DOI: 10.18668/NG.2018.05.07

Jacek Matusik

Instytut Nafty i Gazu – Państwowy Instytut Badawczy

Mikrokontroler – narzędzie bardzo przydatne w pracach badawczych

W artykule przedstawiono charakterystykę mikrokontrolerów 8-bitowych z rodziny AVR. W szczególności położono nacisk na to, aby pokazać ogromne możliwości tych układów w aspekcie wykorzystania ich w codziennym funkcjonowaniu laboratorium badawczego. W pierwszej części dokonano ogólnej charakterystyki mikrokontrolerów oraz podzespołów współpracujących z tymi układami. Następnie przedstawiono sposób praktycznego wykorzystania mikrokontrolera na przykładzie hipotetycznego stanowiska badawczego.

Słowa kluczowe: mikrokontroler, stanowisko badawcze, czujnik, pomiar, programowanie, język C.

Microcontroller – a very useful tool in research

This article presents the characteristics of 8-bit microcontrollers from the AVR family. In particular, emphasis was placed on showing the great potential of these systems in terms of their use in the everyday functioning of a research laboratory. In the first part, the general characteristics of the microcontrollers and the subassemblies cooperating with these systems, were made. An example of the practical use of a microcontroller is shown on the example of a hypothetical test stand.

Key words: microcontroller, test stand, sensor, measurement, programming, language C.

Wprowadzenie

Powszechna automatyzacja i chęć ciągłej poprawy jakości życia sprawiają, że elektronika cyfrowa nieubłaganie wypiera z naszego otoczenia elektronikę analogową. Już niemal każde z otaczających nas urządzeń elektronicznych zawiera w sobie układ mikroprocesorowy, z czego często nawet nie zdajemy sobie sprawy. Nie inaczej jest również w laboratoriach badawczych. Stosowane jeszcze niedawno analogowe mierniki wskazówkowe zastąpione zostały wysokospecjalizowanymi przyrządami cyfrowymi. Podobna zmiana objęła również całe stanowiska badawcze. Układ sterowania, który do tej pory zajmował sporo miejsca, czasami wręcz całe szafy, teraz mieści się w małej „czarnej skrzynce", a obsługa stanowiska sprowadza się do kilku kliknięć myszką komputera. Istnieje jednak istotna wada tego postępu cywilizacyjnego – koszt zakupu wspomnianej „czarnej skrzynki". Kwota, którą trzeba zapłacić za specjalnie wykonane na zamówienie urządzenie dedykowane, może osiągnąć poziom skutecznie zniechęcający niejednego zainteresowanego. Jest to jednak zrozumiałe, gdyż stworzenie takiego urządzenia wymaga niezbędnej wiedzy z zakresu elektroniki, automatyki, informatyki i innych dziedzin pokrewnych. Nic więc dziwnego, że konstruktorzy odpowiednio cenią swoją pracę. Autor artykułu, który miał przyjemność po trosze zgłębić tę wiedzę, twierdzi jednak, że nie zawsze musimy być skazani na korzystanie z cudzych rozwiązań. Umysł analityczny, odrobina samozaparcia i przyswojenie niewielkiej ilości niezbędnej wiedzy z zakresu elektroniki pozwolą konstruować proste, lecz w pełni funkcjonalne urządzenia elektroniczne. Przedstawiony w tym artykule praktyczny przykład może być tego dowodem.

Układ mikroprocesorowy

Każde urządzenie zawierające w sobie mikrokontroler należy rozpatrywać z dwóch punktów widzenia:

- sprzętu (hardware), czyli części związanej z dziedziną elektroniki,

- oprogramowania (software), czyli części związanej z dziedziną informatyki.

Elementy te powinny ze sobą ściśle współpracować, tworząc w efekcie działające urządzenie o określonych cechach użytkowych [5]. Elementem pośredniczącym pomiędzy tymi dwiema płaszczyznami i jednocześnie najważniejszą częścią układu jest mikrokontroler, będący zarządcą i strażnikiem całego urządzenia. Aby mógł jednak pełnić tę funkcję, należy go odpowiednio zaprogramować, tj. zlecić mu wykonywanie jasno określonego zadania (algorytmu), które pozwoli osiągnąć założony cel. Działanie mikrokontrolera nierozerwalnie związane jest z jego otoczeniem, czyli częścią sprzętową. Do portów wejściowych mikrokontrolera może dopływać mnóstwo różnych informacji, które wedle uznania programisty mogą w odpowiedni sposób wpływać na działanie algorytmu. Źródłem takich sygnałów może być na przykład klawiatura lub jakikolwiek inny interfejs pozwalający użytkownikowi na komunikowanie się z urządzeniem. Największą jednak grupę elementów wejściowych stanowią różnego rodzaju czujniki i sensory monitorujące wybrane parametry procesowe. Mikrokontroler analizuje otrzymane informacje wejściowe, a następnie zgodnie z zaimplementowanym algorytmem wpływa na działanie podłączonych do swoich portów wyjściowych elementów wykonawczych lub podejmuje inne niezbędne czynności.

