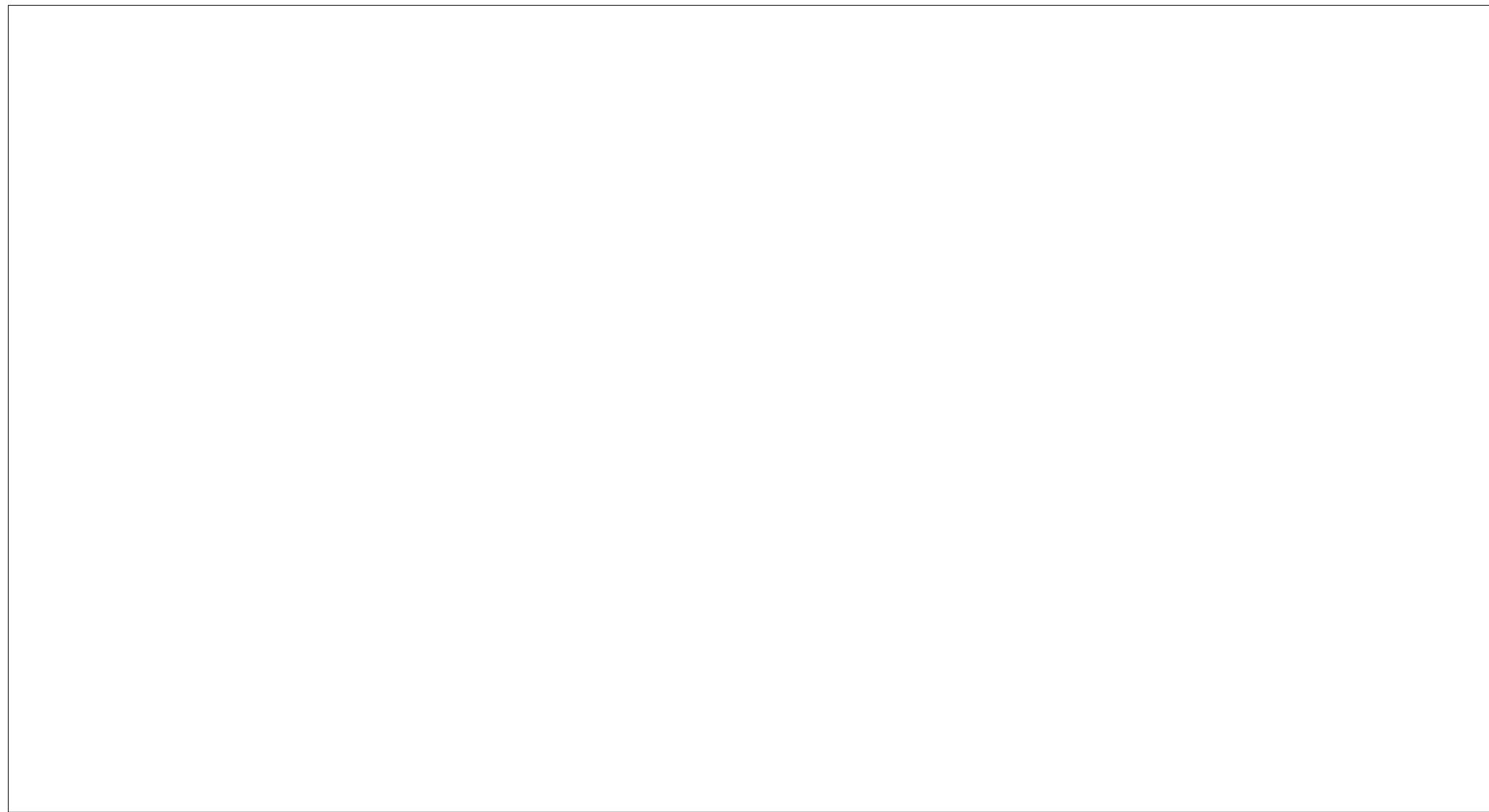


# MCU

## Online Meeting 1

Speaker: Le Quang Phuc



# MCU

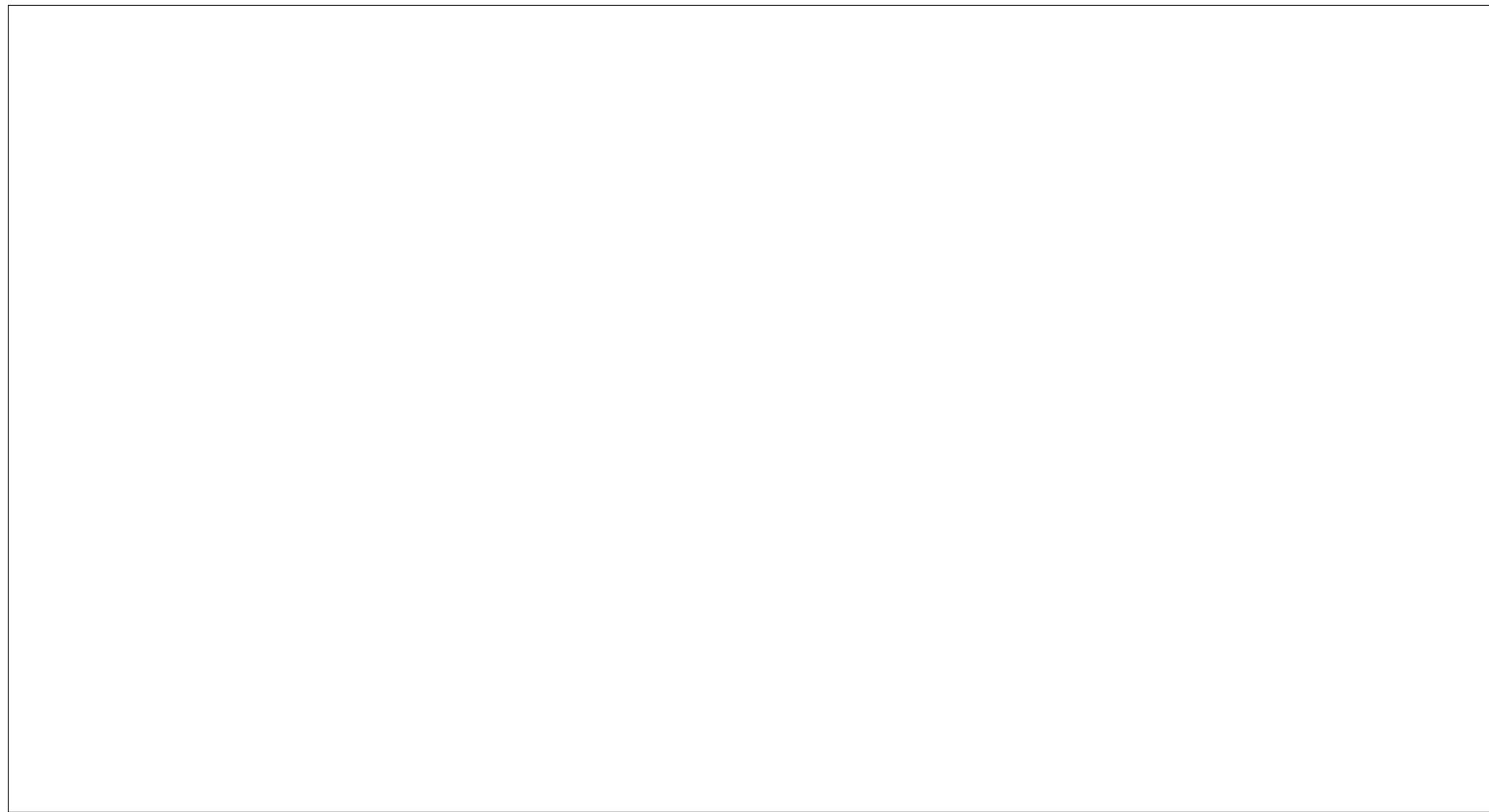
## Online Meeting 2

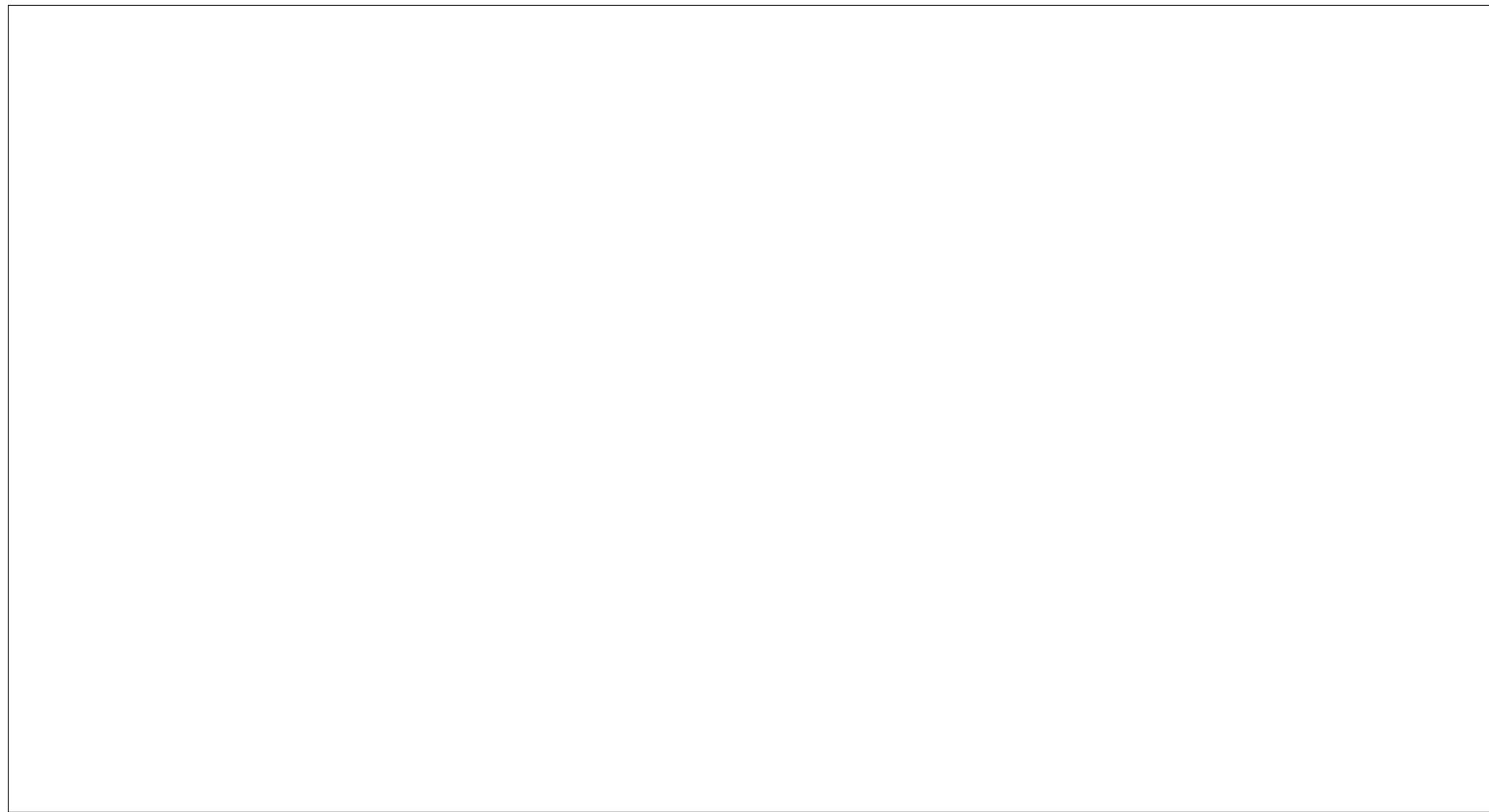
Speaker: Le Quang Phuc

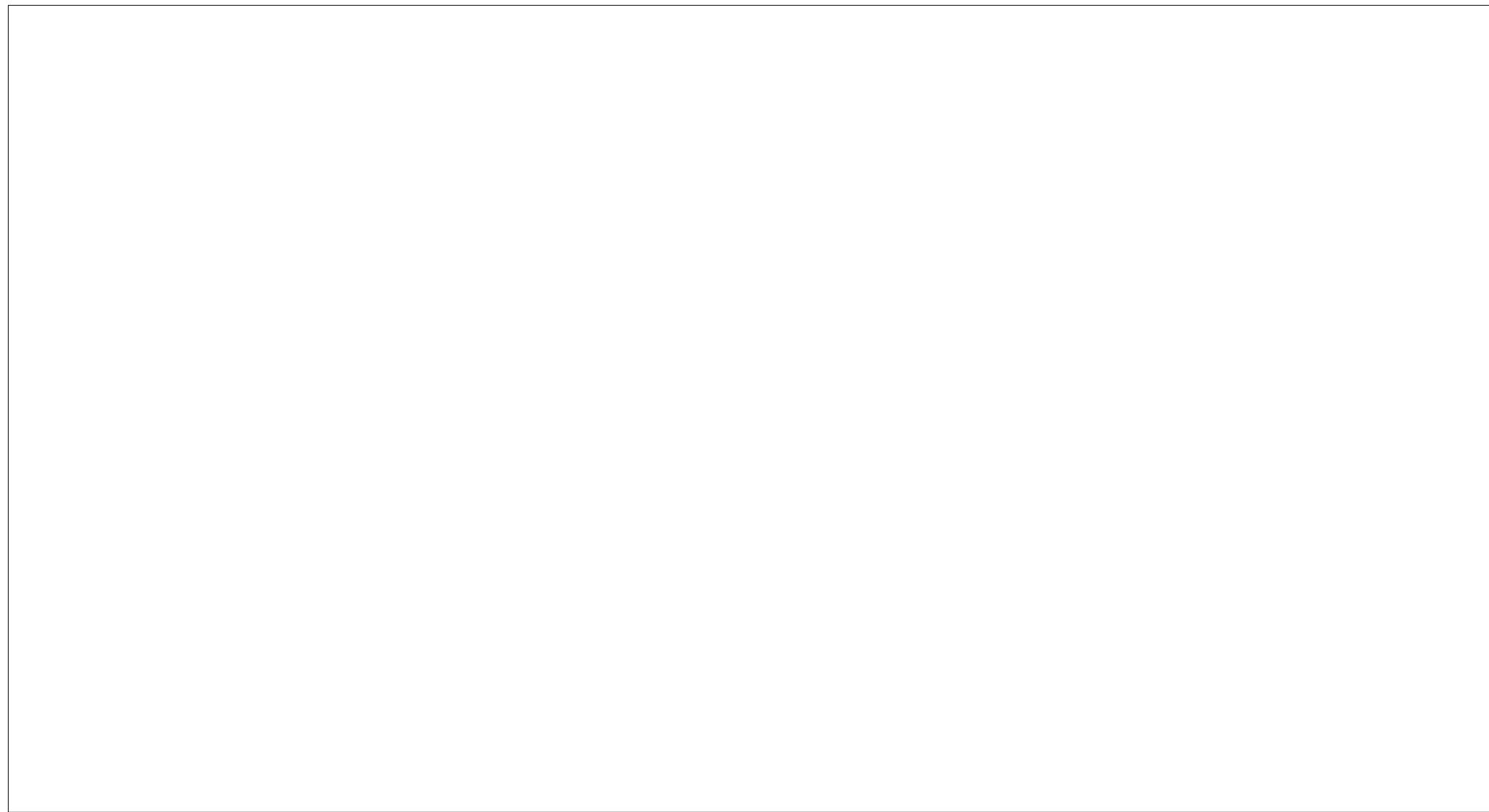
## Câu 2: (3 điểm)

