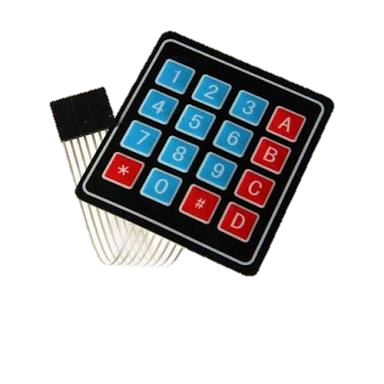


**Keypad Interfacing**

***with 8086***

**Made Simple**

***By Obed Mokweri***

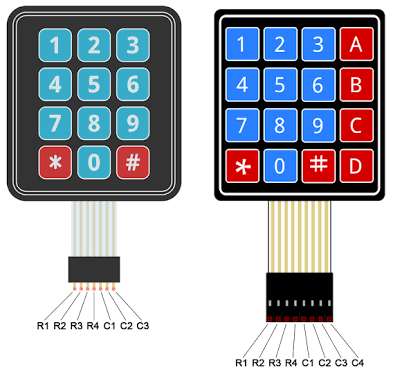


**Contents;**

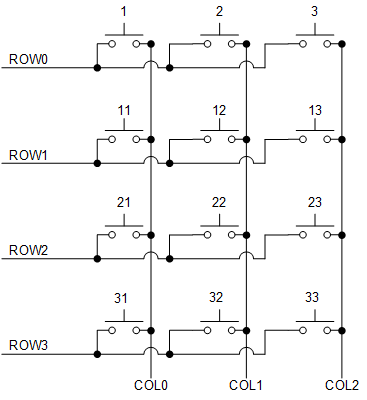
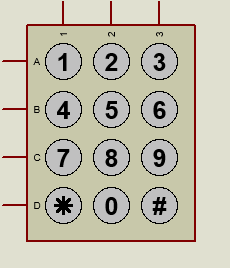
1. **Keypad Layout**
2. **The Logic**
3. **Programming**

**Keypad Layout**

**Keypad Layout**



***Typical Matrix Keypads***



***Keypad Symbol for Proteus***

***4x3 Matrix Keypad Layout***

***Brief Description;***

A matrix keypad consists of arrangement of switches in matrix format in rows and columns with the microprocessor I/O pins connected to the rows and columns of the matrix such that switches in each row are connected to one pin and switches in each column are connected to another pin.

A keypad is generally a matrix arrangement of tact switches which are basically push button switches.

**The Logic**

**How a Matrix Keypad works:**

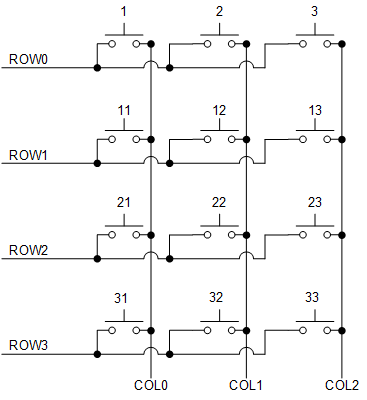
There are numerous techniques depending on the connection keypad with microcontroller, but the fundamental logic making the columns as inputs and drive the rows making them as output. So as to detect which key is pressed from the matrix keypad, the row lines are to be made low one by one and read the columns.

Here we are going to use a 4×3 matrix keypad. It is 12 keys keypad consists of four rows and three columns. Assume that if row1 is made low, then read the columns. If any of the key in row1 is pressed then correspondingly the column 1will give low that is if second key is pressed in row1, then column2 will give low. Suppose, if we press **1** on keypad then ***switch 1****(on Figure on page above)* are switched ON makes the connection and outputs the value through the microprocessor. Similarly, all keys will perform same operation as key 1. We cannot press two keys at the same time. There should be a time difference between to press the key with one other.

**Logic sequence:**

***Step 1:***

The first step involved in interfacing the matrix keypad is to write all logic 1’s to the rows and all logic 1’s to the columns. This indicates no key pressed. Remember, the row pins are inputs and column pins, outputs.