Mikrokontroler

Czym właściwie jest mikrokontroler? Wiele osób mylnie utożsamia go z mikroprocesorem spotykanym w komputerach PC. Mikroprocesor znajdujący się w komputerze do swojej pracy potrzebuje układów peryferyjnych, takich jak: dysk twardy, pamięć RAM, karta graficzna, oprzyrządowanie płyty głównej itd. Dopiero te wszystkie elementy umieszczone w jednej obudowie tworzą funkcjonalną całość w postaci komputera. Wyobraźmy sobie, że mikrokontroler jest właśnie takim komputerem, zamkniętym w małej obudowie w postaci układu scalonego [1]. Porównanie takie jest oczywiście daleko idącym przybliżeniem. W rzeczywistości mikrokontroler jest układem znacznie prostszym i mniej wydajnym niż komputer. Nie ma to jednak większego znaczenia, gdyż jego zastosowanie jest zupełnie inne. Moce obliczeniowe oferowane przez mikrokontrolery są wystarczające do sprostania niemal każdemu, nawet najbardziej wymagającemu zadaniu stawianemu w układach automatyki, sterowania czy w każdym innym zastosowaniu. Mikrokontroler nie posiada systemu operacyjnego i od razu po włączeniu zasilania przystępuje do realizacji algorytmu, który może być zaimplementowany w postaci programu komputerowego.

Możliwości mikrokontrolerów

Obecnie na rynku dostępna jest niezliczona liczba tych układów. Różnią się one od siebie wielkością, mocą obliczeniową, wbudowanymi układami peryferyjnymi oraz oczywiście ceną. Ceny układów wahają się od kilku do kilkudziesięciu złotych. W artykule tym opisana została bardzo szeroka rodzina mikrokontrolerów 8-bitowych AVR produkcji Atmel. Tego typu mikrokontrolery mogą być taktowane częstotliwością nawet 32 MHz. W rzeczywistości tak duża moc obliczeniowa potrzebna jest tylko w najbardziej wymagających zadaniach. Bardziej interesujące z punktu widzenia projektanta układu elektronicznego jest to, jakimi układami peryferyjnymi dysponuje mikrokontroler. W tablicy 1 przedstawiono dostępne w układach AVR wewnętrzne moduły oraz możliwości ich zastosowania [3].

Przedstawione w tablicy 1 opisy wewnętrznych układów pokazują mnogość zastosowań mikrokontrolerów. O prawdziwym

Tablica 1. Peryferia i układy wewnętrzne mikrokontrolerów AVR

Wewnętrzny układ	Możliwości zastosowania
Jednostka arytmetyczno-logiczna	– wykonywanie nawet najbardziej skomplikowanych operacji matematycznych
Pamięć flash (pamięć programu)	– pamięć nieulotna, miejsce na program (algorytm)
Pamięć RAM	– pamięć ulotna
Pamięć EEPROM	– pamięć nieulotna, przechowywanie danych roboczych nawet po zaniku zasilania
Porty (nóżki mikrokontrolera)	– dwustronna komunikacja z otoczeniem mikrokontrolera
Przetwornik analogowo-cyfrowy	– bardzo dokładny pomiar napięcia, prądu, rezystancji, mocy i wielu innych parametrów – rejestracja dźwięku
Przetwornik cyfrowo-analogowy	– generowanie dźwięku – generowanie dowolnego sygnału napięciowego
Komparator analogowy	– porównywanie dwóch wartości napięć

cd. Tablica 1

Wewnętrzny układ	Możliwości zastosowania
Timer/licznik	– zliczanie różnego rodzaju zdarzeń wewnętrznych lub zewnętrznych – licznik impulsów – generator impulsów – generator sygnałów (np. PWM) – pomiar częstotliwości – odmierzanie czasu
Moduł UART	– dwukierunkowa transmisja danych w standardzie RS-232 – dwukierunkowa transmisja danych w standardzie RS-485
Moduł SPI	– dwukierunkowa komunikacja z układami w otoczeniu mikrokontrolera
Moduł TWI (I^2C)	– dwukierunkowa komunikacja z układami w otoczeniu mikrokontrolera
System przerwań	– zwiększenie możliwości funkcjonalnych, poprawa elastyczności pracy
Watchdog	– czuwa nad poprawnością pracy mikrokontrolera