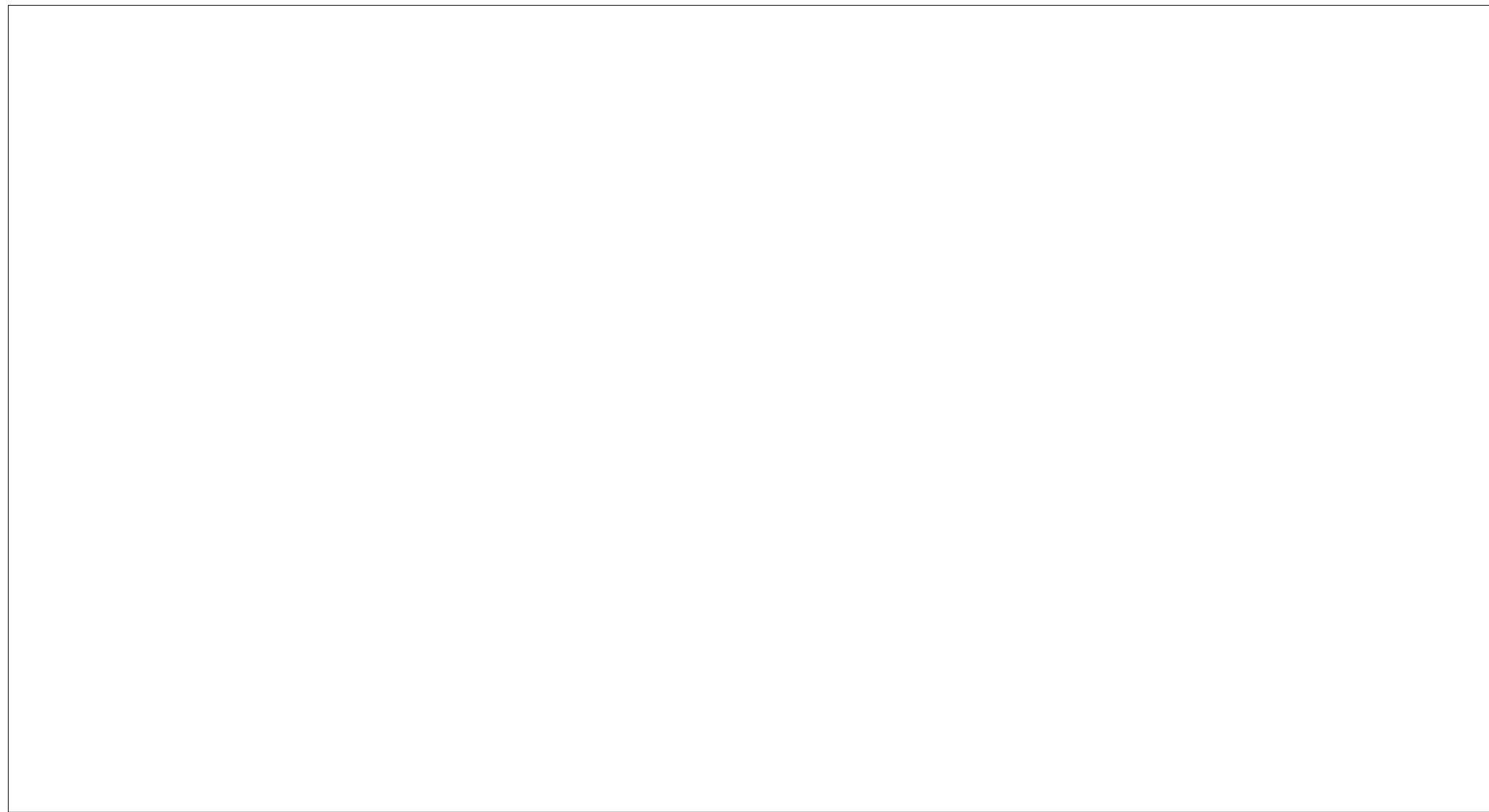
Một ứng dụng đo nhiệt độ 2 kênh, sử dụng cảm biến LM35, môi trường cần đo có nhiệt độ trong phạm vi [0°C --- 95°C]. Nhiệt độ được hiển thị luân phiên trên 2 LED bảy đoạn (được kết nối với **PORTC** & **PORTD**). Một LED bảy đoạn khác (**PORTB**) hiển thị số thứ tự của kênh nhiệt độ: Kênh 1 (**AN5**) hoặc Kênh 2 (**AN6**). Một nút nhấn CHANNEL (**RE2**) được sử dụng để chọn kênh nhiệt độ cần đo. **Mặc định lúc mới bật điện là Kênh 1.** Cho  $F_{OSC} = 20$  MHz. Yêu cầu nhiệt độ phải được đọc và **lấy trung bình giá trị** trong 100 lần trước khi hiển thị ra LED.

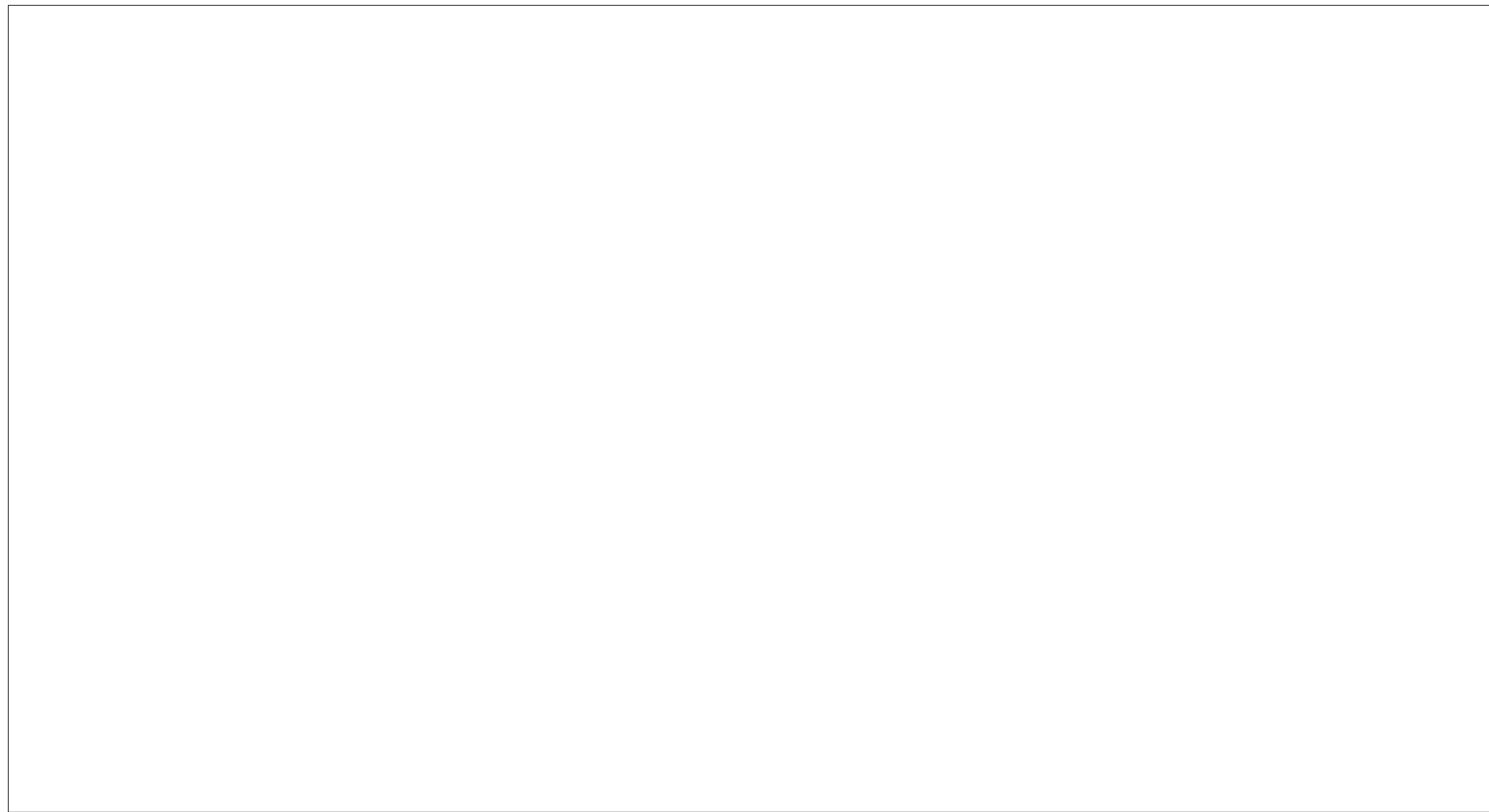
1. Vẽ mạch nguyên lý hoàn chỉnh (1 điểm)
2. Vẽ lưu đồ (1 điểm)
3. Viết chương trình C (1 điểm)











Một hệ thống điều khiển động cơ từ xa gồm 2PIC giao tiếp với nhau thông qua chuẩn UART

- VDK1 giao tiếp với 1 bàn phím ma trận 4x4, 1 màn hình LCD 20x4.
- VDK2 giao tiếp với một động cơ DC 24V có tích hợp sẵn encoder 300 xung/vòng thông qua IC L298,

### **SV thực hiện các yêu cầu sau:**

- a. Vẽ sơ đồ nguyên lý 1 cảm biến đo nhiệt độ động cơ. của hệ thống
- b. Viết chương trình cho VDK1 nhận 2 phím liên tiếp bé hơn 10 để cài đặt giá trị phần trăm duty cho động cơ và hiển thị giá trị này lên hàng 1 LCD. Khi nhấn phím F thì truyền giá trị này qua VDK2 để VDK2 điều khiển động cơ.

Ví dụ: muốn động cơ quay với duty là 79% thì ta nhấn phím 7 rồi phím 9 rồi sau đó phím F.

- c. Bổ xung chương trình cho VDK1 sau khi nhấn D thì dừng động cơ, nhấn C thì chạy lại.
- d. Viết chương trình cho VDK2 nhận giá trị điều khiển từ VDK1 để động cơ quay đúng yêu cầu câu b và c.

Chú ý: Tạo xung PWM với chu kỳ 0.2ms

- e. Viết chương trình cho VDK2 đo tốc độ động cơ và truyền về cho VDK1 để hiển thị lên LCD dòng số 2. **“Toc do: XXXX (rpm)”**
- f. Viết chương trình cho VDK2 đo nhiệt độ động cơ và truyền về cho VDK1 để hiển thị lên LCD dòng số 3. **“Nhiệt do: YY °C”**

