**Microprocessor Lab-work #3uVision**

**4x4 Keypad Scanning** 100-11-14

[1] **Subject and goals**

1. interconnecting structure of a 4x4 keypad, together with scanning and encoding process
2. balance among several IO-control tasks that have to be executed in order repetitively

[2] **Preparations**

(a) **Refer to the ckt schematic diagram**:

(a.1) how the DIP switch, 4x1 switch, and 4x4 keypad module(s) would be scanned.

(a.2) data path(s) from 51CPU to the target switch module(s)?

(b) **Datasheets reading**:

(b.1) none

(c) **Readiness-evaluation:**

Can you or can you not

(c.1) check if the target module (i.e., any switch and any 7-seg LED digit) is working properly or not by manually wiring the circuitry?

(c.2) carry out trouble shooting along the path way when the lab-work isn’t going as expected? How will you do that?

[3] **Lab-work for all:**

(a) **Operating Procedure**

|  |
| --- |
| **TASK 1** encoding for any single key entry during 4x4 keypad scanning and then displaying the key-code on the 7-seg LED module, respectively with one and two 7-seg digit(s)  \*\* 4x4 keypad set up \*\* code preparation \*\* code executing under IDE51 emulation |

(a.1) jumper-wiring for 4x4 keypad module setup

Refer to the schematic circuit diagram, do all jumper-wiring necessary for setting up the circuitry as required below.

C0

C1

C2

**51μ*p*** C3

RN3

EA R3

8

P1

R2

**Pz** R1

JPx JPy

R0

**JP? JP??**

Refer to the sample codes for

proper wiring to set up the

7-segment LED module

(a.2) code preparation:

\*\* edit the following sample 51 assembly codes, …two versions

\*\* get version1 codes ready for execution first, then version2

; =================================

; **version 1** key-code display on one 7-seg

; digit

; **x** **x** **x** **x**

; 8 9 **x** **x**

; 4 5 6 7

; 0 1 2 3

; **X**s: non-decimal 7-seg patterns

; =================================

; port 1 for keypad scanning

; P1: 0-3 output for column scanning

; P1: 4-7 input for row reading

; port 2 for using one digit of 7-seg LED

org 0

mov sp, #50H

mov P2, #0FFH

col0: mov R6, #0

mov P1, #0FEH

mov A, P1

orl A, #0FH

swap A

cpl A

jnz keycoding

col1: mov R6, #1

mov P1, #0FDH

mov A, P1

orl A, #0FH

swap A

cpl A

jnz keycoding

col2: mov R6, #2

mov P1, #0FBH

mov A, P1

orl A, #0FH

swap A

cpl A

jnz keycoding

col3: mov R6, #3

mov P1, #0F7H

mov A, P1

orl A, #0FH

swap A

cpl A

jz col0

keycoding: ; A: 0-9,A,B,C,D,E F

rr A

anl #0FH

jnb A.2, cont

mov A, #3

cont: jz cont2

mov R7, A

clr A

cont1: add A, #4

djnz R7, cont1

cont2: add A, R6

display: ; when A>9 ???

oal A, #0E0H

mov P2, A

jmp col0

end

; =============================

; =================================

; **version 2** key-code display on two 7-seg

; digits

; 12 13 14 15

; 08 09 10 11

; 04 05 06 07

; 00 01 02 03

; =================================

; port 1 for keypad scanning

; P1: 0-3 output for column scanning

; P1: 4-7 input for row reading

; port 2 for using two digits of 7-seg LED

org 0

mov sp, #50H

mov R7, #15

mov P2, #0FFH

col0: mov R6, #0

mov P1, #0FEH

mov A, P1

cpl A

jnz keycoding

call display ; ==??==

col1: mov R6, #1

mov P1, #0FDH

mov A, P1

orl A, #0FH

swap A

cpl A

jnz keycoding

call display ; ==??==

col2: mov R6, #2

mov P1, #0FBH

mov A, P1

orl A, #0FH

swap A

cpl A

jnz keycoding

call display ; ==??==

col3: mov R6, #3

mov P1, #0F7H

mov A, P1

orl A, #0FH

swap A

cpl A

jz col0

kecoding: ; A: 0-9,A,B,C,D,E F

rr A

anl #0FH

jnb A.2, cont

mov A, #3

cont: jz cont2

mov R7, A

clr A

cont1: add A, #4

djnz R7, cont1

cont2: add A, R6

mov R7, A

call display ; ==??==

jmp col0

display:

push PSW

push A

push F0H ; push B??

mov A, R7

mov B, #10

div A, B

anl A, #0FH

orl A, #0D0H

mov P2, A

call delay

mov A, B

anl A, #0FH

orl A, #0E0H

mov P2, A

call delay

pop F0H

pop A

pop PSW

ret

delay: push 2

push 3

mov R2, #20

xxx: mov R3, #250

djnz r3, $

djnz r2, xxx

pop 3

pop 2

ret

end

; =============================

(a.3) task execution:

\*\* start IDE51 emulator for the execution of code version1 and observe circuit behaviors

\*\* start IDE51 emulator for the execution of code version2 and observe circuit behaviors

\*\* start trouble-shooting if necessary

[**Key**: checking along the data path in a stage by stage manner, from the start: inside of 89c51 to the end: the target modules. ]

1. **Observations**

(b.1) Is code version 1 or version 2 running well? If so, it’s a night of yours.

If not, congratulate you that you have a chance for getting more experience in trouble-shooting.

(b.2) While running version 1 sample code, one 7-seg LED will show the key-code when a key is entered and the display would remain so until the next entry of a key. For each key-code display, how many times does 89c51 access the 7-seg LED digit? And how many times of accesses occur in between two key entries? What does that tell about the output nature of P2?

(b.3) Consider the limits imposed by the 7-seg LED module, what could you say regarding the issue of handling both keypad scanning and the display of two 7-seg LEDs concurrently?

(b.4) Instead of jumping to the label **display** as done in version 1 sample, version 2 sample code deals with two 7-segment LED digits showing different patterns by calling the subroutine **display**. Why? Is there batter approach to placing the call instructions other than at the lines marked by **==??==** in version2 code lines?

(b.5) Without **RN3** pull-up resistor array, will the keypad module still function properly? Why or why not?

(b.6) As the code of version 2 starts up, “15” appears on the two 7-seg LED digits even before key entry. Fix the problem!

(b.7) Elongate the duration of **delay** in version2 sample code by 10 times of its original setting. What do you see regarding the change in circuit behaviors? Explain.

[4] **Comprehension evaluation**

(a) Could you modify version-1 codes for a shorter code length or a better coding structure (e.g., handling 4-column scanning with a loop-body)?

(b) How could the version-2 codes be revised in such a way that the handling of the display of two 7-seg LEDs and keypad scanning would be resolved somewhat more elegantly?

(c) Consider the duration of a typical key entry from the moment of key pressing to that of key releasing, how long would it be? When a key is entered while executing either version of the sample codes, how many times the CPU would have detected this very same key entry?

What are the issues to be tackled in coding such that one-detection for one-entry could be achieved?

(d) Consider the case of multiple key entry in which more than one keys are pressed at one time. Could you write the codes using the 4x1 switch module, where each switch is assigned an value, say 1, 2, 3 and 4 respectively, for summing up values resulting from multiple key entry and displaying the sum on any of the 3 LED modules on the circuit board?