Microprocessor Lab-work #2.2

7-segment LEDs 100-11-14

[1] **Subject and goals**

(a) The access of six 7-segment LED for ON/OFF and pattern control in the 7-segment LED module

(b) Organized display patterns in static or dynamic form can be achieved as required.

[2] **Preparations**

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

(a.1) how the 7-segment LED module may operate (its inputs to module for individual 7-seg LED selection as well as display pattern)?

(a.2) functions of 7447 and BJT-2N3906 (discrete bipolar-transistor)?

(a.3) data path from 51CPU to the 7-seg LED module?

(b) **Datasheets reading**:

(b.1) TTL7447

(c) **Readiness evaluation:**

The 7-seg LED module physically consists of six 7-seg LED components, each being powered by a BJT acting as a power switch.

Can you or can you not

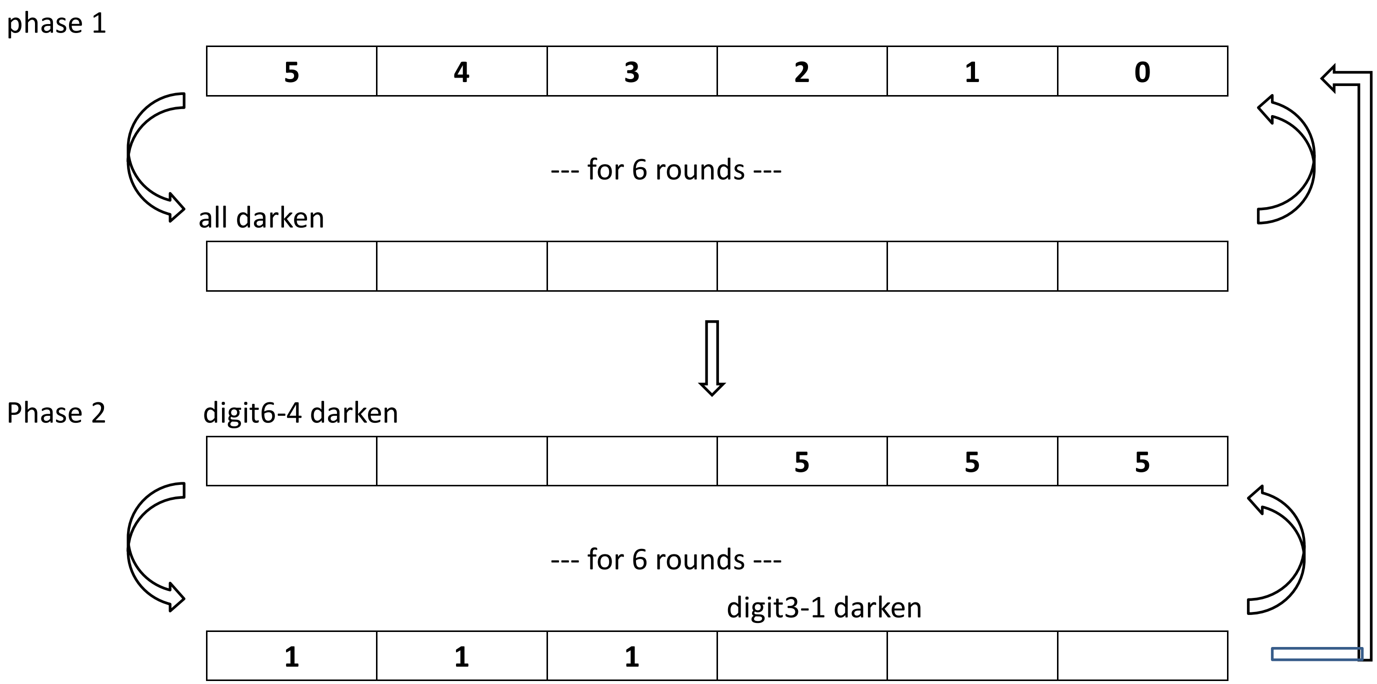
(c.1) check the 7-segment LED module to see if it's working or not by manual wiring the circuitry?

(c.2) write the codes for any static/dynamic pattern display on the module?

(c.3) describe the operational limits of the 7-seg LED module imposed by the circuitry?

