Urządzenie telemetryczne z modułem GSM SIM300D

AVT 5261

Dzięki powszechnej dostępności oraz niskim cenom modułów GSM/GPRS i odbiorników GPS, stało się możliwe skonstruowanie zwartego systemu telemetrycznego. W artykule opisano budowę oraz oprogramowanie urządzenia, które może znaleźć zastosowanie m.in. jako lokalizator pojazdów, element systemu zdalnego nadzoru i kontroli urządzeń lub alarmowego.

Rekomendacje: *urządzenie przyda się każdemu zajmującemu się monitoringiem pojazdów, ochroną obiektów lub automatyką przemysłową.*

AVT-5261 w ofercie AVT:
AVT-5261A – płytka drukowana

Podstawowe informacje:
- Wbudowane moduły GPS i GSM/GPRS,
- Sterowanie przez mikrokontroler z rdzeniem ARM7,
- Interfejsy RS485 i CAN, 4 wejścia cyfrowe, 4 wyjścia typu otwarty kolektor, 2 wejścia analogowe,
- Złącza anten GPS i GSM,
- Transmisja danych z użyciem protokołu TCP/IP,
- Nastawy przez USB lub zdalnie TCP/IP,
- Napięcie zasilania: 6...27 VDC,
- Średni pobór prądu: 110 mA przy 12 VDC

Dodatkowe materiały na CD i FTP:
ftp://ep.com.pl, user: 16719, pass: 8b13241g
- wzory płytek PCB
- karty katalogowe i noty aplikacyjne elementów oznaczonych w wykazie elementów kolorem czerwonym

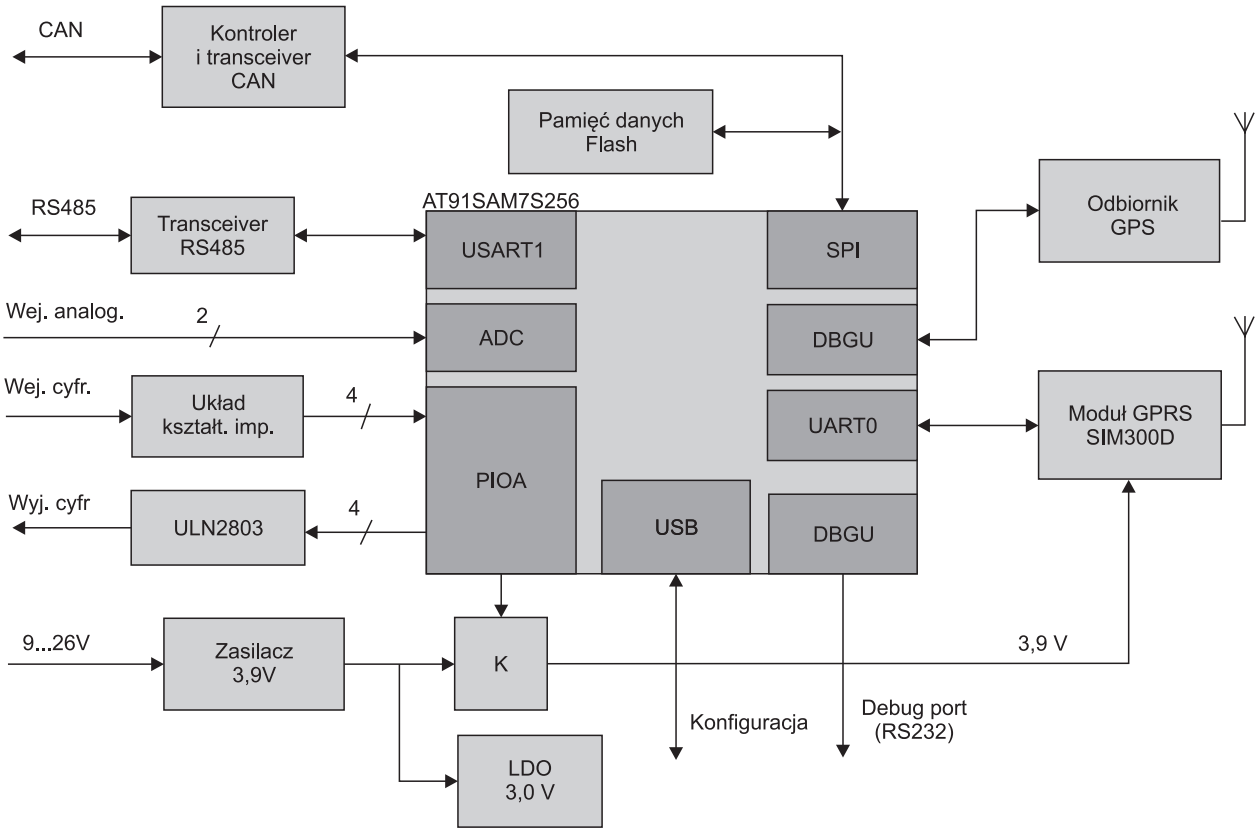
Projekty pokrewne na CD i FTP:
(wymienione artykuły są w całości dostępne na CD)
AVT-5231 Lokalizator samochodu (EP 4/2010)
AVT-2777 Centrala alarmowa GSM (EdW 2/2006)

Zasada działania

Schemat blokowy urządzenia pokazano na **rysunku 1**. Do jego sterowania zastosowano mikrokontroler AT91SAM7S256 z rdzeniem ARM7. Wybór był podyktowany wielkością dostępnej pamięci RAM (64 kB), dużą wydajnością obliczeniową, a przede wszystkim bogatym zestawem układów peryferyjnych. Dzięki temu można było wyposażyć urządzenie w interfejsy: RS485, CAN, 4 wejścia cyfrowe, 4 wyjścia typu otwarty kolektor oraz 2 wejścia analogowe, co znacznie rozszerzyło wachlarz jego zastosowań.

Zadaniem programu mikrokontrolera jest przetwarzanie danych odbieranych z odbiornika GPS oraz interfejsów CAN i RS485, a następnie przesyłanie ich do serwera z wykorzystaniem protokołu TCP/IP (GPRS). W przeciwnym kierunku możliwe jest przesyłanie danych z serwera do interfejsu RS485 lub zdalne sterowanie stanem wyjść dwustanowych.

W zewnętrznej pamięci Flash z interfejsem szeregowym są przechowywane na-



Rysunek 1. Schemat blokowy urządzenia telemetrycznego

stawy urządzenia. Pamięć ta służy także do buforowania danych z odbiornika GPS oraz pomiarowych w momentach, gdy połączenie TCP/IP z serwerem jest niedostępne.

Interfejs USB pracuje w trybie emulacji interfejsu szeregowego RS232 i jest wykorzystywany do wstępnego skonfigurowania urządzenia z poziomu komputera PC.

Na **rysunku 2** zamieszczono główną część schematu ideowego urządzenia zawierającą mikrokontroler, kontroler CAN U10, sterowniki magistrali CAN i RS485 (U11, U9) oraz elementy pomocnicze portu USB. Częstotliwość taktowania mikrokontrolera, określoną przez rezonator kwarcowy XTAL1 oraz wewnętrzną pętlę PLL, ustalono na 48 MHz dla zapewnienia prawidłowej pracy wbudowanego kontrolera USB. Elementy R18, C21, C22 stanowią filtr dolnoprzepustowy pętli PLL. Porty I/O mikrokontrolera oraz część pamięci Flash zasilane są napięciem 3,0 V doprowadzonym do VDDFLASH oraz VDDIO. Rdzeń mikrokontrolera i PLL wymagają napięcia z przedziału 1,65...1,95 V. Jest ono dostarczane przez wewnętrzny stabilizator liniowy, który obniża napięcie 3,0 V podawane na pin VDDIN. Napięcie z wyjścia stabilizatora VDDOUT trafia następnie na piny VDDCORE oraz VDDPLL. Wewnętrzny stabilizator oraz każdy z pinów VDDIO, VDDCORE i VDDPLL wymagają kondensatorów filtrujących. Są to C55...C58, C35...C42, C55...C58.

Dioda LED2 dołączona do portu PA15 spełnia rolę sygnalizatora stanu pracy urządzenia. Na złączu CONN3 wyprowadzony został debug port – wyjście wewnętrznego synchronicznego interfejsu szeregowego SSC. Pracuje on w niestandardowym trybie nadajnika RS232. Zastępuje w tym przypadku interfejs szeregowy DBGU (debug unit), co umożliwia użycie DBGU do komunikacji z odbiornikiem GPS.

