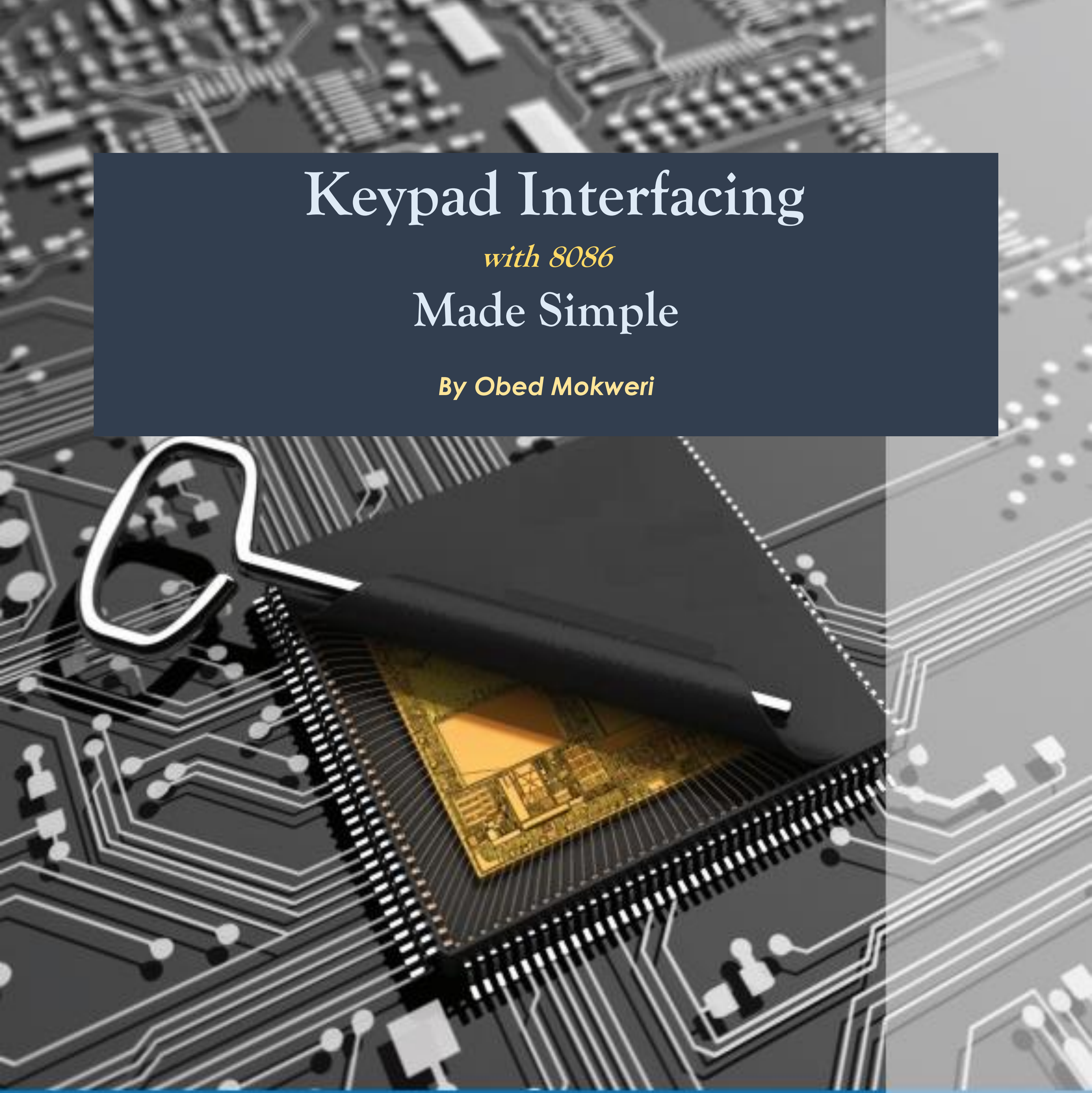
Keypad Interfacing

with 8086

Made Simple

By Obed Mokweri



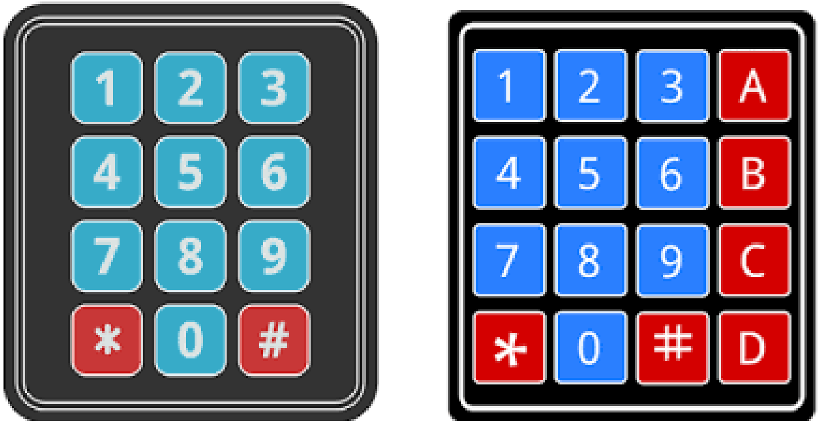
Contents;

Embedded Systems

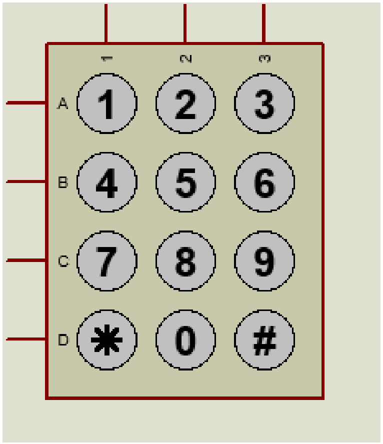
Compiled by:
Eng. Obed © 2018
(Embedded Systems Developer)

Keypad Layout

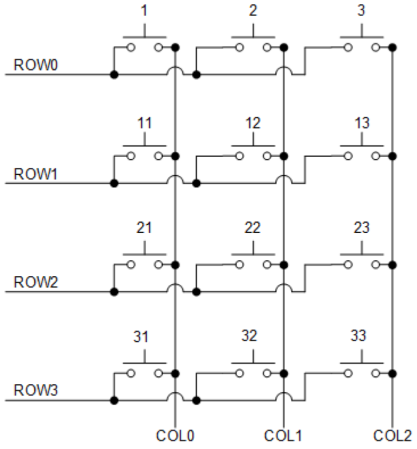
Keypad Layout



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:

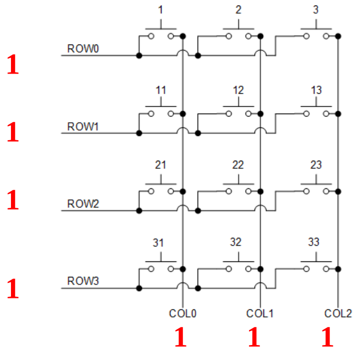
There are numerous techniques depending on the connection keypad with microcontroller, but the fundamental logic making the columns as inputs and drive the rows making them as output. So as to detect which key is pressed from the matrix keypad, the row lines are to be made low one by one and read the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

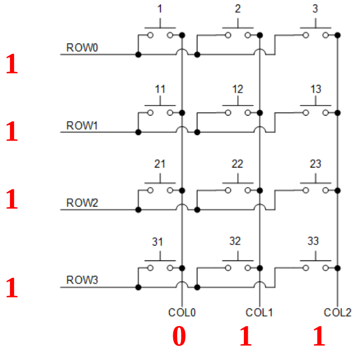
Step 1:

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.

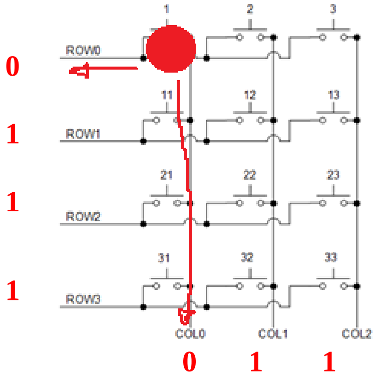


Step 2:

Then let's set column 0 **LOW**.

