8051 (89S52)

(*)Quy định:

Thi trên mô phỏng	Kết quả nộp trên UTEX	Điểm TK <5đ, thi lại buổi chiều
Tài liệu offline (giấy/laptop)	Sai quy định, thì "rớt môn"	Không hỏi bạn khác, có vấn đề thì trao đổi với Thầy (lúc thi)
TG KT = ASM + C (chung)		

[Bài kiểm tra ASM] **Gồm: Led, button, Timer, Led7seg**

Đề bài: Hệ thống gồm 2 led đơn

- Bật/tắt bằng 4 nút nhấn

- Led sáng 3s rồi tự tắt (Led7seg hiện số 3 rồi đếm lùi về 0 thì led đơn tắt)

- Led7seg hiện 2 số (mỗi số là thời gian sáng của led đơn tương ứng) (2 Led đơn có thể sáng cùng lúc.)

(*)Lưu ý:

- Chạy được 100% ý nào thì chấm ý đó.

- Cho phép chọn chân tùy ý (phù hợp với đề bài)

[Bài kiểm tra C] **Gồm: Led, button, Led7seg, Serial, (Nên xem Interrupt Timer (4/4) in C)**

Đề bài: Điều khiển 2 led đơn chớp tắt với tốc độ tùy chỉnh

- Cho led 1 chớp tắt với chu kỳ 2S (On 1s, Off 1s)

- Cho led 2 chớp tắt với chu kỳ 1S

- Thay đổi chu kỳ bằng 2 nút nhấn (1 nút tăng, 1 nút giảm). Mỗi lần tăng/giảm 0.5s. Chu kỳ thuộc đoạn [1: 5] (s).

- Thay đổi chu kỳ từng Led bằng cách gửi từ Terminal. Cấu trúc chuỗi (tùy bạn quy ước.) (VD: gửi "105" tức Led 1, chu kỳ 0.5S)

- Led7seg hiện giá trị chu kì của 2 led đơn.

(*)Lưu ý:

- Chạy được 100% ý nào thì chấm ý đó.

- Cho phép chọn chân tùy ý (phù hợp với đề bài)

Lệnh: xyz

x = số LED (1 hoặc 2)

y,z = hai số của chu kỳ × 0.1s

VD:

"105" → LED1 chu kỳ 0.5s (500 ms)

"110" → LED1 chu kỳ 1.0s

"205" → LED2 chu kỳ 0.5s

"210" → LED2 chu kỳ 1.0s

- Gợi ý làm Bài: "Vừa có Led đơn chớp tắt, vừa có Led 7 đoạn quét", thì nên dùng thêm 1 bộ Timer 0 để:

+ Timer0: chạy tự động Led đơn

+ while(1) main: chạy quét Led 7 đoạn.

(Hoặc ngược lại)

