب خودتان ۹ تا از آن ها را برای پخش ملودی استفاده کنید.

[4] buzzer

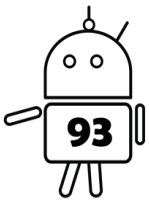
Note	Hz	Note	Hz	Note	Hz	Note	Hz	Note	Hz	Note	Hz	Note	Hz
C1	32.7	C2	65.4	C3	130.8	C4	261.6	C5	523.3	C6	1046.5	C7	2093.0
C#1	34.6	C#2	69.3	C#3	138.6	C#4	277.2	C#5	554.4	C#6	1108.7	C#7	2217.5
D1	36.7	D2	73.4	D3	146.8	D4	293.7	D5	587.3	D6	1174.7	D7	2349.3
D#1	38.9	D#2	77.8	D#3	155.6	D#4	311.1	D#5	622.3	D#6	1244.5	D#7	2489.0
E1	41.2	E2	82.4	E3	164.8	E4	329.6	E5	659.3	E6	1318.5	E7	2637.0
F1	43.7	F2	87.3	F3	174.6	F4	349.2	F5	698.5	F6	1396.9	F7	2793.8
F#1	46.2	F#2	92.5	F#3	185.0	F#4	370.0	F#5	740.0	F#6	1480.0	F#7	2960.0
G1	49.0	G2	98.0	G3	196.0	G4	392.0	G5	784.0	G6	1568.0	G7	3136.0
G#1	51.9	G#2	103.8	G#3	207.7	G#4	415.3	G#5	830.6	G#6	1661.2	G#7	3322.4
A1	55.0	A2	110.0	A3	220.0	A4	440.0	A5	880.0	A6	1760.0	A7	3520.0
A#1	58.3	A#2	116.5	A#3	233.1	A#4	466.2	A#5	932.3	A#6	1864.7	A#7	3729.3
B1	61.7	B2	123.5	B3	246.9	B4	493.9	B5	987.8	B6	1975.5	B7	3951.1





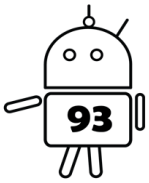
رقص نور(*)

خوردوینو و نندوینو تصمیم به برگزاری یک مهمانی و دعوت دوستان خود(اَندوینو و اُردوینو) گرفتند و برای افزایش هیجان مهمانی قرار است یک سیستم رقص نور پیاده سازی کنند. به آنها کمک کنید که مداری طراحی کنند که حداقل چهار LED با پخش آهنگ و متناسب با آن روشن و خاموش شوند. برای تشخیص صدا از ماژول میکروفون (KY-037) استفاده کنید. (در صورت استفاده از LED بیشتر امتیاز بیشتری کسب خواهید کرد)



لامپ هوشمند(*)

