# راه اندازی و استفاده از دکمه

دکمه از پایه‌ای‌ترین قطعات الکترونیکی است که یادگیری آن مفاهیم پایه‌ای ضروری و مفیدی را در بردارد. در این بخش می‌خواهیم با دکمه و نحوه‌ی راه‌اندازی آن آشنا شویم. همین طور با روش های مختلف بستن دکمه در یک مدار الکترونیکی و نویز و رفع نویز آشنا خواهیم شد.

# نحوه عملکرد دکمه

همان طور که در شکل می‌بینید دکمه قطعه‌ای مکانیکی است که با فشردن آن، ارتباط بین دو پایه‌ی آن برقرار می‌شود یعنی جریان در این مسیر برقرار می‌شود. اما اتفاق دیگری که می‌ا‌فتد و برای ما از دید الکترونیکی اهمیت دارد، هم‌پتانسیل‌شدن دو سر کلید است. فرض کنیم مدار طوری است که با هم‌پتانسیل‌شدن دو سر کلید، پتانسیل یکی از سرها با حالت قبلی‌اش متفاوت شود. در این وضعیت، می‌توانیم با اندازه‌گیری پیوسته پتانسیل آن نقطه متوجه تغییر در وضعیت مدار شویم. اما حالا دو سوال مطرح می‌شود: یک اینکه مدار را چگونه ببندیم تا این وضعیت فراهم شود و دو اینکه چگونه از پتانسیل آن نقطه با خبر شویم؟



مدار را به زودی در همین بخش خواهیم آموخت. همچنین آردوینو برای سوال دوم قابلیت مناسبی دارد. زیرا آردوینو می‌تواند پتانسیل الکتریکی هر نقطه از مدار را برای ما آشکار کند که این توانایی به دو صورت آنالوگ و دیجیتال است. آنالوگ یعنی اندازه‌گیری پتانسیل به صورت پیوسته و دیجیتال یعنی این اندازه‌گیری به صورت گسسته و صفر و یکی. در این بخش دیجیتال را بررسی می‌کنیم و در بخش آنالوگ به بررسی خواندن مقادیر به صورت آنالوگ خواهیم پرداخت.

# دیجیتال

در مدار منطقی دو وضعیت منطقی صفر و یک وجود دارد. این دو وضعیت را می‌توان به هر دو حالت متضاد در دنیای فیزیکی همانند کرد، مثلا وجود یا نبود جریان در مدار و پر یا خالی بودن لیوان. اما در مدارات دیجیتال این دسته‌بندی بر مبنای ولتاژ صورت می‌گیرد. این مدارها عموما بین صفر ولت (Gnd) تا ولتاژ حداکثری 5 ولت (Vcc) یا در برخی موارد 3.3 ولت کار می‌کنند. بر همین مبنا، وضعیت نقطه‌ای را که دارای حداکثر ولتاژ است، یک، True یا High می‌گیریم و نقطه‌ای از مدار را که ولتاژ صفر ولت دارد، صفر، False یا LOW می‌گوییم.

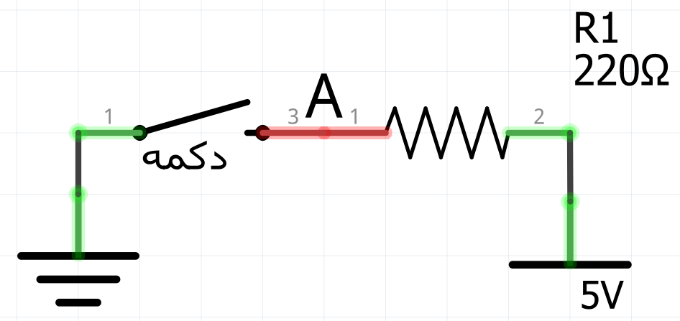
این گونه از متغیر را در آردوینو با نماد bool یا boolean نشان می‌دهیم که می‌تواند 0 یا 1 ، True یا False و HIGH یا LOW باشد.

### مدار الکتریکی دکمه

کاربرد اولیه و بسیار سادۀ مفهوم دیجیتال، راه‌انداختن دکمه است. برای اینکار احتیاج به مدار الکتریکی داریم که در زیر انواع و نکات آن توضیح داده می‌شود.

