MECHANICAL KEYPAD REPLACEMENT WITH CAPACITIVE TOUCH

By Meenakshi Sundaram, Applications Engineer Senior, Cypress Semiconductor Corp.

Gone is the era of mechanical push buttons and membrane switches as more and more of these interfaces are being replaced by innovative capacitive touch sensing technology. This new technology is not only fancier and easier to use but also more robust. Most designs will require only minor modifications to accommodate capacitive touch buttons. This in turn dictates the designers to come up with a ready-to-use, robust and a low touch drop-in replacement for those mechanical buttons.

With simple and easy to use capacitive touch designs that exist today, replacing a button with a touch sensor is straightforward. The sensor detects the presence of a finger and toggles an output pin high or low to mimic a mechanical button's ON/OFF state. However, replacement just doesn't end here. There are special types of mechanical user interfaces which will not work with a 1:1 input-output, including mechanical keypads. Although mechanical keypad interfaces are simple, their replacement does require more work than just replacing individual buttons. This article will cover widely used methods used to replace mechanical keypads with capacitive touch sensors.

Mechanical keypads: Underlying Basics

Conventional mechanical keypads use individual buttons laid out in a matrix fashion with buttons grouped into rows and columns. A key press is found by driving either the row or the column lines (scan lines) and checking the other (read lines) for any continuity (press). This concept is categorized into 2 types:

1. Polling-based
2. Interrupt-based

Polling-based:

This method is usually used by an independent key scan controller which continuously polls the scan lines and checks the read lines for any continuity/press. The read lines are usually pulled up to supply (Vcc) or logic '1' through a resistor, and scan lines are polled one at a time by driving the line being polled to '0' while keeping all other lines at '1'. Whenever there is a button press, a scan-to-read line short occurs where the '0' driven on the scan line travels to the read line – this is inferred as a button press (as shown in Figure 1). On detection of a press, the controller reports the event to the host by means of some communication interface like I2C or SPI to offload the host from performing the key scanning itself. As such, this method is inefficient and impractical for a single-chip system scenario.

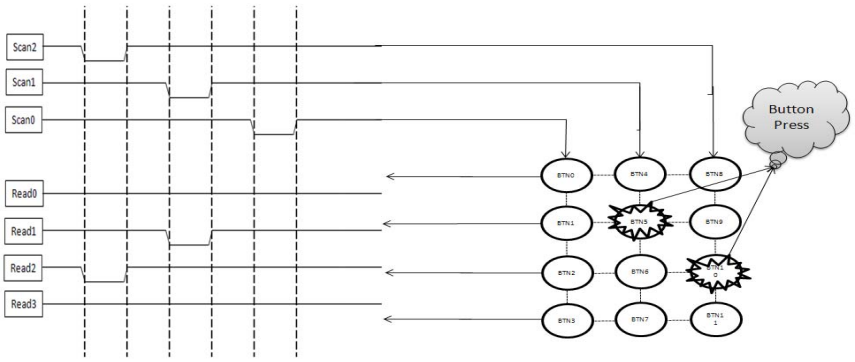

Figure 1. Working of a Mechanical Keypad

Interrupt-based:

Published in ECN Magazine (http://www.ecnmag.com) December 2011

This method is commonly used in systems where the host/master does the key scanning by itself. All the scan lines are grounded and read lines are pulled up to Vcc. When a button is pressed, '0' on the scan lines reach the read lines by means of the physical short (key press) that occurs. Read lines are configured to trigger an interrupt on reading a '0'. In this interrupt, the host does the polling of the scan lines to find the pressed key(s). This frees the host from having to constant poll the scan lines, therefore improving CPU bandwidth utilization.

Mechanical Keypads: 'Cutting edge' replacement

In either of the methods described in the previous section, the scan lines are polled to figure out the pressed buttons – either continuously or when an interrupt has occurred. This forms the basis of the replacement method. Most capacitive touch controllers are microcontrollers which can perform more than just capacitive sensing. The microcontroller part of the capacitive touch controller will be the one playing the replacement/mimic part.

A typical waveform seen on the scan lines for a 4x4 Matrix keypad is as shown below.