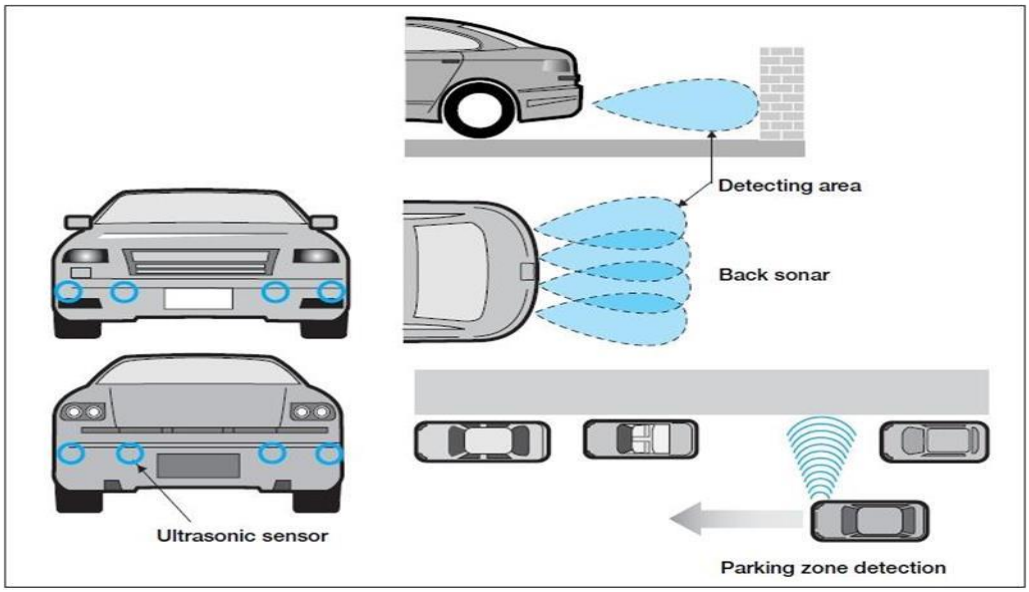
نندوینو اخیرا لامپ هوشمندی خریده است که با تاریک شدن هوا به شکل اتوماتیک روشن میشود. از آنجا که خوردوینو نمیخواهد جلوی نندوینو کم بیاورد و با توجه به مساله بیکاری اخیری که برای وی پیش آمده و قادر به خرید لوازم گران قیمت نیست، تصمیم گرفته که خودش این مدار را طراحی و پیاده سازی کند. به او کمک کنید تا با استفاده از سنسور فتوسل مداری طراحی کند که یک LED بسته میزان تاریک بودن هوا، پرنور تر و یا کم نور تر شود.



تشخیص فاصله(*)

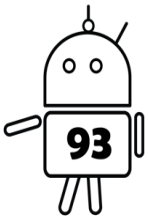
خوردوینو و نندوینو اخیرا دچار مشکل شدند و دعوایی بین آنها صورت گرفته. خوردوینو معتقد است نندوینو زیادی به مسائل به شکل صفر و یکی نگاه می کند و برای همین تصمیم گرفته مدتی از وی دور بماند. به کمک سنسور تشخیص فاصله(اولتراسونیک) و یک المان پیزو و یا زنگ اخبار[5] مداری طراحی کنید که اگر فاصله‌ی نندوینو از خوردوینو از حد معینی(۷۰ سانتی متر) کمتر بود آلارمی به صدا درآید و خوردوینو خبردار شود.

[5] buzzer



از سنسورهای اولتراسونیک در سیستم حسگر پارک ماشین استفاده میشود

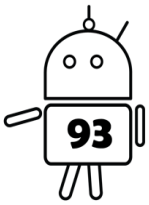
برای مشاهده‌ی عملکرد سنسور، باید جسم کاملا در راستای مقابل سنسور باشد.



ساعت باینری(**)

از آنجا که خوردوینو و نندوینو فقط صفر و یک میفهمند، قادر به خواندن ساعت نیستند(دلیل اینکه همیشه تاخیر دارند نیز یحتمل به همین موضوع برمیگردد) به کمک ماژول RTC و با استفاده از تعداد مناسب LED، ساعت کامپیوتر خود را به شکل باینری نمایش دهید تا این دو گیت متوجه شوند ساعت چند است. توجه کنید که ساعتی که نمایش میدهید باید شامل ثانیه، دقیقه و ساعت باشد.

* برای پیاده‌سازی این سوال کتابخانه‌ی مربوط به RTC (RTClib. h) برای راحتی کار ضمیمه شده است.

