
	<p>Instytut Informatyki Politechniki Śląskiej</p> <p>Zespół Mikroinformatyki i Teorii Automatów Cyfrowych</p>			
Rok akademicki:	Rodzaj studiów*: SSI/NSI/NSM	Przedmiot (Języki Asemblerowe/SMiW):	Grupa	Sekcja
2017/2018	NSI	SMiW		
Imię:	Robert	Prowadzący: OA/JP/KT/GD/BSz/GB	GB	
Nazwisko:	Andrzejczuk			
<h2><i>Raport końcowy</i></h2>				
<p>Temat projektu:</p> <p>Lokalizator GPS do pojazdu, wysyłający wiadomości SMS gdy dany pojazd się poruszy.</p>				
<p>Data oddania: dd/mm/rrrr</p>		<p>10.09.2018</p>		

Temat projektu.

Tematem projektu jest budowa Lokalizatora GPS do pojazdu, wysyłającego wiadomości SMS gdy dany pojazd się poruszy.

Opis założeń.

Urządzenie ma składać się z mikroprocesora, akcelerometru, modułu GSM i modułu GPS. Po wykryciu ruchu przez akcelerometr wysłana zostaje wiadomość SMS do właściciela pojazdu w którym znajduje się urządzenie. Treść wiadomości ma wskazywać aktualne położenie urządzenia.

Analiza zadania.

Aby wykonać zadanie, najlepszym rozwiązaniem wydaje się być podłączanie kolejnych istotnych elementów po kolei, a następnie sprawdzanie czy działają poprawnie. Aby to było możliwe wykorzystano płytke stykową, aby mieć pełną kontrolę nad elementami, oraz możliwość zmiany ułożenia ich jeżeli zajdzie taka potrzeba. Pierwszym krokiem będzie podłączenie mikroprocesora, zaprogramowanie go i sprawdzenie czy wszystko działa poprawnie za pomocą diody lub monitora portu szeregowego. Kolejnym krokiem będzie podłączenie modułu z akcelerometrem, oraz napisanie prostego programu sprawdzającego jego działanie. Następnie podłączony zostanie Moduł GSM i zaprogramowany w taki sposób, aby wysyłał SMS y o wcześniej ustalonej treści pod odpowiedni numer telefonu. Ostatnim elementem do podłączenia pozostanie moduł GPS. Po jego przetestowaniu należy napisać program który będzie spełniał pełną funkcjonalność końcowego urządzenia.

Wykorzystane elementy i narzędzia.

Mikroprocesor wykorzystany w urządzeniu to Atmega 328P Pu. Został on wybrany ze względu na możliwość wykorzystania biblioteki SoftwareSerial, dzięki której możliwe było podłączenie kilku urządzeń wykorzystujących komunikację poprzez UART. Kolejnym powodem wyboru tego mikroprocesora było to, że posiada on wbudowane piny służące do komunikacji przez SPI oraz I2C. Ponadto układ charakteryzuje się niską ceną. Kolejnym wybranym przez mnie elementem był akcelerometr lis3dh, potrafiący kontrolować przyspieszenie w 3 osiach, oraz posiadający możliwość komunikacji zarówno przez SPI jak i I2C. Wykorzystany w urządzeniu moduł GPS to Fibocom G510 wybrany ze względu na niską cenę oraz możliwość komunikacji przez UART. Ostatnim modulem zainstalowanym w urządzeniu jest moduł GPS, którym jest Ublox neo-6m. Został on wybrany ze względu na komunikację przez UART oraz prostotę obsługi. Całość została zmontowana na płytce uniwersalnej. Narzędziami wykorzystanymi do stworzenia urządzenia były płytka stykowa, aby móc zmieniać ułożenie elementów oraz w razie potrzeby poprawić połączenia do nieodpowiednich pinów. Programator avr prog-mkII-eco, służący do wgrywania programów na mikroprocesor, oraz zmianę Fusebitów. Konwerter USB-UART, dzięki któremu możliwe było włączenie monitora portu szeregowego na komputerze, aby sprawdzać, co się dzieje wewnątrz urządzenia. Dodatkowo wykorzystano również diody, aby sprawdzać działanie niektórych funkcji.

Schemat.

