5. Convertor Analog Numeric

Funcționalități ADC
Exemplu de utilizare ADC
Aplicații propuse

Fiind un sistem digital, un microcontroller este capabil să proceseze doar semnale binare. Atunci când acesta este alimentat de la 5V, de exemplu, pe pinii de intrare ieșire percepe valoarea de 5V ca valoarea 1 în binar iar 0V ca 0 în binar. Totuși, se întâmplă deseori ca semnalul conectat la microcontroler să fie unul continuu, de exemplu cuprins în intervalul 0 - 5V. Uzual, senzorilor analogici furnizează un astfel de semnal. Pentru a o procesa un asftel de semnal,microcontrolerul dispune de un modul capabil să transforme un semnal analogic într-o valoare numerică (digitala). Acest circuit se numește convertor Analog Numeric (eng. ADC - A*nalog* to D*igital* C*onvertor*).

Convertoarele A/D integrate pe microcontrolere sunt convertoare cu aproximaţii succesive sau, mai rar, convertoare cu integrare. Rezoluţia este de 8, 10 sau 12 biţi. Modulul de conversie este prevăzut şi cu un multiplexor analogic, fiind astfel disponibile mai multe canale de intrare. Unele microcontrolere sunt echipate şi cu circuit de eşantionare/memorare. În cazul în care circuitul de eşantionare/memorare lipseşte, semnalul analogic trebuie menţinut constant pe durata unei conversii. Tensiunea de referinţă necesară convertorului poate fi generată în circuit sau, dacă nu, este necesar să fie furnizată din exterior [5].

Microcontrolerul ATMega328 deține un singur circuit ADC cu comparații succesive, pe 10 biți. Acest convertor este conectat la un multiplexor analog cu 6 canale, ceea ce permite conectarea a șase intrări distincte de tensiune analogică, la Portul C al microcontrolerului (pinii A0:A5 de pe placa Arduino Uno). Modulul de conversie conține un circuit de eșantionare și reținere integrat, acesta asigurând menținerea tensiunii de intrare la o anumită valoare, pe durata conversiei.

Modulul ADC convertește o mărime analogică de intrare într-o valoare digitală pe 10 biți, prim metoda aproximațiilor succesive [1]. Valoarea minimă care poate fi convertită este valoarea de GND iar valoare maximă este reprezentată de tensiunea de pe pinul AREF (la care se scade 1 LSB). Drept referință, se mai pot folosi AV_{CC} sau o tensiune internă de

1.1V. Atunci când se folosește AV_{CC} drept referință este necesară aplicarea unei tensiuni pe pinul AREF. Pentru folosirea referinței interne, trebuie configurat biții REFSn din registrul ADMUX.

Configurarea convertorului analog numeric este posibilă prin intermediul regiștrilor ADMUX și ADCSRA. Registrul ADMUX este responsabil pentru configurarea multiplexorului modulului ADC. Următorii doi biți fac parte din acest registru (figura 5.1:

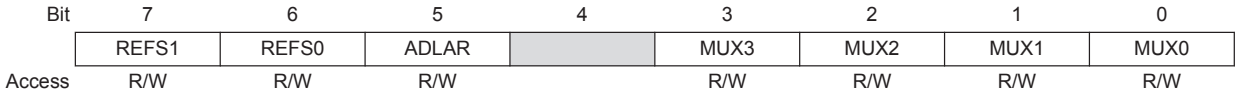

Fig. 5.1 Registrul ADMUX.