spektrum możliwości można przekonać się jednak dopiero po przeanalizowaniu tablicy 2, prezentującej przykłady układów, z którymi mogą one współpracować. Układy te dołączane są do mikrokontrolera za pomocą jego portów. Do komunikacji pomiędzy nimi można zastosować któryś z wbudowanych sprzętowo w mikrokontroler modułów transmisji lub poprzez dołączenie odpowiedniej biblioteki można emulować niemal każdy inny rodzaj wymiany danych. Na uwagę zasługuje również sposób podłączenia układów wykonawczych. Moc, którą można pobrać z portu mikrokontrolera, jest niewielka, wystarczy do zasilenia jedynie niewielkich odbiorników, np. diody LED. Zastosowanie jednak elementów pośredniczących pozwoli na sterowanie każdym odbiornikiem, nawet tym o największej mocy.

Tablica 2. Układy, z którymi może współpracować mikrokontroler AVR

Zewnętrzny układ	Charakterystyka
UKŁADY WEJŚCIOWE	
Czujnik temperatury (np. Pt100, termoelement, czujnik cyfrowy)	Działanie większości czujników polega na przekształceniu nieelektrycznej wartości zmierzonej na sygnał elektryczny. Najczęściej spotykane sygnały wyjściowe czujników to [7]: – zwarcie – rozwarcie styków – sygnał prądowy 4÷20 mA – sygnał napięciowy 0÷10 V – rezystancja – pojemność – częstotliwość – transmisja RS-232 – transmisja RS-485 – transmisja 1-wire – transmisja SPI – transmisja TWI (I^2C) – inne…
Czujnik wilgotności, punktu rosy	
Czujnik ciśnienia, barometr	
Czujnik gazu (np. metanu, wodoru, czadu)	
Czujnik alkoholu	
Czujnik pH	
Czujnik ruchu	
Czujnik przepływu	
Czujnik poziomu	
Czujnik przyspieszenia, żyroskop	
Czujnik odległości	
Czujnik wibracji	
Czujnik nacisku	
Czujnik dźwięku	
Czujnik natężenia i barwy światła	
Czujnik koloru	
Czujnik pola magnetycznego	
Zegar czasu rzeczywistego	Bardzo dokładny pomiar czasu

cd. Tablica 2

Zewnętrzny układ	Charakterystyka
UKŁADY WYKONAWCZE	
Silnik DC	Może osiągać bardzo duże obroty, dostępne różne moce silników
Silnik krokowy	Bardzo duży moment obrotowy, możliwość wykonania precyzyjnego obrotu o dowolny kąt, praca krokowa
Serwomechanizm	Bardzo duży moment obrotowy, możliwość wykonania precyzyjnego obrotu o dowolny kąt
Elektrozawór	Otwieranie i zamykanie przepływu gazu lub cieczy
Kontaktron, przekaźnik, tranzystor MOSFET, optotriak, transoptor	Układy pośredniczące w sterowaniu urządzeniami dużej mocy
Głośnik, buzzer	Generowanie dźwięku, pojedynczych sygnałów dźwiękowych
Dioda laserowa	Źródło bardzo mocnego promieniowania spójnego
INTERFEJSY UŻYTKOWNIKA	
Klawiatura, przycisk, switch	Umożliwiają wydawanie mikrokontrolerowi pojedynczych komend lub wprowadzanie całych ciągów znaków
Dioda LED	Sygnalizacja stanów, zdarzeń
Wyświetlacz ciekłokrystaliczny tekstowy	Umożliwia wyświetlanie tekstu i liczb. Dostępne w różnych rozmiarach, maksymalnie 4 × 40 znaków
Wyświetlacz ciekłokrystaliczny graficzny	Umożliwia wyświetlanie grafiki, wykresów, tekstu i liczb. Dostępne w różnych rozmiarach, maksymalnie 240 × 128 pikseli
MODUŁY TRANSMISJI DANYCH	
Moduł Bluetooth	Umożliwia bezprzewodową komunikację mikrokontrolera z drugim urządzeniem, np. komputerem lub smartfonem (zasięg do kilkudziesięciu metrów)
Dioda IR	Umożliwia bezprzewodową komunikację mikrokontrolera z drugim urządzeniem (zasięg transmisji ograniczony zasięgiem światła podczerwonego)
Moduł Ethernet Moduł Wi-Fi	Umożliwia komunikację za pośrednictwem sieci internetowej z serwerem lub dowolnym innym urządzeniem podłączonym do sieci
Moduł GSM	Umożliwia komunikację bezprzewodową za pośrednictwem sieci komórkowej, np. ze smartfonem lub dowolnym urządzeniem wyposażonym w moduł GSM
Moduł radiowy 433 MHz	Umożliwia komunikację bezprzewodową za pośrednictwem fal radiowych z dowolnym urządzeniem wyposażonym w moduł radiowy 433 MHz
Karta SD	Zapis danych roboczych na zewnętrznej przenośnej pamięci