# Ôn tập VXL

---

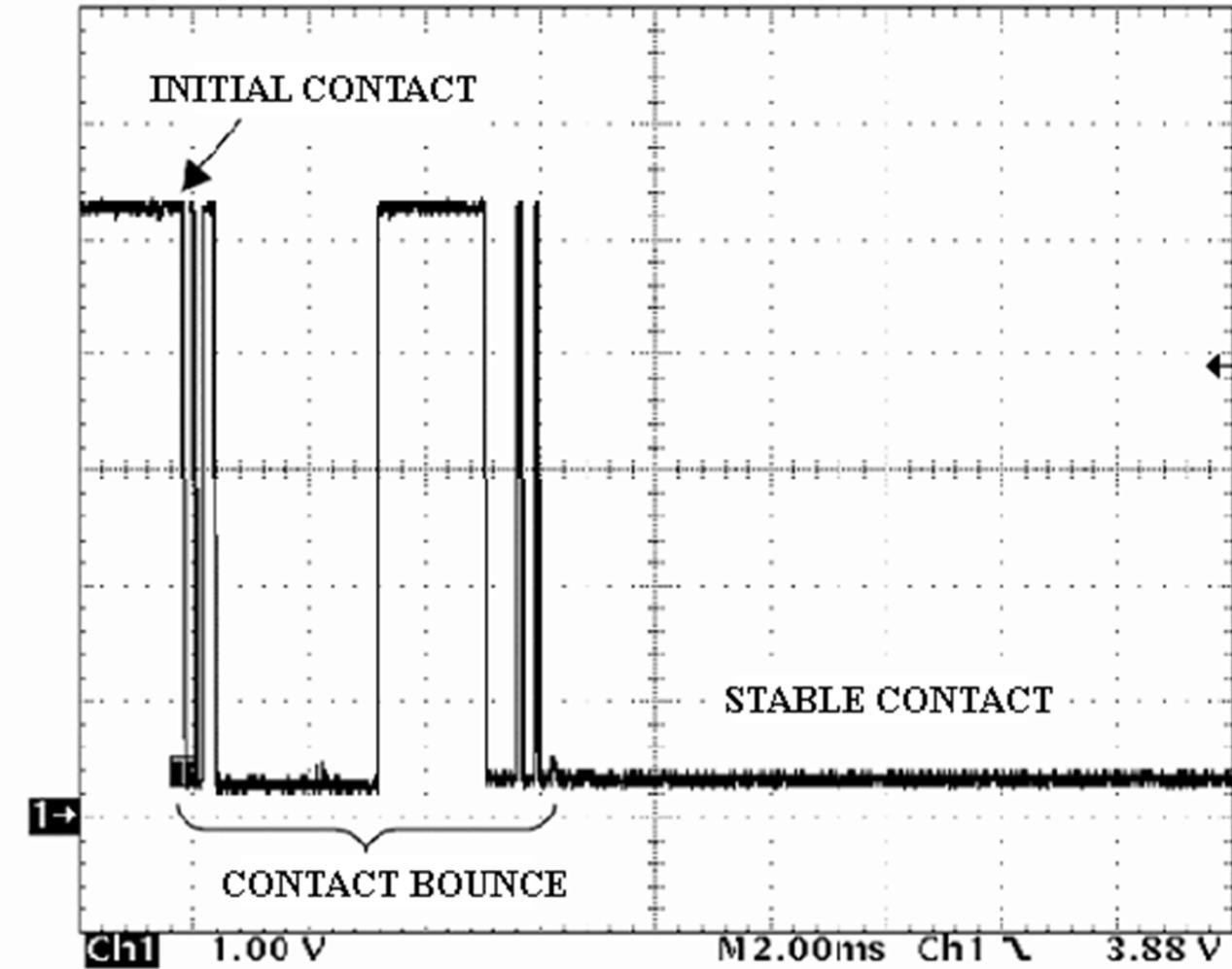
# CẤU TRÚC CHƯƠNG TRÌNH

```
#INCLUDE <TV_16F887.C>
#INCLUDE <TV_LCD.C>

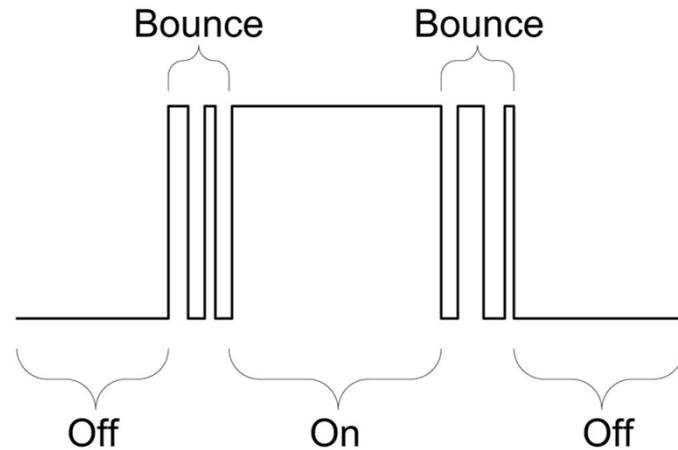
//KHAI BÁO UART
//ĐỊNH NGHĨA - NÚT NHẤN
//KHAI BÁO BIỂN
//VIẾT HÀM NGẮT - NẾU CÓ (UART, TIMER, ADC)
//VIẾT HÀM CON----->
//KHỞI TẠO BAN ĐẦU
SET_TRIS...
//KHỞI TẠO LCD, ADC, PWM, TIMER, COUNTER, NGẮT
//TRẠNG THÁI BAN ĐẦU CỦA HT - khi mới cắp điện

VOID MAIN()
{
    //CÔNG VIỆC THỰC HIỆN LIÊN TỤC
}
```

GM + HT LED 7 ĐOẠN  
GM\_LCD + HT\_LCD  
PHÍM NHẤN  
ĐỌC ADC



## • Chống dội nút nhấn



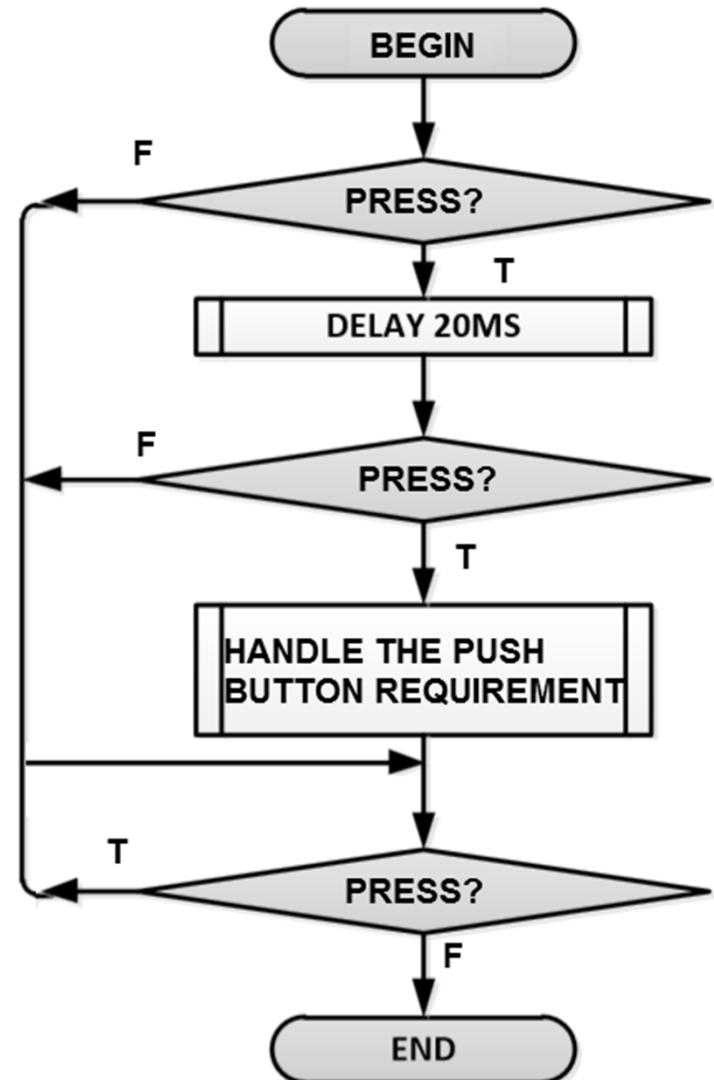
## • Xử lý nút nhấn

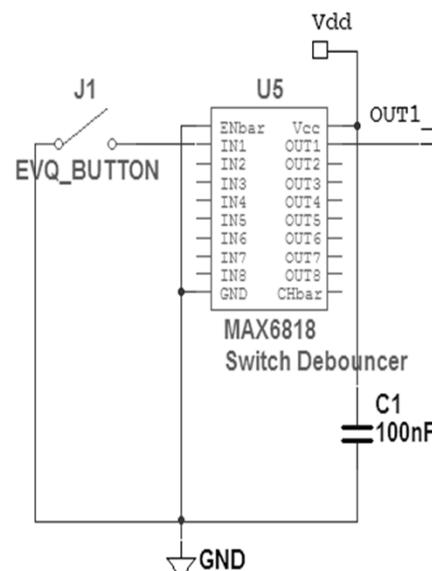
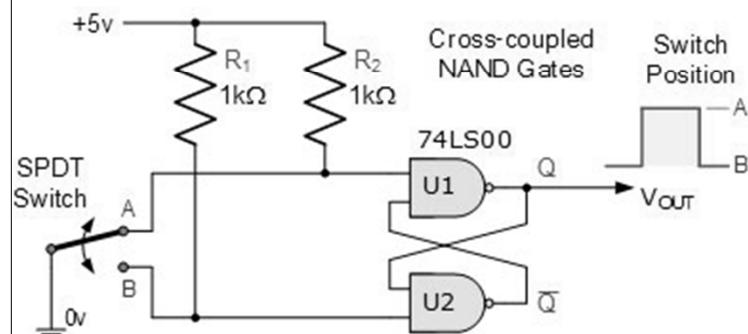
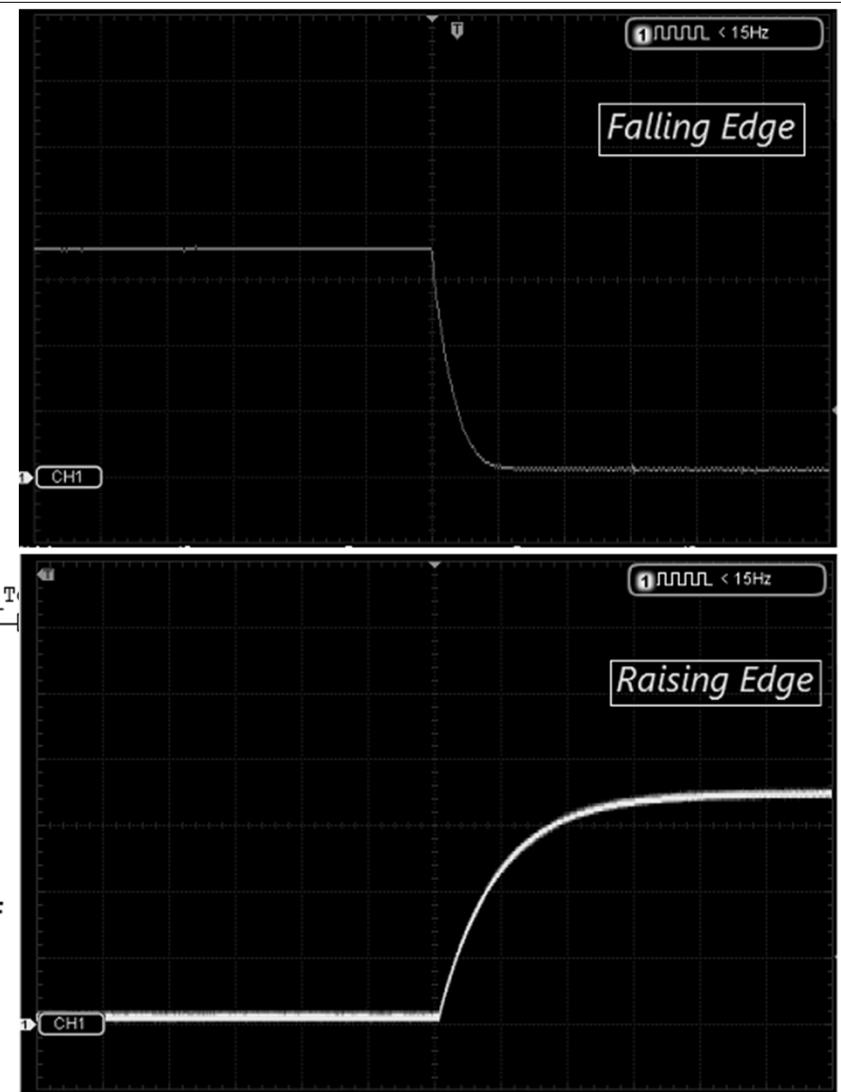
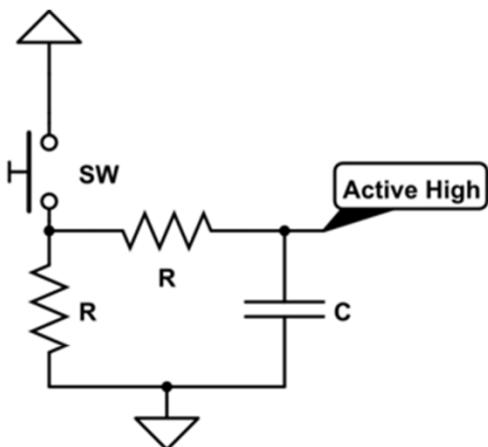
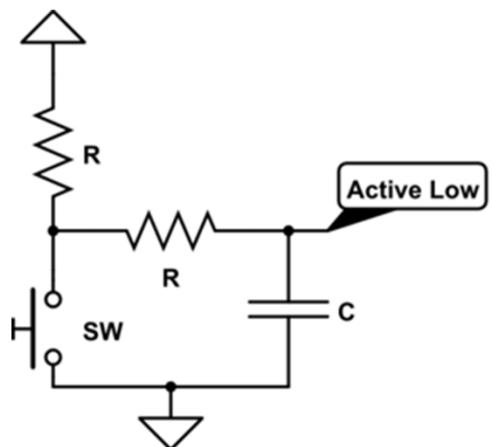
+ Trực tiếp

+ Dùng biến trạng thái

$TT = \sim TT$  (int1 nếu 2TH)

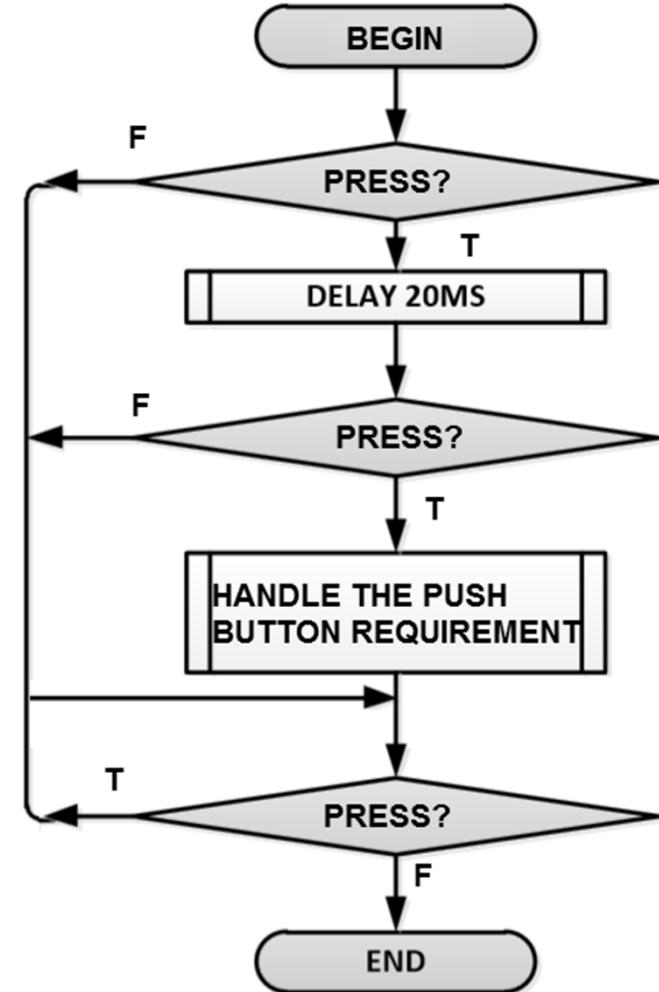
$TT++$  (int8 nếu >2TH)