– Biții 7:6 – REFSn: Selectarea referinței [n = 1:0]: Acești biți permit selectarea tensiunii de referință pentru ADC. Dacă acești biți sunt schimbați pe parcursul unei conversii, efectul o sa fie vizibil doar după terminarea conversiei curente. Dacă o tensiune externă este aplicată pe pinul AREF, tensiunea internă de referință este posibil să nu mai poată fi utilizată. Următoarele combinații pot fi posibile:
  - 00: AREF, tensiunea internă V_{ref} este oprită;
  - 01: AV_{CC} este utilizată (este necesar un condensator pe pinul AREF);
  - 10: neutilizabil;
  - 11: Tensiune internă de 1.1V.
– Bitul 5 – ADLAR: setarea acestui bit afectează rezultatul conversiei. O valoare de 1 va realiza o ajustare la stânga a rezultatului, în timp ce valoarea 0 va ajusta la dreapta. Rezultatul ajustat al conversiei este în registrul de date ADC: ADCL și ADCH;
– Biții 3:0 – MUXn: Selectare canal analogic [n = 3:0]. Permite selectarea canalului pe care se primește semnalul analogic care urmează a fi transformat în numeric. Pentru combinația 0000 avem ADC0, iar combinația 0101 corespunde ADC5. Combinațiile între cele exemplificate corespund celorlalte canale de intrare.

Cel de al doilea registru important pentru lucrul cu ADC este registrul de control și stare ADCSRA. Următorii biți sunt conținuți de acest registru:

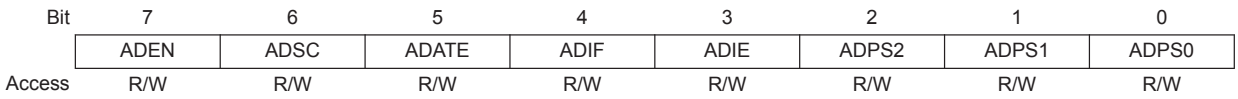

Fig. 5.2 Registrul ADCSRA.

– Bitul 7 – ADEN: Activare ADC: Prin scrierea unui 1 logic în acest bit, este activat modulul ADC. Un zero logic îl dezactivează;
– Bitul 6 – ADSC: Începere conversie ADC: prin scrierea acetui bit este pornită o conversie ADC. Dacă ADC este în modul de conversie "conversie singulară", bitul pornește conversia. În modul "conversie continuă", va determina pornirea primei conversii. În

prima conversie se realizează inițializarea convertorului și procesul de conversie durează mai mult decît în următoarele conversii (aproximativ dublu);
- Bitul 5 – ADATE: Activarea mecanismului de declanșare automată ADC: când acest bit este activat, convertorul analog numeric va porni o conversie pe frontul pozitiv al unui semnal specific de declanșare. Acest semnal este setat din registrul ADCSRB;
- Bitul 4 – ADIF: flag-ul de întrerupere ADC: Acest bit este automat setat când o conversie este finalizată și registrul de date este actualizat. Acest bit este șters în mod automat când este executată întreruperea corespunzătoare. Pentru ștergere manuală, este necesară scrierea unui 1 logic în acest registru;
- Bitul 3 – ADIE: Activare întrerupere ADC. Când acest bit este setat și bitul 7 (I) din registrul SREG este setat, finalizarea procesului de conversie analog numerică va genera o întrerupere;
- Biții 2:0 – ADPSn: selectarea prescalarului pentru ADC [n=2:0]: determină factorul de diviziune între frecvența oscilatorului plăcii de dezvoltare și semnalul de ceas utilizat de ADC. Factorul de divizare variază de la 2 (pentru 000) până la 128 (pentru 111).

Rezultatul conversiei este regăsit în două registre. Aceste registre sunt ADCL și ADCH. Pentru o singură conversie rezultatul este:

$$\mathrm{ADC} = \frac{\mathrm{V_{IN}} \cdot \mathrm{V_{REF}}}{1024} \tag{5.1}$$

5.1 Exemplu de aplicaţie

Să se elaboreze și simuleze, utilizându-se Tinkercad, un program care citește o tensiune variabilă de la un potențiometru și o afișează pe un LCD. Canalul ales pentru intrarea analogică este canalul 0. Tensiunea de referință este de 5V și este aplicată pe pinul AREF. Modificarea tensiunii de alimentare se face prin utilizarea unui potenţiometru. Schema electrică a aplicației propuse este prezentată în figura 5.3. Componentele utilizate în schemă sunt:
- 1x LCD 16x2;
- 1x Arduino Uno R3;
- 1x 220 ohm Resistor;
- 2x Potentiometru;
- 1x TMP36;
- 1x Multimetru;
- 1x Breadboard.