Środowisko i język programowania

W kwestii programowania do wyboru mamy kilka możliwości. W przypadku mikrokontrolerów AVR najlepszym wyborem, według autora, jest skorzystanie z dostarczanego przez producenta tych układów darmowego środowiska do programowania AVR Studio. Co do języka programowania tu też jest kilka możliwości. Najlepszym wyborem będzie jednak język C. Jest to bardzo uniwersalny i intuicyjny język programowania, który doskonale sprawdza się również przy pracy z mikrokontrolerami. Niewątpliwą zaletą języka C, która ułatwia pracę początkującym programistom, jest to, że w sieci WWW można znaleźć wiele przykładowych programów. Analizując je, dużo łatwiej zrozumieć ideę działania mikrokontrolerów. Dodatkowo w księgarniach dostępnych jest sporo książek poruszających tematykę programowania mikrokontrolerów w tym języku (np. [2, 3]). Autorzy książek, widząc coraz większe zainteresowanie mikrokontrolerami, prześcigają się, aby w jak najbardziej przystępny sposób opisać tę mimo wszystko niełatwą tematykę. Przejdźmy zatem do przykładu praktycznego.

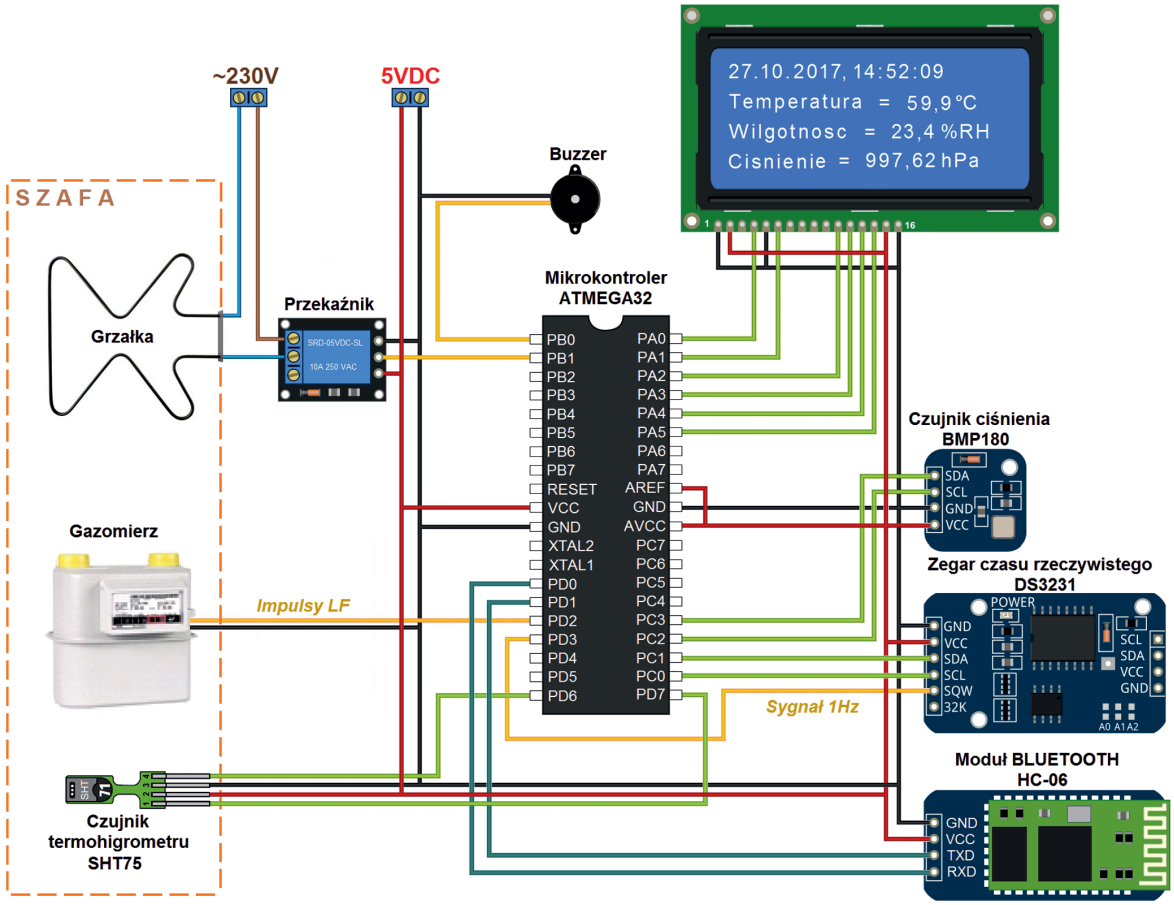
Przykład praktyczny

Rozważmy przykład zastosowania mikrokontrolera w charakterze hipotetycznego sterownika stanowiska badawczego do sprawdzania odporności gazomierzy miechowych na długotrwałe oddziaływanie wysokiej temperatury. Badanie na

tym stanowisku polegać ma na umieszczeniu na okres dwóch miesięcy gazomierza wewnątrz izolowanej termicznie szafy, w której za pomocą zainstalowanej wewnątrz grzałki utrzymywana będzie temperatura na poziomie 60°C (±3°C). Wymagania odnośnie do pracy stanowiska badawczego są następujące:

- w pełni autonomiczna praca,
- utrzymywanie temperatury w szafie grzewczej na poziomie niewykraczającym poza dopuszczalne granice,
- sterowanie grzałką o mocy 500 W,
- pomiar temperatury wewnątrz szafy grzewczej,
- pomiar wilgotności względnej wewnątrz szafy grzewczej,