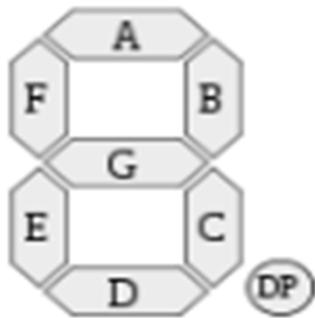


## CHƯƠNG TRÌNH CHỐNG DỘI NÚT NHẤN

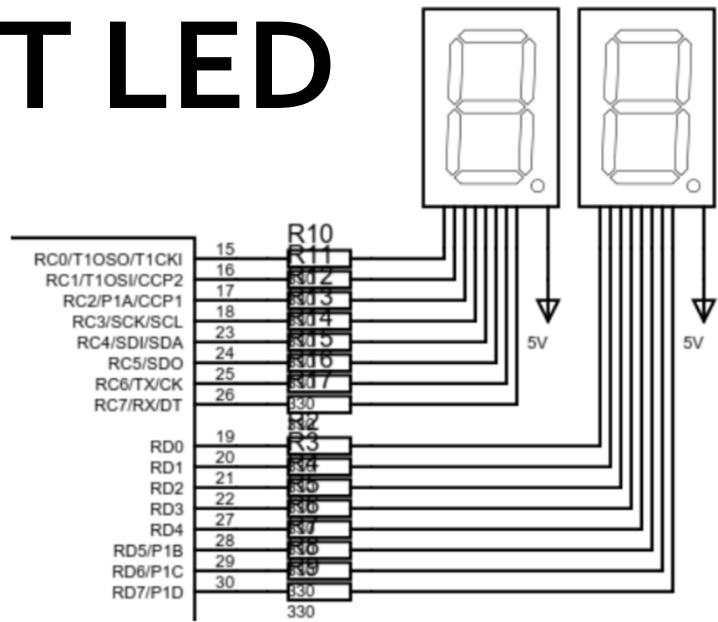
```
INT1 KTNN (INT1 PIN)
{
    IF (INPUT (PIN) == 0)
    {
        DELAY_MS (20);
        IF (INPUT (PIN) == 0)
        {
            WHILE (INPUT (PIN) == 0);
            RETURN 1;
        }
    }
    RETURN 0;
}
```



# SEVEN SEGMENT LED



0	DP	G	F	E	D	C	B	A	C0
1	1	0	0	0	0	0	0	0	0

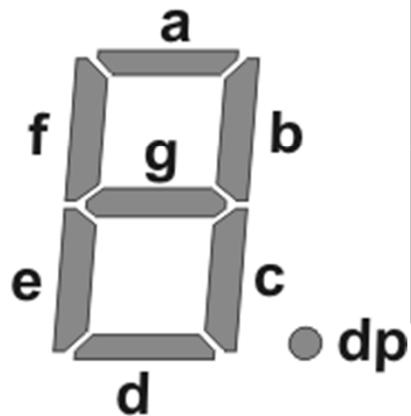


Tìm mã 7 đoạn:

- Đánh dấu seg sáng bằng số 0 (anode chung)
- Chuyển sang HEX

Hiển thị:

- Tách số
- Lấy mã
- Xuất ra GPIO



OCT	BIN								HEX	OCT	BIN								HEX
	7	6	5	4	3	2	1	0			7	6	5	4	3	2	1	0	
	DP	G	F	E	D	C	B	A			DP	G	F	E	D	C	B	A	
0	1	1	0	0	0	0	0	0	C0	5	1	0	0	1	0	0	1	0	92
1	1	1	1	1	1	0	0	1	F9	6	1	0	0	0	0	0	1	0	82
2	1	0	1	0	0	1	0	0	A4	7	1	1	1	1	1	0	0	0	F8
3	1	0	1	1	0	0	0	0	B0	8	1	0	0	0	0	0	0	0	80
4	1	0	0	1	1	0	0	1	99	9	1	0	0	1	0	0	0	0	90

```
#include<TV_16F887.c>

void hienthi7doan ()
{
    output_X(MA7DOAN[t/10%10]);
    output_Y(MA7DOAN[t%10]);
}
void main()
{
    Set_tris_X(0); // Cấu hình các port giao tiếp với LED là ngõ ra
    Set_tris_Y(0);

    // Gọi chương trình con "hienthi7doan()" nếu thực hiện 1 lần
    while(true)
    {
        // Gọi chương trình con "hienthi7doan()" nếu thực hiện nhiều lần
    }
}
```

# LED 7 ĐOẠN

## GIẢI MÃ HIỂN THỊ

```
VOID GIAI_MA()
{
    DV = MA7DOAN [DEM%10]; // TÁCH SỐ + LẤY MÃ
    CH = MA7DOAN [DEM/10%10];
    TR = MA7DOAN [DEM/100];

    IF(TR == 0XC0) // LẤY MÃ 7 ĐOẠN SO SÁNH VỚI MÃ 7Đ SỐ 0
    {
        TR = 0xFF; //MÃ TẮT LED - 1: TẮT (ANODE CHUNG)
        IF(CH == 0XC0)
            CH = 0xFF;
    }
}
```

```
const unsigned int8
ma7doan []={ 0xc0, 0xf9, 0xa4, 0xb0, 0x99, 0x92,
0x82, 0xf8, 0x80, 0x90};
```

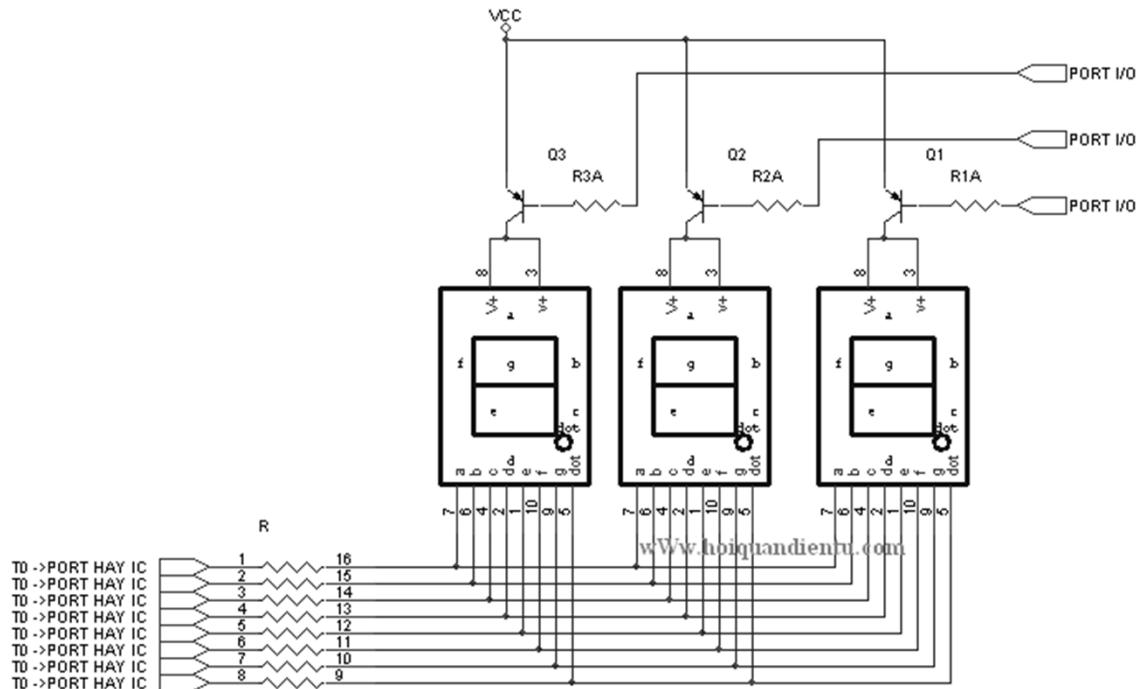
```
VOID HIEN THI_7DOAN()
{
    OUTPUT_B(TR); OUTPUT_C(CH); OUTPUT_D(DV);
}
```

# LED 7 ĐOẠN QUÉT

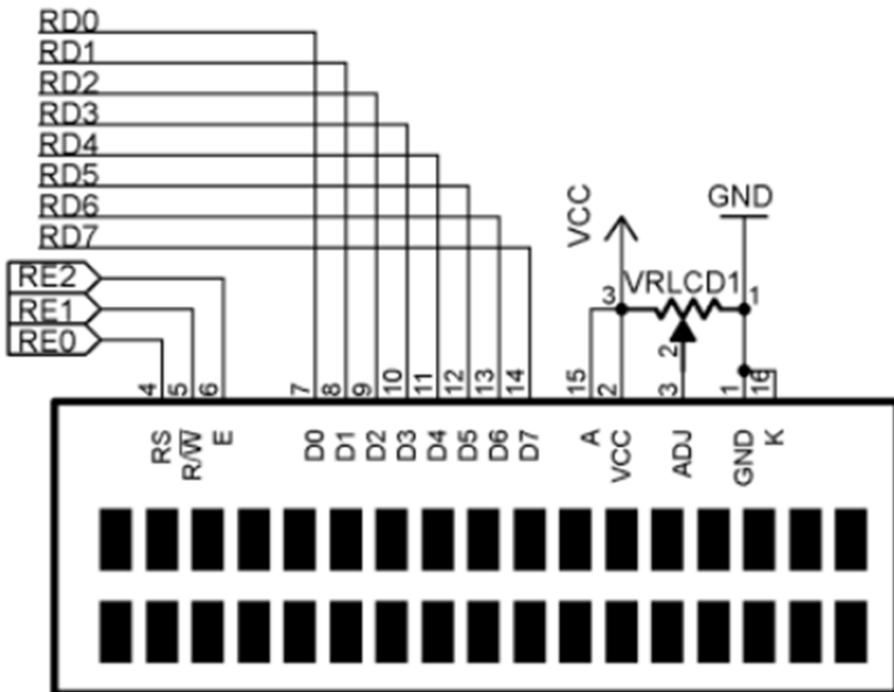
**Hiển thị:**

- Xuất dữ liệu
- Bật led
- Delay
- Tắt led

```
Void quetled()
{
    output_c(ma7doan[t/100]);    output_low(pin_d0); delay_ms(1); output_high(pin_d0);
    output_c(ma7doan[t/10%10]);  output_low(pin_d1); delay_ms(1); output_high(pin_d1);
    output_c(ma7doan[t%10]);     output_low(pin_d2); delay_ms(1); output_high(pin_d2);
}
```



# LCD



```
#INCLUDE <TV_16F887.C>
#define LCD_RS PIN_E0
#define LCD_RW PIN_E1
#define LCD_E PIN_E2
#define OUTPUT_LCD OUTPUT_D
#include <TV_LCD.C>
```

```
#define LCD_RS_PIN PIN_E0
#define LCD_RW_PIN PIN_E1
#define LCD_ENABLE_PIN PIN_E2
#define LCD_DATA4 PIN_D...
#define LCD_DATA5 PIN_D...
#define LCD_DATA6 PIN_D...
#define LCD_DATA7 PIN_D...
#include <LCD.C>
```

# LCD bus 8

1. **LCD\_SETUP();**
2. **LCD\_COMMAND( ); //GỎI MÃ LỆNH ĐIỀU KHIỂN**
3. **LCD\_DATA( ); // GỎI DỮ LIỆU HIỂN THỊ**

<b>0x01</b>	<b>Clear display</b>
0x80	Start address of line 1
0xC0	Start address of line 2

*Note: Định địa chỉ trước khi gọi dữ liệu hiển thị*

# LCD bus 4

```
1. LCD_INIT();  
2. LCD_GOTOXY(X,Y);  
    // X – COLUMN, Y – ROW  
    //Start from 1:1  
3. LCD_PUTC(DATA);  
4. PRINTF(LCD_PUTC,"SP=%03u", sp );  
    // If sp in unsigned int8 → 'u', unsigned int16 → "Lu"
```