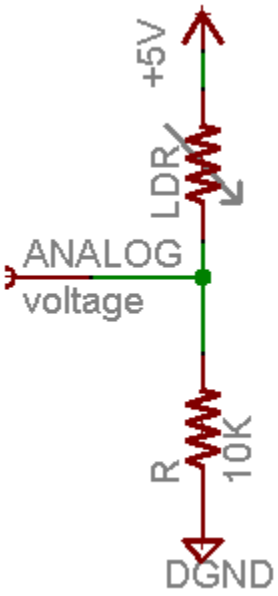


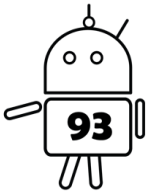
فتوسل

فتوسل نوعی مقاومت است که براساس میزان نوری که به آن تابیده میشود، مقاومت الکتریکی آن تغییر میکند. به عبارت دیگر از خود فوتورسانندگی نشان میدهد. فتوسل از یک نیمه رسانای دارای مقاومت بالا تشکیل شده است و در صورتی که نور تابیده شده بر روی آن از بسامد کافی برخوردار باشد، فوتون های جذب شده توسط نیمه رسانا به الکترون های وابسته اش انرژی کافی برای جهش به نوار رسانش را میدهند. الکترون آزاد به دست آمده و حفره های حاصل جریان الکتریکی را هدایت میکنند و به این شکل مقاومت الکتریکی کاهش میابد.

از فتوسل ها در منازل برای روشن کردن لامپ ها با تاریک شدن هوا استفاده میشود.

برای اتصال این سنسور به آردوینو نیاز به مدار تقسیم ولتاژ مانند شکل زیر خواهید داشت.

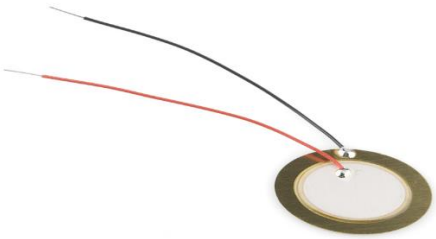




المان پیزو

المان پیزو[6] برای تشخیص ضربه و یا به عنوان زنگ اخبار[7] برای تولید صدا استفاده میشود. سیم قرمز این المان به ورودی آردوینو و سیم مشکی آن به زمین متصل میشود. همچنین بهتر است سیم قرمز بوسیله یک مقاومت یک مگا اهمی به زمین نیز متصل شود[8].

هنگامی که به عنوان سنسور تشخیص ضربه از این المان استفاده میشود خروجی آن عددی بین ۰ تا ۱۰۲۳ خواهد بود که در صورتی که این عدد از آستانه[9]ای بیشتر باشد به معنای ضربه وارد شدن به آن است. هر چقدر که عدد آستانه بیشتر باشد سنسور حساسیت کمتری خواهد داشت. عدد آستانه معمولا بین ۱ تا ۱۰ انتخاب میشود.



نمونه ای از المان پیزو

ماژول میکروفون KY-037

از این ماژول برای تشخیص صدا استفاده میشود و شامل سه پایه اتصال به زمین، اتصال به منبع تغذیه (۵ ولت) و پایه خروجی میباشد که بر اساس بلندی صدا مقدار آن تغییر میکند.

سنسور تشخیص فاصله ultrasonic

این سنسور شامل ۵ پین زمین، vcc، echo، out و تریگر است که در این آزمایش نیازی به استفاده از پین out نیست. دو پین اصلی برای کار این سنسور، پین‌های تریگر و اکو هستند. برای این که سنسور شروع به خواندن

[6] Piezo Element
[7] Buzzer
[8] pull down
[9] Threshold

کند باید پالسی حداقل با طول ۱۰ میکروثانیه به پایه‌ی تریگر آن اعمال شود(high)، از آن پس سنسور وارد حالت listen می‌شود تا پاسخی دریافت کند. اگر جسمی مقابل این سنسور در فاصله‌ای حدود ۲ سانتی‌متر تا ۳ متر باشد، بسته به فاصله‌ای که این جسم از سنسور دارد پالسی در پایه‌ی اکو دریافت میشود(high). (طول این پالس متناسب با فاصله تا سنسور است).

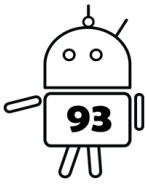
برای اینکه سنسور را به آردینو وصل کنید کافی است دو پین از میکرو را انتخاب کرده و آن دو را به پین‌های تریگر و اکو متصل کنید و از این طریق با سنسور ارتباط برقرار کنید.