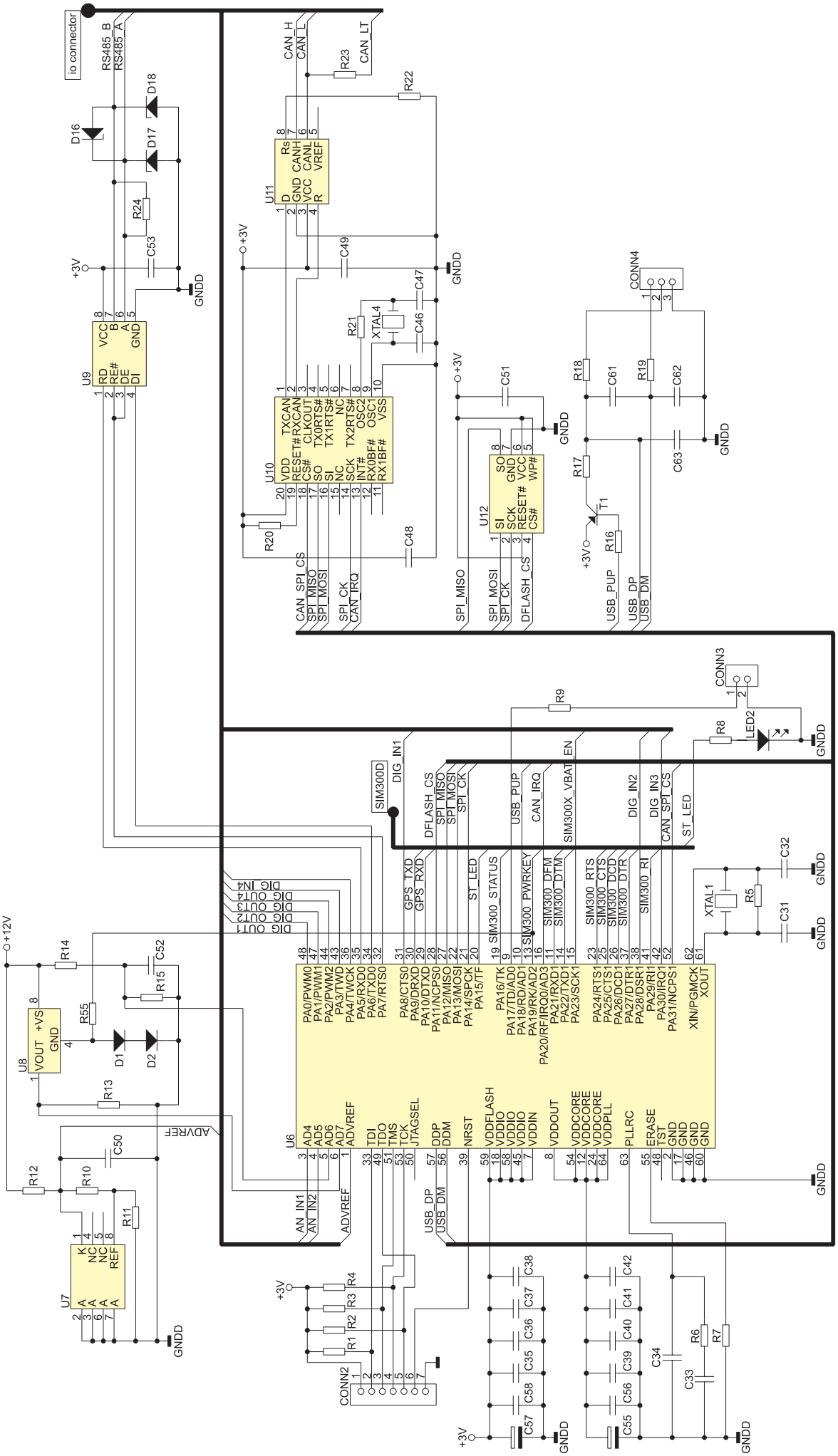
Sygnały interfejsu RS485 po konwersji do postaci różnicowej w sterowniku linii U9 wyprowadzone są na piny 9 i 23 złącza CONN1. Rezystor R24 spełnia rolę terminatora linii RS485, natomiast diody D16..D18 zabezpieczają układ U9 przed przepięciami.

Linie danych złącza USB – USB_DP oraz USB_DM, wyprowadzono na złącze CONN3 poprzez rezystory R37 i R38, które zapewniają dopasowanie impedancji interfejsu USB mikrokontrolera do impedancji szyny. Kondensatory C61...C63 tłumią składowe wysokoczęstotliwościowe na liniach USB D+, D–. Tranzystor T1 umożliwia sterowanie zasilaniem linii USB_DP. Sterownik interfejsu USB (po jego inicjalizacji) wymusza poziom niski na linii USB_PUP, rozpoczynając proces enumeracji USB. Stan linii zasilania USB (VBUS) nie jest monitorowany, w związku z czym urządzenie nie powinno być zasilane po podłączoniu do hosta, gdy ten jest nieaktywny. W przeciwnym wypadku może dojść do przepływu prądu w kierunku hosta poprzez rezystor R36 i nasycony tranzystor T1.

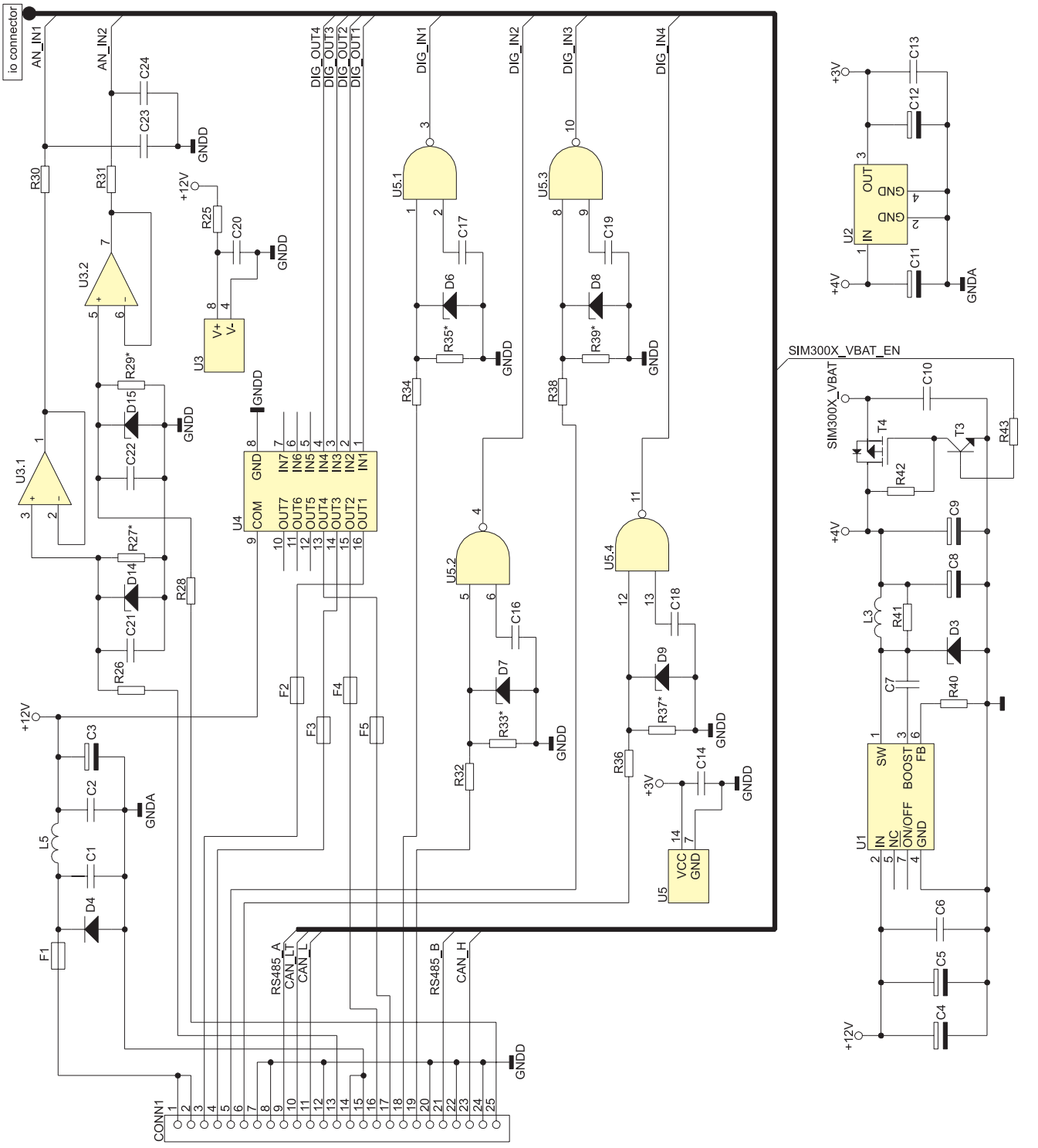
Napięcie odniesienia 2,56 V dla wbudowanego przetwornika A/C mikrokontrolera dostarcza układ U7 (TL431), a jego wartość jest określona przez dzielnik napięcia R10, R12. Do dwóch z czterech używanych wejść przetwornika A/C doprowadzone jest napięcie z pinów 13 i 26 złącza CONN1, poprzez dzielniki napięciowe i buforujące wzmacniacze operacyjne U3.1, U3.2. Wejście AD6 służy do monitorowania napięcia zasilania urządzenia, natomiast na wejścia AD2, AD7 jest podawany sygnał z czujnika temperatury U8 umieszczonego na płytce urządzenia. Dzięki diodom D1, D2 potencjał wyprowadzenia VOUT U8 może być niższy od potencjału wyprowadzenia GND, dlatego jest możliwy pomiar temperatury poniżej 0°C. Taką konfigurację zaczerpnięto z noty katalogowej układu LM35.

