

# System wykrywania ruchu

---

Urządzenie oparte o układ Atmega8

Konrad Grzelczyk

16.03.2019

## Spis treści

Przeznaczenie .....	3
Komponenty .....	4
Schemat układu .....	5
Instrukcja obsługi .....	6
Wybrane funkcje.....	6

## 1. Przeznaczenie

Urządzenie powstało w celu wykrywania ruchu w pomieszczeniach, tudzież innych przestrzeniach zamkniętych. Opracowane zostało tak, aby zapewnić długie i skuteczne działanie.

Zasilania bazuje na bateriach AA (4 lub 6 sztuk) co pozwala nieprzerwanie działać urządzeniu przez co najmniej 14 dni. Kontroler został zaprogramowany w tryb uśpienia co zwiększa czas pracy urządzenia.

Kontroler Atmega8 działający z częstotliwością 2MHz zapewnia obsługę czujnika oraz modułu GSM. Urządzenie po uruchomieniu dokonuje inicjalizacji oraz informuje użytkownika o poprawnym działaniu. System dokonuje pomiaru stanu baterii co godzinę i w razie zbyt niskiego wyniku pomiaru wysyła powiadomienie do użytkownika. W przypadku wykrycia ruchu, system wykonuje podwójne sprawdzanie i w przypadku prawidłowych wykryć powiadamia użytkownika.

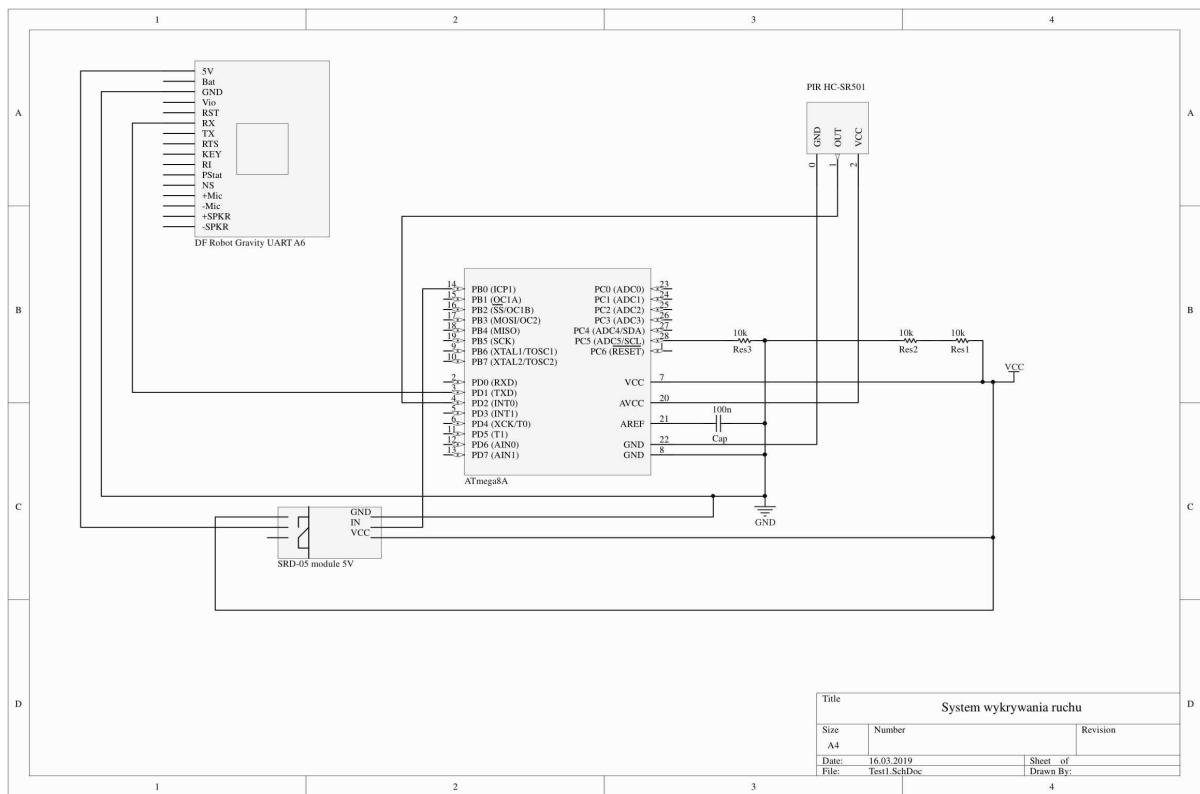
Komponenty zostały umieszczone w uszczelnionej plastikowej obudowie. Docelowo urządzenie będzie pracować w warunkach zewnętrznych.

## 2. Komponenty

Komponenty układu:

- Kontroler Atmega8.
- Moduł przekaźnika SRD-05.
- Moduł GSM DF Robot Gravity UART A6.
- Rezystory (3x 10k).
- Kondensator 100nF.
- Koszyk na baterie.
- Czujnik ruchu PIR HC-SR501.