Programul care realizează cerinţele impuse este următorul:

Algoritmul 5.1 Utilizarea convertorului analog numeric.

```
1 #include <LiquidCrystal.h>
2
3 #define ADC_VREF_TYPE ((0<<REFS1) | (0<<REFS0) | (0<<ADLAR))
4
```

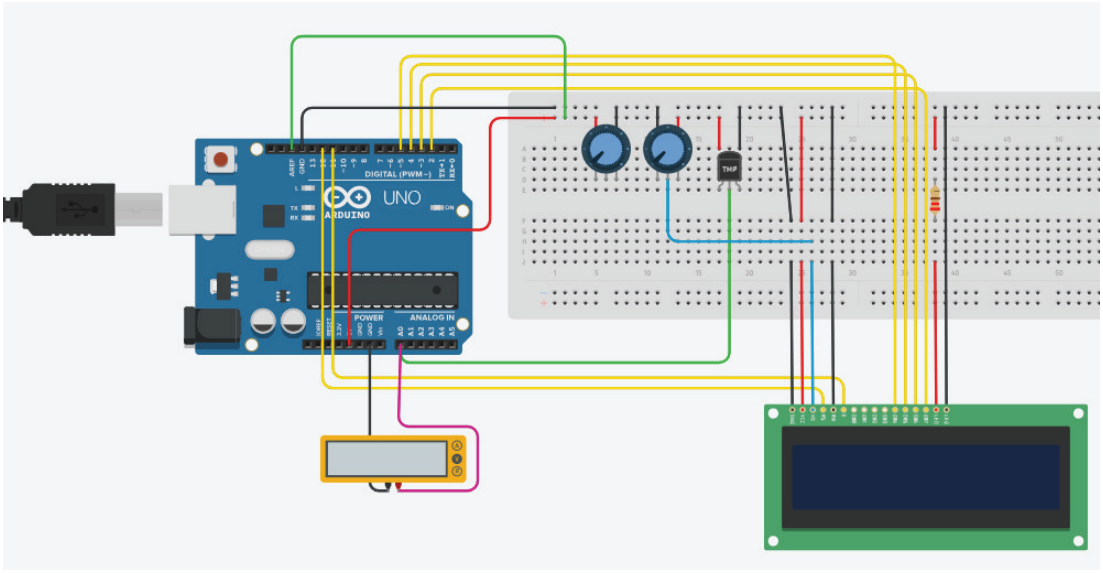

Fig. 5.3 Schema electrică pentru utilizarea modulului ADC.

```
5  /*
6    Conectarea pinilor la placa Arduino UNO:
7   * LCD RS la pinul digital 12
8   * LCD Enable la pinul digital 11
9   * LCD D4 la pinul digital 5
10  * LCD D5 la pinul digital 4
11  * LCD D6 la pinul digital 3
12  * LCD D7 la pinul digital 2
13  * LCD R/W la pinul de masa
14  * LCD VSS la pinul de masa
15  * LCD VCC la pinul 5V
16  * VO la pinul de iesire al potentiometrului
17 */
18
19 // initializare librarie LCD cu pinii corespunzatori
20 LiquidCrystal lcd(12, 11, 5, 4, 3, 2);;//RS,EN,D4,D5,D6,D7
21
22 //stepADC-ul se calculeaza conform formulei (5.1):
23 //fie N rezolutia convertorului (la Arduino N=10, adica 1024) si
     ↪ Vref tensiunea de referinta (la Arduino Vref = Vcc = 5V)
24 //step = Vref / 2^N;
25 //Arduino: step = 5 / 1024 = 0.0048828125
26 float stepADC = 0.0048828125;
27
28 void setup() {
29   // setare numar linii si coloane LCD:
```

```
30    lcd.begin(2, 16);
31    initializareADC();
32  }
33
34  void loop() {
35    // Afisare mesaj pe LCD
36    lcd.print("LCD Display");
37    // setare cursor la coloana 0, lina 1
38    lcd.setCursor(0, 1);
39    // citire intrare analogica
40    unsigned int sensorValue = citesteADC(0);
41    //determinare tenisune de intrare
42    float tensiune = stepADC*sensorValue;
43    //afisare rezultat pe LCD
44    lcd.print(tensiune, 2);
45    //repetare operatiuni la fiecare 100 milisecunde
46    delay(100);
47    //stergere informatie de pe LCD
48    lcd.clear();
49  }
50  unsigned int citesteADC(unsigned char adc_input)
51  {
52    ADMUX= adc_input | ADC_VREF_TYPE;
53    // Delay necesar pentru stabilizarea ADC
54    // pentru tensiunea de intrare analogica
55    delayMicroseconds(10);
56    // Start conversie
57    ADCSRA|=(1<<ADSC);
58    // Asteptare finalizare conversie
59    while ((ADCSRA & (1<<ADIF))==0){}
60    ADCSRA |= (1<<ADIF);
61
62    //Returnare rezultat pe 16 biti
63    return ADCW;
64  }
65
66  void initializareADC()
67  {
68    //dezactivare buffer intrare digitala
69    DIDR0=(0<<ADC5D) | (0<<ADC4D) | (0<<ADC3D) | (0<<ADC2D) | (0<<
        ↪ ADC1D) | (0<<ADC0D);
```

```
70
71    //Configurare multiplexor ADC
72    ADMUX=ADC_VREF_TYPE;
73
74    //configurare registru de control si stare
75    ADCSRA=(1<<ADEN) | (0<<ADSC) | (1<<ADATE) | (0<<ADIF) | (0<<ADIE
          ↪ ) | (1<<ADPS2) | (0<<ADPS1) | (0<<ADPS0);
76 }
```