حسگر مادون قرمز PIR

سنسور PIR که مخفف passive infrared sensor است, نسبت به تابش اشعه مادون قرمز حساس بوده و از آن برای تشخیص حرکت در دوربین های مدار بسته و سایر مدار های الکتریکی استفاده میشود. این حسگر معمولا ۳ پایه دارد و از دو قطعه کریستالی تشکیل شده است, که براثر تابش اشعه مادون قرمز روی آنها شارژ سطحی ایجاد میشود و با تغییر میزان تابش , میزان شارژ هم تغییر میکند.

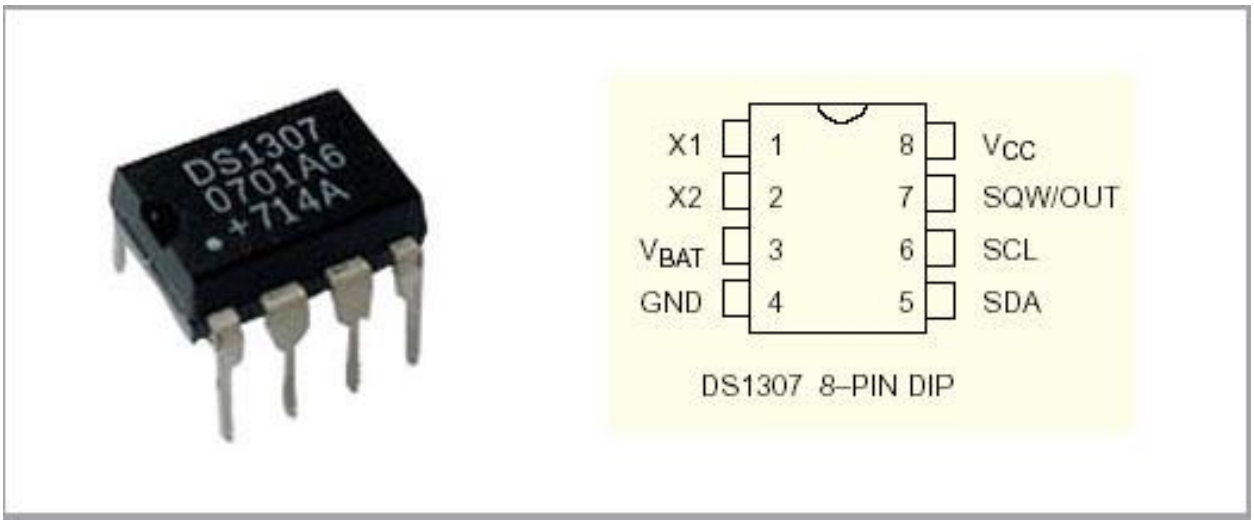
همانطور که انتظار میرود , پایه های GND و VCC این قطعه به پین های GND و VCC برد آردوئینو متصل میشود و پین وسط با یک مقاومت (حدودا ۴۷۰ اهمی) به یک پین ورودی دیجیتال متصل میگردد.

برای استفاده از سنسور باید ابتدا آن را کالیبره کرد به این معنا که به مدت ۶۰ تا ۷۰ ثانیه در مثابل صفحه کریستالی حرکتی انجام نشود و کاملا سنوسر بی حرکت بماند. برای حساسیت بیشتر این سنسور پیشنهاد میشود اطراف آن را با یک صفحه کاغذی استوانه ای شکل بپوشانید.



ماژول RTC، ds1307

RTC که مخفف real time clock است در واقع تراشه‌ای با باتری جداست که به میکرو این امکان را میدهد تا از زمان حال همواره آگاه باشد(در شرایط قطع برق میکرو با پروگرام شدن مجدد و . . .). باتری این تراشه این امکان را به شما میدهد تا زمان سیستم را تا ۵ سال یا بیشتر نگهدارید !