**0x01**

**Clear display**

*Note: Định địa chỉ trước khi gọi dữ liệu hiển thị*

b7- b3- b4 -b0	0000	0010	0011	0100	0101	0110	0111	1010	1011	1100	1101	1110	1111
	CG RAM (1)	Ø	ø	P	~	P		—	ø	ø	ø	ø	p
0001	(2)	!	1	A	Q	a	q	o	P	手	4	ä	q
0010	(3)	"	2	B	R	b	r	「	イ	ウ	×	β	ø
0011	(4)	#	3	C	S	c	s	」	テ	エ	€	ø	ø
0100	(5)	\$	4	D	T	d	t	、	エ	ト	ト	μ	ø
0101	(6)	%	5	E	U	e	u	・	オ	ナ	1	ç	ø
0110	(7)	&	6	F	V	f	v	ヲ	カ	ニ	3	ρ	Σ
0111	CG RAM (8)	*	7	G	W	g	w	ア	ア	ラ	g	π	π
1000	CG RAM (1)	(	8	H	X	h	x	イ	ワ	リ	5	⊗	
1001	(2)	)	9	I	Y	i	y	セ	テ	リ	6	γ	
1010	(3)	*	:	J	Z	j	z	コ	ル	レ	3	≠	
1011	(4)	+	:	K	C	k	c	ア	サ	ヒ	□	×	△
1100	(5)	,	<	L	¥	l	l	フ	シ	フ	○	Φ	ø
1101	(6)	=	=	M	]	m	)	ユ	ス	ヘ	ン	±	ø
1110	(7)	.	>	N	^	n	→	ミ	セ	市	×	ñ	
1111	CG RAM (8)	/	?	O	_	o	←	ø	ø	ø	ø	ø	ø

```
void LCD_GOTOXY(int8 x, int8 y)
{ // hàng X, cột Y để chọn vị trí hiển thị
const unsigned int8 dc[]={0x80,0xC0};
LCD_COMMAND(dc[X]+Y);
}

void main()
{
    SET_TRIS_D(0);
    SET_TRIS_E(0);
    SETUP_LCD(); // CẤU HÌNH LCD
    LCD_COMMAND(0X80); // ĐỊNH ĐỊA CHỈ HIỂN THỊ
    LCD_DATA('A'); // GỬI DỮ LIỆU HIỂN THỊ 1 LẦN DUY NHẤT, HOẶC DỮ LIỆU CỐ ĐỊNH

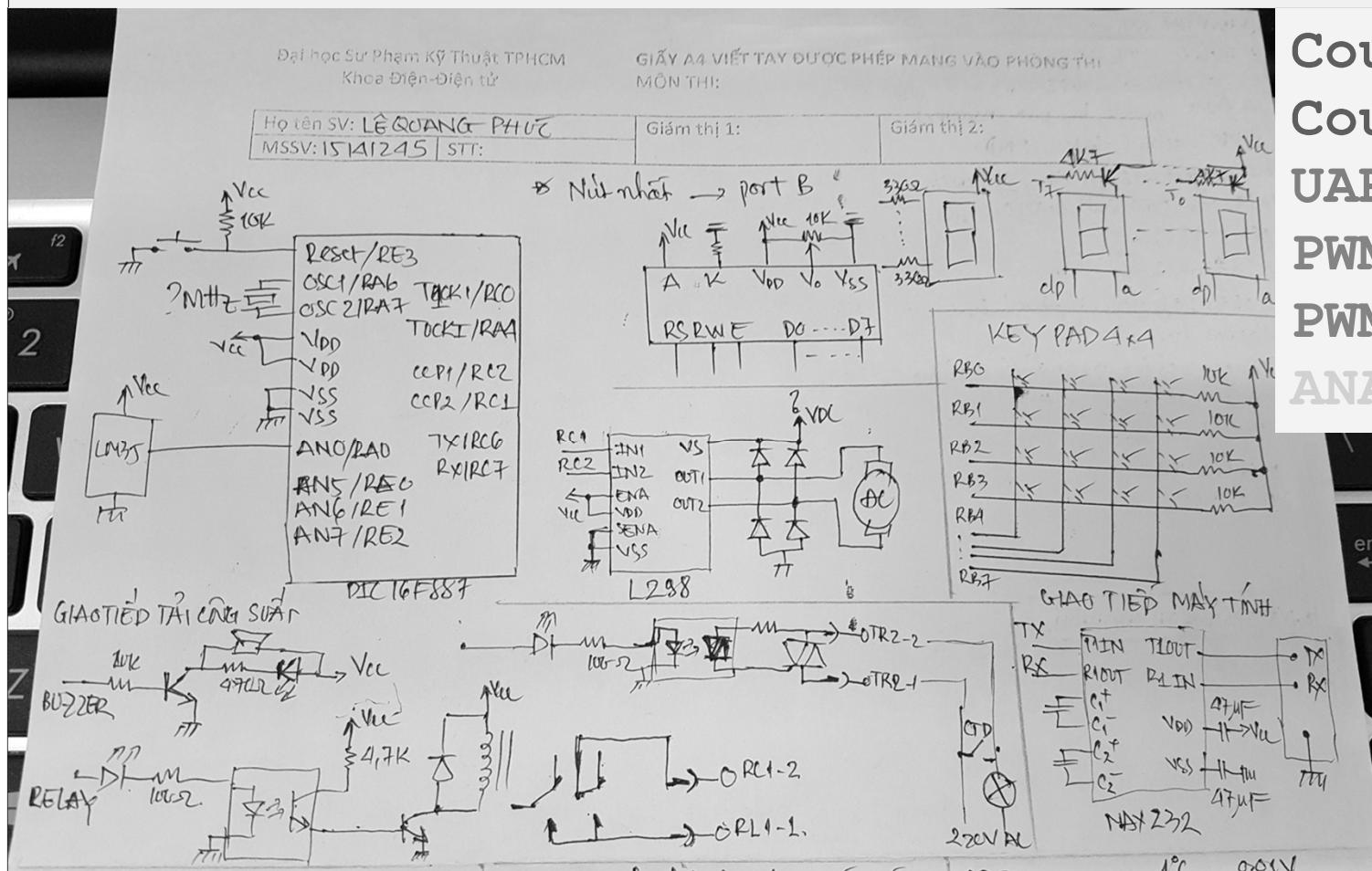
    WHILE(TRUE)
    {
        LCD_GOTOXY(1,0);
        LCD_DATA("HELLO PHUC");
        LCD_DATA(DEM%10 + 0X30); // GỬI DỮ LIỆU HIỂN THỊ THAY ĐỔI
    }
}
```

# LCD bus 4

```
void main()
{
    SET_TRIS_D(0);
    SET_TRIS_E(0);
    lcd_init();      // CẤU HÌNH LCD
    lcd_gotoxy(1,1); // ĐỊNH ĐỊA CHỈ HIỂN THỊ
    lcd_putc('A');   // GỎI DỮ LIỆU HIỂN THỊ 1 LẦN DUY NHẤT, HOẶC DỮ LIỆU CỐ ĐỊNH

    WHILE (TRUE)
    {
        LCD_GOTOXY(1,1);
        lcd_putc("HELLO PHUC");
        lcd_putc(DEM%10 + 0X30); // GỎI DỮ LIỆU HIỂN THỊ THAY ĐỔI
    }
}
```

# CÁC KẾT NỐI CƠ BẢN VỚI VI ĐIỀU KHIỂN



# Counter 0 → RA4

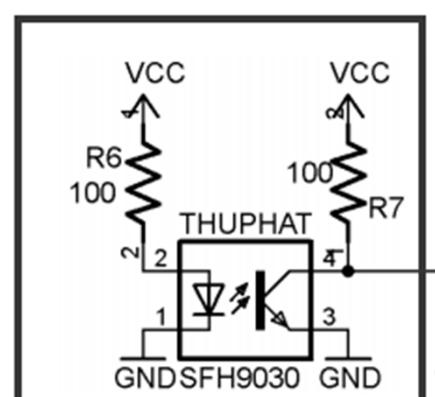
## Counter 1 → R<sub>C0</sub>

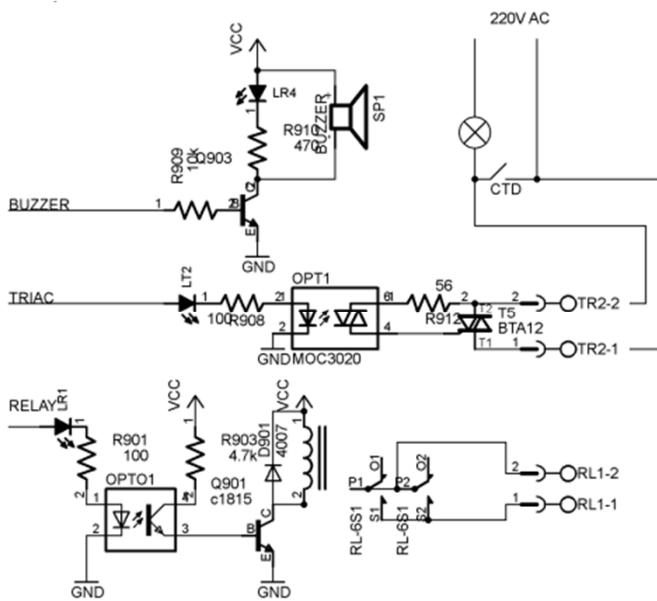
**UART → RC6 RC7**

PWM1 (CCP1) → RC2

**PWM2 (CCP2) → RC1**

**ANALOG → AN0–AN13**





Hình 7-8: Sơ đồ mạch giao tiếp điều khiển relay, triac, buzzer.

**Counter 0 → RA4**

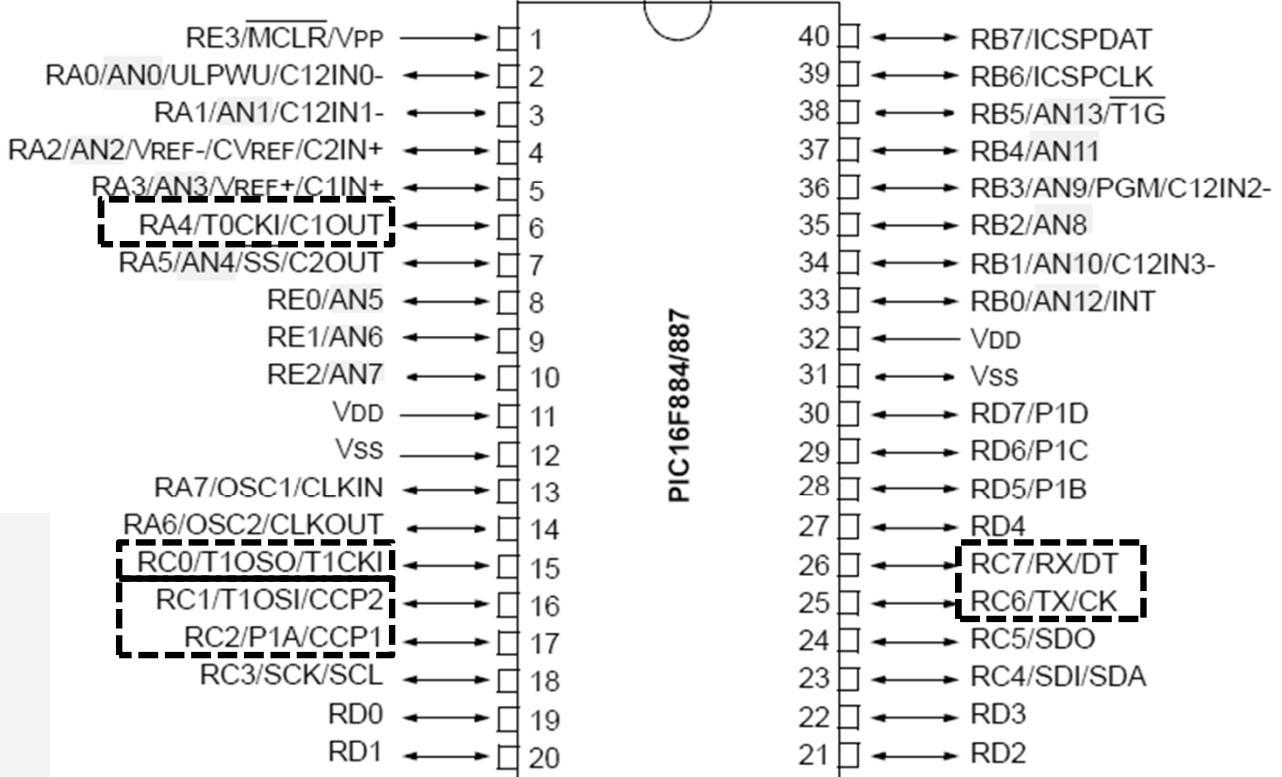
**Counter 1 → RC0**

**UART → RC6 – RC7**

**PWM1 (CCP2) → RC1**

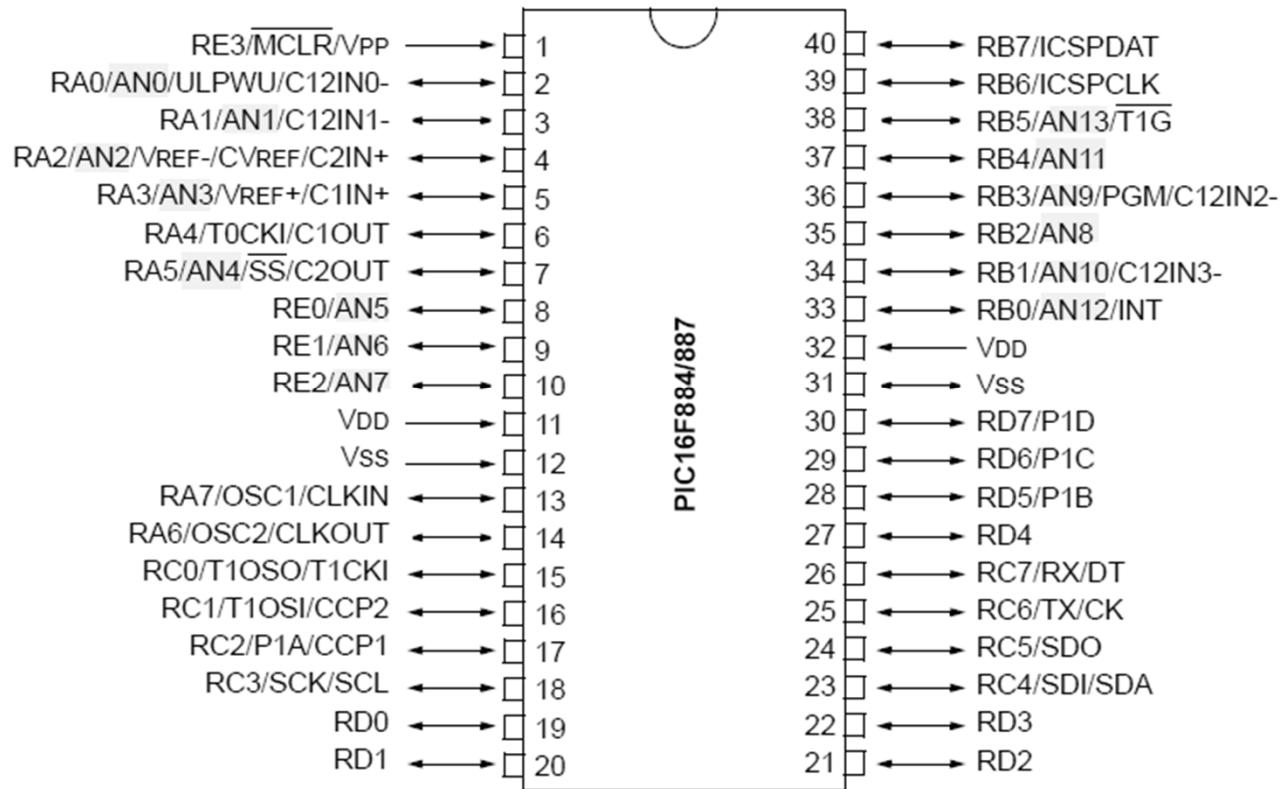
**PWM2 (CCP1) → RC2**

**ANALOG → AN0 – AN13**



# ADC

- **read\_adc(mode)**  
**adc\_read\_only**  
**ADC\_START\_ONLY**
- **setup\_adc(ADC\_CLOCK\_DIV\_32)**
- **set\_adc\_channel( x )** **x : 0-15**



## **setup\_adc\_ports(mode )**

**Configure pins used for ADC. mode can be:**

**sANo** : use pin ANo to be analog pin

...

