**Experiment 8**

**KEYPAD INTERFACING**

**Introduction:**

The keypad is a matrix of push buttons. It can be used in various ways to take an input. Keypad can be scanned using two methods i.e., either ground each row at a time and read the columns or ground each column at a time and read rows, in order to scan the whole keypad for detecting the key pressed. 4x4 Matrix Keypad Interfacing with 8051 Microcontroller. Keypads are widely used input devices being used in various electronics and embedded projects. They are used to take inputs in the form of numbers and albhabets, and feed the same into system for further processing.

**Objective:**

To learn scanning of a simple keypad for input

To learn a method of taking input from keypad and process it using the 89C52 micro control

**Procedure:**

In the first lab task, we were to take input from the keypad and in turn display the squared input as output on the LEDs. We modified the keypad values for this task by assigning decimal value 10 to asterisk **\*** and 11 to hash **#** sign.

In the second lab task, we took input of a 2-digit number from the keypad and if the input is an even number we set the pin P1.3 to high logic otherwise to low logic.

**Applications:**

The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals .Security system with a user changeable password is a practical application in which keypad is used as one of the interfacing devices. Password based door lock system is used for security to house or organization or office or company.

**Issues:**

No issues faced while performing the lab.

**Conclusions:**

In this lab first we learn scanning of a simple keypad for input and also a method of taking input from keypad and process it using the 89C52 microcontroller.

**Post Lab:**

|  |
| --- |
| org 0x0  mov p1,#0 ; port p1 is for rows (output port)  mov p2,#0ffh ; port p2 is for columns (input port)  ;;;;; keypad info  rows equ 4  cols equ 3  ;;;; creating mask for checking columns  mov a,#0h  mov r1,#0h  rot\_again:  setb c  inc r1  rlc a  cjne r1,#cols,rot\_again  mov r0,a  mov P3,#00H  **restart:**  call get\_key  mov 38H,r4  //mov p3,r4  mov A,r4  //anl A,#0FH  //mov R7,A //copy of the input  **zero:**cjne A,#0H,one  clr P3.3  setb p3.6  sjmp zero  **one:**cjne A,#1H,two  setb P3.3  mov 23H,#1  call delay\_1s  clr P3.3  mov 23H,#9  call delay\_1s  sjmp one  **two:**cjne A,#2H,three  setb P3.3  mov 23H,#2  call delay\_1s  clr P3.3  mov 23H,#8  call delay\_1s  sjmp two  **three:**cjne A,#3H,four  setb P3.3  mov 23H,#3  call delay\_1s  clr P3.3  mov 23H,#7  call delay\_1s  sjmp three  **four:**cjne A,#4H,five  setb P3.3  mov 23H,#4  call delay\_1s  clr P3.3  mov 23H,#6  call delay\_1s  sjmp four  **five:**cjne A,#5H,six  setb P3.3  mov 23H,#5  call delay\_1s  clr P3.3  mov 23H,#5  call delay\_1s  sjmp five  **six:**cjne A,#6H,seven  setb P3.3  mov 23H,#6  call delay\_1s  clr P3.3  mov 23H,#4  call delay\_1s  sjmp six  **seven:**cjne A,#7H,eight  setb P3.3  mov 23H,#7  call delay\_1s  clr P3.3  mov 23H,#3  call delay\_1s  sjmp seven  **eight:**cjne A,#8H,nine  setb P3.3  mov 23H,#8  call delay\_1s  clr P3.3  mov 23H,#2  call delay\_1s  sjmp eight  **nine:**cjne A,#9H,ten  setb P3.3  mov 23H,#9  call delay\_1s  clr P3.3  mov 23H,#1  call delay\_1s  sjmp nine  **ten:**jmp restart  ;;;;; start scanning  get\_key:  ; mask is in r0  **again:**  mov r1,#0feh ; ground 0th row  mov r2,#0  mov r3,#0  **next\_row:**  mov p1,r1  mov a,p2  anl a,r0  cjne a,0h,key\_pressed  mov a,r1  rl a  mov r1,a  inc r2 ; r2 will contain the row index  cjne r2,#rows,next\_row  jmp again  **key\_pressed:**  call delay ; debounce time  again1:  rrc a  jnc findkey  inc r3 ; r3 contains the column index  jmp again1  **findkey:**  mov R4,#0H  mov a,#cols  mov b,r2  mul ab  add a,r3  mov dptr,#key  movc a,@a+dptr  mov r4,a  **release\_key:**  mov a,p2  anl a,r0  cjne a,0h,release\_key  call delay ; debounce time  ret  **delay:**  MOV R5,#45  lOOOP:  MOV R6,#255  DJNZ R6,$  DJNZ R5, LOOOP  ret  **delay\_1s:**  loop3:mov 32H,#8  loop0:mov 31H,#250  loop1:mov 30H,#229  loop2:DJNZ 30H,loop2  DJNZ 31H,loop1  DJNZ 32H,loop0  DJNZ 23H,loop3  ret  org 0x100  **key:** db 1,2,3,4,5,6,7,8,9 /\* 1D index = column index + (row index \* total no. of cols)\*/  **end** |