TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH

KHOA ĐÀO TẠO CHẤT LƯỢNG CAO

NGÀNH KTĐT-TT

ĐÁP ÁN THI CUỐI KỲ HK 2 NĂM HỌC 2023
Môn: VI XỬ LÝ
Mã môn học: MICR330363
Đề số/Mã đề: 01 Đề thi có 2 trang.
Thời gian: 90 phút.
Được phép sử dụng tài liệu là một tờ A4 viết tay.

Chú ý: *SV được phép sử dụng tất cả thư viện trong sách và của trình biên dịch CCS C Compiler mà không cần phải viết lại trong bài thi.*

Câu 1 (3.0đ).

Một hệ thống đếm sản phẩm dùng TIMER/COUNTER của PIC16F887 có kết nối với các ngoại vi như sau: 1 LCD 16x2 giao tiếp với PORTD và PORTE; 1 cảm biến phát hiện sản phẩm – SV tùy chọn kết nối dựa vào bộ TIMER/COUNTER được sử dụng; 1 nút nhấn thường hở tên là **R_S** nối với chân RB0 có chức năng bật hoặc tắt bộ TIMER/COUNTER để cho phép đếm hoặc dừng đếm sản phẩm; hệ thống sử dụng dao động thạch anh 12Mhz.

Viết 1 chương trình duy nhất thực hiện các yêu cầu sau: (không cần vẽ sơ đồ nguyên lý)

a. Khi mới bật nguồn hệ thống sẽ đếm sản phẩm từ 0 đến 24. Nếu đếm đến 24 mà thêm 1 sản phẩm nữa thì quay về 1 và tiếp tục đếm chu kỳ mới. Hiển thị ở hàng trên cùng của LCD nội dung theo mẫu dưới với **XX** là số sản phẩm đếm được. ***(2.0đ).***

San Pham=**XX**(Cai)

b. Khi nhấn nút **R_S** trong lúc hệ thống đang đếm sản phẩm thì sẽ dừng đếm và hiển thị hàng cuối LCD nội dung **"DUNG DEM"**. Ngược lại nếu nhấn nút **R_S** trong lúc hệ thống dừng đếm sản phẩm thì sẽ cho phép đếm lại và hiển thị hàng cuối LCD nội dung **"DANG DEM"**. ***(*** *dừng đếm* ***0.5đ,*** *đếm lại* ***0.5đ)***

Chương trình

```
#include<16f887.h>
#fuses hs
#use delay (clock=12M)
#define     LCD_RS          PIN_E0
#define     LCD_RW          PIN_E1
#define     LCD_E           PIN_E2
#define     OUTPUT_LCD      OUTPUT_D
#include<TV_LCD.C>
unsigned int8 kq;
int1 ttdem=1;
void main()
{
      set_tris_b(0x01);     // Khai báo và set tris 0.25đ
      set_tris_c(0x01);
      set_tris_d(0);
      set_tris_e(0);
      lcd_setup();
      setup_timer_1(T1_EXTERNAL | T1_DIV_BY_1);
      set_timer1(0);
      while(true)
      {
            kq= get_timer1();  // cấu hình Timer và đọc được kết quả đếm 0.75đ
            if(kq>24){set_timer1(1);kq=1;} // Đúng giới hạn 0.5đ
            lcd_command(0x80);
            printf(lcd_data,"SAN PHAM=%02u(CAI)",kq);// Hiển thị đúng 0.5đ

            if(input(pin_b0)==0)   // Chống dội 0.25đ
```

```
            {
                delay_ms(20);
                if(input(pin_b0)==0)
                {
                    ttdem=!ttdem;
                    lcd_command(0xC0+4);
                    if(ttdem)   // Cho phép đếm lại đúng 0.25d
                    {
                        setup_timer_1(T1_EXTERNAL | T1_DIV_BY_1);
                        lcd_data("DANG DEM");
                    }
                    else        // Dừng đếm đúng 0.5d
                    {
                        setup_timer_1(T1_DISABLED);
                        lcd_data("DUNG DEM");
                    }
                    while(input(pin_b0)==0);
                }
            }
        }
}
```

Câu 2 (4.5đ).

Một hệ thống điều khiển và giám sát nhiệt độ động cơ gồm 2 vi điều khiển PIC16F887 gọi là VĐKA và VĐKB giao tiếp với nhau theo chuẩn UART với tốc độ baud là 19200.

VĐKA giao tiếp với:

- 1 màn hình LCD 16x2;
- 6 nút nhấn thường hở tên lần lượt là 0%, 20%, 40%, 60%, 80%, 100%;
- VĐKA sử dụng dao động thạch anh 10Mhz.