Kontrolera interfejsu CAN nie ma wśród urządzeń peryferyjnych mikrokontrolera AT91SAM7S256, w związku z czym zastoso-

R E K L A M A



Rysunek 2. Schemat ideowy głównej części urządzenia telemetrycznego



Rysunek 3. Schemat ideowy zasilacza

wano popularny układ MCP2515 (U10), komunikujący się z mikrokontrolerem poprzez interfejs SPI. Interfejs ten jest używany także do komunikacji z układem pamięci Flash U12. Wybór odpowiedniego układu podrzędnego SPI następuje poprzez zmianę stanu portów PA11, PA13 mikrokontrolera.

Linie sygnałowe interfejsu JTAG wyprowadzono na 7-pinowe złącze szpilkowe. W związku z tym jest konieczne użycie odpowiedniej przejściówki na standardowe złącze 10- lub 20-pinowe.

Wykonanie zasilacza modułu SIM300D wymaga szczególnej staranności, ze względu na stosunkowo duże, dochodzące do 2 A, wartości prądu w impulsie pobieranego w krótkich momentach nadawania sygnału GSM. Napięcie na pinach VBAT nie powinno wówczas spadać poniżej 3,4 V.

Schemat ideowy zasilacza impulsowego dostarczającego napięcie zasilania dla modemu GSM zaprezentowano na **rysunku 3**. Zbudowano go w oparciu o układ LM2676 (IC5), który pracuje w standardowej konfiguracji obniżającej napięcie (step-down) z częstotliwością kluczowania 260 kHz. Parametry kondensatorów C4, C5, C8, C9 są dość krytyczne dla prawidłowej pracy układu. Kondensatory powinny

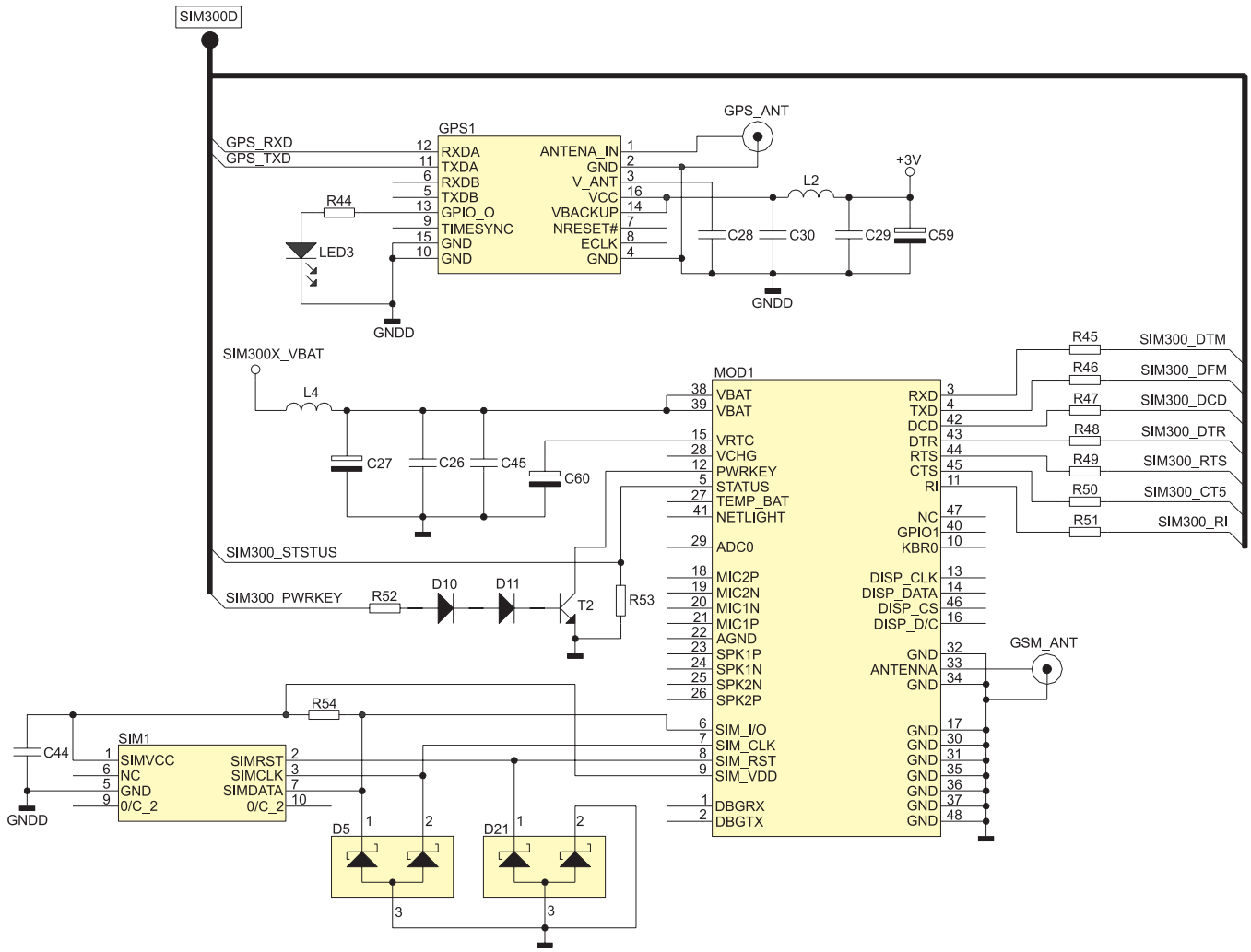
Rysunek 4. Oscylogram napięcia na kondensatorze C27 w trakcie nadawania sygnału GSM przez moduł SIM300D (składowa zmienna)

charakteryzować się niskim ESR ze względu na występujące znaczne prądy impulsowe.

Napięcie 9...18 V, obniżone do poziomu 4 V, zasila moduł GSM poprzez układ kluczujący z tranzystorami T3 i T4, sterowany z wyjścia PA23 portu mikrokontrolera (SIM300X_PWR). Ustawienie poziomu wysokiego na linii SIM300X_PWR powoduje nasycenie tranzystora T3 oraz włączenie tranzystora T4 (P-MOS), przez co jest załączane napięcie zasilania SIM300X_VBAT. Możliwość sterowania zasilaniem modułu SIM300D przydaje się w przypadku konieczności wykonania jego pełnego restartu.

Na **rysunku 4** zamieszczono oscylogram napięcia zasilania modemu GSM zdjęty w trakcie transmisji GPRS. Wyraźnie widoczne są zakłócenia spowodowane impulsowym obciążeniem, przy czym napięcie spada o więcej niż 100 mV. Częstotliwość zakłóceń jest równa odwrotności okresu ramki czasowej GSM (4,615 ms), natomiast czas trwania impulsu wynosi 1/8×4,615 ms=0,577 ms, czyli odpowiada jednej szczelinie czasowej GSM. Stabilizator liniowy LDO – MCP1826S (U2) – dostarcza napięcie o wartości 3,0 V, zasilające pozostałe układy.

Połączenie modułu SIM300D z mikrokontrolerem przedstawiono na **rysunku 5**. Komunikacja odbywa się za pomocą komend AT poprzez USART1. Parametry elektryczne portów we/wy modułu pozwalają na bezpośrednie ich dołączenie do portów mikrokontrolera. Dopuszczalne napięcie na wejściach wynosi ok. 3,23 V. Znaczenie poszczególnych sygnałów RS232 po stronie modułu w trybie GPRS jest następujące: DTR – przełączanie modemu w tryb AT lub tryb danych, DCD – sygnalizacja stanu połączenia TCP, RI – sygnalizacja przychodzącego połączenia głosowego lub wiadomości SMS. Sygnały RTS, CTS zapewniają kontrolę przepływu danych na linii RXD. Rezystory R45...



Rysunek 5. Sposób połączenia modułu FGPMMOSL3 z mikrokontrolerem

R51 ograniczają prąd wypływający z wyjść mikrokontrolera w stanach nieustalonych, gdy jest wyłączone zasilanie modemu. Dynamiczne konfigurowanie trybu pracy portów mikrokontrolera w trakcie wyłączania zasilania modemu pozwala uniknąć takich sytuacji. Podstawka dla karty SIM jest dołączona do pinów 6...9 modułu. Pojemność kondensatora C44, blokującego zasilanie karty, nie powinna być zwiększana, aby nie zakłócić pracy układu detekcji obecności karty. Elementy D5, D21 zapewniają ochronę modułu przed wyładowaniami elektrostatycznymi. Napięcie zasilania 4.0 V, dodatkowo odfiltrowane przez elementy C10, L4, C27, C26, jest podawane na piny VBAT. Kondensator C45 eliminuje ewentualne zakłócenia w.cz.