**1 1 1**

**1**

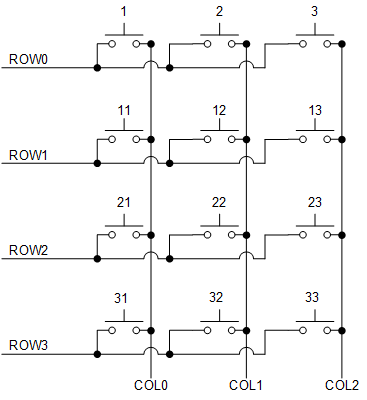
**1**

**1**

**1**

***Step 2:***

Then let’s set column 0 **LOW**.



**0 1 1**

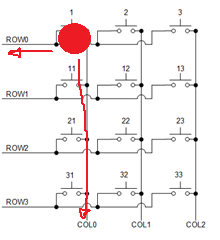
**1**

**1**

**1**

**1**

Now, we loop through the rows in the column to check if any key is pressed. If say key 1 is pressed, then the value of row 0 goes LOW.



**0**

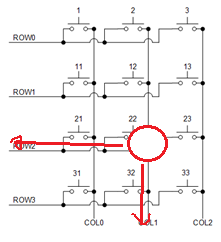
**1**

**1**

**1**

**0 1 1**

By checking the value on the row pins, then we can tell which key is pressed. We then output the value of the key pressed to another port or use however we want. That’s sounds easy, right? I think so.

Also if Key 8 is pressed;

**1 0 1**

**1**

**1**

**0**

**1**

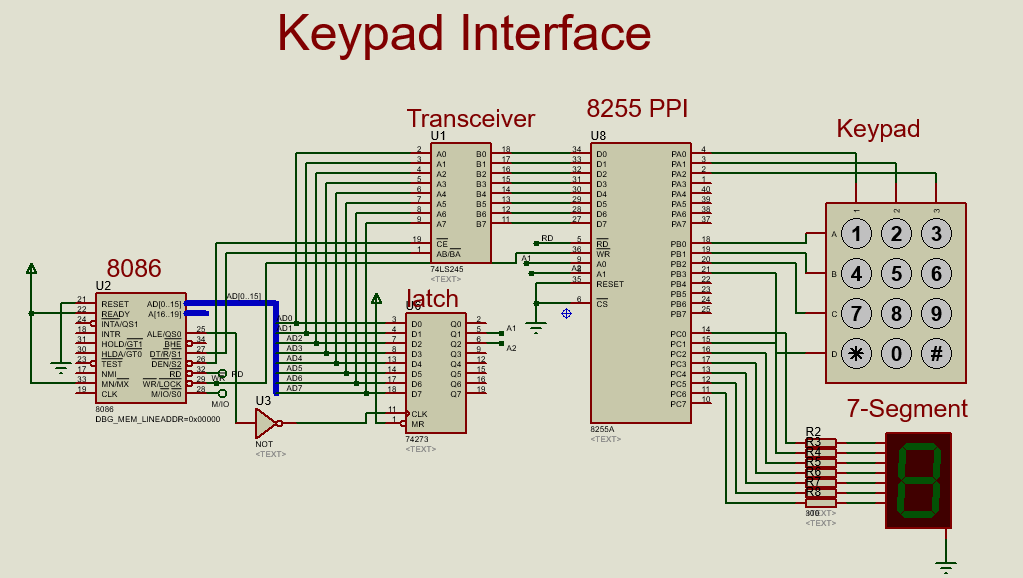
That’s what happens for all the columns and pins. The program code will fully illustrate this concept.