**sAN15** : use AN15 to be analog input

**NO\_ANALOGS** : no use any analog pin

**ALL\_ANALOG** : all ANo to AN15 are analog

**VSS\_VDD** : use vss and vdd as vref- and vref+

**VSS\_VREF** : vref- = vss, vref+ external source

**VREF\_VREF** : vref- and vref+ are external source

**VREF\_VDD** : vref- from external source, vref+ = vdd

Use “|” between multi-choice

**Ex: SETUP\_ADC\_PORTS(SAN0|SAN5|VSS\_VDD);**

**setup\_adc(ADC\_CLOCK\_DIV\_32)**

**setup\_adc\_ports(san0|san1|VSS\_VDD)**

**set\_adc\_channel( x ) X : 0 → 15**

// Đọc 1 kênh thì để ngoài while(true), đọc nhiều kênh thì để trong while(true)

**Value = read\_adc();**

1	SS
Dout	? Vin
$\Rightarrow \text{Vin} = \text{Dout} * \text{SS}$	

LM35: 10mV/1°C	
1°C	0.01V
? t°	Vin

$$t^\circ = \frac{Vin}{0.01} = \frac{Dout}{0.01} \times \frac{(vref^+ - vref^-)}{2^n - 1}$$

$$t^\circ = \frac{Dout}{0.01} \times \frac{5}{2^{10} - 1} = \boxed{\frac{Dout}{2.046}}$$

# TIMER/COUNTER

**Tốc độ:**  $\leq 20\text{MHz}$  (dùng dao động thạch anh bên ngoài)

$$f_{\text{Làm\_việc}} = \frac{1}{4} f_{\text{Thạch\_Anh}}$$

( /4 là do 1 lệnh cần phải có 4 xung dao động của thạch anh)

$f_{\text{thạch\_anh}} = 20\text{Mhz} \rightarrow f_{\text{LV}} = 5 \text{ MHz} \rightarrow T = 200\text{ns}$   
 $\rightarrow$  VĐK thực hiện được **5 triệu lệnh/1s.**

# TIMER/COUNTER

## Timer 0:

Công thức tính thời gian định thời:

$$nạp = 256 - \frac{t * f}{4000 * ct * n}$$

Bộ chia trước từ 1 đến 256 ( $2^n$ )

## Timer 2:

Công thức tính thời gian định thời:

$$nạp = \frac{t * f}{4000 * ct * cs * n}$$

Ct: 1 hoặc 4 hoặc 16

Cs: 1 → 16

## Timer 1:

Công thức tính thời gian định thời:

$$nạp = 65536 - \frac{t * f}{4000 * ct * n}$$

Bộ chia trước 1 2 4 8

f: **tần số thạch anh**

t: **thời gian cần định thời (ms)**

ct: **bộ chia trước**

n: **số lần**

nạp: **Giá trị đưa vào set\_timerX();**

## LỆNH:

```
SETUP_TIMER_X();  
SET_TIMERX(nạp);  
GET_TIMERX();
```

```
SETUP_TIMER_0(MODE|BC);  
SETUP_TIMER_1(MODE|BC);  
SETUP_TIMER_2(BCT,PR2,BCS)
```

T0\_INTERNAL  
T0\_EXT\_L\_TO\_H  
T0\_DIV\_1  
...

T2\_disableD  
T2\_DIV\_BY\_1  
T2\_DIV\_BY\_4  
T2\_DIV\_BY\_16

T1\_DISABLED  
T1\_INTERNAL  
T1\_EXTERNAL  
T1\_DIV\_BY\_1  
T1\_DIV\_BY\_2  
T1\_DIV\_BY\_4  
T1\_DIV\_BY\_8

## Vd:

```
SETUP_TIMER_0(T0_EXT_L_TO_H|T0_DIV_1);  
SETUP_TIMER_1(T1_INTERNAL|T1_DIV_BY_8);  
SETUP_TIMER_2(T2_DIV_BY_8,250,7);
```

## Định thời 1s dùng timer 1

$$f_{Làm\_việc} = \frac{1}{4} f_{Thạch\_Anh}$$

$$f_{thạch\_anh} = 20\text{Mhz} \rightarrow f_{LV} = 5 \text{ MHz}$$

bộ chia

$$f_{LV} = 5 \text{ MHz} \xrightarrow{\div 8} f = 625\text{kHz} \longrightarrow t = 16 \text{ us}$$

Timer 1: 65536  $\xrightarrow{\div 10}$  62500 xung

1 xung	$t = 16 \text{ us}$
? xung	$t = 1s$

Đặt trước:  $65536 - 62500 = 3036$

65536

3036

```
void main ()
{
    setup_timer_1(T1_INTERNAL|T1_DIV_BY ***); → Bộ chia
    set_timer1(***) ;
    while (true) → Giá trị nạp
    {
        if(TMR1IF==1) //Tràn
        {
            set_timer1(***) ; bdt++; TMR1IF=0;
            if(bdt==***)
            {
                // Công việc
                bdt=0;
            }
        }
    }
}
```

#bit TMR0IF =0x0b.2  
#bit TMR1IF =0x0c.0  
#bit TMR2IF =0x0c.1

Số lần tràn

```
void main ()
{
    setup_timer_0(T0_EXT_H_TO_L|T0_DIV_1);
    set_timer0(0);
    while (true)
    {
        dem = get_timer0();
        if (dem==1)
        { set_timer0(0);
          sp++; }
    }
}
```

## Đếm lên/xuống dung counter0/1

```
void main()
{
    set_tris_a(?);
    setup_timer_0(T0_EXT_H_TO_L|T0_DIV_X);
    set_timer0(a); //Đặt trước số đếm
    set_tris_c(?);
    setup_timer_1(T1_EXTERNAL|T1_DIV_BY_X);
    set_timer1(a); // Đặt trước số đếm
    while(true)
    {
        if(sp0>=b) set_timer0(a);
        sp0= get_timer0();
        sp0x= (a+b)-sp0; // Nếu đếm xuống
        if(sp1>=b) set_timer1(a);
        sp1= get_timer1();
        sp1x= (a+b)-sp1; // Nếu đếm xuống
        // Thực thi các công việc khác ở đây
    }
}
```

Mạch đo nhiệt độ 2 kênh dùng cảm biến LM35, vi điều khiển PIC16F887 và 3 led 7 đoạn.

Hãy vẽ mạch, viết lưu đồ và chương trình đo nhiệt độ 2 kênh lần lượt hiển thị trên 3 led, mỗi kênh đo trong khoảng thời gian 1s, định thời 1 giây chuyển kênh dùng Timer T1.

Code: ON\_TAP\_BUOI\_2

**Bài tập 8-6:** Một vi điều khiển PIC16F887 giao tiếp với 3 led 7 đoạn loại anode chung, 1 led hiển thị số thứ tự kênh, 2 led còn lại hiển thị nhiệt độ, 4 cảm biến nhiệt độ LM35 nối với 4 kênh từ AN0 đến AN3, có 3 nút nhấn: 1 nút chọn chế độ tự động hay bằng tay, 2 nút còn lại là UP và DW, nhấn UP là tăng lên để chọn kênh ADC cao hơn, DW thì giảm.

Hãy viết lưu đồ và chương trình chuyển đổi nhiệt độ ở chế độ tự động dùng ngắt của ADC để biết quá trình chuyển đổi xong, mỗi kênh chuyển đổi trong thời gian 5 giây, dùng timer1 đếm thời gian và báo ngắt để chuyển kênh.

Ở chế độ bằng tay thì chuyển kênh khi nhấn UP hoặc DW.

Code: ON\_TAP\_BUOI\_2\_BAI\_2

## Câu 2: (3.5 điểm)

Cho hệ thống đếm sản phẩm: Vi điều khiển Pic16F887 kết nối một cảm biến phát hiện sản phẩm, LCD 16x2, hai nút nhấn thường hở UP, DW. Bình thường ngõ ra cảm biến là 0V, khi có sản phẩm chấn ngang cảm biến ngõ ra sẽ lên mức 5V.

Giao diện hiển thị của LCD như sau:

GIA TRI DAT: AA  
SO SAN PHAM: BB

Giá trị đặt AA là số sản phẩm của một thùng cần đóng gói. Giá trị AA có thể điều chỉnh được bởi hai nút nhấn UP (tăng 1 đơn vị), DW (giảm 1 đơn vị). AA nằm trong phạm vi [20-30]. Mặc định AA = 20.

Giá trị BB là số phẩm đếm được.

Khi sản phẩm đếm được BB bằng AA thì LCD sẽ nhấp nháy (sáng 0.5s, tắt 0.5s) chỉ hai dòng chữ như sau:

DA DU SO LUONG  
DE NGHI DONG GOI

a. **Vẽ mạch nguyên lý kết nối (1 điểm)** (phần cứng kết nối sinh viên tùy chọn chân thích hợp, LCD có thể giao tiếp với vi điều khiển theo kiểu 8bit dữ liệu hoặc 4 bit dữ liệu. Khối cảm biến chỉ cần vẽ khói, không cần vẽ chi tiết mạch cảm biến).

b. **Viết chương trình thực hiện yêu cầu trên (2.5 điểm)**

Code: ON\_TAP\_BUOI\_3\_BAI2

## Câu 2: (3.5 điểm)

Cho hệ thống đếm sản phẩm: Vi điều khiển Pic16F887 kết nối một cảm biến phát hiện sản phẩm, LCD 16x2, hai nút nhấn thường hở START, STOP. Bình thường ngõ ra cảm biến là 0V, khi có sản phẩm chắn ngang cảm biến ngõ ra sẽ lên mức 5V.

Khi mới bật nguồn lên, hệ thống đếm sản phẩm chưa hoạt động, giao diện hiển thị của LCD như sau:

NHAN START  
DE KHOI DONG

Khi nhấn nút START, hệ thống đếm sản phẩm bắt đầu hoạt động. LCD hiển thị số sản phẩm đếm được. Nếu số sản phẩm đủ 30 sản phẩm thì hệ thống đếm sản phẩm ngừng đếm. Giao diện LCD khi đó như sau (XX: sẽ được thay thế bằng số sản phẩm thực tế khi đếm):

DEM SAN PHAM  
SO LUONG: XX

Bắt cú khi nào nhấn nút STOP hệ thống đếm sản phẩm sẽ dừng đếm.

- a. **Vẽ mạch nguyên lý kết nối (1 điểm)**
- b. **Viết chương trình thực hiện yêu cầu trên (2.5 điểm)**

# TRUYỀN THÔNG UART

```
#USE rs232(baud=9600,xmit=pin_c6,rcv=pin_c7)
#use rs232(BAUD=9600,BITS=8,STOP=1,PARITY=N,RCV=PIN_C7,XMIT=PIN_C6)

PUTC(DATA); // data là số 8 bit
PUTC('a'); // Gửi một ký tự ASCII lên máy tính
VALUE = GETC(); // Hàm trả về một ký tự nhận được từ MT
Value = KBHIT(); // cờ báo nhận dữ liệu
printf(string); //hàm gửi một chuỗi ký tự lên máy tính
```