W celu uruchomienia modułu, przy załączonym napięciu VBAT, wejście PWRKEY jest zwierane do masy (poprzez nasycony tranzystor T2) na czas dłuższy niż 2 s. W momencie uruchomienia na wyprowadzeniu STATUS występuje napięcie ok. 2,9 V, a następnie mikrokontroler zeruje linię SIM300_PWRKEY zatykając tranzystor T2. Wyłączenie modułu odbywa się poprzez zwarcie PWRKEY do masy na czas ok. 0,7 s (0,5...1 s).

Program mikrokontrolera ustawia wszystkie wyjścia łączące go z modemem GSM w stan wysokiej impedancji (tryb wejścia, z wyłączonym podciąganiem do napięcia zasilania) przed wyłączeniem zasilania modułu. Zapobiega to przepływaniu zbyt dużych prądów z wyjść mikrokontrolera w kierunku modułu SIM300D.

Moduł SIM300D ma wbudowany zegar czasu rzeczywistego. Podczas pracy modułu kondensator C60 ładowany jest niewielkim prądem do ok. 1,8 V, natomiast po odłączeniu zasilania ładunek zgromadzony w C60 podtrzymuje RTC. Ze względu na niewielki wymagany prąd podtrzymania (rzędu 5 µA) przy pojemności 0,22 F, maksymalny czas podtrzymania wynosi ok. 7 godzin.

Wejścia oraz wyjścia audio nie są używane i zgodnie z zaleceniem producenta modułu pozostają niepodłączone. Nieco więcej uwagi podczas projektowania płytki drukowanej wymaga linia łącząca wejście antenowe z zewnętrznym gniazdem anteny (SMA). Ścieżka prowadząca do pinu 33 SIM300D powinna być jak najkrótsza. Należy unikać zagięć, a jeżeli są konieczne, powinny być owalne – nie należy ich prowadzić pod kątem ostrym. Impedancja falowa linii powinna wynosić 50 Ω. W celu wyznaczenia szerokości ścieżki oraz odstępu do obszaru masy można użyć programu AppCAD, za pomocą którego można przy zadanych wymiarach oraz rodzaju laminatu obliczyć wartość impedancji falowej linii paskowej.

Jako odbiornik GPS zastosowano moduł FGPMMOSL3 z chipsetem MT3318 firmy MediaTek. Charakteryzuje się on dużą czułością (–156 dBm), bardzo małymi wymiarami oraz stosunkowo niską ceną. Jest to odbiornik 51-kanałowy, o dokładności horyzontalnej 3 m (50%) i maksymalnej częstotliwości uaktualniania 5 Hz. W czasie testów sygnał z satelitów odbierany był nawet wewnątrz budynków, jednak z większym błędem wyznaczenia pozycji spowodowanym najpewniej wielodrogowością sygnału oraz mniej korzystnym zestawem widocznych satelitów.

Odbiornik GPS komunikuje się z mikrokontrolerem poprzez interfejs szeregowy. Używany jest popularny protokół NMEA 0183. Dane nawigacyjne przesyłane są przez odbiornik w postaci komend GGA, GSV, GSA, RMC i VTG. Dodatkowo odbiornik obsługuje komendy specjalne, rozpoczynające się od $PMTK, które między innymi pozwalają na skonfigurowanie układu odniesienia ($PMTK330), zestawu generowanych komunikatów NME, ich częstotliwości ($PMTK314) oraz prędkości transmisji interfejsu szeregowego ($PMTK251). Wadą modułu może być brak wsparcia dla protokółu binarnego, który możemy znaleźć w odbiornikach takich firm jak Trimble lub z chipsetem SIRF STAR. Protokół binarny pozwala uniknąć kłopotliwego parsowania danych tekstowych w celu przetworzenia ich na postać binarną dla dalszej transmisji. Dodatkowo, udostępnia bardziej szczegółowo dane nawigacyjne, np. trójwymiarowy wektor prędkości, nie tylko w płaszczyźnie horyzontalnej (jak to ma miejsce w przypadku FGPMMOSL3).

Sposób dołączenia modułu FGPMMOSL3 do mikrokontrolera pokazano na **rysunku 6**. Odbiornik współpracuje z zewnętrzna anteną aktywną GPS o impedancji 50 Ω i zalecanym napięciu zasilania 2,8 V. Antena jest dołączona do pinów 1 i 2 modułu za pośrednictwem złącza SMA. Jak wspomniano ścieżka doprowadzająca sygnał z anteny do modułu powinna być jak najkrótsza, a impedancja falowa powinna wynosić 50 Ω.

Wykaz elementów

Rezystory: (SMD, 1206)
R1...R4: 15 kΩ
R10: 2,7 kΩ
R11: 100 kΩ
R12: 4,3 kΩ
R13: 18 kΩ
R14: 27 kΩ
R15, R30, R31: 3,3 kΩ
R16: 22 kΩ
R17, R6: 1,5 kΩ
R18, R19: 27 Ω
R20, R52...R54: 10 kΩ
R21, R25: 100 Ω
R22: 0 Ω
R23, R24: 120 Ω
R26, R28, R44...R51, R8, R9: 1 kΩ
R27*, R29*, R33*, R35*, R37*, R39*: nie montować
R32, R34, R36, R38: 4,7 kΩ
R40: 1,2 kΩ
R41: 2,7 kΩ
R42: 100 kΩ
R43: 22 kΩ
R5, R7: 1 MΩ
R55: 2 kΩ

Kondensatory:
C1, C2, C23, C24, C28, C33, C52, C7: 10 nF (SMD, 1206)
C10, C13, C14, C29, C30, C35...C42, C48, C49, C51, C53, C58, C6: 100 nF (SMD, 1206)
C11: 10 µF/10 V
C12, C27: 100 µF/6,3 V
C16...C19, C44: 220 nF (SMD, 1206)
C20, C21, C22, C26: 1 µF
C3: 470 µF/35 V
C31, C32, C46, C47: 22 pF (SMD, 1206)
C34, C55: 1 nF (SMD, 1206)
C4, C5: 10 µ/35 V
C45: 10 pF (SMD, 1206)
C50, C56: 2,2 µF/6,3 V
C57: 4,7 µF
C59: 47 µF/6,3 V
C60: 0,25 F
C61: 33 pF (SMD, 0603)
C62, C63: 15 pF (SMD, 0603)
C8, C9: 100 µF/10 V

Półprzewodniki:
D1, D2, D10, D11: 1N4148 (MINIMELF)
D14, D15: dioda Zenera 3,9 V (MINIMELF)
D16...D18: P6SMBJ33CA (39 V)
D21, D5: PESD6V1S2BT
D3: MBRS320T3
D4: 1N4002
D6...D9: diody Zenera 3 V
T1: BC857B
T2, T3: BC847A
T4: IRF7204
U1: LM2676-ADJ
U2: MCP2515-I/ST
U3: LM358D
U4: ULN2003D
U5: CD4093BCM
U6: AT91SAM7S256-LQFP
U7: TL431AID
U8: LM35DM
U9: MAX3430
U10: MCP1826S
U11: SN65HVD230D
U12: AT45DB011B
LED2, LED3: diody LED (SMD, 1206)

Inne:
L2: dławik ferromagnetyczny BLM31AF700SN1L 1206 (TME)
L3: dławik 22 µH YAGEO
L4: dławik ferromagnetyczny BLM31PG500SN1L (TME)
L5: dławik 22 µH
CONN1: DB25F
CONN2: igłowe 7-pin 2,54 mm
CONN3: igłowe 2-pin 2,54 mm
SIM1: podstawka na kartę SIM ATTEND 115A-ADA0-R01
Gniazdo SMA żeńskie kątowe do druku: 2 szt.
MOD1: moduł GSM SIM300DZ
GPS1: moduł odbiornika GPS GPS-FGPMMOSL3
F1...F5: bezpiecznik polimerowy 0,5 A
XTAL1: kwarc 18,432 MHz (HC49S)
XTAL1: kwarc 12 MHz (HC49S)
Antena GSM kątowa na złącze SMA żeńskie
Antena GPS zewnętrzna FGPANE1A5 z kablem zakończonym ze złączem męskim SMA