- pomiar ciśnienia atmosferycznego,
- zliczanie impulsów LF z gazomierza,
- pomiar czasu,
- sygnalizacja dźwiękowa w sytuacji, gdy temperatura w szafie jest zbyt wysoka,
- wyświetlanie na zintegrowanym wyświetlaczu aktualnie zmierzonych parametrów,
- bezprzewodowa transmisja danych do komputera PC.

Wyżej wymienionym zadaniom z powodzeniem sprosta mikrokontroler Atmega32. Układem pośredniczącym w sterowaniu grzałką będzie przekaźnik z wyjściem zwiernym o dopuszczalnym obciążeniu prądowym 5 A. Jako czujnik warunków środowiskowych w szafie zastosowano cyfrowy sensor SHT75 firmy Sensirion, charakteryzujący się stosunkowo dobrą dokładnością pomiarową (niepewność pomiaru temperatury: ±0,3°C, a wilgotności ±2,0% RH). Do pomiaru ciśnienia atmosferycznego wybrano czujnik BMP180 firmy Bosch (niepewność pomiaru: ±0,12 hPa). Pomiar czasu realizowany będzie za pomocą układu DS3231 produkcji Dallas Semiconductor (niepewność pomiaru czasu: ±2 ppm). Wszystkie trzy wymienione powyżej układy posiadają interfejs komunikacyjny I^2C. Źródłem impulsów w gazomierzu jest zwarciowy kontaktron magnetyczny, co umożliwia podłączenie go bezpośrednio do portu mikrokontrolera. Zliczanie impulsów zostanie zrealizowane programowo. Generatorem sygnału alarmowego będzie niewielki buzzer piezoelektryczny, który również może zostać podłączony bezpośrednio do portu mikrokontrolera. Aktualne wyniki pomiarów prezentowane będą na tekstowym wyświetlaczu LCD o rozdzielczości 4 × 20 znaków. Bezprzewodowa transmisja danych do komputera PC zrealizowana zostanie w standardzie Bluetooth za pomocą dołączonego modułu HC-06. Na rysunku 1 przedstawiono poglądowy schemat zaprojektowanego sterownika. Aby nie utrudniać analizy schematu, nie zawarto na nim kilku drobnych, ale niezbędnych do poprawnej pracy układu elementów (kondensatory filtrujące, rezystory podciągające linie sygnałowe, rezystory ograniczające prąd, potencjometr regulacji kontrastu wyświetlacza).