## Chương trình nhận dữ liệu

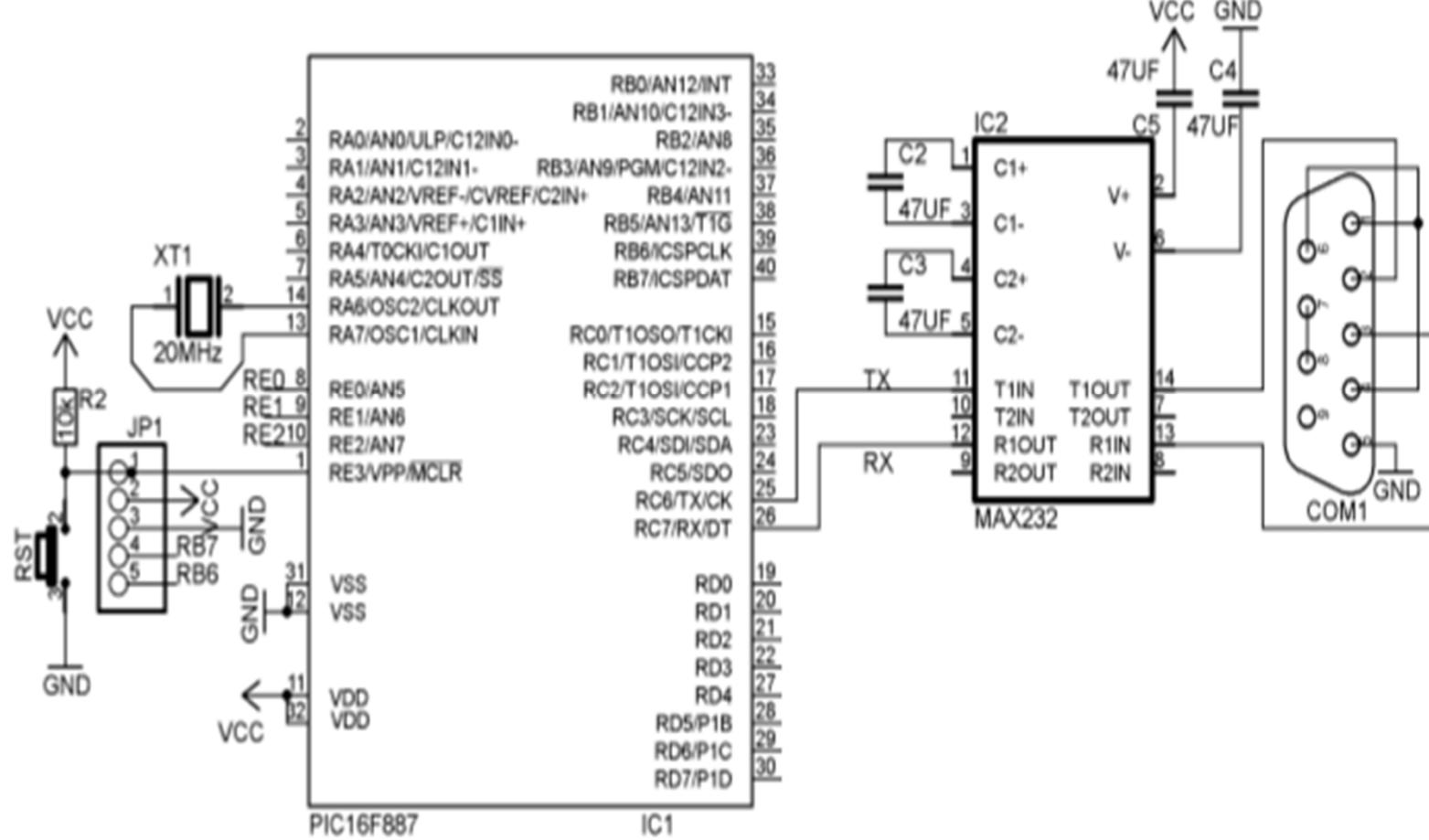
```
if (kbhit() == 1) { //công việc } else;
```

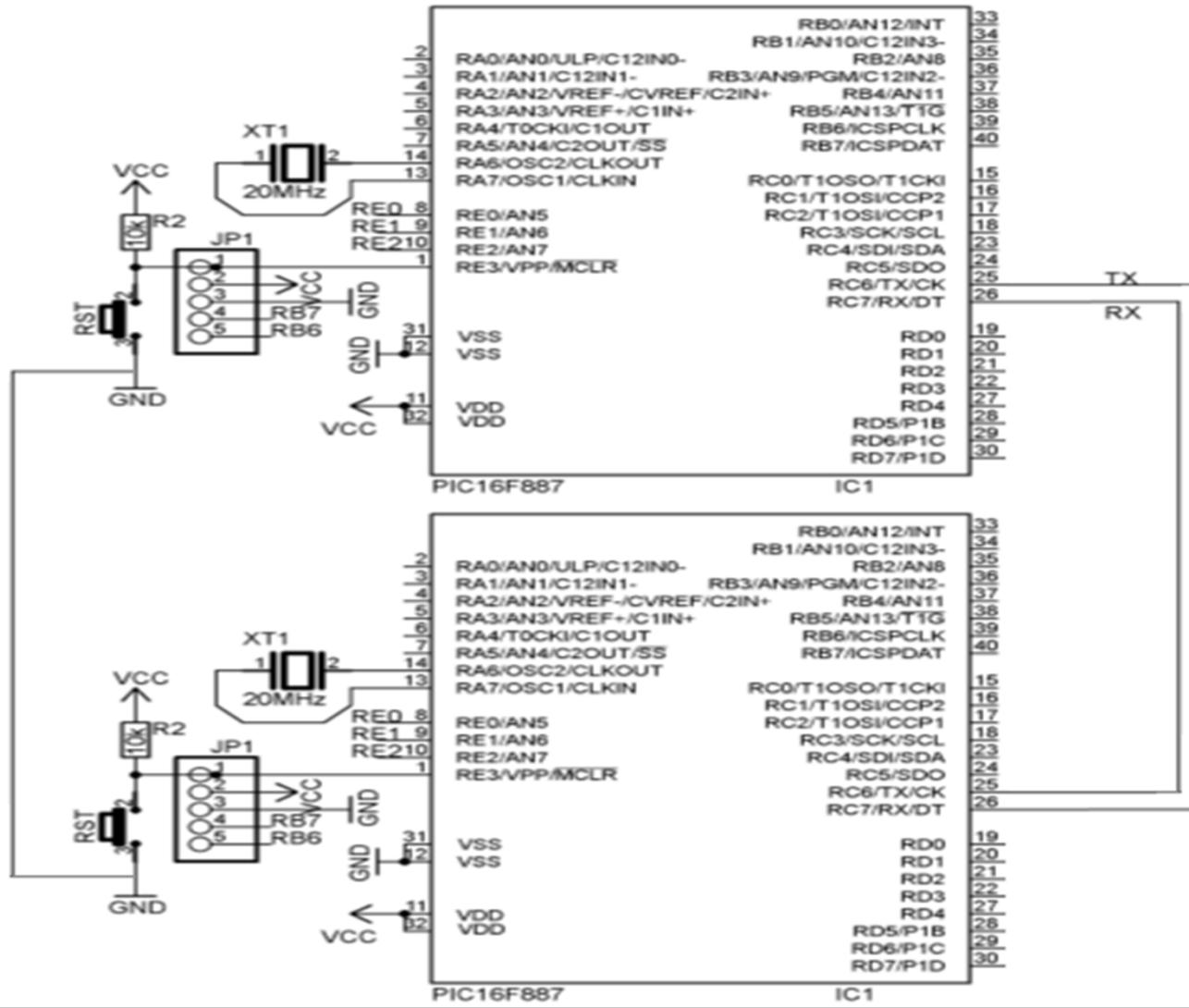
	Example formats:		
u	Unsigned int	Specifier	Value=0x12
d	Signed int		Value=0xfe
Lu	Long unsigned int	%03u	018
		%u	18
Ld	Long signed int	%2u	18
		%5	18
x	Hex int (lower case)	%d	254
X	Hex int (upper case)	%x	-2
Lx	Hex long int (lower case)	%X	fe
LX	Hex long int (upper case)	%4X	FE
		%3.1w	0012
			00FE
			25.4
		* Result is undefined - Assume garbage.	

```
if(kbhit()==1)
{
    //công việc
    data = getc();
} else;
```

### \* Gửi dữ liệu qua UART

```
printf(SP=%03u", sp );
// If sp in unsigned int8 → 'u', unsigned int16 → "Lu"
```





Câu 3: (3đ)

Một hệ thống quản lý bao gồm hệ thống A và hệ thống B như sau:

**Hệ thống A:** có chức năng đếm số sản phẩm thi công, khi thi công xong thì công nhân sẽ nhấn 1 cái nút để số sản phẩm tăng lên 1. Số sản phẩm nằm trong giới hạn từ 00 đến 99 (khi bằng 99 và nếu nhấn nữa thì quay về 0) hiển thị trên 2 led 7 đoạn anode chung.

**Hệ thống B:** Số lượng sản phẩm từ hệ thống A được gửi về phòng quản lý để hiển thị trên 2 led 7 đoạn anode chung. Có 1 nút nhấn reset và khi nhấn thì sẽ xóa sản phẩm đếm được về 0 của cả 2 hệ thống.

- a. Hãy thiết kế phần cứng cho mỗi hệ thống đều dùng vi điều khiển PIC16F887 và tùy chọn port, hai hệ thống giao tiếp với nhau dùng chuẩn UART, tốc độ 9600 baud. (0.75đ)
- b. Hãy vẽ lưu đồ cho 2 vi điều khiển. (0.5đ)
- c. Hãy viết các chương trình thực hiện các yêu cầu trên. (1.75đ)

*Chú ý không sử dụng counter để đếm cho hệ thống A.*

Code: ON\_TAP\_BUOI\_3\_BAI\_4

THÀNH PHỐ HỒ CHÍ MINH  
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO  
NGÀNH CÔNG NGHỆ KỸ THUẬT  
ĐIỆN TỬ TRUYỀN THÔNG

ĐỀ THI CUỐI KỲ HỌC KÌ 1 NĂM HỌC 2017-2018

Môn: VI XỬ LÝ

Mã môn học: MICR330363

Đề số: 1 Đề thi có 2 trang.

Thời gian: 90 phút.

Được phép sử dụng tài liệu trên một tờ giấy A4 viết tay

Câu 1: (3.5 điểm)

Cho hệ thống giám sát sản xuất gồm vi điều khiển PIC16F887 kết nối với máy vi tính giao tiếp theo chuẩn truyền dữ liệu nối tiếp bất đồng bộ RS232, tốc độ baud = 4800. Vi điều khiển kết nối với 2 nút nhấn GOOD, BAD; 1 led 7 đoạn hiển thị số sản phẩm tốt; 1 led 7 đoạn hiển thị số sản phẩm hư; giá trị hiển thị từ 0-9. Tần số dao động cho vi điều khiển tùy chọn.

Vẽ sơ đồ nguyên lý và viết chương trình điều khiển theo các yêu cầu:

a. Khi nhấn nút GOOD thì tăng sản phẩm tốt 1 đơn vị và gửi về máy tính ký tự ‘G’, khi nhấn nút BAD thì tăng sản phẩm hư 1 đơn vị gửi về máy tính ký tự ‘B’; nếu đếm đến 9 thì quay về 0;

b. Khi máy tính truyền ký tự ‘S’ thì hệ thống sẽ tạm ngừng và 2 nút nhấn GOOD và BAD không thay đổi giá trị sản phẩm khi nhấn. Còn khi máy tính truyền ký tự ‘R’ thì cho phép hệ thống tiếp tục hoạt động và 2 nút nhấn GOOD và BAD sẽ thay đổi giá trị sản phẩm khi nhấn.

# **INTERRUPT**

**enable\_interrupts(interrupt)**  
**disable\_interrupts(interrupt)**

Interrupt	Description
GLOBAL	Global interrupt
INT_AD	ngắt chuyển đổi ADC
INT_RB	Port B interrupt
INT_RDA	Receive data from EUART interrupt
INT_TIMER0	Timer 0 interrupt
INT_TIMER1	Timer 1 interrupt
INT_TIMER2	Timer 2 interrupt

## Chương trình con phục vụ ngắn

```
# interrupt  
void name()  
{  
    // handle interrupt requirement  
}
```

## Chương trình nhận dữ liệu

```
if (kbhit() == 1) { //công việc } else;
```

**Bài tập 8-5:** Một vi điều khiển PIC16F887 giao tiếp với 3 led 7 đoạn loại anode chung, 2 cảm biến nhiệt độ LM35 nối với kênh AN0, AN1.

Hãy viết lưu đồ và chương trình chuyển đổi nhiệt độ **dùng ngắt của ADC** để biết quá trình chuyển đổi xong, mỗi kênh chuyển đổi trong thời gian **5 giây**, dùng timer1 đếm thời gian và báo ngắt để chuyển kênh.

3 led 7 đoạn: 1 led hiển thị kênh, 2 led hiển thị nhiệt độ

Note:

```
#int_ad  
void ngatadc()  
{  
    nd=read_adc(adc_read_only);      set_adc_channel(0);  
    nd=nd/2.046;                   read_adc(adc_start_only);  
}  
}
```