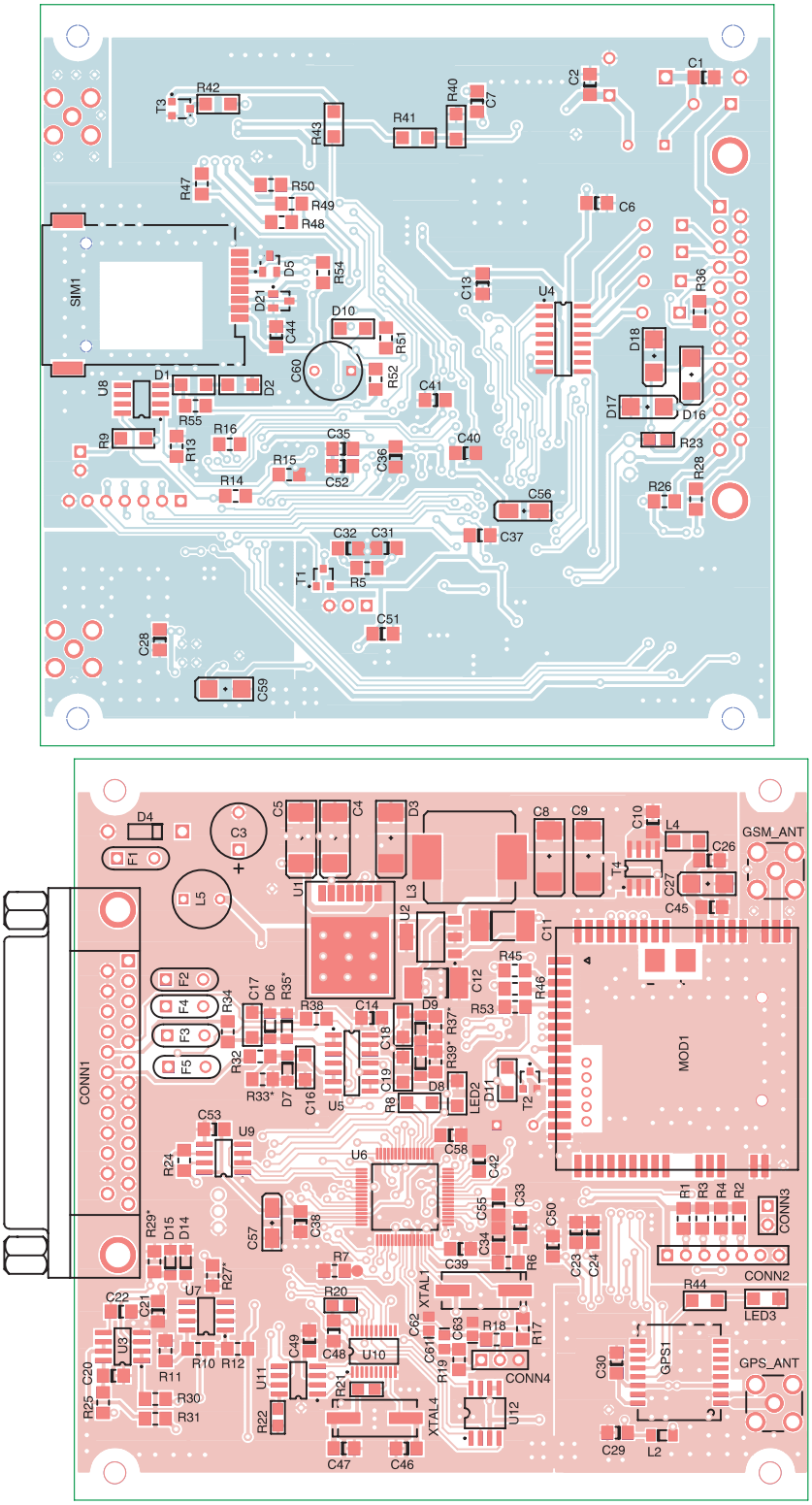
Te wymagania są podobne jak dla obwodu antenowego modułu GSM. Napięcie zasilania, filtrowane poprzez elementy C29, C59, L2, C30, zasila obwody odbiornika GPS poprzez pin 16. Ponadto, jest odsprzężone kondensatorem C28 i doprowadzone do pinu zasilania anteny V_ANT. Napięcie to jest wewnątrz modułu podawane poprzez dławik na pin 1 (ANTENNA_IN).

Prąd zasilania anteny aktywnej jest ograniczany wewnątrz modułu do 30 mA. Dioda LED3 sygnalizuje status odbiornika. Jej migotanie oznacza brak lub zbyt słaby sygnał (np. odłączenie anteny). W momencie rozpoczęcia śledzenia sygnału wystarczającego do wyznaczenia pozycji zostaje zgaszona.

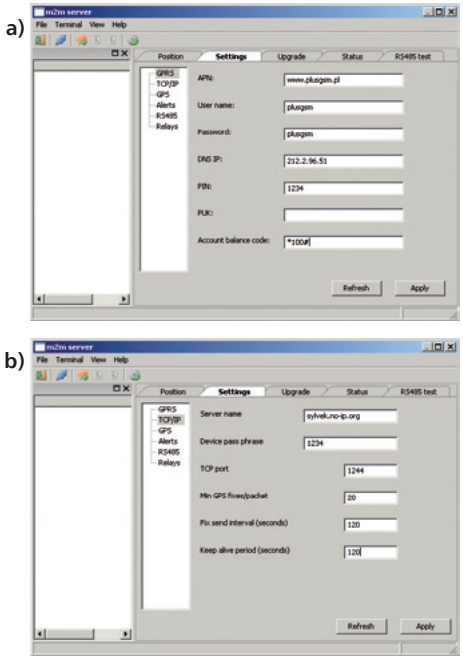
Montaż i uruchomienie

Schemat montażowy urządzenia zamieszczono na **rysunku 7**. Montaż najlepiej rozpocząć od wlutowania najmniejszych rezystorów i kondensatorów po obu stronach płytki. Następnie montujemy wszystkie elementy głównego zasilacza – układy U1, U2 oraz ich otoczenie, przy czym bazę tranzystora T3 zwieramy tymczasowo do masy. Po upewnieniu się, że napięcia zasilania 3,0 V oraz 4,0 V są prawidłowe, możemy przystąpić do wlutowania odbiornika GPS oraz elementów L2, R44, LED3. Po załączeniu zasilania LED3 powinna migotać z częstotliwością 1 Hz. Następnie montujemy pozostałe elementy (jak na rysunku 2). Na tym etapie można pominąć układy U9, U10 i U11. Przylutowanie układu U6 w obudowie LQFP64 wymaga nieco cierpliwości, jednak wspomagając się odpowiednią lupą da się to wykonać bez większych trudności. Pomocne może okazać się nałożenie wstępnie na pola kontaktowe układu U6 niewielkiej ilości cyny z topnikiem.

Za pomocą programatora JTAG dołączonego do złącza CONN2, programujemy pamięć programu mikrokontrolera od adresu 0 kodem wynikowym z pliku *bootloader.bin*, natomiast zawartość pliku *firmware.bin* powinna być zapisana od adresu 0x3000. W przypadku problemów z wykryciem mikrokontrolera przez programator należy sprawdzić poprawność napięć VDDIO, VDDCORE i VDDPLL oraz stan połączeń elementów XTAL1, C31, C32. Po prawidłowym zaprogramowaniu mikrokontrolera dioda LED2 powinna rozbłysnąć na chwilę po załączeniu zasilania. Po dołączeniu konsoli szeregowej (np. *TeraTerm* lub *Minicom*) do wyjścia *debug* portu dostępnego na złączu CONN3, powinny być widoczne komunikaty sygnalizujące start urządzenia. Następnie wlutowujemy wszystkie pozostałe elementy i usuwamy wcześniej wykonane połączenie bazy T3 z masą.