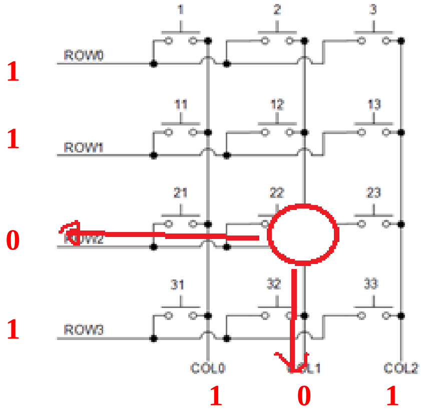


Now, we loop through the rows in the column to check if any key is pressed. If say key 1 is pressed, then the value of row 0 goes LOW.



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;



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

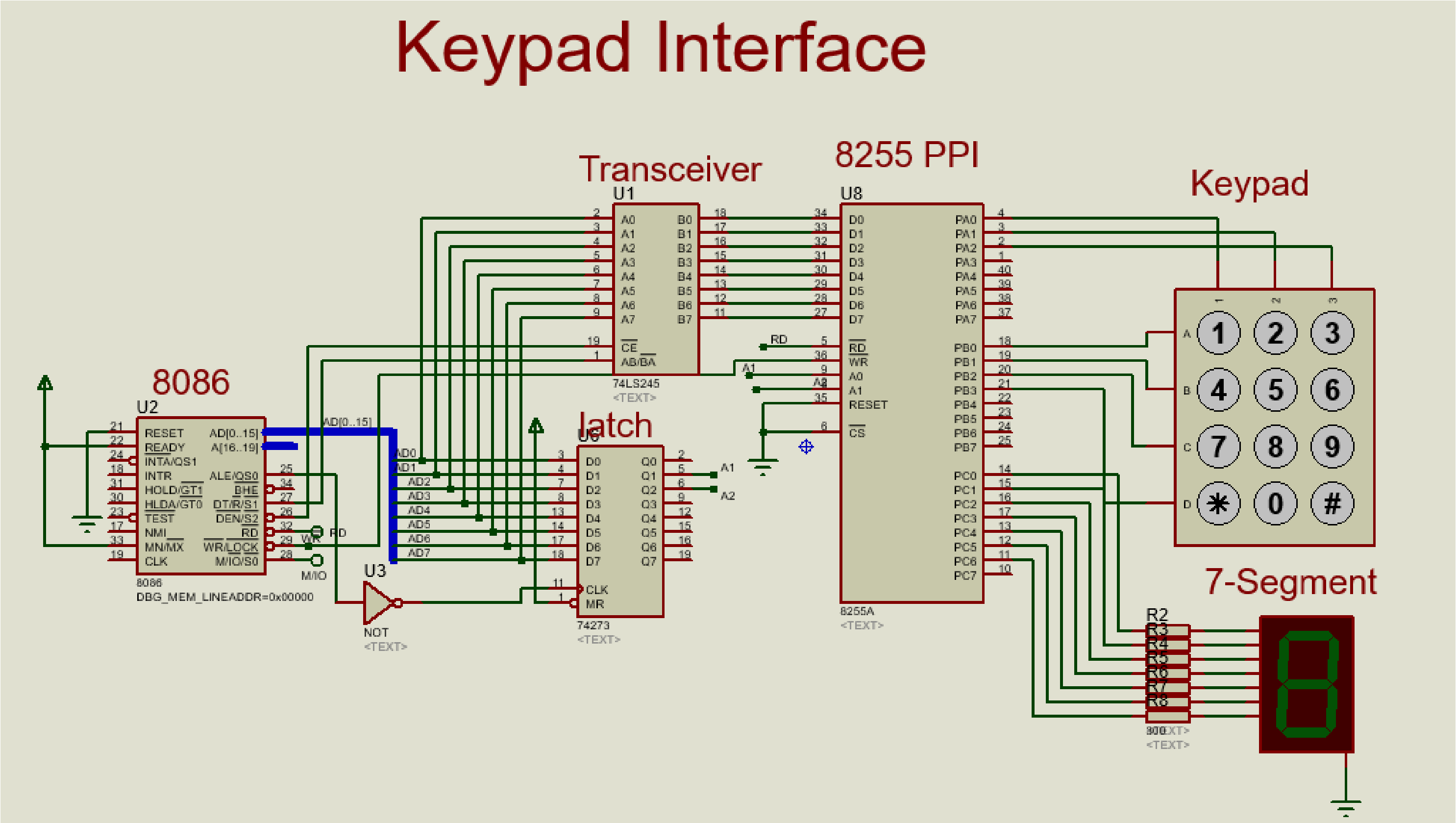
Programming

Sample Application

