# Keypad

The keypad used in this experiment has twelve buttons representing numbers 0 to 9, and symbols * and #, as on a typical telephone keypad configuration. Its buttons and electrical wiring configurations are shown in Figure 1 and Figure 2. You will also need to add pullup or pull-down resistors to always generate a valid input at the CPU input pins. You are advised to read the application note "Implementing An Ultralow-Power Keypad Interface with MSP430" (posted by prof. Jean-Samuel in Experiment 2 directory). You are **not required** to implement the low power features for this lab, therefore, you may remove the diodes presented in the application note and keep the CPU active at all times, but the application note will detail the procedure of interfacing to a keypad.

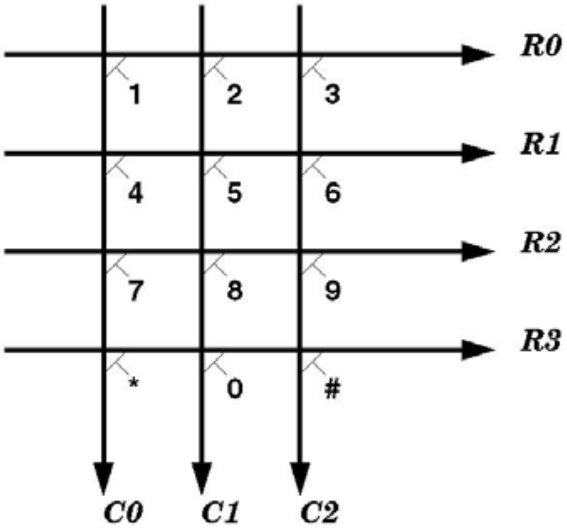


**Figure 1 Keypad Matrix Wiring**

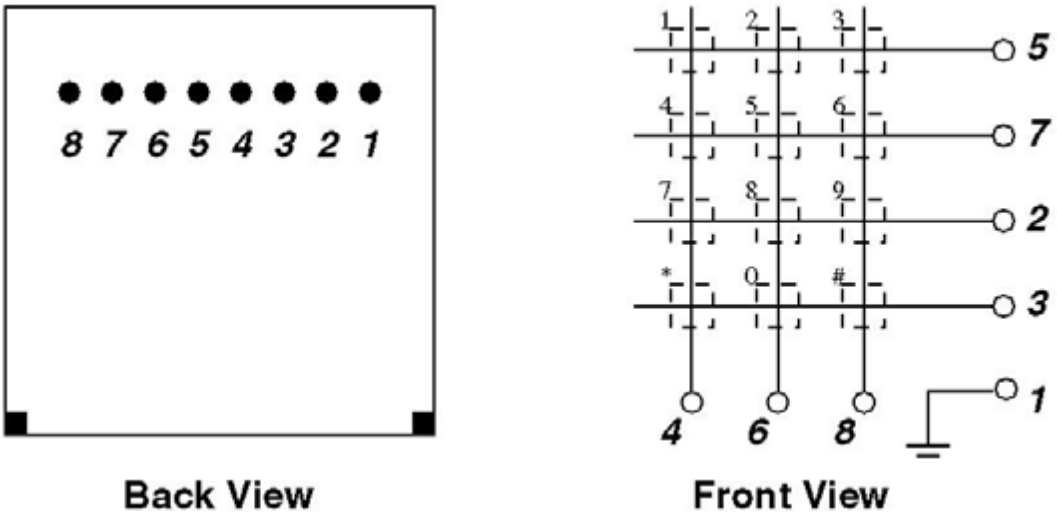


**Figure 2 Keypad Pinout Configuration**

The keypad has seven wires emanating from it; three column wires from C0 to C2, and four row wires from R0 to R3. Each of the twelve keys is lying on a particular intersection (Figure 3). The intersecting wires are normally disconnected. When any key is pressed, a column wire and a row wire will be electrically connected. For example, depressing of button '6' will cause wires C2 and R1 to be connected. All other wires will remain unaffected. The pin-out diagram of the keypad is illustrated in Figure 4.

Unfortunately, mechanical switches are not ideal in that they do not generate a crisp 1 or 0 when they are pressed or released. At the fast sampling speed of the controller, a seemingly brisk and firm switching action becomes comparatively slow. In fact, during a switching action, switch contacts oscillate ("bounce") between the connected and disconnected states after initial contact or separation before it settles. This random process typically lasts about 2-20ms. Figure 5 illustrates the switch bounce behavior when a key is pressed and released.

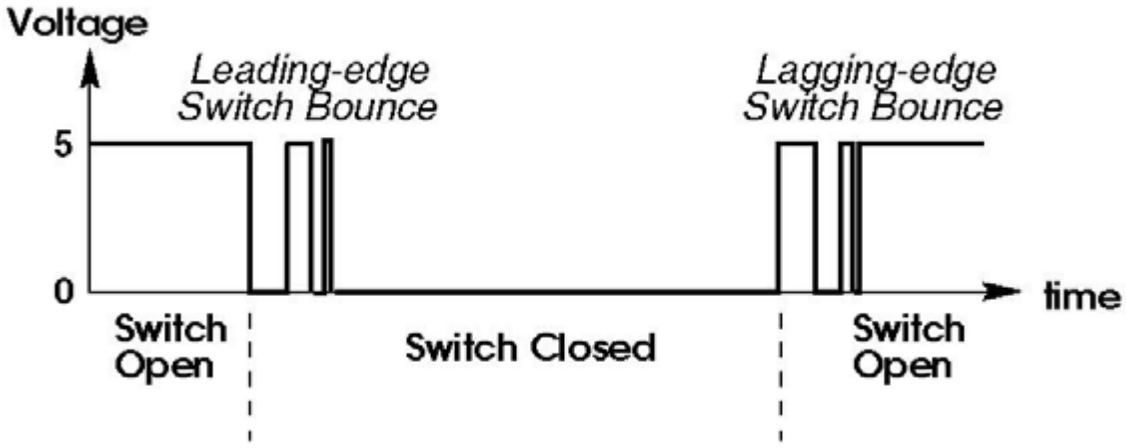


**Fig. 5 Switch Bounce**

Therefore, in determining when a button is pressed, the MCU must ensure that a firm physical connection has established and the voltage level has been stabilized. Waiting about 10 ms after the first connection or separation is detected, and checking again to confirm if the same button has been depressed is a typical approach to this problem. This process is called *debouncing*.

Good Luck !