Rysunek 6. Schemat montażowy urządzenia telemetrycznego

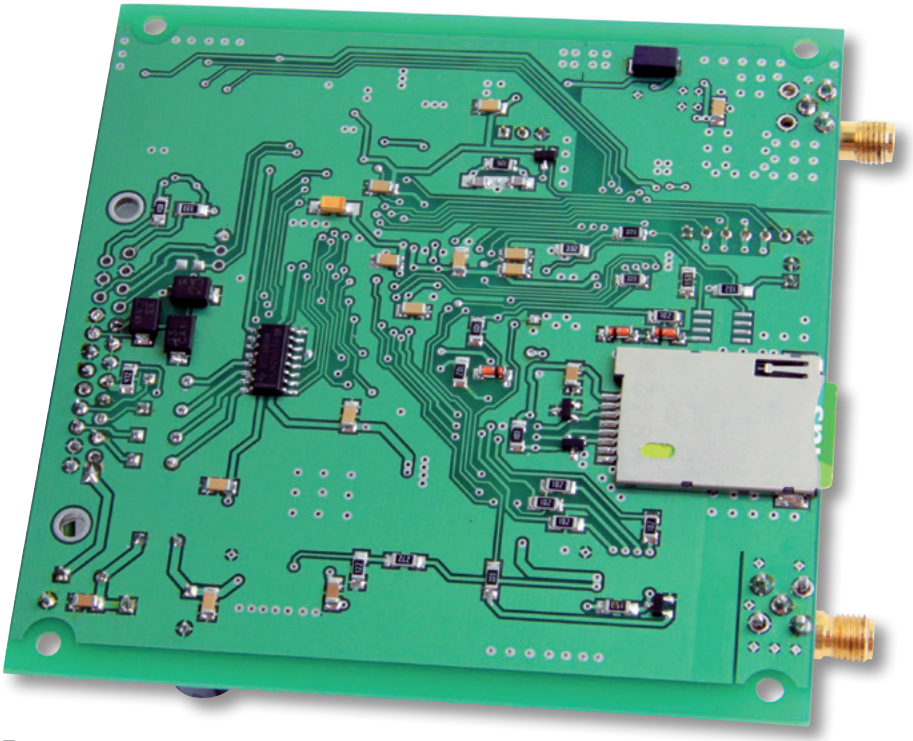


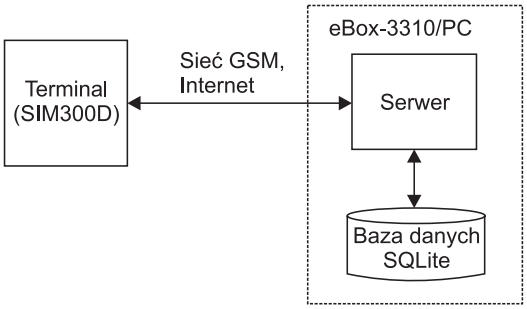
Rysunek 7. Okno programu konfiguracyjnego z przykładowymi ustawieniami dla sieci Plus (a) oraz parametrami TCP/IP (b)

Tabela 1. Sposób dołączenia kabla USB do złącza CONN4

Pin CONN4	Kolor kabla	Opis
1	biały	D- (USB_DM)
2	zielony	D+ (USB_DP)
3	czarny	GND

Do złącza CONN4, zgodnie z **tabelą 1**, należy przylutować kabel USB zakończony wtykiem USB A. W złącze znajdujące się pod modułem MOD1 wsuwamy kartę SIM z wyłączonym żądaniem kodu PIN. Dla uzyskania połączenia GPRS wymagane jest skonfigurowanie profilu GPRS dla danego operatora. Ustawienia te dla Plus GSM (karta prepaid Simdata) pokazano na **rysunku 8**. Konfigurowanie urządzenia jest możliwe po dołączeniu urządzenia do komputera kablem USB i zaznaczeniu w menu File opcji *USB connection mode*. W zakładce *Settings – TCP/IP* należy ponadto wypełnić pole *Server name*, w którym powinna się znaleźć nazwa lub jego adres IP a także *TCP port*, na którym nasłuchuje serwer. W programie konfigurującym jest wbudowany serwer, który można uaktywnić w menu *File – Settings* zaznaczając *Enable internal server*.

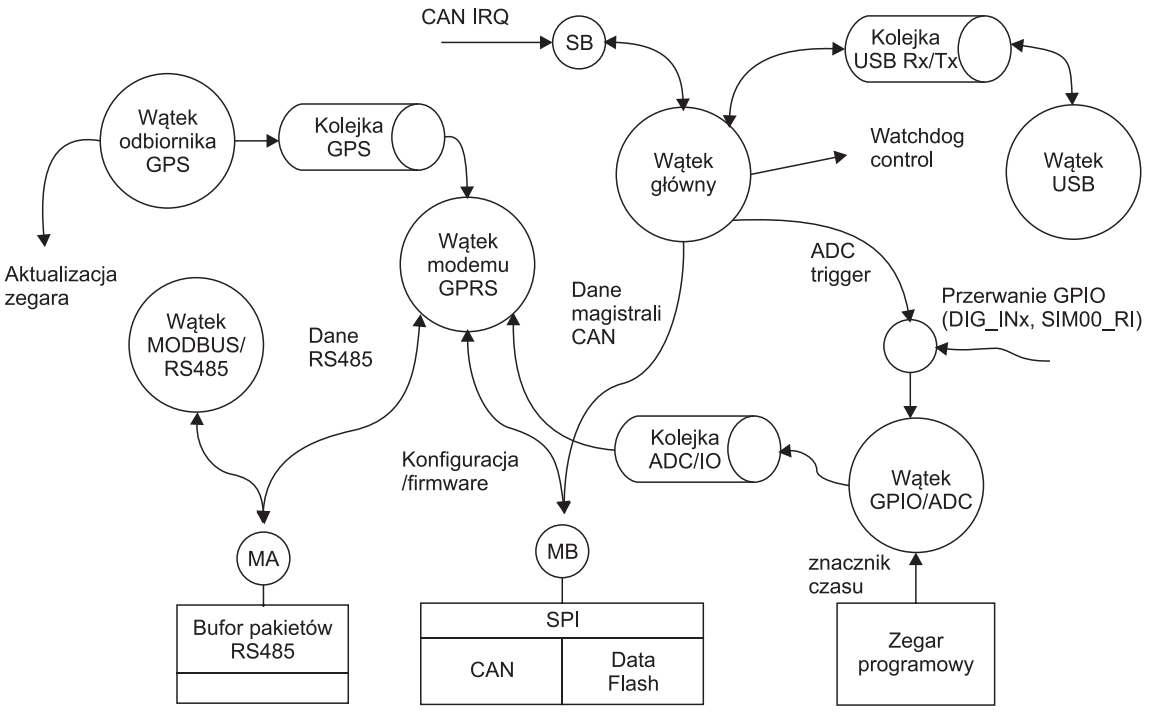




Rysunek 8. Przykład podstawowej konfiguracji klient/serwer

Program serwera współpracującego z urządzeniem

Przykładową konfigurację oprogramowania współpracującego z urządzeniem przedstawia **rysunek 9**. W przypadku korzystania z publicznego APN, komputer pełniący rolę serwera musi być widoczny w Internecie, co wiąże się z koniecznością posiadania publicznego adresu IP. W przypadku, gdy nie jest to adres statyczny, możemy skorzystać z usługi DDNS (np. *www.no-ip.com* lub *www.dyndns.org*) i odwoływać się do serwera korzystając z jego nazwy domenowej, gdyż moduł SIM300D wspiera rozwiązywanie nazw poprzez DNS. Dodatkowo, korzystając z translacji adresów (NAT) w lokalnej sieci z routerem dostępowym, należy zatroszczyć się o przekierowanie odpowiedniego portu TCP na komputer, na którym będziemy uruchamiać serwer. Ze względu na stosunkowo niewielkie wymagania sprzętowe serwera w minimalnej konfiguracji, może nim być np. router Linksys z serii WRT54 z dystrybucją Linuksa *OpenWrt*. Program konfiguracyjny *m2m_server.exe* pozwala na wstępne skonfigurowanie urządzenie nie tylko za pomocą USB, ale również zdalnie poprzez GPRS. Zdalnie nie jest możliwa tylko zmiana nazwy serwera oraz numeru portu TCP, gdyż mogło-



Rysunek 9. Uproszczony model wątków

Listing 1. Modyfikacja skryptu w celu alokacji programu w pamięci RAM

```
SECTIONS
{
  .text:    ...
  .rodata:  ...
  .data:    /* dane inicjalizowane */
  {
            _data = . ;
            *(.data)
            . = ALIGN(4);
  /* kod alokowany w RAM */
            *(.ramfunc)
  } >DATA
  . = ALIGN(4);
  ...
}
```

by to spowodować zerwanie połączenia bez możliwości jego przywrócenia.

Oprogramowanie mikrokontrolera

Jako *toolchain* wykorzystano pakiet *arm-gcc* (Yagarto) oraz środowisko *Eclipse CDT* z wtyczką *Zylin*, umożliwiającą debugowanie programu z poziomu Eclipse poprzez serwer JTAG OpenOCD. Dokładny opis środowiska oraz wszystkie wymagane pliki do pobrania można znaleźć na stronie *www.yagarto.de*.

Urządzenie pracuje pod kontrolą FreeRTOS (do ściągnięcia ze strony *freertos.org*). Jest to oprogramowanie otwarte, oficjalnie wspierające 23 architektury, a jego główny kod w języku C jest zawarty w zaledwie trzech plikach. Użycie systemu operacyjnego znacznie uprościło strukturę programu, nieznacznie tylko uszczuplając zasoby pamięci RAM czy Flash. Uproszczony model wątków pokazano na **rysunku 10**. Poszczególne wątki obsługują odpowiadające im urządzenia. Komunikacja pomiędzy wątkami odbywa się poprzez kolejki i semafory S_A, S_B. Muteksy M_A, M_B zapewniają synchronizację dostępu odpowiednio do interfejsu SPI i bufora pakietów RS485. Wątek odbiornika GPS odczytuje dane z kolejki odbiorczej interfejsu szeregowego DBGU, dekoduje komunikaty NMEA, formuje rekordy zawierające współrzędne geograficzne, prędkość oraz czas wyznaczenia pozycji i zachowuje tak przygotowane rekordy w kolejce GPS. Ponadto, co minutę dokonuje aktualizacji czasu zegara programowego, zrealizowanego w przerwaniach od licznika T0 (przerwanie schedulera).