Rys. 1. Schemat ideowy sterownika szafy grzewczej

```
#include <avr/io.h>              // dołączenie pliku nagłówkowego z definicjami rejestrów i bitów mikrokontrolera
#include <avr/interrupt.h>       // dołączenie pliku nagłówkowego obsługi przerwań
#include "HD44780.h"             // dołączenie do programu biblioteki wyświetlacza LCD
#include "usart.h"               // dołączenie do programu biblioteki z funkcjami transmisji RS232
#include "DS3231.h"              // dołączenie do programu biblioteki układu DS3231
#include "SHT75.h"               // dołączenie do programu biblioteki czujnika temperatury i wilgotności SHT75
#include "BMP180.h"              // dołączenie do programu biblioteki czujnika ciśnienia BMP180

TDATETIME datetime;              // stworzenie w pamięci struktury przechowującej datę i czas
float temperatura;               // stworzenie zmiennej przechowującej wartość temperatury
float wilgotnosc;                // stworzenie zmiennej przechowującej wartość wilgotności względnej
float cisnienie;                 // stworzenie zmiennej przechowującej wartość ciśnienia atmosferycznego
volatile unsigned int impulsy;   // stworzenie zmiennej przechowującej liczbę zliczonych impulsów

int main (void)                  // program główny
{
    LCD_init();                  // inicjalizacja wyświetlacza LCD
    TWI_init();                  // inicjalizacja interfejsu TWI
    UART_init();                 // inicjalizacja interfejsu UART
    DS3231_init();               // inicjalizacja układu DS3231
    SHT75_init();                // inicjalizacja czujnika SHT75
    BMP180_init();               // inicjalizacja czujnika BMP180

    DDRB |= (1<<PB0);            // przygotowanie nóżki 0 w porcie B do współpracy z grzałką
    DDRB |= (1<<PB1);            // przygotowanie nóżki 1 w porcie B do współpracy z buzzerem

    MCUCR |= (1<<ISC01)|(1<<ISC00);  // konfiguracja wejścia współpracującego z wyjściem impulsowym gazomierza
    GICR  |= (1<<INT0);          // cd. konfiguracji wejścia współpracującego z wyjściem impulsowym gazomierza
    GIFR  |= (1<<INT0);          // cd. konfiguracji wejścia współpracującego z wyjściem impulsowym gazomierza

    sei ();                      // globalne zezwolenie na przerwania

    while(1)                     // główna pętla programu
    {
        if( GIFR & (1<<INTF1) )                  // sprawdzanie, czy układ DS3231 przysłał informację o upłynięciu kolejnej sekundy
        {
            Get_Date_Time();                     // odczytanie z układu DS3231 aktualnej daty i godziny

            LCD_GoTo(0, 0);                      // ustaw kursor wyświetlacza LCD w pozycji: kolumna 0, wiersz 0
            Display_Date_Time();                 // wyświetlenie na wyświetlaczu LCD aktualnej daty i godziny

            Get_temp_SHT75(temperatura);         // odczytanie z czujnika SHT75 aktualnie zmierzonej temperatury

            LCD_GoTo(0, 1);                      // ustaw kursor wyświetlacza LCD w pozycji: kolumna 0, wiersz 1
            LCD_Write_Text(„Temperatura = ”)     // wyświetlenie na wyświetlaczu LCD tekstu „Temperatura = ”
            LCD_Write_Float(temperatura);        // wyświetlenie na wyświetlaczu LCD wartości temperatury
            LCD_Write_Data(223);                 // wyświetlenie na wyświetlaczu LCD symbolu „°”
            LCD_Write_Text("C");                 // wyświetlenie na wyświetlaczu LCD symbolu „C”

            Get_hum_SHT75(wilgotnosc);           // odczytanie z czujnika SHT75 aktualnie zmierzonej wilgotności względnej

            LCD_GoTo(0, 2);                      // ustaw kursor wyświetlacza LCD w pozycji: kolumna 0, wiersz 2
            LCD_Write_Text(„Wilgotnosc = ”)      // wyświetlenie na wyświetlaczu LCD tekstu „Wilgotnosc = ”
            LCD_Write_Float(wilgotnosc);         // wyświetlenie na wyświetlaczu LCD wartości wilgotności względnej
            LCD_Write_Text("%RH");               // wyświetlenie na wyświetlaczu LCD jednostki „%RH”

            Get_press_BMP180(cisnienie);         // odczytanie z czujnika BMP180 aktualnie zmierzonego ciśnienia

            LCD_GoTo(0, 3);                      // ustaw kursor wyświetlacza LCD w pozycji: kolumna 0, wiersz 3
            LCD_Write_Text(„Cisnienie = ”)       // wyświetlenie na wyświetlaczu LCD tekstu „Cisnienie = ”
            LCD_Write_Float(cisnienie);          // wyświetlenie na wyświetlaczu LCD wartości ciśnienia
            LCD_Write_Text("hPa");               // wyświetlenie na wyświetlaczu LCD jednostki „hPa”
```