VĐKB giao tiếp với:

- 1 động cơ DC 24V-1A thông qua IC L298;
- 1 cảm biến LM35 (10mV/1^{o}C, tầm đo -55^{o}C đến 150^{o}C) thông qua kênh **AN0**;
- 1 biến trở 10K để tạo điện áp tham chiếu dương 1.5V cho bộ ADC;
- VĐKB sử dụng dao động thạch anh 10Mhz.

Mô tả hoạt động của hệ thống:

- Khi mới cấp nguồn động cơ không quay.
- Khi nhấn 1 trong 6 nút nhấn thì VĐKA sẽ gởi mã (SV tự quy ước) qua VĐKB để VĐKB điều khiển động cơ bằng xung PWM xuất ra từ chân CCP2 có độ rộng xung tương ứng với tên nút được nhấn. Ví dụ nhấn nút 20% thì VĐKB sẽ điều khiển động cơ bằng xung PWM xuất ra từ CCP2 có độ rộng xung là 20%.
- VĐKB đo nhiệt độ động cơ thông qua cảm biến LM35 được gắn trên vỏ động cơ rồi gởi giá trị nhiệt độ đo được qua VĐKA để hiển thị trên hàng đầu LCD.

SV thực hiện các yêu cầu sau:

a. Biết rằng hệ thống sử dụng ADC 10 bit, Vref$^-$ =0V, Vref$^+$ = 1.5V. SV hãy lập công thức tính giá trị nhiệt độ theo kết quả chuyển đổi ADC (gọi là **kqadc**) theo mẫu sau: ***(0.5đ)***

t^o = **kqadc* A**

Trong đó t^o là nhiệt độ theo đơn vị ^{o}C và **A** là giá trị SV cần tìm.

$t^o = \frac{kqadc}{0.01} x \frac{(vref^+ - vref^-)}{2^n - 1} = \frac{kqadc}{0.01} x \frac{(1.5-0)}{2^{10}-1} = kqadc * 0.147$

Tính đúng 0.5đ

b. Hệ thống sử dụng xung PWM có chu kỳ là 0.8ms. SV hãy tính toán giá trị PR2, bộ chia trước của TIMER2 và các giá trị hệ số chu kỳ tương ứng với các độ rộng xung 0%, 20%, 40%, 60%, 80%, 100%. ***(0.5đ)***