BÀI TẬP ASM

```
SEGMENT_PORT EQU P0
VALUE        DATA 30H
COUNT_END EQU 7

RUN_FLAG     EQU 31H

LED_CTRL     BIT P2.0
LED_MANUAL   BIT P2.1

BTN_ON     BIT P3.0
BTN_OFF    BIT P3.1
BTN_START    BIT P3.3
BTN_STOP     BIT P3.4


ORG 0000H
    SJMP MAIN

MAIN:
    SETB LED_MANUAL
    MOV VALUE, #COUNT_END       ; start
    MOV RUN_FLAG, #00H
```

```
MAIN_LOOP:
    JNB BTN_ON, TURN_ON
    JNB BTN_OFF, TURN_OFF

    JNB BTN_START, START_COUNTDOWN
    JNB BTN_STOP, STOP_COUNTDOWN

    MOV A, RUN_FLAG
    JZ MAIN_LOOP
    SJMP DO_COUNT

START_COUNTDOWN:
    CLR LED_CTRL
    MOV VALUE, #COUNT_END
    MOV RUN_FLAG, #01H
    LCALL DEBOUNCE
    SJMP MAIN_LOOP

STOP_COUNTDOWN:
    SETB LED_CTRL
    MOV RUN_FLAG, #00H
    LCALL DEBOUNCE
    SJMP MAIN_LOOP

TURN_ON:
    CLR LED_MANUAL
    SJMP MAIN_LOOP

TURN_OFF:
    SETB LED_MANUAL
    SJMP MAIN_LOOP

DEBOUNCE:               ; 20ms debounce
    MOV R2, #40
DB1:
    MOV R1, #250
DB2:
    DJNZ R1, DB2
    DJNZ R2, DB1
    RET

DO_COUNT:
    JNB BTN_STOP, STOP_COUNTDOWN

    MOV DPTR, #TABLE
    MOV A, VALUE
    MOVC A, @A+DPTR
    MOV SEGMENT_PORT, A
```

```
    JNB BTN_ON, TURN_ON
    JNB BTN_OFF, TURN_OFF

     MOV R4, #COUNT_END
WAIT_LOOP:
        LCALL DELAY_200MS
        JNB BTN_STOP, STOP_COUNTDOWN
        JNB BTN_START, START_COUNTDOWN
        DJNZ R4, WAIT_LOOP

    DJNZ VALUE, MAIN_LOOP
    MOV SEGMENT_PORT, #3Fh ; end = 0
    SETB LED_CTRL
    LCALL DELAY1S

    MOV VALUE, #COUNT_END
    CLR LED_CTRL
    SJMP MAIN_LOOP

;--------------------------------delay
DELAY_200MS:
    MOV R2, #200
D1:
    MOV R1, #200
D2:
    DJNZ R1, D2
    DJNZ R2, D1
    RET

DELAY1S:
    MOV R3, #5
L1:
    LCALL DELAY_200MS
    DJNZ R3, L1
    RET
;------------------------------------------------------------ delay timer
DELAY_10MS:
    MOV TMOD, #01H      ; Timer0 Mode1 (16-bit)
    MOV TH0, #0D8H
    MOV TL0, #0F0H

    SETB TR0            ; b?t Timer0

WAIT_T0:
    JNB TF0, WAIT_T0
    CLR TR0
    CLR TF0
    RET

DELAY_1S:
```

```
    MOV R7, #100
DELAY_LOOP:
    LCALL DELAY_10MS
    DJNZ R7, DELAY_LOOP
    RET


TABLE:
    DB 03FH
    DB 006H
    DB 05BH
    DB 04FH
    DB 066H
    DB 06DH
    DB 07DH
    DB 007H
    DB 07FH
    DB 06FH

END
```