Stan kolejki GPS jest cyklicznie sprawdzany przez wątek modemu GPRS i gdy liczba zapisanych w niej rekordów przekroczy ustalony podczas konfigurowania próg, to są one formowane w pakiet i przesyłane do serwera. Port PIOA jest skonfigurowany tak, aby zmiana stanu na którejkolwiek z linii RI, DIG_IN1..DIG_IN4 spowodowała wygenerowanie przerwania.

W obsłudze przerwania jest zapisywany stan portu i „podnoszony" semafor S_A. Pobudzony wątek GPIO/ADC tworzy rekord alarmu zawierający stan wejść cyfrowych, wartości napięć na wejściach analogowych oraz znacznik czasu i dodaje rekord do kolejki ADC/IO. Dodanie rekordu nastąpi tylko wtedy, gdy dany pin nie jest maskowany w globalnej konfiguracji urządzenia. Identyczny rekord jest tworzony, gdy wątek główny cyklicznie „podnosi" semafor S_A w celu wyzwolenia procedury pomiaru napięcia na wejściach analogowych, zgodnie z ustalonym czasem próbkowania.

Listing 2. Sekwencja komend AT inicjalizująca pracę modemu SIM300D (pogrubiono komendy przysłane do modemu)

```
01      AT
02      OK
03      ATE1
04      OK
05      AT+GSN
06      356895030323***
07      OK
09      AT+CPIN?
10      SIM PIN
11      OK
12      AT+CPIN="2528"
13      OK
14      AT+CPIN?
15      +CPIN:
16      READY
17      AT+CMGF=1
18      OK
19      AT+CIPFLP=0
20      OK
21      AT+CGATT=1
22      OK
23      AT+CIPMODE=1
24      OK
25      AT+CIPCCFG=3,2,1024,0
26      OK
27      AT+CDNSORIP=1
28      OK
29      AT&D1
30      OK
31      AT&C1
32      OK
33      AT+CGDCONT=1,"IP","www plusgsm pl"
34      OK
35      AT+CSTT="www plusgsm pl","plusgsm","plusgsm"
36      OK
37      AT+CDNSCFG="212.2.96.51"
38      OK
39      AT+CIICR
40      OK
41      AT+CIFSR
42      77.113.175.70
43      AT+CNUM
44      +CNUM: „NUMER WLASNY","691386xxx",129,7,4
45      OK
46      ATD*100#
47      +CUSD: 0,"Aktualny stan konta:  19.85  PLN.
48      Konto wazne do dnia  31/01/2010",15
49      OK
50      AT+CIPSTART="TCP","sylvek.no-ip.org","1244"
51      OK
52      CONNECT
```

Zdalna aktualizacja oprogramowania urządzenia

W celu usprawnienia procedury zdalnej aktualizacji zawartości pamięci Flash mikrokontrolera, z oprogramowania urządzenia został wydzielony bootloader, który zawiera procedury weryfikujące oraz przeprogramowujące pamięć Flash. Jest on zapisany w początkowym obszarze adresowym pamięci Flash o wielkości 12 kB (0x000...0x2FF). Pozostałe 244 kB jest do dyspozycji głównej aplikacji. Aktualizacja oprogramowania polega na przepisaniu danych z zewnętrznej pamięci Flash do obszaru pamięci Flash mikrokontrolera powyżej adresu 0x0300. Na żądanie aplikacji bootloader dokonuje aktualizacji poprzez sprawdzenie odpowiednich flag w pamięci DFLASH. Warunkiem rozpoczęcia przeprogramowania jest pomyślny wynik porównania sumy kontrolnej pliku z wartością odebraną z serwera.

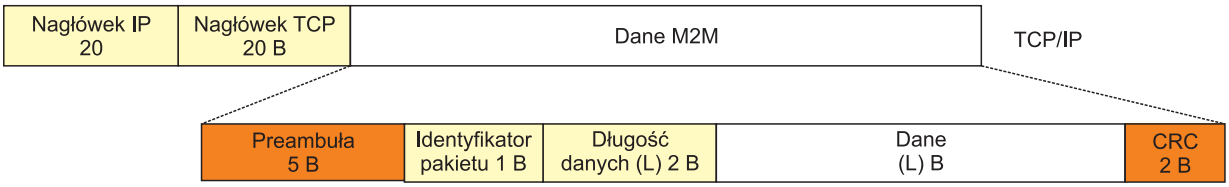
Pamięć Flash mikrokontrolera AT91SAM7S256 jest typu *single plane*, w związku z czym właściwy kod programujący jej zawartość musi być uruchamiany z pamięci SRAM. Alokację kodu w RAM uzyskano przypisując funkcję do odpowiedniej sekcji, tzn. stosując mechanizm *__attribute__* dostępny w pakiecie GCC. Na przykład chcąc umieścić ciało funkcji w sekcji o nazwie *.ramfunc*, wystarczy jej deklarację poprzedzić zwrotem: ***__attribute__*** *((long_call, section („.ramfunc")))*. Dla ułatwienia możemy zdefiniować makrodefinicję: ***#define*** *RAMFUNC* ***__attribute__*** *((long_call, section („.ramfunc")))*. W takim przypadku, aby funkcja *foo()* znalazła się w sekcji *.ramfunc*, wystarczy zapis *RAMFUNC void foo();*. Następnie modyfikując skrypt linkera, umieszczamy sekcję *.ramfunc* w obszarze danych zainicjowanych (sekcja *.data*). Modyfikacja polega na dodaniu **(.ramfunc)* w sekcji *.data*, np. jak we fragmencie skryptu zamieszczonym na **listingu 1** (definicje sekcji *.text* oraz *.rodata* zostały pominięte). W tak zbudowanym programie, kod startowy inicjalizacji zmiennych, dodawany domyślnie przez kompilator, dokonuje automatycznie skopiowania ciał funkcji z pamięci Flash do SRAM. Wadą takiego rozwiązania mogłaby

Listing 3. Struktury służące do wymiany danych z modułami GSM i GPS

```
/* Struktura danych reprezentująca odbierany pakiet M2M */
typedef struct {
        unsigned short type;
        unsigned short len;
        unsigned short bytes_rxed;
        unsigned short bytes_to_read;
        short status;
        short pdata_maxlen;
        void *pdata;
} xPacket_t;

/* Definicja rekordu GPS */
typedef struct {
        unsigned int ts; /* unix timestamp */
        float     fLat;    /* latitude in degrees */
        float     fLon;    /* longitude in degrees */
        float     fAlt;    /* altitude in m */
        float     fVel;    /* peak velocity */
        float  fDistance;  /* distance */
} xFixRecord_t;
```

rowane do serwera. Dane odebrane z serwera będą pojawiać się na linii TxD modemu. Komenda AT+CIPCCFG powoduje określenie parametrów trybu transparentnego – odpowiednio liczbę retransmisji pakietu IP, czas oczekiwania (2×200 ms) oraz liczbę bajtów odebranych z portu szeregowego przed wysłaniem pakietu (1024). W wierszu 21 następuje przyłączenie terminala do sieci GPRS (*GPRS attach*). Komendą AT+CDNSORIP wskazujemy, że zestawienie połączenia będzie się odbywać z użyciem nazwy domenowej, a nie adresu IP. Wymaga to podania adresu IP serwera DNS, co jest wykonywane w wierszu 37. W wierszach 33...36 i 39 następuje kon-



Rysunek 10. Podstawowa ramka protokołu M2M

być utrata części pamięci RAM o rozmiarze równym rozmiarowi sekcji *.ramfunc*. Jednak w tym przypadku nie ma to większego znaczenia, gdyż bootloader działa tylko w momencie startu urządzenia (w głównej aplikacji nie ma potrzeby stosowania funkcji alokowanych w RAM). Pełny skrypt linkera zawiera dołączony plik *AT91SAM7S256-ROM.ld*.

Sterowanie modułem GSM

Na **listingu 2** zamieszczono przykładową sekwencję komend AT, którą mikrokontroler inicjalizuje moduł SIM300D przy starcie urządzenia. Moduł SIM300D jest domyślnie skonfigurowany w trybie automatycznej detekcji prędkości bitowej interfejsu szeregowego (autobauding). Ustalenie określonej prędkości umożliwia komenda AT+IPR. Warunkiem prawidłowej detekcji prędkości transmisji przez modem jest wysłanie przez DTE (mikrokontroler), po ok. 5 s od załączenia zasilania, ciągu znaków „AT” lub „at”. Potwierdzenie przyjęcia większości komend AT następuje jeszcze przed realizacją komendy poprzez zwrócenie przez moduł ciągu znaków „\n\rOK\n\r”.

