
**Sprawozdanie z laboratorium**

**Architektury Komputerów**

**Ćwiczenie numer: 4**

Temat: ADC

Wykonujący ćwiczenie:

- Zaborowska Magda
- Wójtowicz Patryk

Studia dzienne I stopnia

Kierunek: Informatyka

Semestr: III Grupa zajęciowa: Lab 15

Prowadzący ćwiczenie:
Dr inż. Mirosław
Omieljanowicz

.....

OCENA

Data wykonania ćwiczenia

13.12.2021r.

.....

Data i podpis prowadzącego

# Cel zadania

Celem zadania było zapoznanie się z 12-bitowym przetwornikiem analogowo-cyfrowym ADC12 oraz napisanie programu, który umożliwi wyświetlanie temperatury otoczenia na wyświetlaczu. Temperatury mają być wyświetlane w trzech różnych jednostkach: stopnie Celcjusza, Kelwiny i Farenheit.

# Informacje potrzebne do wykonania zadania

Rejestr ADC12 umożliwia szybką, 12-bitową konwersję analogowo-cyfrową. Moduł ten wyposażony jest w 12-bitowy rdzeń SAR, kontrolę wyboru kanałów, generator odniesienia i 16-bitowy bufor, który umożliwia przetworzenie 16 niezależnych próbek bez udziału procesora obliczeniowego. Moduł ADC12 przetwarza napięcie na odpowiadającą mu miarę liczbową. Przetwornik ten w zakresie od -50*C do 100*C ma charakterystykę liniową. Można ją opisać wzorem:

$$V_{TEMP} = 0{,}00355 * T_C + 0{,}986$$

Vtemp to napięcie wytwarzane przez czujnik w temperaturze Tc stopni Celcjusza. Aby użyć czujnika, należy wybrać kanał **INCHx = 1010**, a czas próbkowania ustawić na niemniejszy niż 30µs.

Przystępując do zadania należało ustalić rejestry konfiguracyjne **ADC12CTL0 i ADC12CTL1.** Rdzeń jest aktywowany za pomocą bitu **ADC12ON.** Bity kontrolne mogą być konfigurowane tylko gdy bit **ENC = 0.** W przypadku gdy **ENC = 1** można przejść do konwersji.

**Konfiguracja rejestrów kontrolnych:**

```
ADC12CTL0 = ADC12ON | REFON | SHT0_15; //wł. rdzenia, wł.gen. nap. odniesienia, wybór nap. odniesienia
ADC12CTL1 = SHP | CSTARTADD_0;         //próbkowanie impulsowe, wynik składany w ADC12MEM0
ADC12MCTL0 = INCH_10 | SREF_1;         //kanał 10, źródło nap. odniesienia - wew. generator (1,5V)
```

*Rysunek 1 Konfiguracja*

**ADC12ON** - Rdzeń włączany jest bitem **ADC12ON**. **0** – ADC12 off, **1**- ADC12 on
**REFON** – włącznik generatora napięcia odniesienia, **0** - Reference off, **1** - Reference on
**SHT0x –** bit wybierający czas próbkowania. Definiuje on ilość cykli ADC12CKL w okresie próbkowania. W naszym przypadku x **= 1111** co odpowiada 1024 cyklom.
**SHP -** Ten bit wybiera źródło sygnał próbkowania (SAMPCON), który może być albo wyjściem timera próbkowania, albo bezpośrednio sygnałem wejściowym próbki. **0** - Sygnał SAMPCON pochodzi z sygnału wejściowego próbki, **1**- Sygnał SAMPCON pochodzi z timera próbkowania.
**CSTARTADDx -** Adres początkowy konwersji. Te bity wybierają, który ADC12 rejestr pamięci konwersji jest używany do pojedynczej konwersji lub do pierwszej konwersji w sekwencji. Wartość **CSTARTADD0** oznacza wybór **ADC12MEM0**.

Istnieje 16 rejestrów pamięci konwersji **ADC12MEMx** do przechowywania konwersji wyników. Każdy **ADC12MEMx** jest skonfigurowany z powiązanym **ADC12MCTLx** rejestrem kontrolnym. Ponieważ wybraliśmy **ADC12MEM0,** należy skonfigurować **ADC12MCTL0.**
**INCHx –** bit ten wybiera kanał wejścia. Zgodnie z teorią należało wybrać INCH_10.
**SREFx –** wybiera źródło napięcia odniesienia. W naszym przypadku jest 1,5V.

Po konfiguracji należy odczekać, aby ustabilizował się generator. Zanim przejdziemy do konwersji wartości należy włączyć możliwość konwersji. Służy do tego bit **ENC,** w przypadku **0** zabroniona jest konwersja, w przypadku 1 można wykonać konwersję.

```
ADC12CTL0 |= ENC; //uaktywnienie konwersji
while (1)
{
  P2OUT ^= BIT1;
  ADC12CTL0 |= ADC12SC; //start konwersji
  while ((ADC12CTL1 & ADC12BUSY) == 1);  //czekanie na koniec konwersji
  temp = ADC12MEM0 * 1.0318 - 2777.4647; //wartość temperatury z dok. 1-miejsce po przecinku
  show();                  //wyświetla temperature na LCD
}
```

*Rysunek 2 Konfiguracja*

# Realizacja zadania

```
1   #include <msp430x14x.h>
2   #include "LCD.h"
3   #include "portyLcd.h"
4   #define INTERVAL 50000 //okres licznika 0,5s
5   #define _5s 10         //okres pomiarów
6
7   long long int temp;
8   unsigned int cntr;
9   void show(void);
```