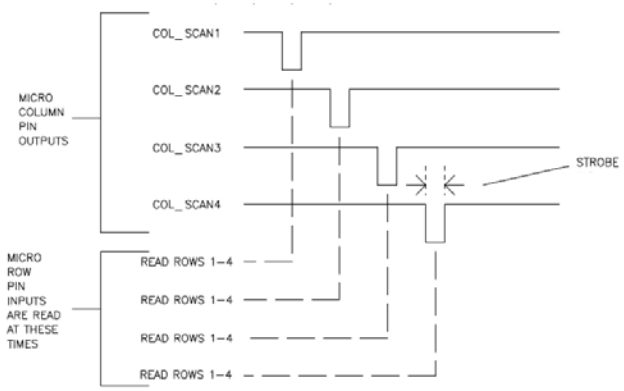

Figure 2. Scan Line Waveform

As can be seen, the scan lines are pulled to GND one after the other and at any given time not more than one scan line is at logic '0'. Having multiple scan lines at logic '0' simultaneously will impair accurate detection of which buttons are being pressed.

To serve as a replacement keypad, the capacitive sensing controller has to act like a mechanical keypad to the host – the output lines (scan lines) of the host will become inputs and the input lines (read lines) of the host will become outputs of the capacitive sensing controller. Based on the scan line status and the buttons touched, the read lines need to be updated by the controller. This can be done as described below. Consider a scenario of 12 buttons (see Figure 3) that need to be laid out in 3x4 matrix fashion, where 3 is the number of scan lines and 4 is the number of read lines.

1. Enable the interrupts on the scan lines.
2. Configure the interrupts to be edge-triggered (on both edges).
3. Initialize a '3' byte array for the 3 scan lines. BYTE is chosen as we need to update 4 read lines (1 bit for each). For more than 8 read lines, WORD should be used.
4. Read the 12 buttons for any finger touch using the capacitive sensing algorithm and update the byte array with the corresponding values. *Note:* the read lines should be active low, as whenever a button is pressed mechanically on a keypad, a '0' on the scan line travels to the read line and the read lines are pulled to Vcc by default.
5. All the scan line interrupts need not be enabled. Only if a button press is to be reported on a scan line should that interrupt be enabled, otherwise it should be disabled. This helps reduce CPU loading in the capacitive touch controller.
6. In the scan line interrupt, read those lines and update the read lines accordingly (refer to the figure below).

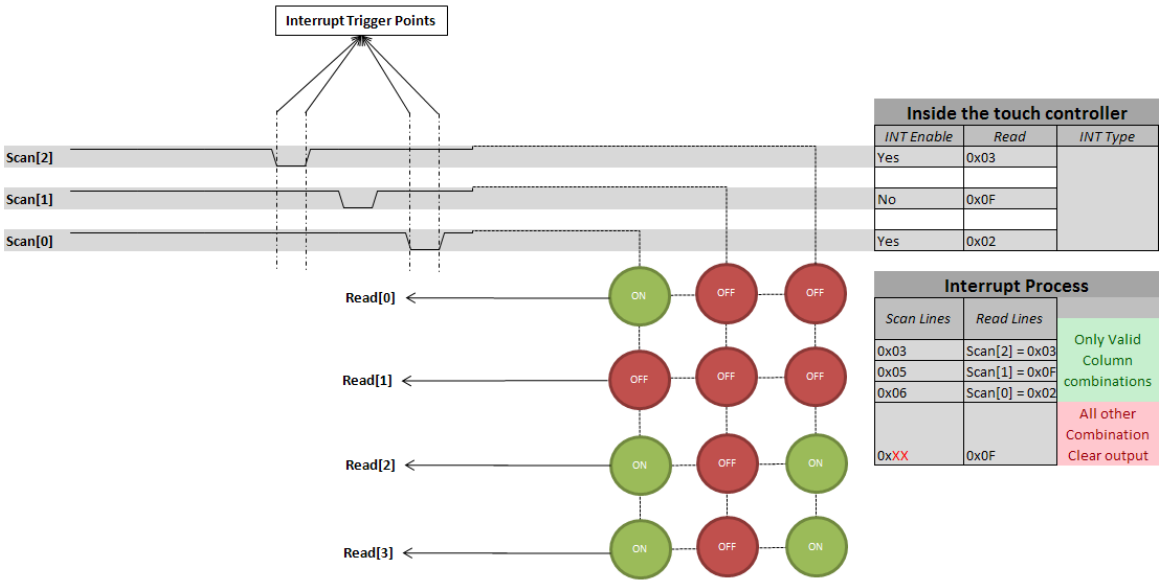

Figure 3. Keypad Mimic Implementaion

The above method helps in replacing polling-based keypad scans. In order to replace an interrupt-based keypad, the same logic can be used with a slight adjustment: whenever a button is pressed, all values stored in the read line array can be AND'ed and sent out on the read lines to the host. Once the host senses the signal and starts scanning, the above steps can be followed.