În programul anterior au fost realizate următoarele setări:

- linia 8: setarea rezoluției ADC conform ecuației 5.1;
- linia 31: configurarea canalului de intrare a semnalului analogic;
- linia 53: configurarea registrului de stare: activare ADC, activarea mecanismului de declanșare automată ADC la funcționare continuă, factor de prescalare de 16.

Drept exemplu concret de utilizare a modulului ADC, exemplul anterior poate fi modificat prin înlocuirea potențiometrului cu un senzor de temperatură analogic. În figura 5.3 este deja adăugat senzorul de temperatură, dar trebuie conectat la canalul 0 al microcontrolerului. În lucrarea de față s-a optat pentru senzorul de temperatura TMP36. Caracteristicile sale sunt următoarele:

- tensiune de operare (2.7V - 5.5V);
- tensiunea de ieșire este proporțională cu valoarea temperaturii, exprimată în °C;
- 10mV/°C;
- ± 2°C acuratețe;
- temperatura de operare: -40°C până la +150°C;
- nu este necesară calibrarea senzorului.

Pentru a înlocuii valoarea citită de la potențiometru cu valoarea efectivă returnată de un senzor de temperatură TMP36, este nevoie de adăugarea următoarei porțiuni de cod:

Algoritmul 5.2 Utilizarea sistemului de întreruperi.

```
1 // calculare temperatura
2 float fTemperature = (stepADC*sensorValue-0.5)*100;
3 //afisare temperatura
4 lcd.print(fTemperature,3);
```

Astfel, pe ecranul LCD se va afișa valoarea temperaturii citite de la senzorul analogic utilizat. Precizia de calcul este de 10 biți, corespunzătoare capabilității maxime a microcontrolerului avut în vedere.

5.2 Aplicaţii propuse

1. Să se implementeze un mecanism de alarmă bazat pe un senzor de gaz (se poate utiliza senzorul analogic MQ2). Aplicația trebuie să fie capabilă să detecteze nivelul de gaz, iar în funcție de concentrație să se afiseze diferite mesaje pe un LCD. Se va utiliza foaia de catalog

a senzorului, pentru determinarea caracteristicilor acestuia.

2. Să se realizeze o stație de monitorizare a temperaturii, umidităţii și a presiuni. Informaţiile colectate sunt afișate pe un ecran LCD. Pentru fiecare mărime trebuie setată o valoare de prag. La depășirea pragului trebuie aprins un led roșu. Atâta timp cât valoarea măsurată este sub prag, un led verde este aprins. Valorile de prag se pot seta din butoane, iar informația este afișată pe un LCD.