پین‌های X1 و X2 فرکانس ۳۲. ۷۶۸ کیلوهرتز مربوط به اسیلاتور را تامین میکند(در این آزمایش نیازی به استفاده از اسیلاتور نیست). SCL و SDA به ترتیب ورودی سریال کلاک و دیتا را مشخص میکنند. Vbat مربوط به ولتاژ باتری مورد استفاده در RTC است. برای مشاهده‌ی موج خروجی نیز میتوانید از پایه‌ی SQW/OUT استفاده کنید. این ماژول برای برقراری ارتباط از پروتکل I2C که یک پروتکل برای انتقال سریال اطلاعات است استفاده میکند که پین‌های نظیر آن در آردینو، پین‌های سریال A4 و A5 است.

قابل ذکر است که خروجی RTC به صورت BCD می‌باشد. اطلاعات مربوط به پین‌های نظیر در آردینو نیز در جدول زیر زیر قابل مشاهده است.

DS1307 PIN	Arduino PIN	Use
6	A5	SCL
5	A4	SDA
4	Arduino Ground	Ground
8	Arduino +5V	+5V

توابع کاربردی

```
pinMode(pinNumber, pinMode)
```

تعیین نوع پین به عنوان ورودی یا خروجی (INPUT | OUTPUT)

```
digitalWrite(pin, value)
```

نوشتن مقدار HIGH یا LOW روی پین دیجیتال

```
digitalRead(pin)
```

خواندن مقدار پین دیجیتال(خروجی = HIGH | LOW)

```
analogRead(pin)
```

خواندن مقدار پین آنالوگ(خروجی بین ۰-۱۰۲۳)

```
analogWrite(pin, value)
```

نوشتن مقدار روی پین آنالوگ(تولید موج PWM)

```
tone(pin, frequency, duration)
```

یک موج به اندازه فرکانس گفته شده روی پین تولید میکند. مدت زمان تولید موج را نیز میتوان مشخص کرد(دلخواه)

```
noTone(pin)
```

در صورتی که مدت زمان برای تابع tone() تعریف نشده باشد با صدا زدن این تابع تولید موج متوقف میشود.

```
millis()
```

مدت زمان گذشته از لحظه شروع به کار آردوینو را به میلی ثانیه به عنوان خروجی میدهد.

```
delay(ms)
```

ایجاد تاخیر به میلی ثانیه

```
delayMicroseconds(us)
```

ایجاد تاخیر به میکرو ثانیه

```
map(value, fromLow, fromHigh, toLow, toHigh)
```

مقدار value که بین fromLow تا fromHigh میباشد را به عددی بین toLow تا toHigh مپ میکند و به عنوان خروجی برمیگرداند.

```
Serial. begin(buad)

Serial. print()
```

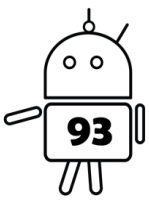
Serial. read()

Serial. readBytes(buffer, length)

برای برقراری ارتباط سریال بین آردوینو و کامپیوتر از این سه تابع استفاده میشود. (توسط کابل یو اس بی). ابتدا تابع begin() را با مقدار 9600 به عنوان مثال صدا بزنید. سپس برای گرفتن اطلاعات از کامپیوتر و یا نمایش آنها روی کامپیوتر از توابع read() و print() استفاده کنید.

Wire. begin(address)

این تابع برای initiate کردن کتابخانه‌ی wire استفاده میشود و تعیین می‌کند که از باس I2C به عنوان master استفاده شود یا slave. آرگومان ۷ بیتی آدرس برای مشخص کردن آدرس slave استفاده میشود. بنابراین اگر این آدرس مشخص نشده باشد یعنی I2C در مد master کار می‌کند. این تابع تنها یک بار فراخوانی میشود.



پین های آردوینو نانو

ARDUINO NANO Version 3.0 Pin Layout