### 3. Schemat układu



## 4. Instrukcja obsługi

Aby uruchomić urządzenie należy odkręcić 4 śruby zabezpieczające obudowę oraz zdjąć górną pokrywę. Następnie należy włożyć odpowiednią ilość baterii do koszyka i aktywować przełącznik. Po 2-3 minutach od włączenia przełącznika, użytkownik powinien otrzymać wiadomość z potwierdzeniem poprawnego działania systemu. Kończąc obsługę urządzenia należy nałożyć oraz przykręcić górną część obudowy.

## 5. Wybrane funkcje

Opis najważniejszych funkcji realizowanych przez system wraz z kodem aplikacji. Software został utworzony w języku C w środowisku Eclipse (wykorzystując AVR-Dude). Za instalowanie oprogramowania odpowiedzialny jest programator USB-asp.

Opis wybranych funkcji:

- Inicjalizacja przetwornika ADC

```
//////// ADC
ADCSRA = (1<<ADEN) | (1<<ADPS0) | (1<<ADPS1) | (1<<ADPS2);
ADMUX = (1<<REFS1) | (1<<REFS0) | (1<<MUX2) | (1<<MUX0); // wybór kanału ADC5 na pinie PC5
DDRC=0xff; //Ustawienie pinu jako wejście
DDRC &=~ (1<<PC5);
```

- Konfiguracja przerwań

```
MCUCR |= (1<<ISC01) | (1<<ISC00); // przerwanie wywołane poprzez zbocze narastające INT0
GICR |= (1<<INT0);

sei(); //Globalne uruchomienie przerwań
```

- Konfiguracja komunikacji UART

```
void USART_Init( unsigned int ubrr)
{
    UBRRH = (unsigned char) (ubrr>>8);
    UBRL = (unsigned char)ubrr;
    UCSRB = (1<<TXEN);
    UCSRC = (1<<URSEL) | (3<<UCSZ0);
}
```

- Komunikacja UART

```
void Usart_Transmit(unsigned char data)
{
    while ( !( UCSRA & (1<<UDRE)) );
    UDR = data;
}
```

```
void Send_clause(char * napis)
{
    while(*napis)
        Usart_Transmit(*napis++);
}
```

- Sprawdzanie stanu baterii

```
void CheckBattery()
{
    ADCSRA = (1<<ADEN);

    int pomiary = 0, srednia=0;

    for(int i=0; i<3; i++)
    {
        ADCSRA |= (1<<ADSC); //ADSC: uruchomienie pojedynczej konwersji

        while(ADCSRA & (1<<ADSC)); // czeka na zakończenie konwersji

        pomiary += ADC;
    }

    srednia = pomiary / 3; // wynik pomiaru to srednia z 3 prob

    if(srednia <= 736) // ADC = 750 ~ 5,62V
    {
        USART_Init(__UBRR);
        PORTB &= ~(1<<PB0);
        Delay(2);
        Initialization_GSM();
        sendSMS("Miejsce 1. Niski stan baterii!");
    }
}
```

- Konfiguracja timera oraz jego przerwanie

```
void TMRO_init( void )
{
    TIMSK |= (1<<TOIE0);           //Przerwanie overflow (przepełnienie timera)
    TCCR0 |= (1<<CS02) | (1<<CS00); // źródłem CLK, preskaler 1024
    TCNT0 = 0;                      //Początkowa wartość licznika
}

ISR(TIMER0_OVF_vect)
{
    TCNT0 = 0;                      //Początkowa wartość licznika

    cnt++;                          //zwiększa zmienna licznik
    cnt2++;                         //zwiększa zmienna licznika ponownego wykrycia
}
```

- Przerwanie zewnętrzne (czujnik)

```
ISR (INT0_vect) // czujnik
{
    if(wykrycie == 0)
    {
        czy_wyslano = 0;

        if ((PIND & (1<<PD2))) // pierwsze wykrycie
        {
            czy_wyslano = 1;
        }

        if(czy_wyslano == 1)
        {
            Delay(2);
            USART_Init(__UBRR);
            PORTB &= ~(1<<PB0);
            Delay(2);
            Initialization_GSM();

            if ((PIND & (1<<PD2))) // drugie wykrycie
            {
                sendSMS("Miejsce 1. Wykryto ruch");
                wykrycie = 1;
                cnt2 = 0;
            }
            PORTB |= (1<<PB0);
            USART_Off();
            czy_wyslano = 0;
        }
    }
}
```



- Usypianie procesora

```
void Go_to_sleep()  
{  
    sleep_enable();  
    set_sleep_mode(SLEEP_MODE_IDLE);  
    sleep_cpu();  
}
```