Connect a 4×3 matrix keyboard to a microcontroller using two I/O pins

Aruna Rubasinghe - May 08, 2013

Matrix keyboards are common as an input device in microcontroller-based projects. A conventional way of connecting a matrix keyboard to a microcontroller is to use multiple I/O pins of the MCU. The MCU then uses a scanning algorithm to identify which keys are pressed. A drawback of this method is that it requires a large number of the MCU’s I/O pins to connect the keyboard. For example, to connect a 4×3 keyboard requires seven digital I/O pins. This becomes a problem when the project is based on a low-pin-count MCU or when the MCU being used does not have enough free I/O pins.

Two solutions for this issue are available: Use readily available I/O expanders, or assign a unique voltage to each key using a resistor network and then use an analog pin to read the voltage and determine which key is pressed. Each solution has its own disadvantages.

Since most of the time I/O expanders require a special communication protocol (I2C or SPI, for example) to read and write data, the MCU should have built-in communication modules, or the user has to implement the relevant communication-protocol software wisely, which adds significantly to the overhead of the MCU. On the other hand, assigning a unique voltage to each key using a resistor network becomes troublesome as the number of keys becomes high, which will lead to tight voltage margins. Then, as resistor values tend to change with temperature, the use of tight voltage margins can cause incorrect readings. Even switch bouncing can play a major role in producing incorrect voltages with this method. Another major drawback of this method is that it requires the presence in the MCU of an analog input pin.

The Design Idea described here addresses all of the above problems in an efficient manner and has several advantages: It requires only two I/O pins regardless of the number of switches connected; it does not require a special communication protocol; and it does not require an analog pin. The idea is based on two CD4017 Johnson counters, which are both common and inexpensive.

Figure 1 shows the circuit for a 4×3 keyboard. R1, R4, R5, and R6 are used for current limiting; R7, D4, D5, and D6 form an OR gate.
The example described here shows how to implement this method to read a 4×3 matrix keyboard. One CD4017 is used to control the keyboard rows, while the other is used to control the columns.

The MCU generates a clock signal and feeds it to the counter IC controlling the columns. Initially, the 0th output of the column counter and row counter is at logic high, and the column counter increments as it receives clock pulses. At the fourth clock pulse, the column counter resets and simultaneously increments by one the counter controlling the rows. As the column controller resets, the row controller increments and the row controller resets with the fifth clock pulse from the column controller. As clock pulses generate, a count variable on the MCU should be incremented and should reset to one upon the fifth clock pulse to the row controller. The output of the keyboard is OR’ed and connected to an external interrupt pin of the MCU.

An interrupt occurs only if a button pressed when both the row and the column of the respective button are at the logic-high level. If either row or column of the button is logic zero, an interrupt will not occur.

When an interrupt occurs, the MCU reads the count value at the moment; that value is equal to the button just pressed.
The clock count kept in the MCU increments as it generates clock pulses in intervals; this count is equal to the switch number on the keypad that could generate an interrupt if pressed. The flow chart in Figure 2 illustrates this scenario.

![Flow chart](image)

**Figure 2** The clock count kept in the MCU increments as it generates clock pulses in intervals; this count is equal to the switch number focused at the moment.

Note that even though this example shows a 4×3 keyboard, you can also read a 10×10 keyboard by using the remaining outputs of both 4017 counters. Furthermore, you can cascade additional 4017 ICs to expand the keyboard size as necessary.

**Reference**

1. Rubasinghe, Aruna, "Read 10 or more switches using only two I/O pins of a microcontroller," *EDN*, Feb 28, 2013.

**Also see:**

- [Use an LED matrix horizontally](#)
- [Two-wire, four-by-four-key keyboard interface saves power](#)
- [Single-wire keypad interface frees microcontroller-I/O pins](#)
- [Keyboard links to microcomputer through one-wire interface](#)