**Programming**

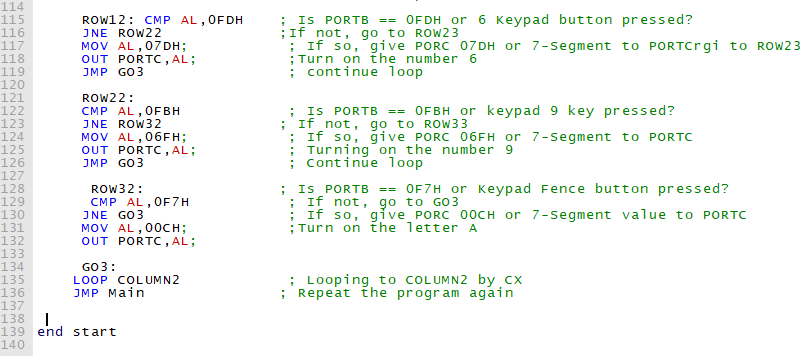
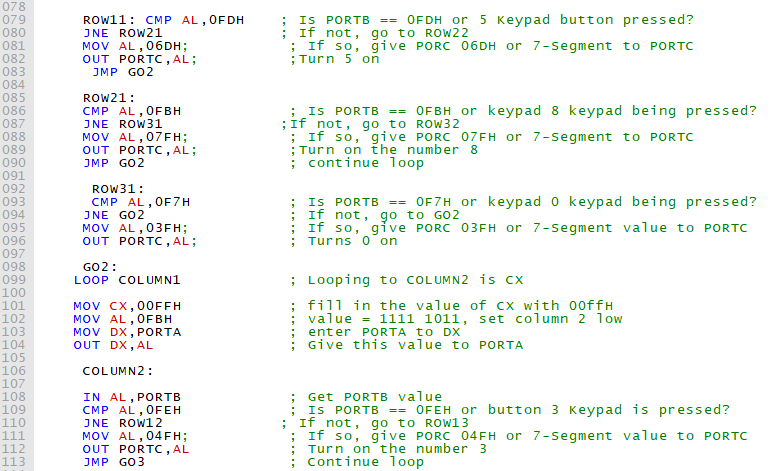
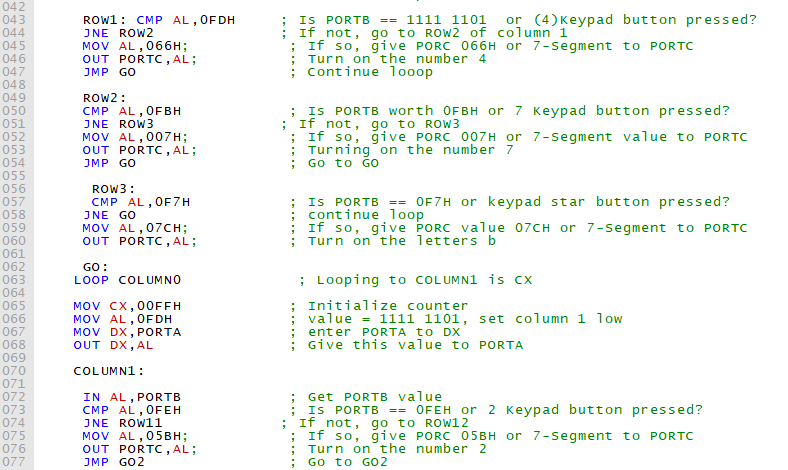
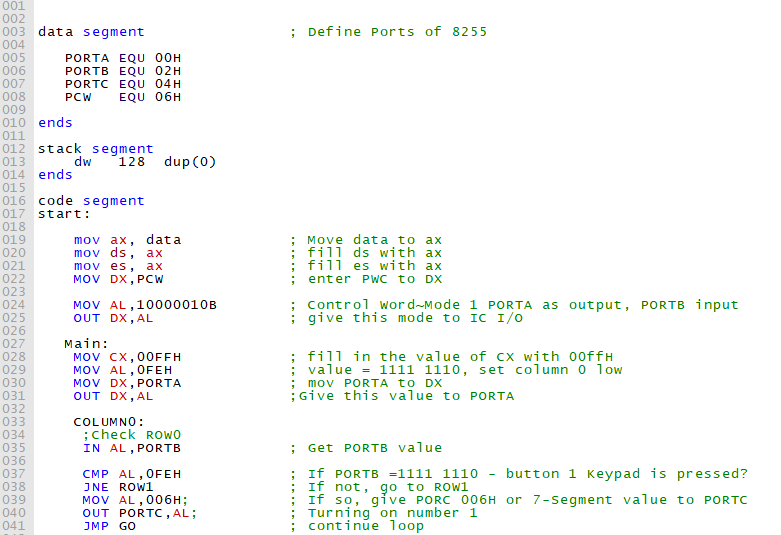
**Sample Application**

A typical application of the keypad is entry of values to a system. To illustrate this, we shall a seven-segment display to display the key pressed on the keypad. This can also be done with an LCD but since we have not handled LCD interfacing, I prefer simpler concepts first. Oh, am not sure if myself I remember how the segment display works and I will refresh and attach it as an appendix.