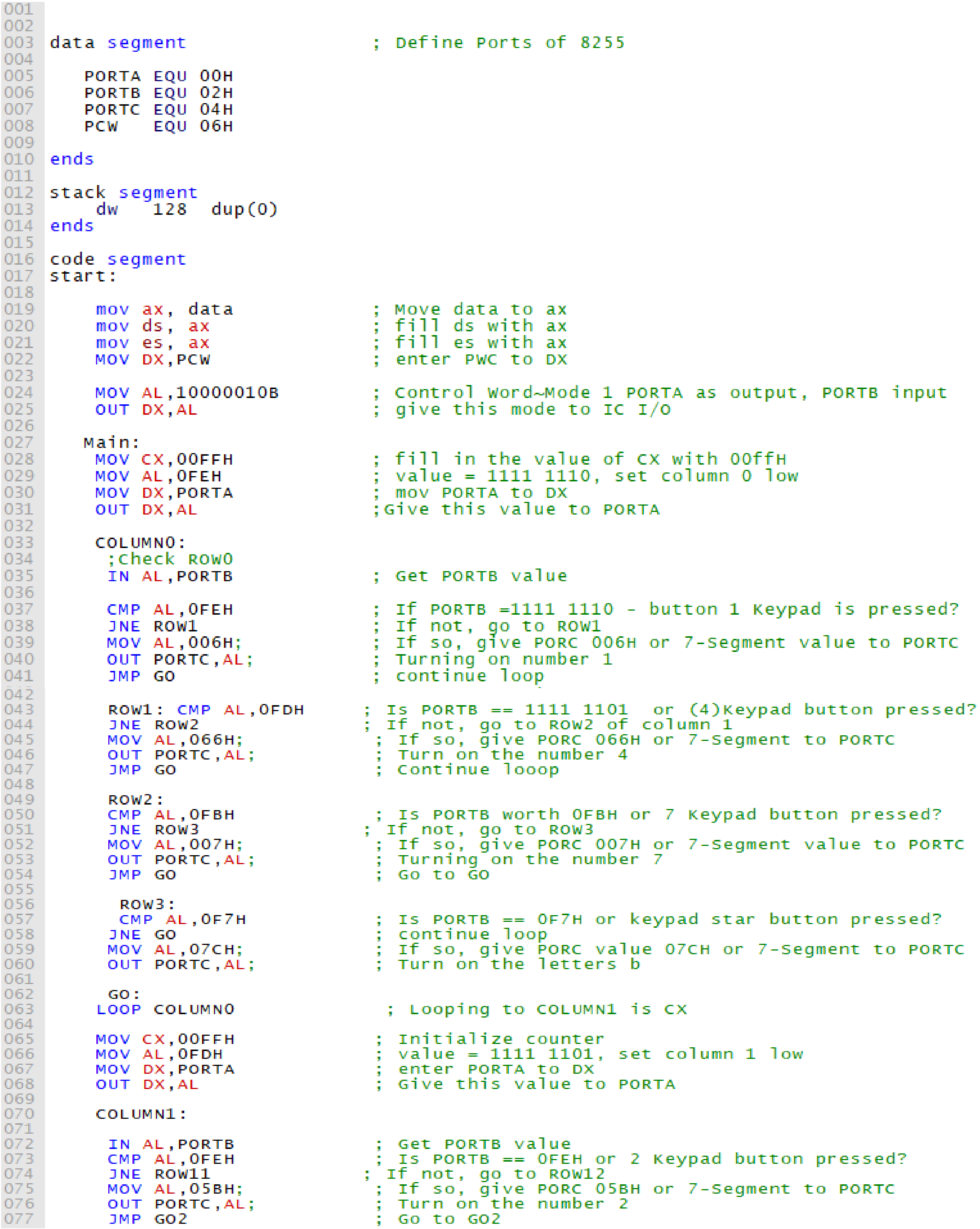
A typical application of the keypad is entry of values to a system. To illustrate this, we shall a seven-segment display to display the key pressed on the keypad. This can also be done with an LCD but since we have not handled LCD interfacing, I prefer simpler concepts first. Oh, am not sure if myself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