Schemat blokowy urządzenia, oraz sposób rozłożenia elementów na płytce uniwersalnej znajduje się na githubie pod następującym adresem:

<https://github.com/RobotSwastaken/Lokalizator-SMIW> Na schemacie płytki połączenia niebieskie oznaczają połączenia, lutowane, oraz istotne ścieżki, które już znajdowały się na płytce, natomiast połączenia czerwone oznaczają kable znajdujące się nad płytką

Opis elementów układu.

1. Moduł GSM Fibocom G510

Połączony przez UART moduł otrzymuje komendy AT, które w przypadku tego urządzenia wykorzystywane są jedynie do ustawienie modułu w tryb tekstowy i wysyłania wiadomości pod odpowiedni numer telefonu.

Moduł znajduje się na płytce

Najważniejsze Piny:

Vin – napięcie zasilania (5-12V)

GND – masa

Vcc – napięcie zasilania logiki(zasilanie mikrokontrolera)

Rx – UART Rx

Tx – UART Tx

On – uruchamia modem GSM

Moduł wyposażony jest w złącze anteny oraz złącze karty SIM

2.Moduł GPS

Połączony przez UART moduł pobiera swoją lokalizację i wysyła ją w formacie NMEA do mikrokontrolera.

Moduł znajduje się na płytce

Najważniejsze Piny:

Vcc– napięcie zasilania (3,3-5V)

GND – masa

Rx – UART Rx

Tx – UART Tx

Moduł wyposażony jest w złącze anteny oraz zasilanie zapasowe

3.Akcelerometr

Podłączony przez I2C moduł przesyła aktualne wartości przyspieszenia do mikrokontrolera. Moduł znajduje się na płytce.

Najważniejsze Piny:

Vin– napięcie zasilania (3,3-5V)

GND – masa

3Vo-wyjście zasilania 3.3V

SCL pin wykorzystywany do podłączenia I2C. Ma podłączony rezystor podciągający 10k

SDA pin wykorzystywany do podłączenia I2C. Ma podłączony rezystor podciągający 10k

SDO pin wykorzystywany do połączenia I2C, służy do zmiany adresu

Opis działania urządzenia.

Po uruchomieniu urządzenia mikrokontroler próbuje nawiązać kontakt z siecią GSM. W tym czasie moduł GPS rozpoczyna próby określenia swojej lokalizacji. Po zakończeniu czasu na nawiązanie komunikacji z siecią GSM mikrokontroler rozpoczyna odczyt danych z akcelerometru. Każdy kolejny odczyt jest porównywany z poprzednim i jeżeli różnica między 2 kolejnymi odczytami jest wystarczająco duża mikrokontroler rozpoczyna przygotowania do wysłania wiadomości SMS. Pierwszym zadaniem mikrokontrolera jest uzyskanie poprawnego odczytu z modułu GPS. Jeśli GPS jeszcze nie określił swojej lokalizacji mikrokontroler czeka aż moduł to wykona. Po uzyskaniu odpowiedniego odczytu mikrokontroler przekształca uzyskane dane na długość i szerokość geograficzną, a następnie rozkazuje modułowi GSM wysłanie wiadomości pod wskazany numer telefonu. Następnie moduł odczeka 5 minut po czym ponownie rozpoczyna odczytywanie wartości akcelerometru.