**Circuit Diagram**

The circuit for this application was done in Proteus.

**Assembly Code**



Quite a lengthy code, yes? Not really.

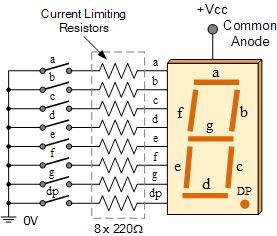
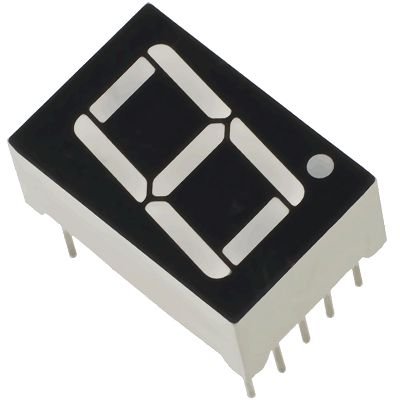
You remember how to generate the executable file for use in Proteus? I hope you do.

I simulated mine and oh yees, it worked.

By the way, this nice quote “If it works, its obsolete”. We should now do something else much better.

**Appendix:**

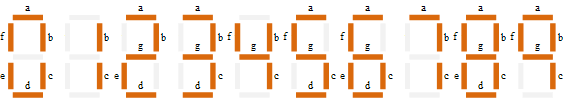
**Seven Segment Display**

**How it Works;**

As shown in the above images of a 7-segment display, it consists of 8 LEDs, each LED used to illuminate one segment of unit and the 8thLED used to illuminate DOT in 7 segment display. We can refer each segment as a LINE, as we can see there are 7 lines in the unit, which are used to display a number/character. We can refer each segment "a,b,c,d,e,f,g" and for dot character we will use "h". There are 10 pins, in which 8 pins are used to refer a,b,c,d,e,f,g and h/dp, the two middle pins are common anode/cathode of all he LEDs. These common anode/cathodes are internally shorted so we need to connect only one COM pin.

### 7-Segment Display Segments for all Numbers.



Then for a 7-segment display, we can produce a truth table giving the individual segments that need to be illuminated in order to produce the required decimal digit from 0 through 9 as shown below.

|  |  |  |
| --- | --- | --- |
| Digit to Display | h g f e d c b a | Hex code |
| 0 | 00111111 | 3F |
| 1 | 11111001 | F9 |
| 2 | 01011011 | 5B |
| 3 | 01001111 | 4F |
| 4 | 01100110 | 66 |
| 5 | 01101101 | 6D |
| 6 | 01111101 | 7D |
| 7 | 00000111 | 07 |
| 8 | 01111111 | 7F |
| 9 | 01101111 | 6F |

***References***;

<https://www.elprocus.com/matrix-keypad-interfacing-with-microcontroller/>

<http://pcbheaven.com/wikipages/How_Key_Matrices_Works/>

<https://circuitdigest.com/microcontroller-projects/keypad-interfacing-with-avr-atmega32>

<https://embedjournal.com/interface-4x4-matrix-keypad-with-microcontroller/>

<https://www.electronics-tutorials.ws/blog/7-segment-display-tutorial.html>

<https://circuitdigest.com/article/7-segment-display>

*NOTE: This document and its accompanying files are also available for download in* [*Github*](http://www.github.com)*, a tool I recommend to you guys. The particular link will be provided.*

*Look forward to the next document release. I will always share any good stuff I manage to chew as long as God gives breathe. Should you find any challenges, I would be glad to help or we can Google it together. But always try finding a solution yourself first.*

[*mogsobd@gmail.com*](mailto:mogsobd@gmail.com)

*NEXT Release: LCD Interface*

*HAVE FUN MEMBERS!!!!!*