# 06. Mô đun chuyển đổi tín hiệu tương tự - số (Analog-to-Digital Converter)

# GIỚI THIỆU VỀ MODULE ADC (1)

**Module ADC cho phép chuyển đổi một tín hiệu tương tự ở đầu vào thành một biểu diễn dưới dạng tín hiệu số 10 bit**

- Module ADC chuyển tín hiệu tương tự vào một mạch lấy mẫu và giữ mẫu đơn

- Ngõ ra của mạch lấy và giữ mẫu này sẽ làm ngõ vào của bộ chuyển đổi ADC

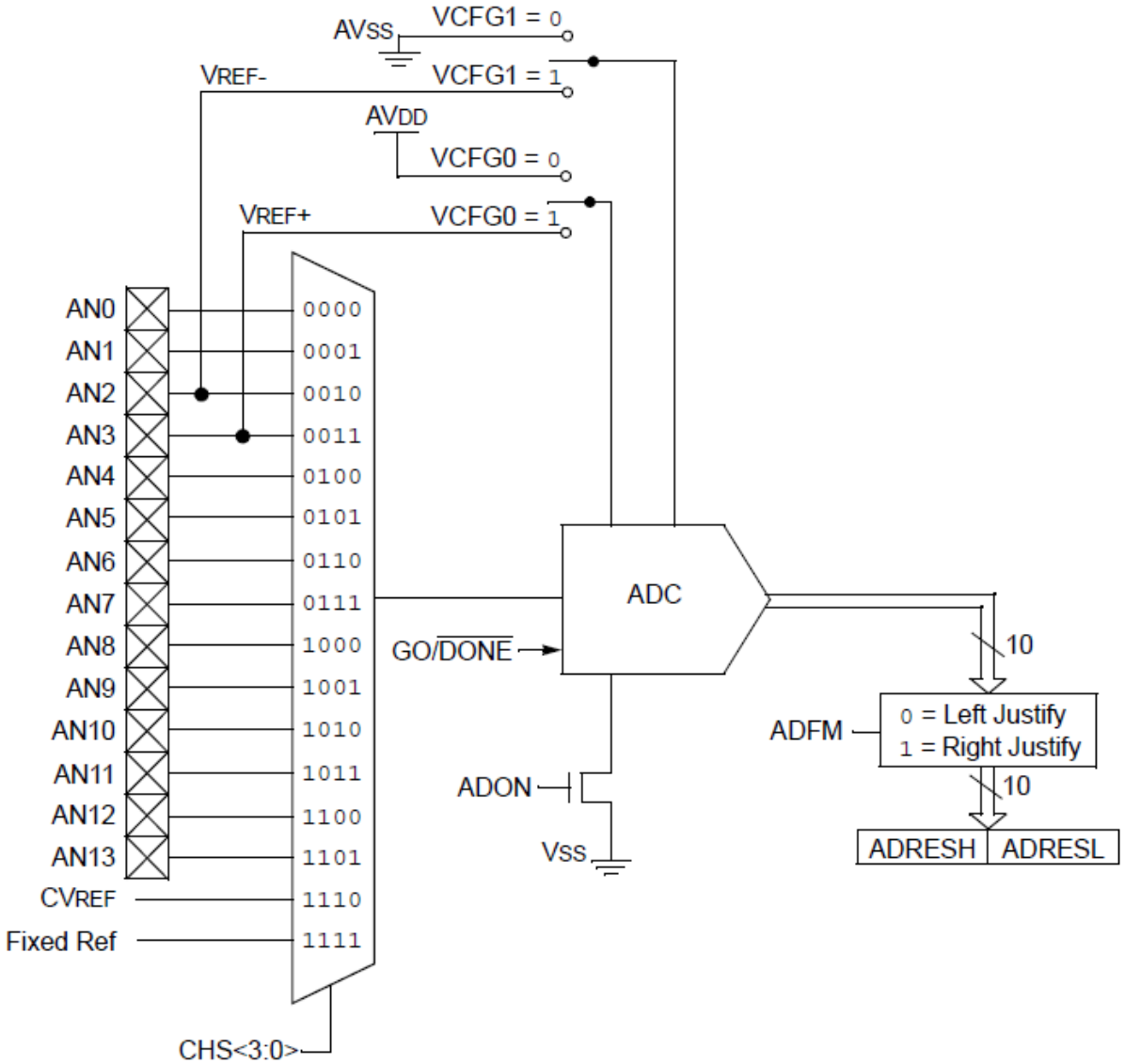
- Bộ chuyển đổi ADC tạo ra một giá trị số 10 bit bằng phương pháp gần đúng liên tiếp và lưu giá trị này vào các thanh ghi ADRESL và ADRESH

# GIỚI THIỆU VỀ MODULE ADC (2)

- Giá trị điện áp đưa vào module ADC có thể được lựa chọn bởi phần mềm, hoặc là giá trị VDD, hoặc bất cứ giá trị điện áp nào đưa vào chân điện áp tham chiếu

- Module ADC có thể tạo ra một ngắt sau khi hoàn thành phép biến đổi. Ngắt này có thể dùng để đánh thức VĐK khỏi trạng thái Sleep

# SƠ ĐỒ KHỐI MODULE ADC



# KHAI BÁO MODULE ADC

**Trước khi sử dụng module ADC chúng ta cần khai báo các chức năng sau:**

- Port configuration: Khai báo chân sử dụng
- Channel selection: Lựa chọn kênh
- ADC voltage reference selection: Lựa chọn điện áp tham chiếu cho module ADC
- ADC conversion clock source: Nguồn xung clock dùng cho module ADC
- Interrupt control: Điều khiển ngắt
- Results formatting: Định dạng kết quả

# PORT CONFIGURATION

- Khi tín hiệu vào là tín hiệu tương tự, chúng ta phải khai báo chân I/O thích hợp để không bị nhầm lẫn với tín hiệu vào là tín hiệu số

- Các thanh ghi: TRISA, TRISB, ANSEL, ANSELH

# CHANNEL SELECTION

- Các bit CHS của thanh ghi ADCON0 quyết định kênh nào sẽ được đưa vào mạch lấy mẫu và giữ mẫu

- Khi thay đổi kênh sẽ có một khoảng trễ trước khi thực hiện việc chuyển đổi tiếp theo

# ADC VOLTAGE REFERENCE

- Các bit VCFG của thanh ghi ADCON0 cung cấp các điều khiển độc lập của điện áp tham chiếu âm và dương

- Điện áp tham chiếu dương có thể là VDD hay nguồn điện áp bên ngoài

- Điện áp tham chiếu âm có thể là VSS hay nguồn điện áp bên ngoài

# CONVERSION CLOCK (1)

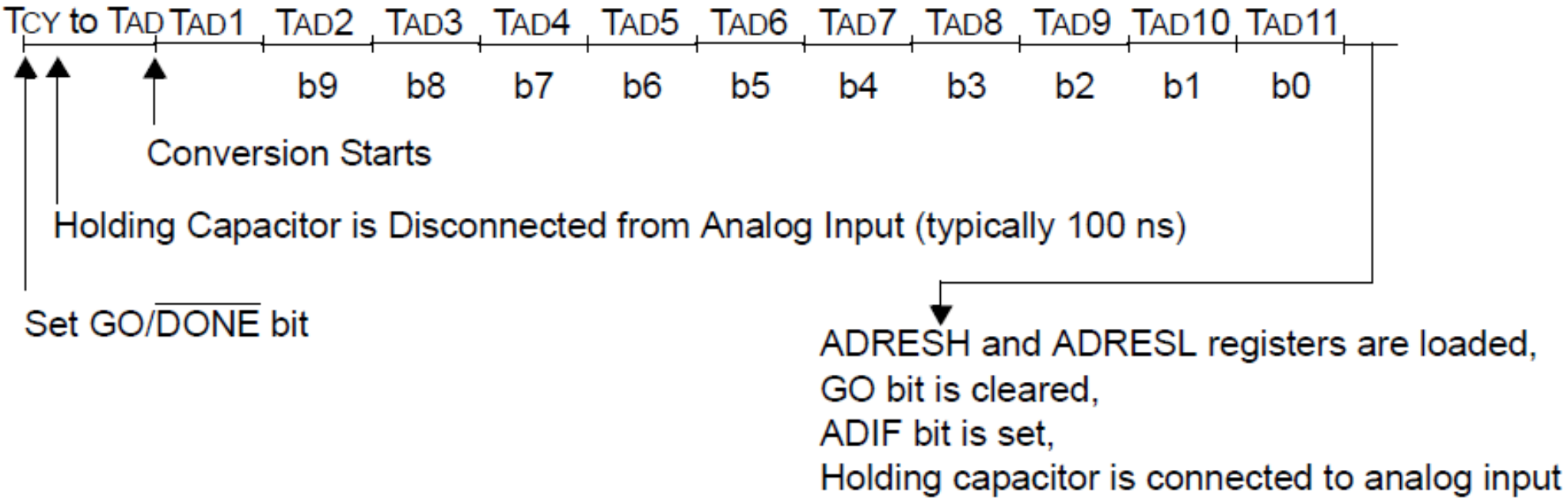
Nguồn của xung clock chuyển đổi thì được lựa chọn bởi phần mềm từ các bit ADCS của thanh ghi ADCON0

Có 4 tần số có thể lựa chọn

- $F_{OSC}/2$
- $F_{OSC}/8$
- $F_{OSC}/32$
- $F_{RC}$

# CONVERSION CLOCK (2)

- Thời gian để thực hiện việc chuyển đổi 1 bit là $T_{AD}$
- Để thực hiện việc chuyển đổi 10 bit cần 1 chu kỳ là 11 $T_{AD}$



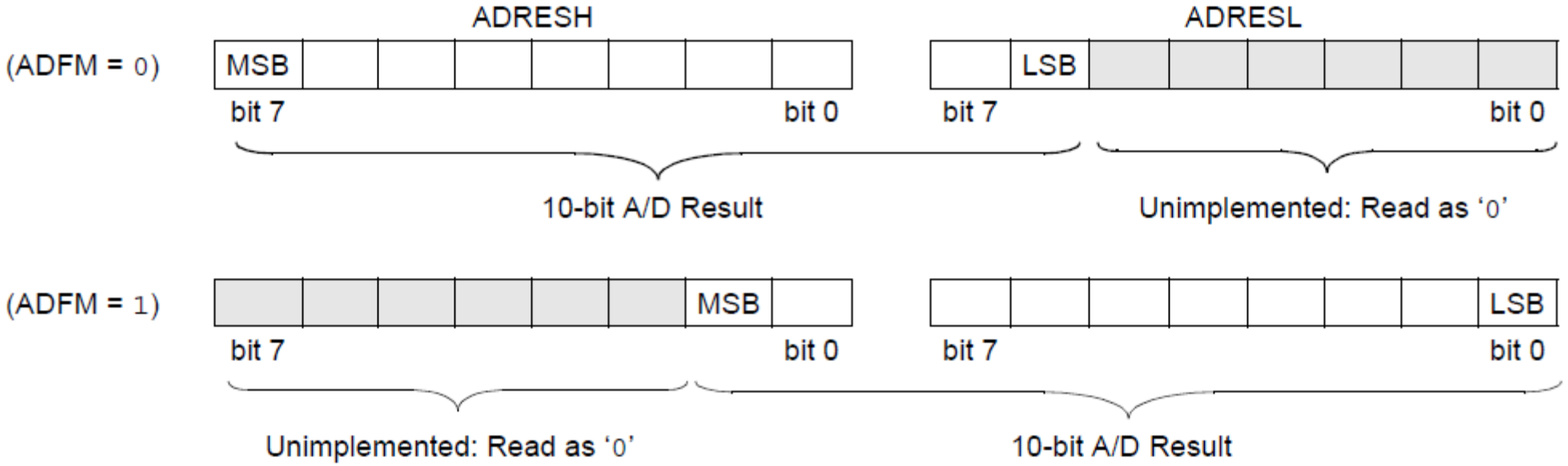
# INTERRUPTS

- Module ADC cho phép tạo ra 1 ngắt sau khi hoàn thành việc chuyển đổi Analog-to-Digital
- Cờ ngắt ADC là bit ADIF trong thanh ghi PIR1
- Cho phép ngắt ADC là bit ADIE trong thanh ghi PIE1
- Bit ADIF phải được xóa bằng phần mềm

# RESULT FORMATTING

- Kết quả chuyển đổi ADC 10 bit có thể được chia thành 2 định dạng: canh lề trái hay canh lề phải

- Bit ADFM của thanh ghi ADCON0 điều khiển định dạng ngõ ra



# CÁC BƯỚC SỬ DỤNG MODULE ADC (1)

**1. Khai báo thông số các chân**

- Chọn các chân là input
- Chọn cấu hình chân là analog

**2. Khai báo thông số module ADC**

- Lựa chọn xung clock chuyển đổi cho ADC
- Khai báo điện áp tham chiếu
- Lựa chọn kênh vào cho bộ ADC
- Lựa chọn định dạng kết quả
- Bật module ADC

**3. Khai báo ngắt ADC**

- Xóa cờ ngắt ADC
- Cho phép ngắt ADC

# CÁC BƯỚC SỬ DỤNG MODULE ADC (2)

- Cho phép ngắt ngoại vi
- Cho phép tất cả các ngắt

**4. Chờ đợi thời gian nhận dữ liệu**

**5. Bắt đầu chuyển đổi bằng cách set bit GO/DONE**

**6. Chờ đợi chuyển đổi ADC hoàn thành**

- Set bit GO/DONE
- Chờ đợi ngắt ADC

**7. Đọc kết quả ADC**

**8. Xóa cờ ngắt ADC**

# BẮT ĐẦU ADC

- Để cho phép thực hiện module ADC, bit ADON của thanh ghi ADCON0 phải set lên “1”

- Set bit GO/DONE của thanh ghi ADCON0 lên “1” sẽ bắt đầu thực hiện chuyển đổi A/D

# HOÀN THÀNH ADC

**Khi hoàn thành việc chuyển đổi ADC thì module ADC sẽ**

- Xóa bit GO/DONE
- Set bit cờ ngắt ADIF
- Cập nhật kết quả chuyển đổi mới vào các thanh ghi ADRESH và ADRESL

# KẾT THÚC ADC

- Nếu 1 chuyển đổi phải kết thúc trước khi hoàn thành, ta có thể xóa bit GO/DONE bằng phần mềm

- Cặp thanh ghi ADRESH:ADRESL sẽ không cập nhật kết quả của phép chuyển đổi dang dở mà sẽ lưu giá trị của phép chuyển đổi trước đó

- Một khoảng thời gian trễ $2T_{AD}$ sẽ xuất hiện trước khi chúng ta có thể nhận tín hiệu vào tiếp theo

# THANH GHI ADCON0 (1)

**REGISTER 9-1: ADCON0: A/D CONTROL REGISTER 0**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADCS1 | ADCS0 | CHS3 | CHS2 | CHS1 | CHS0 | GO/$\overline{\text{DONE}}$ | ADON |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-6 **ADCS<1:0>:** A/D Conversion Clock Select bits
00 = FOSC/2
01 = FOSC/8
10 = FOSC/32
11 = FRC (clock derived from a dedicated internal oscillator = 500 kHz max)

bit 5-2 **CHS<3:0>: Analog Channel Select bits**
0000 = AN0
0001 = AN1
0010 = AN2
0011 = AN3

# THANH GHI ADCON0 (2)

0100 = AN4
0101 = AN5
0110 = AN6
0111 = AN7
1000 = AN8
1001 = AN9
1010 = AN10
1011 = AN11
1100 = AN12
1101 = AN13
1110 = $CV_{REF}$
1111 = Fixed Ref (0.6 volt fixed reference)

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit

1 = A/D conversion cycle in progress. Setting this bit starts an A/D conversion cycle.
This bit is automatically cleared by hardware when the A/D conversion has completed.
0 = A/D conversion completed/not in progress

bit 0 **ADON:** ADC Enable bit

1 = ADC is enabled
0 = ADC is disabled and consumes no operating current

# THANH GHI ADCON1

**REGISTER 9-2: ADCON1: A/D CONTROL REGISTER 1**

| R/W-0 | U-0 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| ADFM | — | VCFG1 | VCFG0 | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7 **ADFM:** A/D Conversion Result Format Select bit
1 = Right justified
0 = Left justified

bit 6 **Unimplemented:** Read as '0'

bit 5 **VCFG1:** Voltage Reference bit
1 = $V_{REF}$- pin
0 = $V_{SS}$

bit 4 **VCFG0:** Voltage Reference bit
1 = $V_{REF}$+ pin
0 = $V_{DD}$

bit 3-0 **Unimplemented:** Read as '0'

# THANH GHI ADRESH

**REGISTER 9-3: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 0**

| R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x |
|---|---|---|---|---|---|---|---|
| ADRES9 | ADRES8 | ADRES7 | ADRES6 | ADRES5 | ADRES4 | ADRES3 | ADRES2 |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-0 **ADRES<9:2>**: ADC Result Register bits
Upper 8 bits of 10-bit conversion result

# THANH GHI ADRESL

**REGISTER 9-4: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 0**

| R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x | R/W-x |
|---|---|---|---|---|---|---|---|
| ADRES1 | ADRES0 | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-6 **ADRES<1:0>**: ADC Result Register bits
Lower 2 bits of 10-bit conversion result

bit 5-0 **Reserved**: Do not use.

# MỘT SỐ LỆNH TRÊN CCS

- **Setup_ADC ( mode )**

  ADC_OFF : not use ADC

  ADC_CLOCK_INTERNAL: sampling time = MCU clock (2-6 us)

  ADC_CLOCK_DIV_2: sampling time = MCU clock/ 2 (0.4 us - when using OSC 20MHz )

  ADC_CLOCK_DIV_8 : sampling time = MCU clock/ 8 (1.6 us )

  ADC_CLOCK_DIV_32 : sampling time = MCU clock/ 32 (6.4 us)

- **Setup_ADC_ports ( value )**

  ALL_ANALOGS ; NO_ANALOG ; AN0_AN1_AN3

- **Set_ADC_channel ( channel )**
- **Read_ADC ( mode )**

  ADC_START_AND_READ : default

  ADC_START_ONLY

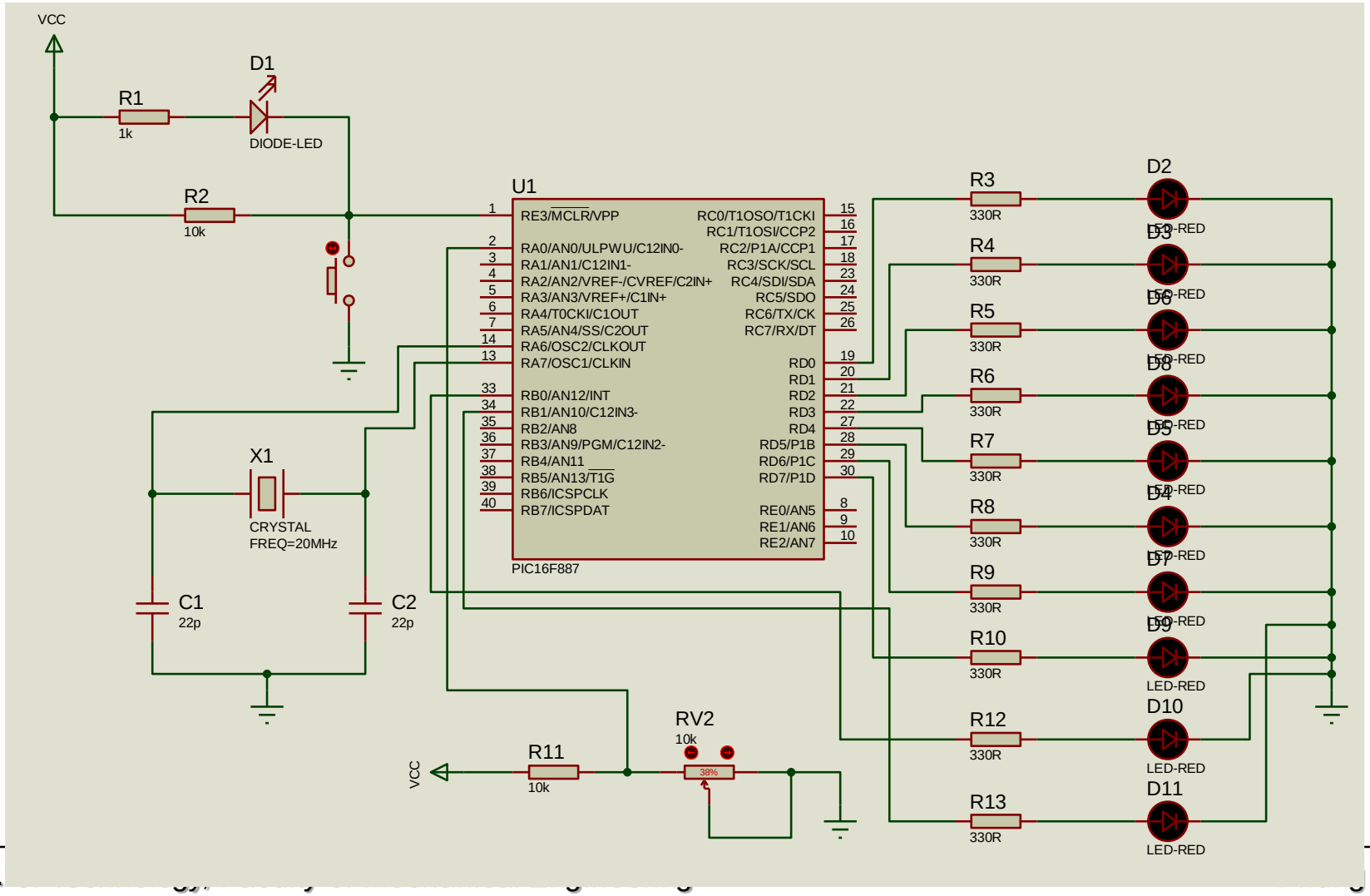
# KHAI BÁO ADC

```
// ADC 8 bit , ADC : 0-255
#device *= 16  ADC = 8
Int8 adc


 Main( )
 {
      Delay_ms (100 );
      Setup_ADC ( ADC_CLOCK_INTERNAL) ;
      Setup_ADC_ports (AN0);
      Set_ADC_channel ( 0 ) ;
      Delay_us (10 );
     While (true )
     {
        adc = read_adc ( ) ;
      }
 }
```

# VÍ DỤ 1: SỬ DỤNG ADC (1)

**Thiết kế mạch điện sử dụng module ADC và chuyển đổi mức điện áp từ tương tự sang số trong khoảng 0-2.5V ?**



# VÍ DỤ 1: SỬ DỤNG ADC (2)

**Thiết kế mạch điện sử dụng module ADC và chuyển đổi mức điện áp từ tương tự sang số trong khoảng 0-2.5V ?**

```
#include <16F887.h>
#device *=16 ADC=10
#fuses NOWDT,PUT,HS,NOPROTECT
#use delay(clock=20000000)

#byte PORTB = 0x06
#byte PORTD = 0x08

int8 adc;
```

```
void main(void)
{
  set_tris_b(0xF0);
  set_tris_a(0x0F);
  set_tris_d(0);
  PORTD = 0;
  setup_ADC(ADC_CLOCK_INTERNAL) ;
  setup_ADC_ports(ALL_ANALOG);
  set_ADC_channel(0) ;
  delay_us(10);
  While (true)
  {
    adc = read_adc();
    PORTD = adc;
  }
}
```