Rys. 2. Kod programu sterownika szafy grzewczej

```
                if (temperatura < 57)                  // jeśli temperatura jest mniejsza niż 57°C...
                  PORTB |= (1<<PB1);                   // ...to włącz grzałkę
                if (temperatura > 63)                  // jeśli temperatura jest większa niż 63°C...
                  PORTB &= ~(1<<PB1);                  // ...to wyłącz grzałkę

                if(temperatura > 70)                   // jeśli temperatura jest większa niż 70°C...
                  PORTB |= (1<<PB0);                   // ...włącz buzzer (alarm sygnalizujący przegrzanie)
                else                                   // w przeciwnym wypadku...
                  PORTB &= ~(1<<PB0);                  // ...wyłącz buzzer

                if(PC_send)                            // jeśli otrzymano od komputera PC żądanie...
                  send_parameters_to_PC();             // ...to parametry zostają wysłane do komputera

                GIFR |= (1<<INTF1);                    // przygotowanie układu na oczekiwanie upłynięcia kolejnej sekundy
            }
        }
}

ISR(INT0_vect)                 // przerwanie zewnętrzne (w momencie wykrycia impulsu z gazomierza mikrokontroler przerywa
{                              // wykonywanie programu głównego i przechodzi do tego przerwania)
    impulsy++;                 // zmienna zliczająca liczbę impulsów zwiększa się o 1
}

void send_parameters_to_PC(void)       // funkcja wysyłająca do komputera wartości wszystkich parametrów roboczych
{
    unsigned char buf[23];     // stworzenie tymczasowego bufora na parametry do wysłania

    buf[0] = 253;              // bajt startu

    buf[1]  = datetime.year;   // rok
    buf[2]  = datetime.month;  // miesiąc
    buf[3]  = datetime.day;    // dzień
    buf[4]  = datetime.hh;     // godzina
    buf[5]  = datetime.mm;     // minuta
    buf[6]  = datetime.ss;     // sekunda

    buf[7] = (impulsy & 0xFF);                         // impulsy (młodszy bajt)
    buf[8] = (impulsy & 0xFF00)>>8;                    // impulsy (starszy bajt)

    buf[9]  = (temperatura & 0xFF);                    // temperatura (pierwszy bajt)
    buf[10] = (temperatura & 0xFF00)>>8;               // temperatura (drugi bajt)
    buf[11] = (temperatura & 0xFF0000)>>16;            // temperatura (trzeci bajt)
    buf[12] = (temperatura & 0xFF000000)>>24;          // temperatura (czwarty bajt)

    buf[13] = (wilgotnosc & 0xFF);                     // wilgotność (pierwszy bajt)
    buf[14] = (wilgotnosc & 0xFF00)>>8;                // wilgotność (drugi bajt)
    buf[15] = (wilgotnosc & 0xFF0000)>>16;             // wilgotność (trzeci bajt)
    buf[16] = (wilgotnosc & 0xFF000000)>>24;           // wilgotność (czwarty bajt)

    buf[17] = (cisnienie & 0xFF);                      // ciśnienie (pierwszy bajt)
    buf[18] = (cisnienie & 0xFF00)>>8;                 // ciśnienie (drugi bajt)
    buf[19] = (cisnienie & 0xFF0000)>>16;              // ciśnienie (trzeci bajt)
    buf[20] = (cisnienie & 0xFF000000)>>24;            // ciśnienie (czwarty bajt)

    unsigned int CRC;                                  // stworzenie zmiennej sumy kontrolnej
    CRC = suma_kontrolna(buf, 21);                     // obliczenie sumy kontrolnej

    buf[21] = (unsigned char)(CRC & 0x00FF);           // suma kontrolna (młodszy bajt)
    buf[22] = (unsigned char)(CRC >> 8);               // suma kontrolna (starszy bajt)

    for(unsigned char i=0; i <= 22; i++)
        uart1_putc(buf[i]);                            // wysłanie danych
```

Rys. 2 cd. Kod programu sterownika szafy grzewczej

Układ przedstawiony na rysunku 1 można fizycznie wykonać na kilka sposobów. Najprostszym z nich jest wykorzystanie tzw. płytki uniwersalnej lub stykowej. Rozwiązania te są jednak mało estetyczne. Autor poleca skorzystanie z programu KiCad. Choć jest to oprogramowanie typu *open source*, umożliwia ono jednak zaprojektowanie w pełni profesjonalnej płytki drukowanej. W tablicy 3 zawarto zestawienie kosztów wszystkich części niezbędnych do samodzielnego wykonania opisanego przykładu. Podane ceny są cenami uśrednionymi na podstawie ofert kilku popularnych sklepów internetowych. Na rysunku 2 przedstawiono najprostszy kod programu, który może realizować założone cele. Do każdej linii kodu dołączono komentarz wyjaśniający znaczenie poszczególnych fragmentów programu.