Algorytm oprogramowania urządzenia.

```
#include <Adafruit_LIS3DH.h>
#include <SoftwareSerial.h>
#include <TinyGPS.h>
#include <Wire.h>
```

```
TinyGPS gps; // Reprezentacja modułu GPS
Adafruit_LIS3DH ada = Adafruit_LIS3DH(); // Reprezentacja akcelerometru
```

```
SoftwareSerial nss(2, 3); // RX, TX Softwarowy port UART wykorzystywany do komunikacji z
modułem GPS
SoftwareSerial mySerial(A2, A3); // RX, TX Softwarowy port UART wykorzystywany do
komunikacji z modułem GSM
```

```
int _pinGSMOn = A1; //Pin do podłączenia pinu POWER_ON z modułu GSM
int _pin = A0; //Pin umożliwiające podpięcie diody, sygnalizującej niektóre zdarzenia w
urządzeniu
int addr = 0x18; //Adres akcelerometru
int x,y,xl,yl; //Zmienne przechowujące wartości odczytane przez akcelerometr
```

```
#define DIFFERENCE 5000 //Ustawienie czułości akcelerometru
void Blink() // Funkcja migająca diodą podłączoną pod zmienną _pin
{
    digitalWrite(_pin, LOW);
    delay(1000);

    digitalWrite(_pin, HIGH);
    delay(1000);
}
```

```
void GSM_G510_init() //Funkcja rozpoczyna działanie modułu GSM
{

    mySerial.begin(9600);
    pinMode(_pinGSMOn, OUTPUT);
    digitalWrite(_pinGSMOn, LOW);

    Serial.println("Laczenie z siecia GSM");
```

```
    delay(20000);  
}
```

```
boolean GSM_G510_isConnecting() // Funkcja sprawdza czy moduł GSM jest połączony z siecią  
{
```

```
    if (GSM_G510_getSignalStrength() > 0)  
    {  
        return true;  
    }  
    else  
    {  
        return false;  
    }  
}
```

```
float GSM_G510_getSignalStrength() // Funkcja zwraca siłę sygnału sieci GSM  
{  
    mySerial.flush();  
    mySerial.write("\r\n");  
    delay(100);  
    mySerial.write("AT+CSQ\r\n");  
    mySerial.flush();  
  
    return mySerial.parseFloat();  
}
```

```
boolean GSM_G510_sendSms(char* phoneNumber, char* message) { // Funkcja wysyłająca wiadomości SMS
```

```
    mySerial.setTimeout(10000);
```

```
    mySerial.flush();  
    mySerial.write("\r\n");  
    delay(10);
```

```
    mySerial.write("AT+CMGF=1\r\n");
```

```
    if (mySerial.findUntil("OK", "ERROR\r\n"))  
    {  
        mySerial.flush();
```

```
        mySerial.write("AT+CMGS=\"");  
        mySerial.write(phoneNumber);  
        mySerial.write("\r\n");  
        mySerial.flush();
```

```
        if (mySerial.findUntil(">", "ERROR\r\n"))  
        {
```

```
            mySerial.write(message);  
            mySerial.write("\x1A\r\n");
```

```
            mySerial.setTimeout(10000);
```

```

    mySerial.flush();

    if (mySerial.findUntil("+CMGS: ", "ERROR\r\n"))
    {
        mySerial.setTimeout(1000);
        return true;
    }
}

mySerial.setTimeout(1000);
return false;
}

void setup() { //Funkcja wykonywana raz na początku działania urządzenia

    // put your setup code here, to run once:
    Serial.begin(9600);
    GSM_G510_init();
    nss.begin(9600);

    pinMode(_pin, OUTPUT);
    digitalWrite(_pin, LOW);
    ada.begin();
    ada.read();

    xl=abs(ada.x);
    yl=abs(ada.y);
}

void loop() { //Funkcja wykonywana w pętli po wykonaniu funkcji setup

    long lat, lon;
    unsigned long fix_age;

    ada.begin();
    ada.read();

    x=abs(ada.x);
    y=abs(ada.y);

    if(x>xl+DIFFERENCE||x<xl-DIFFERENCE||y>yl+DIFFERENCE||y<yl-DIFFERENCE)
    {
        nss.begin(9600);
        digitalWrite(_pin, HIGH);
        while(1){
            if (nss.available()){
                int c = nss.read();
                Serial.println(c);
                if (gps.encode(c)){
                    gps.get_position(&lat, &lon, &fix_age);
                    break;
                }
            }
        }
        Serial.println("not available");
    }
}

```

```

}

char szerokosc [12];
  ltoa(lat,szerokosc, 10);
  char dlugosc [12];
  ltoa(lon,dlugosc, 10);

  char tresc[200] = "Twój pojazd znajduje się na następujących współrzędnych: Szerokosc="
";
  strcat(tresc,szerokosc);
  strcat(tresc," Dlugosc=");
  strcat(tresc,dlugosc);

int _Counter = 0;
int _CounterMax = 10;
delay(1000);
GSM_G510_init();
while (!GSM_G510_sendSms("+48601941981", tresc))
{
  _Counter++;

  Serial.println("Proba wyslania sms");
  delay(500);
  if(_Counter >= _CounterMax)
  {
    break;
  }

}
if(_Counter < _CounterMax)
{
  Serial.println("Wyslano sms");
}
else
{
  Serial.println("Blad wysylania");
}

delay(300000);
ada.begin();
ada.read();

x=abs(ada.x);
y=abs(ada.y);
}
else{
  digitalWrite(_pin, LOW);
}
xl=x;
yl=y;
}

```

Specyfikacja zewnętrzna.