Circuit Diagram

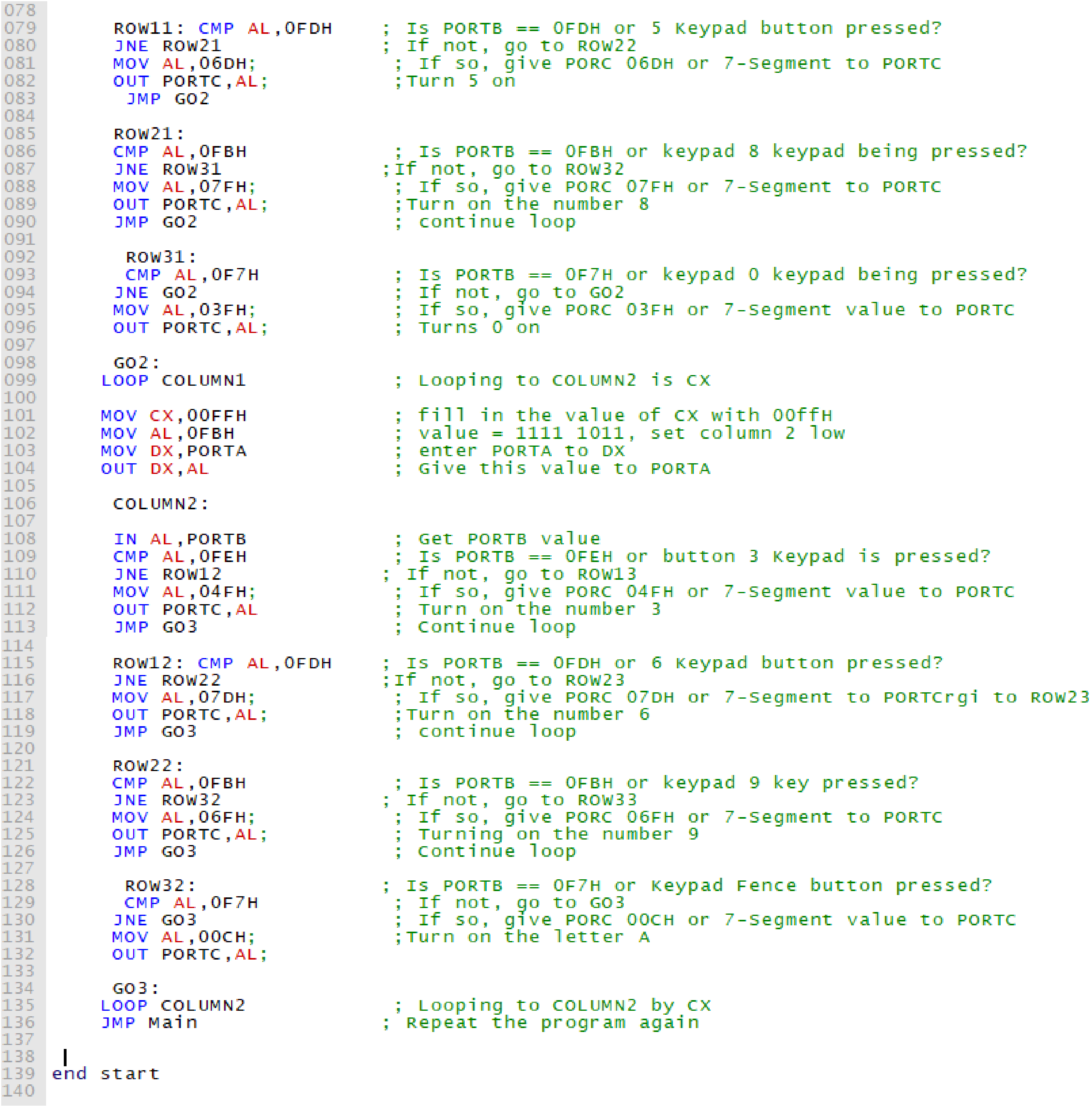
The circuit for this application was done in Proteus.