*Rysunek 3 Biblioteki i definicje*

### Funkcja show() wyświetlająca temperaturę:

```
39  void show(void)
40  {
41    long long int cyfra, waga = 10, cyfrafar, wagafar = 10, far = ((2 * temp) + 284), cyfrakel, wagakel = 10, kel = temp + 2730;
42    LCD_cmd(CLR_LCD);
43    wait(_2ms);
44    // stopnie celcjusza
45    if (temp < 0) //jeśli temperatuja jest ujemna
46    {
47      LCD_char('-'); //wyświetlamy ją w minusem
48      temp *= -1;
49    }
50    if (temp < 10) //jeśli jest cyfrą
51      LCD_char('0'); //z przodu wpisujemy 0
52
53    if (temp >= 1000) // jeślli przekroczy 1000
54    {
55      LCD_char('?'); //błąd
56      return;
57    }
58    while (waga <= temp)
59    {
60      waga *= 10;
61    }
62    while ((waga /= 10) > 1) //wyliczanie temperatury
63    {
64      cyfra = temp / waga;
65      LCD_char((int)('0' + cyfra));
66      temp -= cyfra * waga;
67    }
68    LCD_char('.');
69    LCD_char((int)('0' + temp)); //wyświetlanie temperatury
70    LCD_char(' ');
71    LCD_char('C');
72    LCD_char(' ');
```

*Rysunek 4 Show() - Wyświetlanie stopni Celcjusza.*

```
73    // farenhighty analogiczne jak wyżej
74    if (far < 0)
75    {
76      LCD_char('-');
77      far *= -1;
78    }
79    if (far < 10)
80      LCD_char('0');
81    while (wagafar <= far)
82    {
83      wagafar *= 10;
84    }
85    while ((wagafar /= 10) > 1)
86    {
87      cyfrafar = far / wagafar;
88      LCD_char((int)('0' + cyfrafar));
89      far -= cyfrafar * wagafar;
90    }
91    LCD_char('.');
92    LCD_char((int)('0' + far));
93    LCD_char(' ');
94    LCD_char('F');
95    LCD_char(' ');
```

*Rysunek 5 Show() - Wyświetlanie Farenheightów*

```
96    // kelwiny analogicznie jak wyżej
97    LCD_cmd(DD_RAM_ADDR2); //wyświetlanie w 2 wierszu
98    if (kel < 0)
99    {
100     LCD_char('-');
101     kel *= -1;
102   }
103   if (kel < 10)
104   {
105     LCD_char('0');
106   }
107   while (wagakel <= kel)
108   {
109     wagakel *= 10;
110   }
111   while ((wagakel /= 10) > 1)
112   {
113     cyfrakel = kel / wagakel;
114     LCD_char((int)('0' + cyfrakel));
115     kel -= cyfrakel * wagakel;
116   }
117   LCD_char('.');
118   LCD_char((int)('0' + kel));
119   LCD_char(' ');
120   LCD_char('K');
121   LCD_char(' ');
122 }
```

*Rysunek 6 Show() - Wyświetlanie Kelwinów*

**Funkcja main()**

```
main(void)
{

  unsigned int k;
  WDTCTL = WDTPW + WDTHOLD;
  ini_display();                          //inicjalizacja LCD
  ADC12CTL0 = ADC12ON | REFON | SHT0_15; //wł. rdzenia, wł.gen. nap. odniesienia, wybór nap. odniesienia
  ADC12CTL1 = SHP | CSTARTADD_0;          //próbkowanie impulsowe, wynik składany w ADC12MEM0
  ADC12MCTL0 = INCH_10 | SREF_1;          //kanał 10, źródło nap. odniesienia - wew. generator (1,5V)
  for (k = 0; k < 0x3600; k++)
    ; //czas na ustabilizowanie generatora nap. odniesienia

  CCR0 = INTERVAL;                //ustala nowy okres licznika
  TACTL = TASSEL_2 | ID_3 | MC_1; //źródło taktowania SMCLK, dzielone przez 8,tryb UP
  CCTL0 = CCIE;                   //uaktywnienie przerwania od TACCR0 CCIFG
  _BIS_SR(GIE);                   //właczenie przerwan

  ADC12CTL0 |= ENC; //uaktywnienie konwersji
  while (1)
  {
    P2OUT ^= BIT1;
    ADC12CTL0 |= ADC12SC; //start konwersji
    while ((ADC12CTL1 & ADC12BUSY) == 1);  //czekanie na koniec konwersji
    temp = ADC12MEM0 * 1.0318 - 2777.4647; //wartość temperatury z dok. 1-miejsce po przecinku
    show();                   //wyświetla temperature na LCD
    _BIS_SR(LPM0_bits);        //wejście w tryb oszczedny
  }
}
```

```
#pragma vector = TIMERA0_VECTOR
__interrupt void Timer_A(void)
{
  if (++cntr == _5s)
  {
    _BIC_SR_IRQ(LPM0_bits); //wyjocie z trybu oszczednego
    cntr = 0;
  }
}
```

# Wnioski

Rejestr ADC12 umożliwia szybką, 12-bitową konwersję analogowo-cyfrową. Wymaga niewielu konfiguracji, aby móc odczytać wartość temperatury otoczenia. Można powiedzieć, że rejestr ma wolny czas reakcji. Po wystawieniu płytki EasyWeb za okno, temperatura spadała powoli. Potrzebne było odczekanie kilku minut, aby z temperatury pokojowej przejść w wartości ujemne. Niemniej jednak finalny odczyt był poprawny.

# Bibliografia

[1] Informator Laboratoryjny
[2] http://krzysztof.halawa.staff.iiar.pwr.wroc.pl/wyswietlacz_LCD_HD44780.pdf
[3] http://std2.phys.uni.lodz.pl/mikroprocesory/wyklad/asem_10_lcd.pdf