Code: ON\_TAP\_BUOI\_3\_BAI\_1

# PWM

$$PR2 = \frac{t(s)*f}{4*CT} - 1$$

$$\text{MAX duty} = (PR2+1)*4$$

`setup_timer_2(mode, PR2, CS)`

`mode:`

`T2_DIV_BY_1` : prescaler 1

`T2_DIV_BY_4` : prescaler 4

`T2_DIV_BY_16` : prescaler 16

`PR2: 0-255`

`CS: 0-16`

`VD: SETUP_TIMER_2(T2_DIV_BY_16, 250, 1);`

**SETUP\_CCPX(CCP\_PWM)**

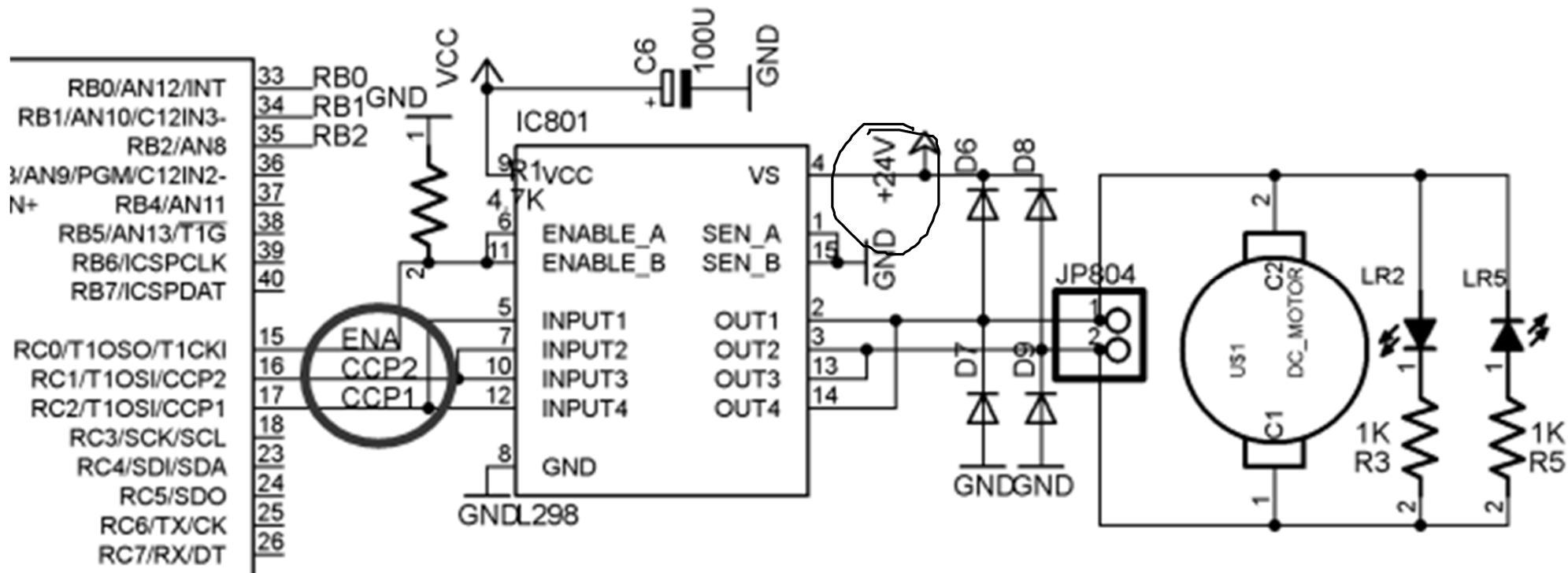
**SET\_PWMX\_DUTY(VALUE)**

*value: 10 bit number*

// nếu lớn hơn 10 bit thì ép kiểu hoặc đặt biến ngoài

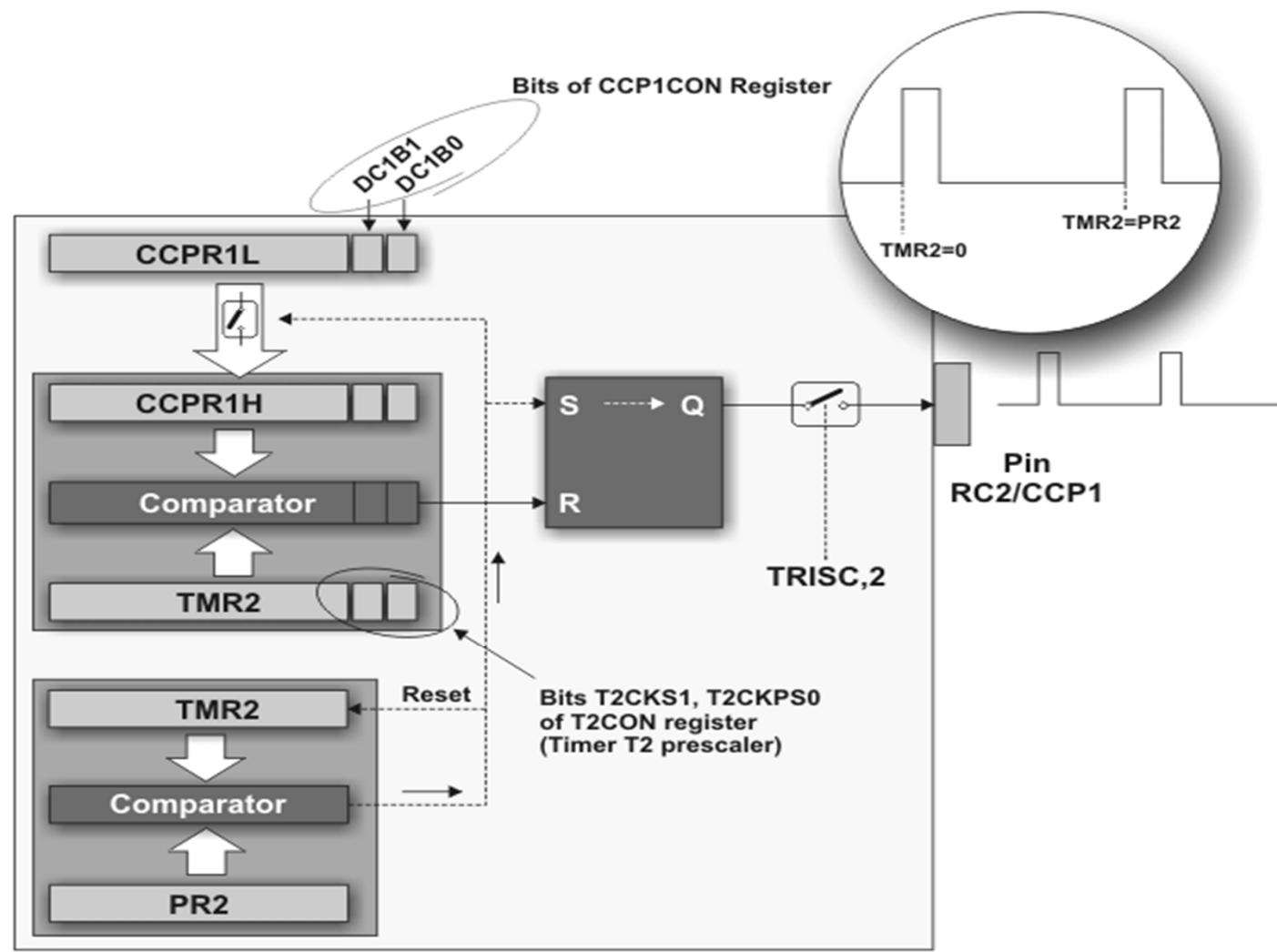
**SETUP\_CCPx(CCP\_OFF);**

# PWM



```
unsigned int16 td;
signed int8 capdo=0;

void main ()
{
    SETUP_TIMER_2(T2_DIV_BY_16,250,1);
    set_pwm1_duty(0);
    set_timer2(0);
    while (true)
    {
        td=capdo*100;
        set_pwm1_duty( (unsigned int16) capdo*100); //simulation
    }
}
```



**Bài 10-3:** Sử dụng PWM của PIC 16F887 để điều khiển thay đổi tốc độ động cơ DC. Cho tần số tụ thạch anh là 20MHz. Cho chu kỳ PWM là 0,8ms. Động cơ dùng nguồn 24V DC, dòng 3A.

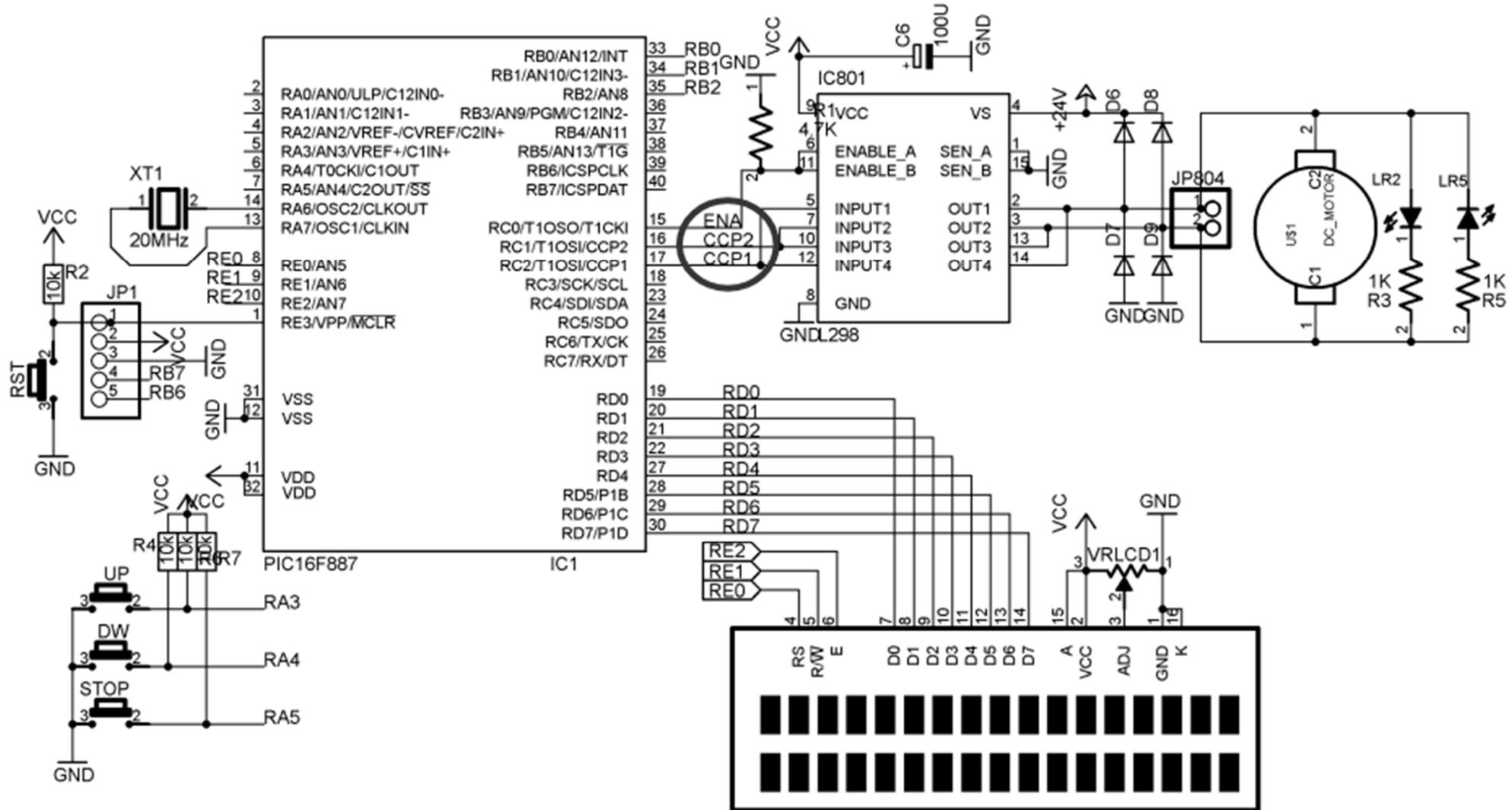
Hãy tính toán các thông số và viết chương trình điều khiển ~~Tốc độ động cơ~~ 10 cấp bằng 2 nút nhấn UP và DW. Nút Stop khi nhấn thì động cơ ngừng, không tính cấp độ 0.

Hiển thị cấp độ trên LCD.

Giá trị thay đổi cho 1 cấp là 100.

**Tốc độ động cơ**

Code: ON\_TAP\_BUOI\_2\_BAI\_4



Câu 3: (3 điểm)

PWM

Cho hệ thống điều khiển giải nhiệt dùng động cơ quạt DC 24VDC, dòng 3A kết nối với vi điều khiển PIC16F887 qua IC L298, có 2 nút nhấn SPEED, RUN\_STOP, có 1 led 7 đoạn hiển thị cấp độ động cơ từ 1-4. Vi điều khiển sử dụng tần số dao động 20 Mhz.

- a. Vẽ sơ đồ nguyên lý kết nối.
- b. Viết chương trình thực hiện các yêu cầu sau:

Dùng PWM của PIC để điều khiển thay đổi tốc độ động cơ DC, 4 cấp độ tương ứng 25%, 50%, 75%, 100% khi nhấn nút SPEED và hiển thị cấp tốc độ từ 1-4 trên led 7 đoạn. Chu kỳ tín hiệu PWM là 400 us. Nhấn nút RUN\_STOP để ngừng động cơ và cho phép động cơ chạy lại theo tốc độ trước khi ngừng.

Ghi chú: Cán bộ coi thi không được giải thích đề thi.

Câu 2: (4đ)

Một hệ thống vi điều khiển PIC16F887 dùng PWM của khối CCP2 để điều khiển 1 động cơ (DC) thay đổi tốc độ 10 cấp (không tính cấp 0), dùng IC giao tiếp công suất L298, 6 nút nhấn thường hở (BTN\_1\_6, BTN\_2\_7, BTN\_3\_8, BTN\_4\_9, BTN\_5\_10, BTN\_STOP) và 1 switch gạt SW (tạo 2 mức logic 0 và 1 tương ứng 2 vị trí on và off). Chu kỳ PWM là 0.8ms, tần số sử dụng là 10MHz.

Khi mới cấp điện hoặc khi nhấn BTN\_STOP thì động cơ ngừng.

Khi SW ở vị trí on và nếu nhấn BTN\_1\_6 thì động cơ sẽ chạy cấp tốc độ 1.

Khi SW ở vị trí off và nếu nhấn BTN\_1\_6 thì động cơ sẽ chạy cấp tốc độ 6.

Tương tự cho các nút nhấn còn lại (2 số theo sau chính là tốc độ 2 cấp tương ứng).

- a. Hãy vẽ sơ đồ mạch giao tiếp vi điều khiển với L298, động cơ, nút nhấn và switch. (0.5đ)
- b. Hãy tính toán các thông số PR2, PV và hằng số tương ứng với tốc độ cực đại. (0.5đ)
- c. Hãy vẽ lưu đồ giải thuật điều khiển theo yêu cầu. (1đ)
- d. Hãy viết chương trình. (2đ)

Code: ON\_TAP\_BUOI\_3\_BAI\_3

## Bài tập tổng hợp:

### VDKA:

Kết nối với 2 nút nhấn UP, DW, Buzzer (mức 1 kêu, mức 0 tắt), 1 led đơn, 2 led 7 đoạn .

VDKA gởi thông số tốc độ động cơ cho VDKB thông qua nút nhấn UP, DW.

Cấp độ bằng 10 thì led sáng.

VDKA nhận dữ liệu từ VDKB để mở buzzer.

### VDKB:

Kết nối với 1 nút nhấn STOP, động cơ DC 24V thông qua IC L298.

Nhận dữ liệu từ VDKA để điều khiển tốc độ động cơ.

Khi nhấn nút STOP thì động cơ ngừng, gửi dữ liệu về VDKA để mở kèn

Code: ON\_TAP\_BUOI\_2\_BAI\_5