## مدار با مقاومت Pull-up

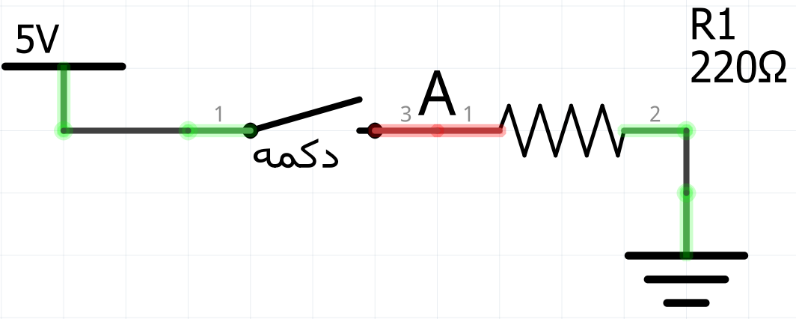
مدار زیر را در نظر بگیرید که در آن مقاومتی مستقیما به Vcc وصل است. به مقاومتی که به Vcc مدار وصل باشد، مقاومت Pull-up می‌گویند. قبل از فشردن کلید، مدار وصل نیست یعنی جریانی در مدار وجود ندارد. به همین دلیل تمام نقاط متصل به Vcc، پتانسیل یکسان 5 ولت را دارند که این شامل نقطه‌ی A نیز می‌شود. وقتی کلید را فشار می‌دهیم، مدار متصل و جریان در آن برقرار می‌شود. به دلیل وجود جریان، پتانسیل دو سر مقاومت طبق قانون اهم متفاوت می‌شود. پس پتانسیل A دیگر 5 ولت نیست. اما اگر دقت کنیم، می‌بینیم این نقطه در حالت کلید فشرده مستقیما به زمین مدار متصل است، پس پتانسیل آن صفر است.



در تمام بردهای آردوینو، پین‌هایی با عنوان دیجیتال وجود دارند که یکی از وظایف آن‌ها تشخیص حالت منطقی نقاط مدار است که با اتصال پین به هر نقطه و چند دستور نرم‌افزاری می‌توان این کار را انجام داد. به این پین‌ها پین‌های دیجیتال می‌گویند. با اتصال نقطه‌ی A به یکی از اینها و با استفاده از دستوری که به زودی خواهیم آموخت، حالت‌های A را ردیابی می‌کنیم.

## مدار با مقاومت Pull-down

اکنون به مدار زیر توجه کنید. در این مدار، مقاومت به جای وصل بودن به Vcc به زمین (Gnd) متصل است. به این مقاومت، مقاومت Pull-down می‌گویند. با کمی تحلیل مثل مدار قبل متوجه می‌شویم عملکرد این دو مدار شبیه یکدیگر است، با این تفاوت که در حالت مدار باز وضعیت منطقی A، یک و در حالت مدار متصل وضعیت منطقی آن صفر است.



## اهمیت وجود مقاومت

تا اینجا توانستیم به خواسته‌ی خود یعنی پیاده‌سازی صفر و یک منطقی در مدار برسیم. اما سوالی که پیش می‌آید این است که دلیل وجود مقاومت‌ها چیست؟ به بیان دیگر، اگر نباشند چه اتفاقی برای مدار می‌افتد؟ مشکلی که به وجود خواهد آمد، معین نبودن وضعیت منطقی نقطه‌ی A خواهد بود.

مدار دوم را در نظر بگیرید و فرض کنید مقاومتی نیست. در حالت مدار باز، مشکلی وجود ندارد و وضعیت منطقی A یک است. اما اگر کلید وصل شود، این نقطه مستقیما به هر دو پتانسیل 5 ولت و صفر ولت متصل خواهد بود یعنی نمی‌توان برای آن وضعیت ثابتی در نظر گرفت. از سوی دیگر، مشکل بزرگتری که ایجاد خواهد شد، جاری شدن جریان بسیار زیاد به دلیل اتصال کوتاه Vcc و Gnd خواهد بود که به شدت به مدار آسیب فیزیکی می‌زند که باید از این نوع مدار به شدت پرهیز کرد. همین شرایط برای مدار یک نیز امکان‌پذیر است.

## digitalRead()

دستور digitalRead() با گرفتن یک پین دیجیتال مقدار منطقی آن پین را به ما برمی‌گرداند. این دستور همان دستوری است که پاسخ سوال دوم مطرح‌شده در ابتدای بخش را می‌دهد. این دستور به ما کمک می‌کند که حالت A را در مدارهای Pull-up و Pull-down شناسایی کنیم.

دستورهای مورد نیاز عبارتند از تعریف pinMode آن پین به روی حالت INPUT و استفاده از دستور digitalRead() برای خواندن وضعیت. نحوه استفاده از آن را در زیر می‌بینید.



حال می‌توانیم کدی بنویسیم که وضعیت دکمه را به ما در سریال مانیتور نشان دهد.