Limitations:

Replacing an entire keypad with this technique does have some limitations. Since the capacitive touch controller just tries to mimic the mechanical key scan interface by using interrupts, extra delay is introduced – the interrupt latency + processing the read lines inside the interrupt. While typically small, this latency is nonetheless a limitation over physical buttons where the physical short happens instantly (typically within nanoseconds). In our replacement system, the delay may extend into microseconds, depending on the controller chosen. It is worth noting that the host need not worry about button debounces as the touch controller has taken care of this before reporting the button as ON. This saves processing time on the host end but requires modification of the host firmware. Figure 4 shows how to analyze and arrive at a minimum waveform on the scan lines that can be interfaced.

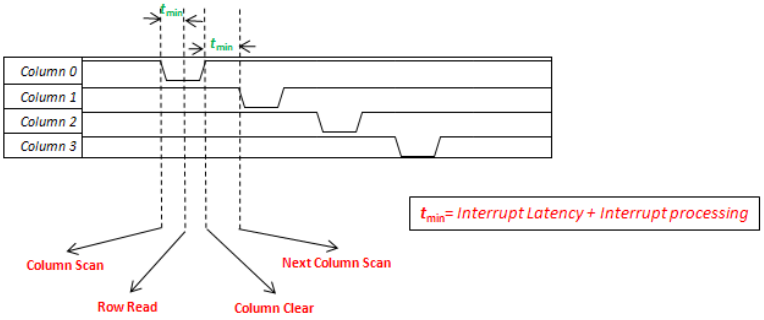

Figure 4. Limitation on the Scan line waveform

There are readily available products on the market which enable developers to replace mechanical keypad with capacitive touch buttons. One such device is the CY8CMBR2016, a 48-pin QFN Capacitive Sensing Express controller from Cypress.

The device can be configured to mimic 4 scan lines and 4 read lines (as shown in Figure 5) – for a total of 16 buttons to replace a 4x4 mechanical keypad. The device can mimic 'polling-based' scanning with a dedicated interrupt line (a separate pin, in addition to the 8 SCAN/READ pins). If your system uses polling-based keypad scanning, then the device can be used to replace the mechanical keypad with capacitive sensing touch buttons. If your system uses interrupt-based scanning mechanism, then the interrupt line offered by the device can be used/configured by the system to start the polling on the SCAN lines.

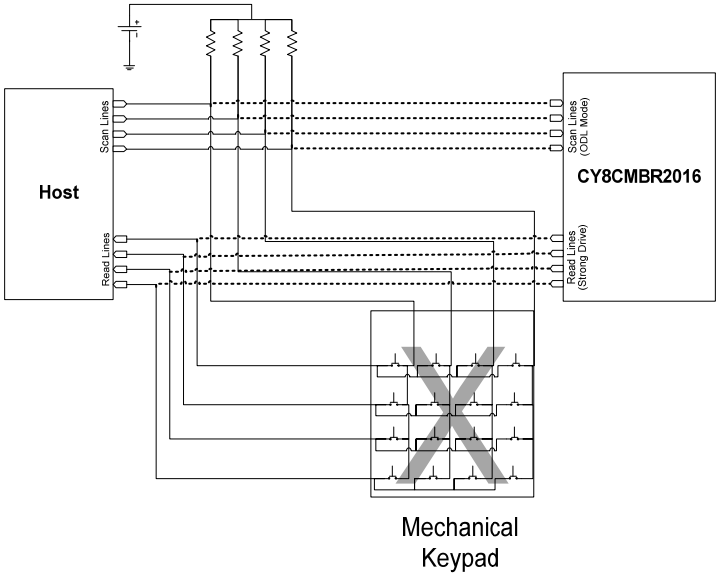

Figure 6. Using CY8CMBR2016

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com