Ta có: $T = (PR2+1)*4*T_{osc}*PV_{T2}$

Chọn: $PV_{T2} = 16$ *(bộ chia trước của TIMER2)*

⇨ PR2 = 124 // Tính đúng PR2 0.25đ

Gọi HSCK là hệ số chu kỳ: $HSCK_{Max} = (PR2+1)*4 = 500$

⇨ $HSCK_{0\%} = 500*0/100 = 0$

⇨ $HSCK_{20\%} = 500*20/100 = 100$

⇨ $HSCK_{40\%} = 500*40/100 = 200$

⇨ $HSCK_{60\%} = 500*60/100 = 300$

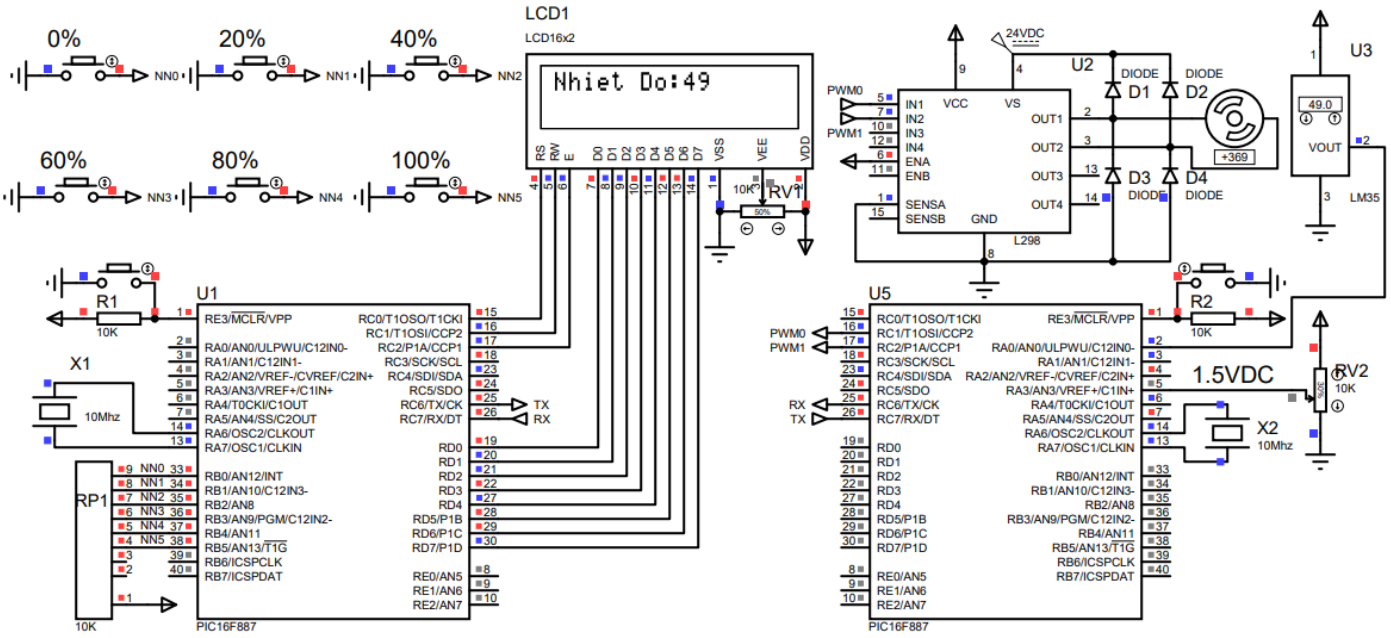
⇨ $HSCK_{80\%} = 500*80/100 = 400$

⇨ $HSCK_{100\%} = 500*100/100 = 500$

// Tính đúng các HSCK 0.25đ

c. Vẽ sơ đồ nguyên lý của hệ thống. ***(1.0đ)***

Sơ đồ nguyên lý



- Giao tiếp LM35 + UART 0.25đ
- Giao tiếp 6 nút nhấn 0.25đ
- Giao tiếp LCD 0.25đ
- Giao tiếp động cơ 0.25đ

-

d. Viết chương trình cho VĐKA theo mô tả trên. ***(****Nhấn các nút gởi mã điều khiển* ***0.5đ,*** *nhận nhiệt độ và hiển thị* ***0.5đ).***

Chương trình

```
#include<16f887.h>
#fuses hs
#use delay (clock=10M)
#use RS232(baud =19200, xmit = pin_c6, rcv = pin_c7)
#define      LCD_RS       PIN_C0
#define      LCD_RW       PIN_C1
#define      LCD_E        PIN_C2
#define      OUTPUT_LCD   OUTPUT_D
#include<TV_LCD.C>
unsigned int8 nhan;
void main()
{
```

```
        set_tris_b(0xff);
        set_tris_c(0x80);
        set_tris_d(0);
        lcd_setup();                              // Khai báo và set tris 0.25đ
        enable_interrupts(INT_RDA);
        enable_interrupts(GLOBAL);
        while(true)
        {
              if(input(pin_b0)==0) putc(0);   // Nhấn phím gởi mã đúng 0.25đ
              if(input(pin_b1)==0) putc(1);
              if(input(pin_b2)==0) putc(2);
              if(input(pin_b3)==0) putc(3);
              if(input(pin_b4)==0) putc(4);
              if(input(pin_b5)==0) putc(5);
              lcd_command(0x80);                  // Hiển thị đúng nhiệt độ 0.25đ
              printf(lcd_data,"Nhiet Do:%u   ",nhan);
        }
}
#INT_RDA
void ngatnhanuart()                               // Nhận được dữ liệu UART 0.25đ
{
        nhan = getc();
}
```

e. Viết chương trình cho VĐKB theo mô tả trên. ***(****Nhận mã điều khiển và điều khiển động cơ* ***0.75đ,*** *đo và gởi nhiệt độ* ***0.75đ).***

Chương trình

```
#include<16f887.h>
#device    adc = 10
#fuses hs
#use delay (clock=10M)
#use RS232(baud =19200, xmit = pin_c6, rcv=pin_c7)
unsigned int8 nhan=0,kqadc;
void main()
{
        set_tris_a(0x09);
        set_tris_c(0x80);                         //Khai báo và set tris 0.25đ
        enable_interrupts(INT_RDA);
        enable_interrupts(GLOBAL);
        setup_adc(ADC_CLOCK_DIV_32);              // Cấu hình ADC đúng 0.25đ
        setup_adc_ports(SAN0|VSS_VREF);
        set_adc_channel(0);
        setup_timer_2(T2_DIV_BY_16,124,1);        // Cấu hình PWM đúng 0.25đ
        setup_ccp2(CCP_PWM);
        output_low(pin_c2);
        while(true)
        {
              kqadc = read_adc()*0.147;
              putc(kqadc);                        // Đo và gởi đúng nhiệt độ 0.25đ
              delay_ms(100);
              set_pwm2_duty((int16)nhan*100);// Điều khiển động cơ đúng 0.25đ
        }
}
#INT_RDA
void ngatnhanuart()    // Nhận được dữ liệu UART 0.25đ
{
        nhan = getc();
}
```

Câu 3 (2,5đ).