Assembly Code



```
data segment                ; Define Ports of 8255

   PORTA EQU 00H
   PORTB EQU 02H
   PORTC EQU 04H
   PCW   EQU 06H

ends

stack segment
    dw   128   dup(0)
ends

code segment
start:

    mov ax, data            ; Move data to ax
    mov ds, ax              ; fill ds with ax
    mov es, ax              ; fill es with ax
    MOV DX,PCW              ; enter PWC to DX

    MOV AL,10000010B        ; Control Word~Mode 1 PORTA as output, PORTB input
    OUT DX,AL               ; give this mode to IC I/O

   Main:
    MOV CX,00FFH            ; fill in the value of CX with 00ffH
    MOV AL,0FEH             ; value = 1111 1110, set column 0 low
    MOV DX,PORTA            ; mov PORTA to DX
    OUT DX,AL               ;Give this value to PORTA

    COLUMN0:
     ;Check ROW0
     IN AL,PORTB            ; Get PORTB value

     CMP AL,0FEH            ; If PORTB =1111 1110 - button 1 Keypad is pressed?
     JNE ROW1               ; If not, go to ROW1
     MOV AL,006H;           ; If so, give PORC 006H or 7-Segment value to PORTC
     OUT PORTC,AL;          ; Turning on number 1
     JMP GO                 ; continue loop

     ROW1: CMP AL,0FDH     ; Is PORTB == 1111 1101  or (4)Keypad button pressed?
     JNE ROW2              ; If not, go to ROW2 of column 1
     MOV AL,066H;           ; If so, give PORC 066H or 7-Segment to PORTC
     OUT PORTC,AL;          ; Turn on the number 4
     JMP GO                 ; Continue looop

     ROW2:
     CMP AL,0FBH            ; Is PORTB worth 0FBH or 7 Keypad button pressed?
     JNE ROW3              ; If not, go to ROW3
     MOV AL,007H;           ; If so, give PORC 007H or 7-Segment value to PORTC
     OUT PORTC,AL;          ; Turning on the number 7
     JMP GO                 ; Go to GO

      ROW3:
      CMP AL,0F7H           ; Is PORTB == 0F7H or keypad star button pressed?
     JNE GO                 ; continue loop
     MOV AL,07CH;           ; If so, give PORC value 07CH or 7-Segment to PORTC
     OUT PORTC,AL;          ; Turn on the letters b

     GO:
    LOOP COLUMN0             ; Looping to COLUMN1 is CX

    MOV CX,00FFH            ; Initialize counter
    MOV AL,0FDH             ; value = 1111 1101, set column 1 low
    MOV DX,PORTA            ; enter PORTA to DX
    OUT DX,AL               ; Give this value to PORTA

    COLUMN1:

     IN AL,PORTB            ; Get PORTB value
     CMP AL,0FEH            ; Is PORTB == 0FEH or 2 Keypad button pressed?
     JNE ROW11             ; If not, go to ROW12
     MOV AL,05BH;           ; If so, give PORC 05BH or 7-Segment to PORTC
     OUT PORTC,AL;          ; Turn on the number 2
     JMP GO2                ; Go to GO2
```



```

     ROW11: CMP AL,0FDH       ; Is PORTB == 0FDH or 5 Keypad button pressed?
     JNE ROW21                ; If not, go to ROW22
     MOV AL,06DH;              ; If so, give PORC 06DH or 7-Segment to PORTC
     OUT PORTC,AL;             ;Turn 5 on
      JMP GO2

     ROW21:
     CMP AL,0FBH               ; Is PORTB == 0FBH or keypad 8 keypad being pressed?
     JNE ROW31                ;If not, go to ROW32
     MOV AL,07FH;              ; If so, give PORC 07FH or 7-Segment to PORTC
     OUT PORTC,AL;             ;Turn on the number 8
     JMP GO2                   ; continue loop

      ROW31:
      CMP AL,0F7H              ; Is PORTB == 0F7H or keypad 0 keypad being pressed?
     JNE GO2                   ; If not, go to GO2
     MOV AL,03FH;              ; If so, give PORC 03FH or 7-Segment value to PORTC
     OUT PORTC,AL;             ; Turns 0 on

     GO2:
    LOOP COLUMN1               ; Looping to COLUMN2 is CX

    MOV CX,00FFH               ; fill in the value of CX with 00ffH
    MOV AL,0FBH                ; value = 1111 1011, set column 2 low
    MOV DX,PORTA               ; enter PORTA to DX
    OUT DX,AL                  ; Give this value to PORTA

     COLUMN2:

     IN AL,PORTB               ; Get PORTB value
     CMP AL,0FEH               ; Is PORTB == 0FEH or button 3 Keypad is pressed?
     JNE ROW12                ; If not, go to ROW13
     MOV AL,04FH;              ; If so, give PORC 04FH or 7-Segment value to PORTC
     OUT PORTC,AL              ; Turn on the number 3
     JMP GO3                   ; Continue loop

     ROW12: CMP AL,0FDH       ; Is PORTB == 0FDH or 6 Keypad button pressed?
     JNE ROW22                ;If not, go to ROW23
     MOV AL,07DH;              ; If so, give PORC 07DH or 7-Segment to PORTCrgi to ROW23
     OUT PORTC,AL;             ;Turn on the number 6
     JMP GO3                   ; continue loop

     ROW22:
     CMP AL,0FBH               ; Is PORTB == 0FBH or keypad 9 key pressed?
     JNE ROW32                ; If not, go to ROW33
     MOV AL,06FH;              ; If so, give PORC 06FH or 7-Segment to PORTC
     OUT PORTC,AL;             ; Turning on the number 9
     JMP GO3                   ; Continue loop

      ROW32:                  ; Is PORTB == 0F7H or Keypad Fence button pressed?
      CMP AL,0F7H              ; If not, go to GO3
     JNE GO3                   ; If so, give PORC 00CH or 7-Segment value to PORTC
     MOV AL,00CH;              ;Turn on the letter A
     OUT PORTC,AL;

     GO3:
    LOOP COLUMN2               ; Looping to COLUMN2 by CX
    JMP Main                  ; Repeat the program again


end start