Do elementów sterujących urządzeniem można zaliczyć przycisk podłączony do pinu reset i uziemienia, który po naciśnięciu powoduje że urządzenie rozpoczyna program od nowa. Dodatkowo urządzenie posiada wtyki żeńskie do pinów RXD i TXD, umożliwiające podłączenie konwertera USB, a także wtyki żeńskie do pinów RESET MOSI MISO i SCK, służące do podłączenia programatora.

Obsługa urządzenia polega na włączeniu go oraz odczekaniu około 2 minut. Po tym czasie urządzenie powinno być gotowe do wysyłania wiadomości SMS jeśli się poruszy. Aby upewnić się że moduł GSM otrzyma rozkazy związane z inicjalizacją, należy najpierw podłączyć zasilanie baterią 9V, a następnie zasilanie główne, lub po włączeniu zasilania nacisnąć przycisk reset.

Problemy podczas uruchamiania i montażu.

1.Brak komunikacji z Modułem GSM

Po podłączeniu zrobionej na zamówienie płytki z modułem GSM nie było możliwe uzyskanie od niego jakiegokolwiek odpowiedzi

Rozwiązanie: Problemem prawdopodobnie było spalenie portu RX w module GSM ponieważ okazało się że logika Atmegi 32A Pu działa na napięciu 5V podczas gdy logika modułu GSM działała na napięciu 3V, w związku z czym zamówiony został fabrycznie zrobiony moduł z płytką

2.Dalsze Problemy z komunikacją z modułem GSM

Po podłączeniu modułu GSM pod fizyczny port UART i wysyłaniu do niego poleceń nie udało się uzyskać odpowiedzi

Rozwiązanie:

Zamiast podłączać urządzenie pod fizyczny UART podłączono je pod UART Softwerowy, co rozwiązało problem.

3.Brak obsługi biblioteki SoftwareSerial przez mikrokontroler Atmega 32A Pu

Rozwiązanie: Zmieniona mikrokontroler na Atmege328P Pu

4. Niepoprawne działanie funkcji delay.

Funkcja delay powodowała opóźnienie 16 razy dłuższe niż powinna

Rozwiązanie:

Po podłączeniu oscylatora kwarcowego o częstotliwości 16MHz, odpowiednich kondensatorów oraz zmienienu ustawienia FuseBitów w mikrokontrolerze udało się rozwiązać problem.

5.Moduł GSM nie wysyłał SMS ów

Gdy możliwa była komunikacja między mikrokontrolerem a modułem GSM, niektóre polecenia w tym Wysyłanie SMS ów powodowały brak odpowiedzi Modułu, oraz gaśnięcie diody na nim
Rozwiązanie:

Dodano baterie 9V i podłączono ją pod pin modułu GSM odpowiadający za zasilanie, gdyż problemem okazało się za niskie natężenie prądu.

Testy poprawności działania urządzenia.

Urządzenie było pierwotnie konstruowane na płycie stykowej. Podczas tego etapu do testowania wykorzystywano odpowiednio zaprogramowane diody, lub konwerter USB-UART, aby sprawdzić poprawność fizycznego podłączenia urządzenia oraz czy dany moduł działa poprawnie. Po przeniesieniu urządzenia na płytkę uniwersalną ponownie został podłączony konwerter, aby na komputerze za pomocą monitora portu szeregowego sprawdzić poprawność wykonywanych zadań. Kolejnym testem było odłączenie konwertera i sprawdzenie czy SMS zostanie poprawnie wysłany. Ostatnim testem była próba urządzenia w samochodzie. Sprawdzono czy po włączeniu urządzenia i ruszeniu samochodem w którym się ono znajdowało zostanie wysłany SMS z prawidłową lokalizacją.

Wnioski z uruchamiania i testowania

Po wykonaniu testów urządzenie działało prawidłowo, jednak jeżeli urządzenie znajduje się w budynku może mieć problemy z określeniem swojej lokalizacji, co uniemożliwia mu wysyłanie SMS a, dlatego najlepiej jest je testować na otwartej przestrzeni. Czasem jednak zdarza się, że urządzenie dosyć szybko określi swoją lokalizację nawet w budynku.