

11.2.DFM și DFT în sisteme EMBEDDED

11.2.1 Perifericul Convertor Analog Digital.

Convertorul analog digital (ADC- Analog to Digital Converter) permite realizarea conversiei unui semnal analogic într-un număr digital pe 10 biți. Perifericul se folosește de intrări analogice ce sunt multiplexate într-un singur circuit de eșantionare. Ieșirea circuitului de eșantionare este conectată la intrarea convertorului analog digital. Acesta furnizează un rezultat pe 10biți folosind un registru de aproximări succesive iar rezultatele sunt stocate în regiștrii ADRESL și ADRESH.

Tensiunea de referință a ADC-ului este selectabilă din program și poate fi fie VDD fie o tensiune externă. ADC-ul poate genera o întrerupere la terminarea unei conversii analog digitale. Aceasta poate trezi și microcontrolerul din starea de consum redus. Schema bloc a ADC-ului este prezentată în Figura 11. 1.

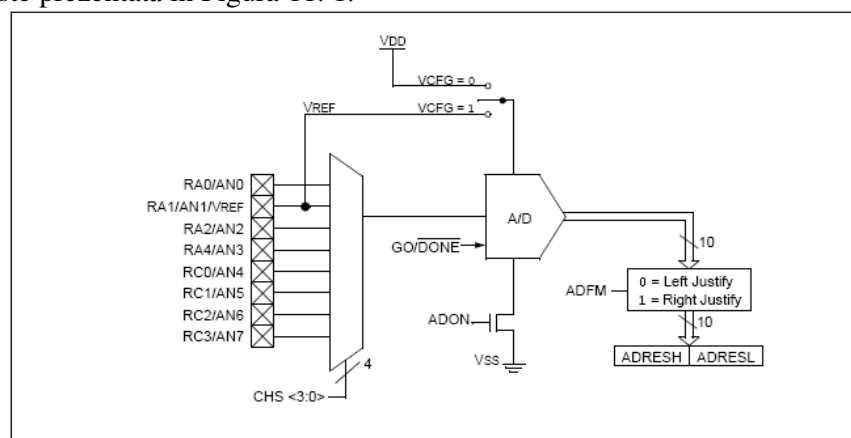


Figura 11. 1. Schema bloc a convertorului ADC

11.2.2 Configurarea ADC-ului.

Când se dorește folosirea ADC-ului acesta trebuie configurat ținând cont de următoare funcții: configurarea pinilor portului, Selecția canalului, selectarea tensiunii de referință, selecția ceasului ADC-ului, controlul întreruperilor, formatarea rezultatelor.

ADC-ul poate converti atât semnale analogice cât și semnale digitale. Când se dorește conversia semnalelor analogice, pinii portului trebuie configurați ca pin analogic. Acest lucru se face cu ajutorul regiștrilor TRIS și ANSEL. Pentru selectarea canalului analogic folosit se folosesc biții CHS din regiștrul ADCON0. Bitul VCFG din regiștrul ADCON0 este folosit pentru selecția tensiunii de referință a ADC-ului.

Structuri hardware si algoritmi specifici microsistemelor EMBEDDED

Ceasul ADC-ului este foarte important deoarece cu ajutorul lui se stabilește viteza de conversie. Acesta ceas trece printr-un divizor programabil prin biții ADCS din ADCON1, și sunt mai multe combinații posibile în funcție și de ceasul microcontrolerului (vezi tabelul 11.1)

Tabelul 11.1

ADC Clock Period (TAD)		Device Frequency (Fosc)			
ADC Clock Source	ADCS<2:0>	20 MHz	8 MHz	4 MHz	1 MHz
Fosc/2	000	100 ns ⁽²⁾	250 ns ⁽²⁾	500 ns ⁽²⁾	2.0 μs
Fosc/4	100	200 ns ⁽²⁾	500 ns ⁽²⁾	1.0 μs ⁽²⁾	4.0 μs
Fosc/8	001	400 ns ⁽²⁾	1.0 μs ⁽²⁾	2.0 μs	8.0 μs ⁽³⁾
Fosc/16	101	800 ns ⁽²⁾	2.0 μs	4.0 μs	16.0 μs ⁽³⁾
Fosc/32	010	1.6 μs	4.0 μs	8.0 μs ⁽³⁾	32.0 μs ⁽³⁾
Fosc/64	110	3.2 μs	8.0 μs ⁽³⁾	16.0 μs ⁽³⁾	64.0 μs ⁽³⁾
Frc	x11	2-6 μs ^(1,4)	2-6 μs ^(1,4)	2-6 μs ^(1,4)	2-6 μs ^(1,4)

1: sursa Frc are un timp de conversie tipic de 3μs.