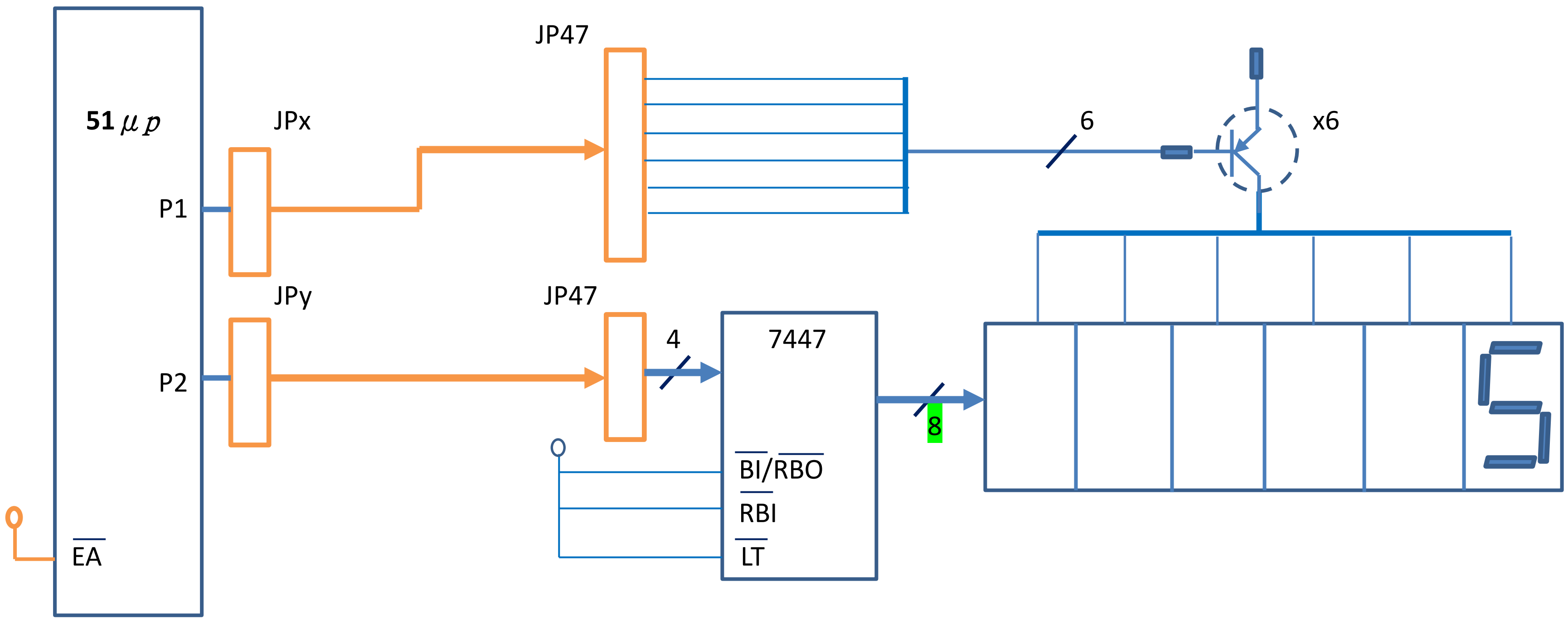
[3] **Lab-work for all:**

The task here is to use the 7-seg LED module for the dynamic display sequence as graphically depicted below



(a)**Operating Procedure**

(a.1) **jumper-wiring for ckt setup** [refer to the schematic circuit diagram for wiring details]



(a.2) **code preparation**:

** edit the following sample 51 assembly code

** get the code ready for execution under IDE51 emulation

```
        org     0
        mov     SP, #50H
start:  mov     R7, #6
next1:
        mov     R5, #250
next11:
        mov     R6, #6
        mov     R1, #0FEH
        mov     R2, #0
next12:
        mov     A, R1
        mov     P1, A
        RL      A
        mov     R1, A
        mov     A, R2
        inc     R2
        mov     P2, A
        call    delay1          ; ===KKK===
        djnz    R6, next12
        djnz    R5, next11
        mov     P1, #0FFH
        call    delay2          ; ===LLL===
        djnz    R7, next1
        mov     R7, #6
next2:
        mov     P1, #0F8H
        mov     P2, #5
        call    delay2          ; ===III===
        mov     P1, #0C7H
        mov     P2, #1
        call    delay2          ; ===JJJ===
        djnz    R7, next2
        jmp     start
delay1:
        push    1
        mov     R1, #200
        djnz    R1, $
        pop     1
```

```
        ret
delay2:                         ; appx. 0.5sec delay, why?
        push    1
        push    2
        push    3
        mov     R1, #100
dd22:   mov     R2, #250
dd21:   mov     R3, #10
        djnz    R3, $
        djnz    R2, dd21
        djnz    R1, dd22
        pop     3
        pop     2
        pop     1
        ret
        end
```

** The sample code guarantees neither syntax err-free nor runtime err-free. Fix all syntax errs due to typo or whatever causes.

(a.3) **task execution:**

** start IDE51, download sample code (in HEX file format) from code preparation step (a.2)

** start execution and trouble-shooting if necessary

(b) Observations

(b.1) Is the code running well? Why or why not?

(b.2) What may happen to the display if the instruction marked by **===III===** being omitted? Why so? And what about the consequence of omitting the line marked by **===JJJ===**? And **===LLL===**?

(b.3) Is the delay provided by **delay1** appropriate considering execution time balance between the 4 phases? Too short? Too long?

(b.4) What might happen if the *R1-push* and *R1-pop* instructions are omitted in **delay1**? And the omitting of *push-pop* instructions in **delay2**?

(b.5) Can you modify the code so as to make it shorter, quicker, in better code structure or smoother execution?

[4] **Comprehension evaluation:**

(a) If the code line marked by ===KKK=== are removed, do you still see the pattern 5-4-3-2-1-0 **VERY** clearly? Explain why so or not so. What would you expect to see if the delay offered by **delay1** is made 1000 times of the original value?

(b) Do the code lines marked by ===KKK=== actually resolved the problem in [4](a)? With the circuitry unchanged, could the problem be solved purely by S/W measure alone? Suppose the problem could be really remedied by a small modification on the circuitry, what would it be?

[5] **Designated Assignment**

Let **k**=mod(T, 6), where T is the table# in the laboratory. Please fulfill the assignment given in Q (**k**).

Q(0) Manually wire-up the circuit so that all 7-seg digits display “0”, without code-driving. Rewire the circuit so that “0” appears on digit-0, and again without code-driving. Explain the difference in the display intensity perceived in the two cases.

Q(1) Do the same as in (0), except for “1” appearing respectively on all digits and on digit-1 alone.

Q(2) Do the same as in (0), except for “2” appearing respectively on all digits and on digit-2 alone.

Q(3) Do the same as in (0), except for “3” appearing respectively on all digits and on digit-3 alone.

Q(4) Do the same as in (0), except for “4” appearing respectively on all digits and on digit-4 alone.

Q(5) Do the same as in (0), except for “5” appearing respectively on all digits and on digit-5 alone.