Tablica 3. Zestawienie kosztów wykonania opisanego stanowiska badawczego

Element	Cena, koszt wykonania
Mikrokontroler Atmega32	10 zł
Czujnik termohigrometru SHT75	110 zł
Czujnik ciśnienia BMP180	14 zł
Zegar czasu rzeczywistego DS3231	6 zł
Moduł Bluetooth HC-06	20 zł
Wyświetlacz LCD 4 × 20 znaków	30 zł
Przekaźnik 5 A	2 zł
Buzzer 5 V	1 zł
Elementy drobne	2 zł
Zasilacz +5VDC	15 zł
Płytka drukowana (wykonana „domowym sposobem”)	10 zł
Grzałka	80 zł
ŁĄCZNIE:	**300 zł**
Możliwość zaprojektowania działania układu ściśle według naszych potrzeb	bezcenne!

Podsumowanie

Celem, który przyświecał autorowi podczas pisania tego artykułu, było zainteresowanie tematyką mikrokontrolerów osób niemających do tej pory styczności z tymi układami. W szczególności autor chciał przedstawić ogromne możliwości tych układów w aspekcie zastosowania ich w codziennym funkcjonowaniu laboratorium badawczego. W artykule zaprezentowany został praktyczny sposób wykorzystania mikrokontrolera na przykładzie prostego stanowiska badawczego. Choć przedstawione stanowisko jest nieskomplikowane, to jednak trudno byłoby znaleźć na rynku gotowe rozwiązanie, które zrealizowałoby założone cele. Przy nieco bardziej złożonych procedurach badawczych niezbędne byłoby zlecenie wykonania stanowiska badawczego osobie lub firmie specjalizującej się w tego typu projektach. Obie możliwości, choć dobre, wiążą się jednak ze znacznymi kosztami. Przedstawiony przykład pokazuje, że każdy może podjąć wyzwanie tworzenia prostych układów opartych na mikrokontrolerach. Dobre opanowanie tej sztuki z pewnością przyniesie wymierne korzyści wszędzie tam, gdzie zachodzi konieczność wykonywania żmudnych, czasochłonnych, powtarzalnych i dokładnych pomiarów, jak również innych czynności. Przykłady innych stanowisk badawczych, na których doskonale sprawdziłby się mikrokontroler, opisano w artykułach: [4, 6, 8].

Prosimy cytować jako: Nafta-Gaz 2018, nr 5, s. 391–398, DOI: 10.18668/NG.2018.05.07

Artykuł nadesłano do Redakcji 9.01.2018 r. Zatwierdzono do druku 20.03.2018 r.

Literatura

[1] Borkowski P.: *AVR i ARM7. Programowanie mikrokontrolerów dla każdego.* Helion 2010.
[2] Francuz T.: *Język C dla mikrokontrolerów AVR od podstaw do zaawansowanych aplikacji*. Helion 2011.
[3] Kardaś M.: *Mikrokontrolery AVR. Język C – podstawy programowania*. Atnel 2013.
[4] Kuśnierczyk J.: *Badania przepuszczalności rdzeni wiertniczych z użyciem różnych płynów złożowych.* Nafta-Gaz 2015, nr 2, s. 87–96.
[5] Pawluczuk A.: *Sztuka programowania mikrokontrolerów AVR*. BTC, Warszawa 2006.
[6] Szuflita S.: *Badania laboratoryjne oddziaływania gazów kwaśnych na skałę zbiornikową w procesach sekwestracji CO_2*. Nafta-Gaz 2016, nr 7, s. 520–527, DOI: 10.18668/NG.2016.07.04.
[7] Szymczyk P., Szymczyk M., Gajer M.: *Cyfrowe czujniki do pomiarów wielkości nieelektrycznych w automatyce*. Pomiary Automatyka Robotyka 2010, nr 4, s. 18–21.
[8] Warnecki M.: *Badania procesów wypierania metanu przy udziale sekwestracji CO_2*. Nafta-Gaz 2015, nr 3, s. 159–166.

Mgr inż. Jacek MATUSIK
Starszy specjalista inżynieryjno-techniczny;
Zastępca kierownika Laboratorium Wzorcującego,
Zakład Metrologii Przepływów
Instytut Nafty i Gazu – Państwowy Instytut Badawczy
ul. Lubicz 25 A, 31-503 Kraków
E-mail: *jacek.matusik@inig.pl*