2: aceste valori încalcă timpul minim de conversie.

3: se recomanda timp de conversie mai scurți.

4: pentru operații de conversie în modul de curent redus doar sursa Frc poate fi folosită.

Dacă se dorește folosirea întreruperilor la terminarea unei conversii atunci în bitul ADIE (ADC interrupt enable) din registrul PIE1 trebuie scrisă valoarea 1, iar stegulețul este bitul ADIF din registrul PIR1. rezultatul obținut este pe 10 biți și este scris în doi regiștrii de 8 biți, astfel că sunt posibile două moduri de formatare a rezultatului (vedeți figura 5.4.2)

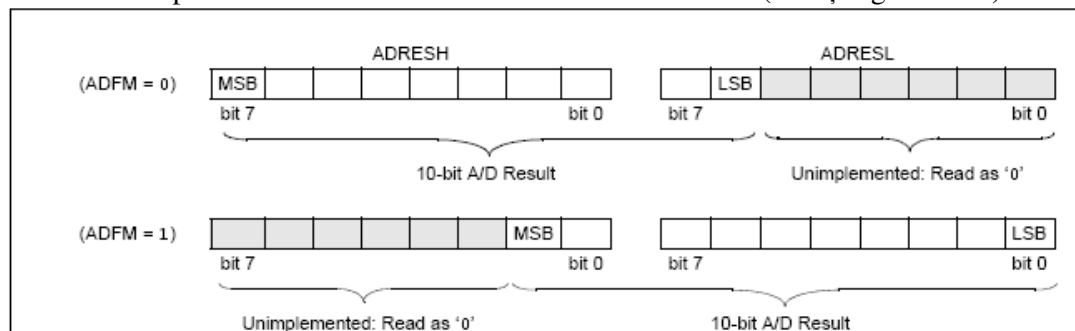


Figura 11. 2. Formatarea rezultatelor ADC-ului

11.2.3 Regiștrii de control ai ADC-ului

Registrul ADCON0

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	VCFG	—	CHS2	CHS1	CHS0	GO/DONE	ADON
bit 7							bit 0

bit 7 **ADFM:** Bitul de formatare a rezultatului

1 = deplasat la dreapta

0 = deplasat la stânga

bit 6 **VCFG:** bitul de selecție a tensiunii de referință

1 = pinul VREF

Structuri hardware si algoritmi specifici microsystemelor EMBEDDED

$$0 = \text{VDD}$$

bit 5 Ne implementat: La citire va indica mereu '0'

bit 4-2 CHS<2:0>: Biții de selecție a canalului analog

$$000 = \text{AN0}$$

001 = AN1

010 = AN2

011 = AN3

100 = AN4

101 = AN5

110 = AN6

111 = AN7

bit 1 **GO/DONE:** bitul de stare a conversiei AD

1 = A/D un ciclu de conversie se află în derularea. Scrierea valorii 1 va porni un nou ciclu de conversie

Acest bit se șterge automat la terminarea conversiei

0 = A/D ciclul de conversie este încheiat și nu se executa nici o conversie.

bit 0 **ADON:** bitul de pornire al ADC-ului

1 = ADC este pornit

0 = ADC este oprit și nu consuma curent

11.2.4 Program exemplu

[illegible]