BÀI TẬP C

```
#include <REGX51.H>

#define SEGMENT_PORT P0
#define SELECT_PORT  P2
sbit LED1 = P1^0;
sbit LED2 = P1^7;

volatile unsigned int period1 = 1000;
volatile unsigned int period2 = 500;
unsigned int c1 = 0, c2 = 0;
char cmd[3];
unsigned char uart_index = 0;

void small_delay()
{
    unsigned int i;
    for(i = 0; i < 200; i++);
}
void delay_ms(unsigned int ms) {
    unsigned int i, j;
    for (i = 0; i < ms; i++)
        for (j = 0; j < 123; j++);
}
void timer0_isr(void) interrupt 1
```

```
{
    TH0 = 0xFC;
    TL0 = 0x66;   // 1ms tick
    c1++;
    c2++;
}

void timer0_init()
{
    TMOD |= 0x01;  // Mode 1
    TH0 = 0xFC; TL0 = 0x66;
    ET0 = 1;
    TR0 = 1;
}
unsigned char digit_patterns[] = {
    0x3F,   // 0
    0x06,   // 1
    0x5B,   // 2
    0x4F,   // 3
    0x66,   // 4
    0x6D,   // 5
    0x7D,   // 6
    0x07,   // 7
    0x7F,   // 8
    0x6F    // 9
};
void display_digit(unsigned char index, unsigned char value, unsigned char dot)
{
        if (dot == 0){
            SELECT_PORT = (index << 2);
            SEGMENT_PORT = digit_patterns[value];
            delay_ms(1);
        }
        else
        {
            SELECT_PORT = (index << 2);
            SEGMENT_PORT = digit_patterns[value] | 0x80;
            delay_ms(1);
        }

}
void external0_isr(void) interrupt 0
{

    period1 += 250;
    period2 += 250;

    if(period1 > 2500) period1 = 2500;
    if(period2 > 2500) period2 = 2500;
}
```

```c
void external1_isr(void) interrupt 2
{
    period1 -= 250;
    period2 -= 250;

    if(period1 < 500) period1 = 500;
    if(period2 < 500) period2 = 500;
}

void UART_Init(void) {
    SCON = 0x50;        // UART mode 1, enable receiver
    TMOD &= 0x0F;       // Clear Timer1 config
    TMOD |= 0x20;       // Timer1 mode 2
    TH1 = 0xFD;         // Baud 9600 (11.0592 MHz)
    TR1 = 1;            // Start Timer1
    TI = 1;             // Allow first transmit
}

void UART_SendChar(char c) {
    while (!TI);
    TI = 0;
    SBUF = c;
}

void UART_SendString(char *str) {
    while (*str) {
        UART_SendChar(*str);
        str++;
    }
}
void Process_Command(void)
{
    unsigned char led = cmd[0] - '0';
    unsigned int val  = (cmd[1]-'0') * 10 + (cmd[2]-'0');
    unsigned int new_period = val * 50;

    if(new_period < 500) new_period = 500;
    if(new_period > 2500) new_period = 2500;

    if(led == 1) period1 = new_period;
    else if(led == 2) period2 = new_period;

    UART_SendString("\r\nSet OK\r\n");
}

void UART_Receive_Handler(void)
{
    if(RI)
    {
```

```
        RI = 0;
        cmd[uart_index] = SBUF;
        uart_index++;
        if(uart_index >= 3)
        {
            Process_Command();
            uart_index = 0;
        }
    }
}

void main()
{

    LED1 = 1;
    LED2 = 1;

    IT0 = 1;
    IT1 = 1;
    EX0 = 1;
    EX1 = 1;
    EA  = 1;
        timer0_init();
        UART_Init();
        UART_SendString("\r\n=== DIEU KHIEN LED QUA UART ===\r\n");
    UART_SendString("Nhap lenh (VD: 105, 210)\r\n");

    while(1)
    {
                small_delay();
                c1++;
                c2++;
                UART_Receive_Handler();
        // LED1
        if(c1 >= period1)
        {
            LED1 = !LED1;
            c1 = 0;
        }
        // LED2
        if(c2 >= period2)
        {
            LED2 = !LED2;
            c2 = 0;
        }
                display_digit(1,period1/500,1);
                display_digit(0,period1%500/50,0);
                display_digit(7,period2/500,1);
                display_digit(6,period2%500/50,0);
    }
```

```
}
```

Bảng tóm tắt sơ đồ nối chân ngoại vi (trên mô phỏng)

VĐK	**8 Led đơn**	**8 nút nhấn đơn**	**Led 7seg**	**Serial**
80C51	P2	P3	P0, P2.2, P2.3, P2.4	P3.0, P3.1 (Timer1)
(*)Lưu ý: 1 số chân trùng lặp, khi dùng nhớ để ý, ví dụ: - Led 7seg: dùng 3 pin ở Port2, nên Led đơn nhớ tránh xài lại 3 pin này (Led đơn còn 5 led) - Serial: dùng Timer1, dùng 2 pin ở Port3, nên nút nhấn nhớ tránh xài lại 2 pin này (Nút nhấn còn 6 nút)				