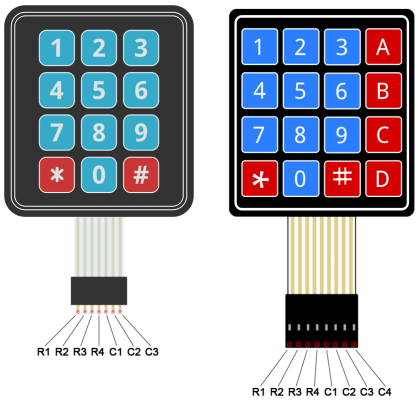
پیش‌گزارش آزمایش ۲:

۱. کدهای برنامه‌ریزی برد به پیوست ضمیمه شده است.

۲. انواع Keypadهای ماتریسی:

یک دسته از Keypadهای ماتریسی به شکل زیر هستند. به پین‌های خروجی این Keypadها ولتاژ High متصل می‌شود. در صورت فشرده شدن شدن یک کلید شماره‌ی j ,i جریان از پین سطر i به پین ستون j متصل می‌شود. برای تشخیص این که کدام کلید فشرده شده، به طور پیوسته کلیدها scan می‌شوند تا تشخیص داده شود که بین کدام دو پین جریان برقرار شده.



دسته‌ی دیگر Keypadها، Keypadهای خازنی هستند. در این نوع صفحه‌کلیدها، در صورت فشرده شدن یک کلید، خازن آن کلید شارژ شده و باعث ایجاد جریان می‌شود.

۳. یک کلید از دو صفحه‌ی رسانا تشکیل شده است. در زمانی که این دو صفحه به هم متصل باشد، جریان برقرار و در غیر این صورت جریان قطع خواهد بود. اما به جز این دو حالت، یک حالت دیگر هم ممکن است. زمانی که دو صفحه‌ی جدا از هم به مقدار کافی به یک دیگر نزدیک شوند، برای هوای بین آن‌ها، پدیده‌ی شکست الکتریکی اتفاق می‌افتد. این به معنی است که مولکول‌های هوا می‌توانند در زمان‌های بسیار بسیار کوتاه در نقش یک رسانا عمل کنند. به این پدیده‌ی برقرار شدن جریان به صورت نوسانی Bounce گفته می‌شود. این جریان نوسانی ممکن است به قطعات مدار آسیب وارد کند.
برای جلوگیری از این پدیده، یک خازن را به صورت موازی با کلید می‌بندیم. به این ترتیب حتی اگر پدیده‌ی Bounce اتفاق بیفتد، جریان نوسانی تنها صرف شارژ شدن خازن می‌شود و آسیبی به دیگر قطعات نمی‌رساند.

۴. توابع مورد نیاز کتاب‌خانه‌ی Keypad:

-()Keypad: Constructor کلاس Keypad. این تابع با گرفتن نام کلیدها، شماره‌ی پین‌های سطرها و ستون‌ها، و تعداد سطرها و ستون‌ها، یک شئ از کلاس Keypad می‌سازد.

-()getKey: کاراکتر کلید فشرده شده را برمی‌گرداند.

-()getKeys:یک آرایه از کارکتر کلیدهای فشرده شده را برمی‌گرداند.

-()waitForKey: این تابع به صورت Blocking منتظر فشرده شدن کلید می‌ماند. به این معنی که تا زمانی که کلیدی فشرده نشود، برنامه به خط بعدی نمی‌رود. همچنین در صورت فشرده شدن کلید، کاراکتر آن کلید برگردانده می‌شود.

-()getState: این تابع state هر کدام از کلیدهای Keypad را برمی‌گرداند. هر کلید می‌تواند در یکی از ۴ حالت Idle, Pressed, Released, Hold باشد.

-()keyStateChanged: این تابع در state که مقدار یک کلید تغییر کند، مقدار True و در غیر این صورت مقدار False برمی‌گرداند.

۵. نحوه و کاربرد ارتباطات سریال در Arduino:

ارتباطات سریال در Arduino از طریق پین‌های RX, TX و تحت منطق TTL اتفاق می‌افتد. از سریال برای ارتباط برد Arduino با کامپیوتر و دیگر دستگاه‌ها استفاده می‌شود. تمام بردهای Arduino حداقل یک پورت سریال (UART یا USART) دارند. برای استفاده از این پورت‌ها می‌توان از IDE خود Arduino استفاده کرد. به این ترتیب که از نوار بالا، گزینه‌ی Serial Monitor در tools را انتخاب می‌کنیم.

۶.تعریف و نحوه‌ی کار با تابع‌های ارتباطات سریال:

-()begin: نرخ انتقال داده را برحسب بیت بر ثانیه مشخص می‌کند.

-()end: ارتباط سریال را می‌بندد. به این ترتیب می‌توان از پین‌های RX, TX به عنوان GPIO استفاده کرد.

-()find: از serial buffer داده را تا زمانی می‌خواند که داده‌ی آرگومان را پیدا کند. در صورت پیدا شدن داده‌ی ورودی در buffer، مقدار True و در غیر این صورت مقدار False برمی‌گرداند.

-()parseInt:داده را تا زمانی که Integer بعدی را تشخیص دهد می‌خواند. در صورت تشخیص ندادن و منقضی شدن Timer، Terminate می‌شود.

-()println: داده را به صورت متن ASCII را روی سریال خروجی می‌نویسد. همچنین در پایان متن از \n می‌فرستد.

-()read:داده‌ی ورودی را دریافت می‌کند.

-()readStringUntil: کاراکترهای ورودی را از serial buffer دریافت و به string تبدیل می‌کند. در صورت منقضی شدن Timer، Terminate می‌شود.

-()write: داده‌ی باینری را می‌فرستد.