Polecenie w wierszu 3 włącza echo odbieranych przez moduł znaków. W wierszach 5...7 wykonywane jest odczytanie numeru seryjnego, natomiast w wierszach 9...16 wprowadzany jest kod PIN po uprzednim sprawdzeniu, czy jest wymagany. Komenda AT+CMGF ustala tryb SMS na tekstowy. Następnie polecenie AT+CIPFLP konfiguruje niezmienny port TCP przy każdym połączeniu. Inkrementacja portu przy kolejnych połączeniach pozwala na przyspieszenie procesu zestawiania połączenia, jednak ta możliwość funkcjonalna jest bezużyteczna, ponieważ serwer nasłuchuje zawsze na tym samym porcie. Polecenie AT+CIPMODE ustala tryb transmisji na „*transparent transfer mode*”, tzn. po zestawieniu połączenia TCP (co nastąpi w wierszu 50) dane wysyłane do portu szeregowego nie będą traktowane jak komendy AT, a jako strumień binarny i zostaną przekierowane do modułu GSM. Dane odebrane z modułu GSM będą pojawiać się na linii RxD mikrokontrolera. Komenda AT+CIPCCFG powoduje określenie parametrów trybu transparentnego – odpowiednio liczbę retransmisji pakietu IP, czas oczekiwania (2×200 ms) oraz liczbę bajtów odebranych z portu szeregowego przed wysłaniem pakietu (1024). W wierszu 21 następuje przyłączenie terminala do sieci GPRS (*GPRS attach*). Komendą AT+CDNSORIP wskazujemy, że zestawienie połączenia będzie się odbywać z użyciem nazwy domenowej, a nie adresu IP. Wymaga to podania adresu IP serwera DNS, co jest wykonywane w wierszu 37. W wierszach 41...45 mają charakter czysto informacyjny. Komenda AT+CIFSR zwraca lokalny adres IP modemu, natomiast AT+CNUM numer telefonu przypisany do modemu. ATD*100# wykonuje połączenie na numer specjalny *100#, po czym w wierszach 47 i 48 pojawia się odpowiedź sieci. W ostatnim kroku jest wydawane polecenie AT+CIPSTART ze wskazaniem protokołu (TCP), nazwy zdalnego komputera (*sylvek.no-ip.org*) oraz portu TCP, na którym nasłuchuje serwer (1244). Jeżeli serwer jest uruchomiony, otrzymujemy odpowiedź CONNECT. Od tego momentu wszelkie dane pojawiające się na liniach danych interfejsu szeregowego są traktowane jako *payload TCP*, a poziom linii DCD zmienia się na niski. Przejście do trybu komend następuje tylko po ustawieniu linii DTR. Początkowo na linii DTR powinien występować poziom wysoki i w momencie otrzymania odpowiedzi „CONNECT” powinna ona być wyzerowana.

Dekodowanie danych z odbiornika GPS

Sterownik portu szeregowego DBGU, do którego dołączony jest odbiornik GPS, alokuje podczas inicjalizacji bufory Rx i Tx odpowiednio dla bajtów odbieranych i wysyłanych. W obsłudze przerwania generowanego po odebraniu pojedynczego znaku odebrany bajt jest dodawany do kolejki Rx. Wątek odbiornika GPS (funkcja *vGpsRxTask()*, plik *gps_receiver.c*) wykonuje blokujący odczyt kolejki Rx. Po odczytaniu znaku jest wywoływana funkcja *nmea_input_proc()*, która wstępnie wydziela pojedynczy komunikat NMEA i zapisuje w strukturze typu *_NmeaRpt*. Wydzielony pakiet RMC jest z kolei przetwarzany przez funkcję *rpt_rmc()*, która formuje rekord *xFixRecord_t* (jak na **listingu 3**) i dodaje go do kolejki GPS (**rysunek 10**). Odbiornik GPS w czasie normalnej pracy generuje raporty NMEA co 1 sekundę, natomiast zapis do kolejki GPS może odbywać się z mniejszą częstotliwością, określoną w konfiguracji urządzenia. Dodatkowo, jest stosowany prosty algorytm filtracji, którego celem jest eliminacja redundantnych danych oraz zapobieżenie nadmiernej utracie informacji przy małych częstotliwościach zapisu pozycji.

Protokół klient-serwer

Komunikacja urządzenia z serwerem odbywa się przez dedykowany protokół M2M używający jako warstwy transportowej TCP/IP. Podstawowy format ramki protokołu przedstawiono na **rysunek 11**.

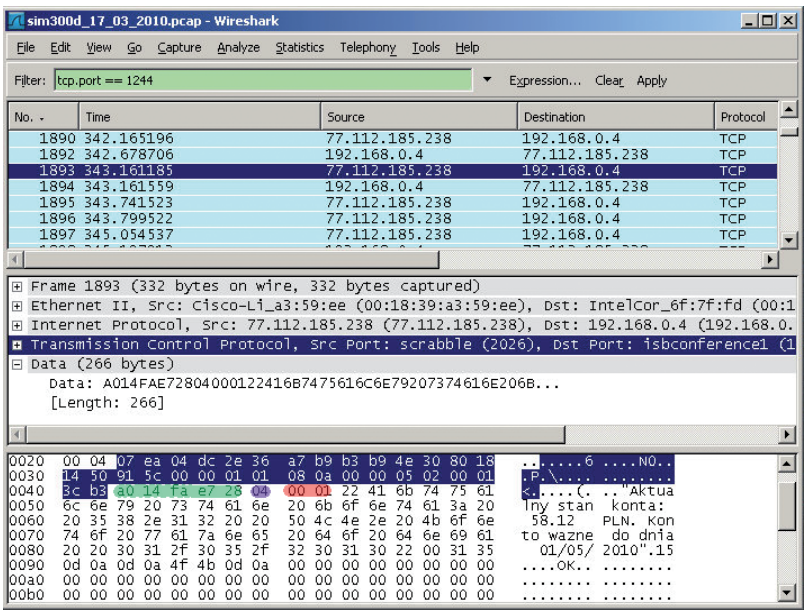
Każda z warstw w stosie protokołów dodaje własny nagłówek, w związku z tym rzeczywista wielkość pakietu danych będzie powiększona o długość nagłówka TCP/IP (najczęściej 40 bajtów). Zasady enkapsulacji pakietów IP na drodze od sieci Internet poprzez sieć GSM/GPRS do modemu nie będą tutaj opisywane.

Każda ramka danych M2M rozpoczyna się preambułą o długości 5 bajtów, pozwalającą na jednoznaczną detekcję początku pakietu po stronie odbiorcy. Wartość preambuły powinna być dobrana tak, aby wykluczyć lub maksymalnie obniżyć prawdopodobieństwo wystąpienia takiego ciągu bajtów w ramce danych konkretnego pakietu. Bajt następujący bezpośrednio po preambule stanowi 8-bitowy identyfikator pakietu, natomiast kolejne 2 bajty określają długość pola danych L. 16-bitowa suma kontrolna jest umieszczana z offsetem (8 + L) w stosunku do początku pakietu.

Dekodowanie pakietu jest wykonywane dwustopniowo. W pierwszym etapie jest wydzielany pakiet ze strumienia danych z interfejsu szeregowego lub gniazda sieciowego BSD po stronie serwera i umieszczenie go w strukturze typu *xPacket_t* (**listing 4**). Pole *type* określa typ pakietu, natomiast **pdata* jest wskaźnikiem na początek bufora danych o długości *len*. W drugim etapie następuje rzutowanie bufora *pdata* na odpowiednią strukturę danych, jednoznacznie identyfikowaną polem *type* lub kopiowanie z ewentualną konwersją *big-endian –> little-endian* składowych struktury. Na przykład, dla pakietu zawierającego pojedynczy rekord danych GPS będzie to struktura *xFixRecord_t*.

Bezpośrednio po zestawieniu połączenia z serwerem, urządzenie przesyła identyfikujący je pakiet danych. Serwer w odpowiedzi wysyła pakiet potwierdzający jego gotowość i oczekuje na kolejne pakiety. Wymiana pakietów pomiędzy terminalem a serwerem odbywa się zazwyczaj z każdorazowym potwierdzeniem odbioru krótkim pakietem, ze względu na ograniczone możliwości buforowania przychodzących pakietów, spowodowane ograniczonym rozmiarem pamięci RAM mikrokontrolera. Należy zwrócić uwagę, że dane pomiędzy urządzeniem i serwerem przesyłane są bez jakiegokolwiek szyfrowania i w związku z tym transmisja jest podatna na próby podsłuchu (**rysunek 12**). Ma to szczególnie znaczenie gdy transmisja odbywa się z użyciem publicznego APN, za pośrednictwem sieci Internet, tak jak w przypadku przedstawionego urządzenia. Problem ten można rozwiązać np. poprzez dodanie nieskomplikowanych procedur szyfrowania, dla których zasoby mikrokontrolera AT91SAM7S256 są wystarczające.

Sylwester Nawrocki
sylvester.nawrocki@gmail.com



Rysunek 11. Okno programu Wireshark obrazujące odebrany i „podsłuchany" pakiet