UNILINE A EUROPEANĂ



MINISTERUL MUNCII, FAMILIEI ȘI
PROTECȚIEI SOCIALE
AMPOSORU



FONDUL SOCIAL EUROPEAN
POSDRU
2007-2013

INSTRUMENTE STRUTTURALE
2007-2013

Structuri hardware si algoritmi specifici microsystemelor EMBEDDED

```

    {
//      nol();
    porta.4=0;          //stingem ledurile care au fost aprinse
                        //si aprindem altele

    porta.5=1;
    porta.2=1;
    porta.1=0;
        i=0;          // contorul i este facut 0, pentru ca la
                        //o noua chemare a functiei Led_control
                        //sa se execute if-ul initial
    tmr0=adc_data;      // in timer-ul 0 se va scrie valarea adc-
                        //ului, daca aceasta este mica va trece
                        //un timp mai lung pana la "overflow" si
                        //asfel aceste led-uri vor sta mai mult
//aprinse (opus ca mai sus)
    }
}

void interrupt( void )    //rutina ce deservește intreruperile
{
    //Handle timer0 interrupt
    if( intcon & (1<<T0IF) )    //daca intreruperea a fost cauzata
                                //de un "overflow" al timer-ului 0
                                //se va executa ce este la acest
                                //"if"
    {
        clear_bit( intcon, T0IF ); //se sterge flagul timer-ului 0,
        // tmr0=adc_data;
        led_control();            //se executa functia "led_control"
                                //care va varia intensitatea unor
                                //led-uri in functie de pozitia
                                //potentiometrului care este citita
                                //cu ajutorul ADC-ului
    }

    if(pir1 & (1<<ADIF))        //daca intreruperea este cauzata de
                                //o terminarea unei conversii
                                //analog -> digitale
    {
        clear_bit(pir1, ADIF); // se sterge flag-ul ADC-ului
        adc_data=adresh;        //iar valaorea "high" a adc-ului se
                                //salveaza intr-o variabila globala,
                                //pozitiva pe 8 biti.
//      adci=256*adresh+adresl;    //save the higher byte or
                                //the ADc result
        adcon0.1=1;            //se porneste o noua conversie
                                //analog digitala
    }
}

void main( void )
{

```



Structuri hardware si algoritmi specifici microsistemelor EMBEDDED

```

oscon=0b01110000;          //oscilatorul intern merge la 8 MHz
osctune=0b00001111;        //frecveta lui este putin marita
ansel=0b00000001;          //porta0 is an anlog input
//Configure port A
trisa = 0b00000001;         //pinul RA0 este in pin de intrare (
                             //si analogic din cauza setarii lui
                             //ansel)
//Configure port C
trisc = 0xFF;               //toti pinii portului C sunt pini
                             //de intrare

//Initialize port A
porta = 0x00;               //pinii din portul A ce sunt setati
                             //ca iesirii au valoarea 0 si led-
                             //urile vor fi stinse
//Initialize port C
portc = 0x00;
i=0;                        //valoarea initiala a variaabilei
                             //globale i este 0;
option_reg = 0b00000011;    //ceasul timer-ului 0 se obtine
                             //divizand ceasul intern cu 16 ->
                             //500khz pentru a nu se observa
                             //licarirea led-urilor

//Enable interrupts (Timer0)
intcon = 0b11100000;        //enable global interrupts,
                             //nonmaskable interrupts and Timer 0
                             //interrupt
piel = 0b01000000;          //enable ADC interrupt
adcon0 = 0b00000001;        //enable ADC
adcon1 = 0b00100000;        //ADC clock =1/32 system clock
adcon0.l=1;                 // (start adc sampling)

//Endless loop
while( 1 )                  //nu se face nimic in bucla
                             //principala. totul se petrece in
                             //intreruperi
{
}
}

```

Bibliografie:

1. Istvan Sztojanov, Sever Pașca, Elisabeta Buzoianu, Aplicații hardware și software cu microcontrolerul PIC12F675, Editura Cavallioti, ISBN 978-973-7622-54-9, Bucuresti 2008
2. Istvan Sztojanov, Alexandru Vasile, Elisabeta Buzoianu, Sever Pașca, *Programarea microcontrolerelor din familia Intel, Aplicații practice hardware cu 80C552*, Editura Man-Dely, ISBN 973-85681-5-3, București 2004.
3. <http://vega.unitbv.ro/~romanca/EmbSys/>
4. <http://facultate.regielive.ro/cursuri/electronica/>
5. www.microcip.com