```

Quite a lengthy code, yes? Not really.

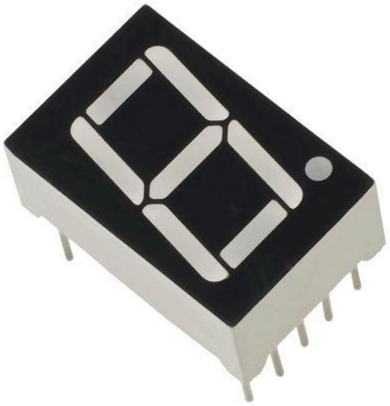
You remember how to generate the executable file for use in Proteus? I hope you do.

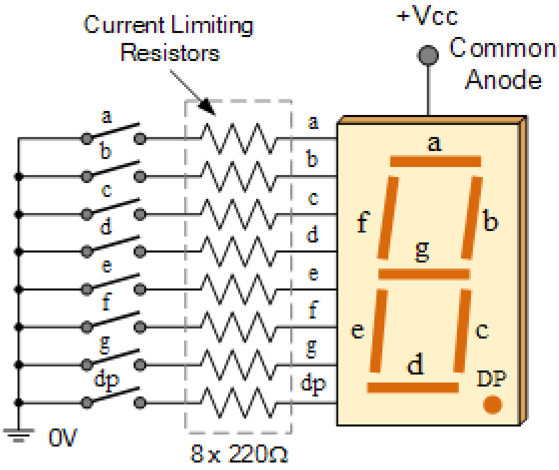
I simulated mine and oh yees, it worked.

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

Appendix:

Seven Segment Display

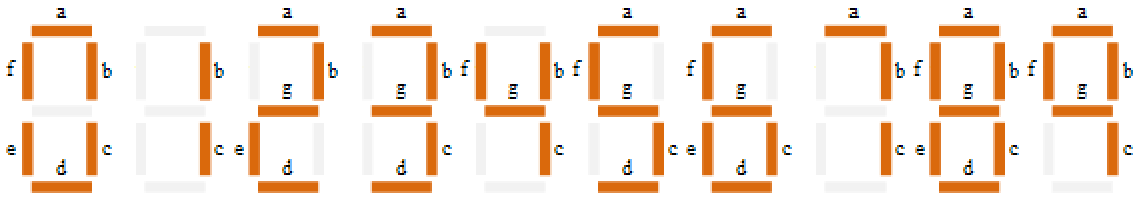




How it Works;

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, 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

NEXT Release: LCD Interface

HAVE FUN MEMBERS!!!!!