Một bảng điện tử dùng để thay cầu thủ trong bóng đá gồm PIC16F887 giao tiếp với các ngoại vi như sau: 1 nút nhấn thường hở **MOD** nối với chân RE0; 1 nút nhấn thường hở **UP** nối với chân RE1; 1 nút nhấn thường hở **DW** nối với chân RE2; 4 LED 7 đoạn anode chung trong đó có 2 LED 7 đoạn màu đỏ để hiển thị số áo cầu thủ thay ra và 2 LED 7 đoạn màu xanh lá để hiển thị số áo cầu thủ vào sân. **SV chọn 1 trong 2 cách** giao tiếp sau cho 4 LED 7 đoạn này:

- **Cách 1:** Giao tiếp trực tiếp thì sử dụng các PORT A, B, C, D.
- **Cách 2:** Giao tiếp theo phương pháp quét thì 8 đường dữ liệu nối với PORTD, 4 đường điều khiển quét được nối với các chân từ RB0 đến RB3.

Viết 1 chương trình duy nhất thực hiện các yêu cầu sau: (không cần vẽ sơ đồ nguyên lý)

Khi mới bật nguồn tất cả các LED 7 đoạn đều hiển thị số 0. Hệ thống hoạt động ở chế độ không chỉnh (nhấn **UP**, **DW** đều không tác dụng). ***(0.25đ)***. Khi nhấn nút **MOD** dấu chấm xuất hiện ở LED hàng đơn vị của 2 LED 7 đoạn màu đỏ để báo là đang trong chế độ chỉnh cầu thủ bị thay ra. Nhấn UP (để tăng), DW (để giảm) số áo của cầu thủ thay ra trong phạm vi từ 00-99 và hiển thị giá trị này trên 2 LED 7 đoạn đỏ ***(1.0đ)***. Nhấn **MOD** lần nữa sẽ chuyển sang chế độ chỉnh cầu thủ vào sân, ở chế độ này thì dấu chấm màu đỏ mất đi và thay vào đó là dấu chấm màu xanh xuất hiện ở LED hàng đơn vị của 2 LED 7 đoạn màu xanh. Tương tự nhấn UP, DW để chỉnh số áo cầu thủ vào sân trong phạm vi từ 00-99 và hiển thị trên 2 LED 7 đoạn xanh ***(1.0đ).*** Nếu nhấn **MOD** lần nữa thì xóa dấu chấm xanh và hệ thống trở về chế độ không chỉnh như ban đầu. ***(0.25đ)***

Chương trình

```
#include<16f887.h>
#fuses intrc_io
#use delay (clock=8M)
signed int8 mode=0,ra=0,vao=0;
const unsigned int8 m7d[]={0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90};
int1 inputcd(int16 pin)
{
      if(input(pin)==0)
      {
            delay_ms(20);
            if(input(pin)==0)
            {
               while(input(pin)==0);
               return 0;
            }
      }
      return 1;
}
void main()
{
      set_tris_a(0);
      set_tris_b(0);                    //Khai báo và set tris 0.25đ
      set_tris_c(0);
      set_tris_d(0);
      set_tris_e(0xff);
      while(true)
      {                                 // Đúng chế độ không chỉnh  0.25đ
                                        // chống dội 0.25đ + chỉnh mode đúng 0.25đ
            if(inputcd(pin_e0)==0){mode++;mode%=3;}
            if(inputcd(pin_e1)==0)      // chống dội 0.25đ + chỉnh UP đúng 0.25đ
            {
```

```
                ra +=((mode==1)&&(ra<99)) *1;
                vao+=((mode==2)&&(vao<99))*1;
            }
            if(inputcd(pin_e2)==0) // chống dội 0.25d + chỉnh DW đúng 0.25d
            {
                ra -=((mode==1)&&(ra>0)) *1;
                vao-=((mode==2)&&(vao>0))*1;
            }
            output_a(m7d[ra/10]); // Hiển dấu "." đúng yêu cầu 0.25d
            output_b(m7d[ra%10]-128*(mode==1));
            output_c(m7d[vao/10]); // Hiển thị giá trị vào-ra đúng 0.25d
            output_d(m7d[vao%10]-128*(mode==2));
        }
}
```