## نویز

یکی از مشکلات اساسی دکمه و به طور کلی هر گونه وسیله‌ی مکانیکی که در الکترونیک استفاده می‌شود، نویز است. نویز یعنی اثرات ناخواسته‌ای که از بیرون یا درون سیستم روی وضعیت یا عملکرد سیستم تاثیر می‌گذارند.

نویز این گونه دکمه‌ها از پدیده‌ی bounce که به معنی نوسان ورودی هنگام زدن دکمه است نتیجه می‌شود. این پدیده موجب می‌شود میکرو اینطور فکر کند که دکمه چندبار فشرده شده‌است، مانند نمودار زیر. دلیل ایجاد این مشکل نیز به واسطه خطاهای ساختار مکانیکی دکمه‌ها است و برای رفع آن دو راهکار وجود دارد: سخت‌افزاری و نرم‌افزاری.

نمودار

### نویزگیری نرم‌افزاری

می‌توان با انداختن تاخیری کوچک در خواندن دیجیتال، از بازه زمانی bounce اجتناب کرد. کد زیر این کار را انجام می‌دهد.



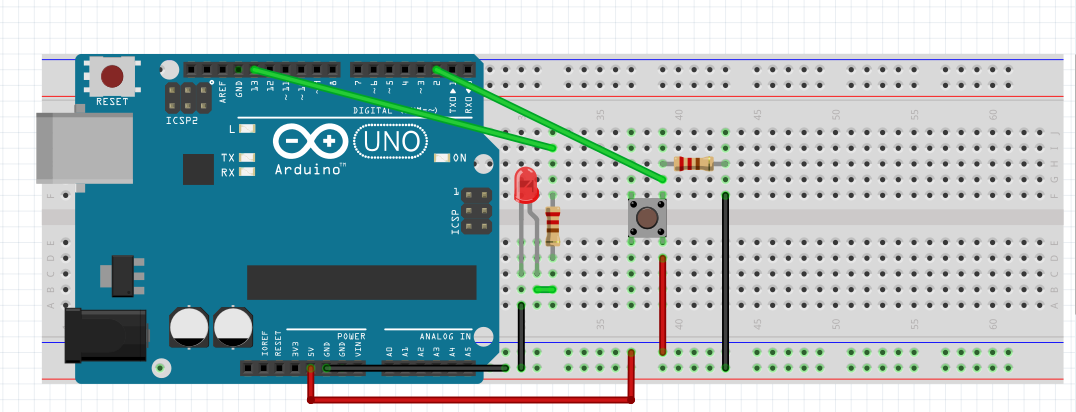
### نویزگیری سخت‌افزاری

مدار زیر نمونه‌ای ساده از این راه حل است. ایده‌ی اجرایی استفاده از خازن به عنوان فیلتر نویزگیر است. برای اطلاعات بیشتر به بخش پیش‌نیازهای الکتریکی مراجعه کنید. کد آن همانند کد شماره 1 است.

مدار

**LED فشاری**

حالا می‌خواهیم با ترکیب ورودی دیجیتال که در این بخش آموختیم با خروجی دیجیتال (ال ای دی) مداری کاربردی بسازیم. ایده این است که LED هنگامی که دکمه فشرده می‌شود روشن شود و در غیر این صورت خاموش باشد. مدار و کد را در زیر می‌بینید. منطق کد بسیار ساده است و برای نویزگیری از روش نرم‌افزاری استفاده شده‌است.





**LED وضعیتی**

اکنون تمرین را کمی متفاوت می‌کنیم. می‌خواهیم با فشردن دکمه، وضعیت کلید نسبت به حالت قبلش تغییر کند و تا زمانی که دکمه دوباره فشرده نشده‌است، ثابت بماند. مدار همانند قبل است و تفاوت در کد هست.

نکته‌ی مهم این کدنویسی، درک متغیرهای موجود در سیستم است. می‌توان الگوریتم‌های متفاوتی را نوشت که اکثرا به دلیل جامع نبودن به شکست می‌خورند. (امتحان کنید!) اما با درک حالت‌های مختلف سیستم می‌توان به الگوریتم زیر رسید:

سه متغیر وضعیت سیستم را تعیین می‌کنند: حالت LED که خروجی سیستم است و باید با فشردن دکمه به نقیضش تبدیل شود یعنی اگر HIGH است به LOW تبدیل شود و برعکس پس متغیری boolean است. دیگر وضعیت کنونی دکمه است که باید با مقایسه با وضعیت قبلی دکمه LED را تغییر دهد، پس دو متغیر وضعیت کنونی و وضعیت قبلی دکمه را نیز داریم. به این ترتیب کد ما به صورت زیر در می‌آید:

