

# Lucrarea 1. Configurarea pinilor intrare/ieșire

Microcontrolerele dsPIC sunt dispozitive integrate puternice care integrează capacități DSP (Digital Signal Processing) și control. Configurarea pinilor de intrare/ieșire a acestor microcontrolere este un aspect crucial al programării embedded, influențând interacțiunea dispozitivului cu mediul extern.

Fiecare microcontroler dsPIC are un număr specific de pini, identificați prin numere și funcții specifice. De exemplu, un pin poate îndeplini rolul de intrare digitală, ieșire digitală, intrare analogică, ieșire PWM sau poate fi dedicat diferitelor interfețe de comunicare.

Configurarea funcțiilor pinilor este o etapă importantă în dezvoltarea unui proiect cu microcontrolerul dsPIC. Prin intermediul mediului de dezvoltare specific, programatorii pot seta funcțiile dorite pentru fiecare pin. Aceste funcții includ direcția (intrare/ieșire), nivelul logic, activarea sau dezactivarea anumitor funcții specifice ale pinilor.

Funcționalitățile pinilor pot fi analogice sau digitale. La rândul lor, se împart în două categorii, de intrare sau de ieșire.

## Lucrul cu pini digitali

Pinii digitali sunt pinii cei mai răspândiți la un microcontroler. În funcție de utilitatea lor, pinii digitali se împart în două categorii:

### - *Ieșiri:*

- controlul unui led
- releu
- buzzer activ
- semnale de comandă de tip enable către alte periferice
- generarea unui semnal PWM software

### - *Intrări:*

- butoane
- senzori cu ieșire digitală (0 logic sau 1 logic) (ex.: senzor de proximitate)
- alte microcontrolere
- module cu ieșiri digitale etc.

Obs:

Un caz particular este reprezentat de dispozitivele ce au nevoie atât de pini de intrare, cât și de pini de ieșire pentru o funcționare corectă. Câteva exemple pentru aceste cazuri ar fi:

- Tastaturi
- Module LCD
- Senzori de temperatură digitali

Procesul de identificare a pinilor relevanți pentru proiectul specific este primul pas în configurarea corectă a microcontrolerului.

Deoarece majoritatea pinilor unui microcontroler au și funcționalitatea de pini digitali, pentru a facilita adresarea lor este introdusă noțiunea de port. Portul unui microcontroler

reprezintă un set de pini, în mod uzual cu 8 sau 16 pini. În cazul microcontrolerului dsPIC, porturile sunt notate cu o literă (A, B etc.), pinii acestora fiind identificați ușor cu sintaxa RXi, unde X este litera portului, iar I este numărul pinului.

### Exercițiu:

Identificați în foaia de catalog a microcontrolerului folosit în laborator numărul porturilor și numărul pinilor pe fiecare port.

### 28-Pin SDIP, SOIC

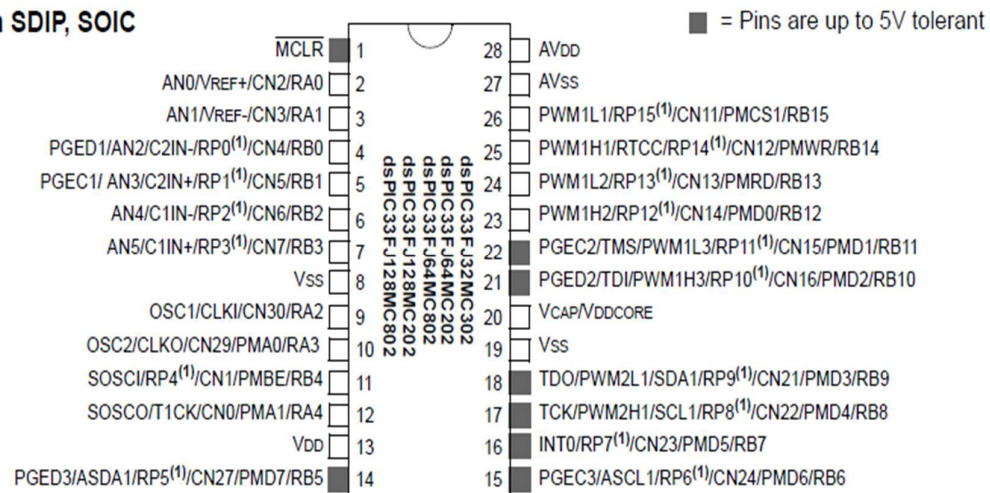


Fig. 1 Reprezentarea pinilor pentru dsPIC

Din exercițiul anterior reiese că există două porturi de pini disponibile pe varianta de dsPIC utilizată în laborator:

- RA (5 pini de la RA0 la RA4)
- RB (16 pini de la RB0 la RB15)

De menționat este faptul că pinii portului A nu sunt scoși.

### Configurarea pinilor ca intrare sau ca ieșire:

Funcționarea fiecărui pin, independentă de restul pinilor din porturile I/O este configurată folosindu-se două registre. Aceste registre sunt TRISA și PORTA pentru primul port ce are 5 pini și respectiv TRISB și PORTB pentru cel de-al doilea port ce are 16 pini.

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISA	02C0	—	—	—	—	—	TRISA10	TRISA9	TRISA8	TRISA7	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	079F
PORTA	02C2	—	—	—	—	—	RA10	RA9	RA8	RA7	—	—	RA4	RA3	RA2	RA1	RA0	XXXX

Fig.2 Regiștrii de configurare I/O pentru portul A

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISB	02C8	TRISB15	TRISB14	TRISB13	TRISB12	TRISB11	TRISB10	TRISB9	TRISB8	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	FFFF
PORTB	02CA	RB15	RB14	RB13	RB12	RB11	RB10	RB9	RB8	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	XXXX

Fig.3 Regiștrii de configurare I/O pentru portul B

Valoarea din registrul TRISX determină sensul de transmitere a datelor pe PORTX. Pinii ce au ca valoare corespunzătoare în registrul TRISX 1 logic sunt configurați ca pini de intrare, iar cei care au ca valoare corespunzătoare 0 sunt configurați ca pini de ieșire.

TRISBi

- 0 - pinul RBi este configurat ca pin de ieșire
- 1 - pinul RBi este configurat ca pin de intrare

În cazul în care un pin RBi este setat ca ieșire:

RBi

- 0 – 0L pe pinul RBi
- 1 – 1L pe pinul RBi

Pentru cazul în care un pin RBi este setat ca intrare, verificarea se poate face în program cu un if, ca în exemplul următor:

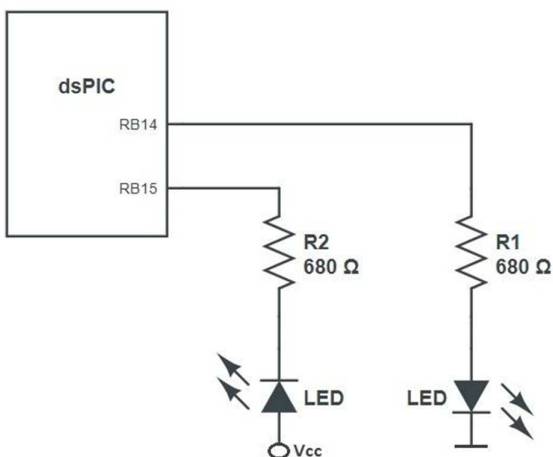
```
_TRISB15 = 1      //pinul 15 al portului B este setat ca intrare
if(_RB15 == 0)    //verificarea nivelului logic de pe pinul RB15
{
    //instructiune;
}
```

O modalitate ușoară de a vă aminti acest lucru este aceea că un „1” arată ca un I (intrare) și un „0” arată ca un O (ieșire).

După un reset toți pinii sunt setați ca intrări, în registrul TRISB introducându-se automat valoarea 0xFFFF, așa cum este marcat în Fig.3. În practică ne putem folosi de acest aspect, profitând de faptul că toți pinii sunt setați ca intrare, configurând pinii care ne interesează să fie setați ca ieșire.

Un caz de luat în considerare ar fi modul în care poate fi comandat un LED cu ajutorul unui pin digital setat ca ieșire. Nivelul logic ce trebuie trimis pe pinul digital pentru a aprinde/stinge un LED depinde foarte mult de modul în care LED-ul respectiv este conectat la microcontroler.

După cum se poate observa în Fig.4, pentru a aprinde LED-ul conectat la pinul RB15 trebuie emis 0 logic pe pinul RB15, pe când pentru a aprinde LED-ul conectat la pinul RB14 trebuie emis 1 logic pe pinul RB14.



**Fig.4 Comandarea LED-urilor cu un microcontroller**

#### **Fișierul header f33fj128mc802.h**

Aici se găsesc etichetele corespunzătoare tuturor regiștrilor de control ai microcontrolerului (TRISB, PORTB) precum și cele ale grupurilor de biți de control (RB15, RB14) din regiștrii de control.

Pentru fiecare registru de control există o structură alcătuită din numele registrului și sufixul "bits" structură ai cărei membri au numele biților de control din acel registru.

De exemplu, PORTBbits.RB15 face referire la bitul de control RB15 din registrul de control PORTB.

Pentru fiecare structură de tipul PORTBbits.RBx există o prescurtare cu același rol, notată \_RBx.

Compilerul preia informațiile din acest fișier, unde sunt scrise informații despre regiștri de control și grupuri de biți de control.

Ex.:

1. *Setarea pinului RB15 ca intrare:*

```
_TRISB15 = 1      //pinul 15 al portului B este setat ca intrare
if(_RB15 == 0)    //verificarea nivelului logic de pe pinul RB15
{
    //instructiune;
}
```

2. *Setarea pinului RB7 ca ieșire:*

```
_TRISB7 = 0      //pinul 7 al portului B este setat ca ieșire
_RB7 = 1;        //atribuirea nivelului logic pe pinul setat ca ieșire
```

3. *Setarea întregului port ca intrare/ieșire:*

```
TRISB = 0xFFFF      //intregul pot B a fost setat ca intrare
                    //(0 pentru a seta ca iesire)
```

Compilerul recunoaște intenția de a lucra cu un singur pin, ca în primul exemplu, sau cu toți pinii disponibili pe port, ca în cel de-al treilea exemplu.

### **Lucrul cu pini analogici**

Pe lângă funcționalitatea de pini digitali, o parte dintre pinii dsPIC-ului au și funcții de intrări analogice notate cu AN<sub>x</sub>.

#### ***Exercițiu:***

*Identificați în foaia de catalog a microcontrolerului pinii ce corespund intrărilor analogice.*

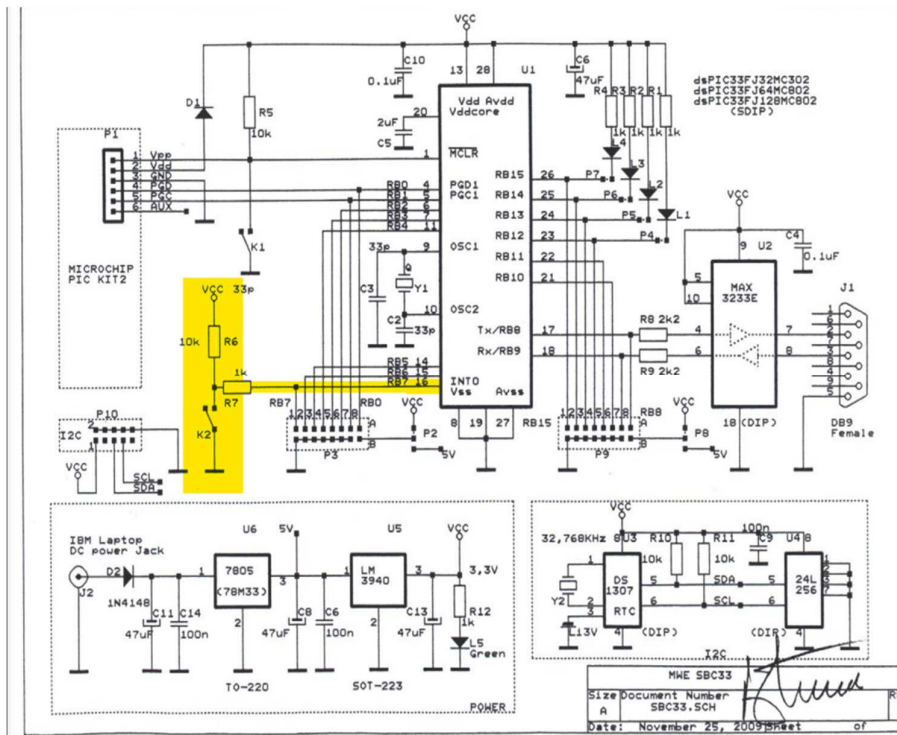
Pinii ce suportă această funcție, în mod implicit sunt configurați ca intrări analogice. Dacă se dorește funcționarea ca intrări/ieșiri digitale, pinii trebuie configurați explicit prin program.

Pentru a folosi un pin ca și intrare analogică trebuie ca acesta să fie setat prin intermediul lui TRISX ca intrare iar bitul corespunzător din AD1PCFGL să fie 0. Dacă se citește valoarea unui port pinii configurați ca intrări analogice sunt citiți ca 0 logic.

Din exercițiul anterior reiese faptul că acești pini coincid cu pinii RA0, RA1, RB0, RB1, RB2 și RB3.

#### **Program exemplu:**

Pe placă există două butoane, K1 și K2. K1 este butonul de reset, iar K2 este un buton conectat la RB7. În mod predefinit este 1, fiind conectat la Vcc. Când este apăsător, circuitul dintre masă și pin se închide, iar la pin se transmite 0.



**Fig. 5 Evidențierea butonului în schema plăcuței de dezvoltare cu dsPIC33FJ128MC802.**

În aplicația din exemplu se configurează pinul RB7 ca intrare, acesta primind semnalul logic de la butonul de pe placă (legătura este deja făcută). Pinul RB4 este configurat ca ieșire, iar pe acesta se poate monitoriza apăsarea pe butonul conectat la RB7, conectând unul din LED-urile disponibile la pinul RB4.

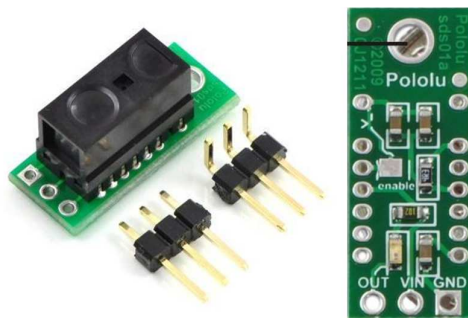
```
#include <p33fxxx.h>
int main( void )
{
    _TRISB7 = 1;
    _TRISB4 = 0;

    while (1)
    {
        if(_RB7 == 1)
        {
            _RB4 = 1; //semnalizeaza apasarea butonului
        }
    };
    return 0;
}
```

#### Aplicație de laborator:

Utilizând un senzor de obstacol POLOLU1134, să se facă legăturile pe placa de dezvoltare, folosind pinul RB15 ca pin de intrare și RB14 ca pin de ieșire și să se modifice codul astfel încât, momentul detectării unui obstacol să fie semnalizat folosind unul din LED-urile disponibile.

Obs: POLOLU1134 este un senzor digital de proximitate ce detectează obiecte aflate între 2 și 10 cm aflate în fața sa. În momentul în care acesta detectează un obstacol, la ieșire avem valoare 0 logic.



**Fig. 6 Senzorul de obstacol POLOLU1134.**