

WOJSKOWA AKADEMIA TECHNICZNA

WYDZIAŁ MECHATRONIKI I LOTNICTWA

ĆWICZENIE LABORATORYJNE

Przedmiot: Systemy bezprzewodowe w automatyce



Temat *Dekodowanie protokołu NMEA*

Prowadzący: dr inż. Wojciech Kaczmarek

Grupa A1A1S1

Violetta Munar Ernandes

Michał Bokiniec

Wstęp

Celem ćwiczenia było zapoznanie się z protokołem NMEA oraz przeprowadzenie procedury dekodowania poszczególnych pakietów protokołu w języku C++.

Protokół NMEA

Protokół transmisji NMEA-0183 został opracowany przez National Marine Electronics Association i jest standardowym protokołem wykorzystywanym przez odbiorniki GPS do komunikacji z urządzeniami elektronicznymi tj. komputer, echosonda lub autopilot. W oparciu o ten protokół utworzono już szereg innych formatów zapisu danych.

Wiadomości NMEA-0183 używają zestawu znaku ASCII i mają precyzyjnie zdefiniowany format. Każda z nich zaczyna się od znaku \$, a kończy się znakami końca linii <CR><LF> (ang. carriage return i line feed). Liczba widocznych znaków w pojedynczej sekwencji nie może przekraczać 80, a poszczególne dane oddzielone są od siebie za pomocą znaku przecinka. Pierwsze litery tworzą nagłówek, który określa, kod urządzenia oraz informacje, jakie znajdują się w danej linii. Dla urządzeń GPS kodem urządzenia jest GP, a informacje zawarte w linii definiowane są przez takie sekwencje nagłówków jak np. GLL - Geographic Position – Latitude/Longitude, GSV Satellites in View czy GGA – Global Positioning System Fixed Data.

Przykład transmitowanych danych:

\$GPGGA,115324.242,5347.6824,N,02754.3478,E,1,11,1.2,26.3,M,-24.7,M,,0000*6E

Dekodowany przez nas pakiet to pakiet \$GPRMC -Recommended Minimum Specific GNSS Data – Raporty pozycji, prędkości, kurs względem ziemi w stopniach i data.

Tab. 3. Format danych w wiadomości RMC			
Numer pola	Nazwa	Przykład	Format/Opis
1	ID wiadomości	\$GPRMC	nagłówek wiadomości RMC
2	Czas UTC	031312.876	hhmmss.sss – godziny, minuty, sekundy, ułamkowe części sekundy
3	Status	A	A – dane poprawne, V – dane niepoprawne
4	Szerokość geograficzna	2446.5270	ddmm.mmmm – stopnie, minuty, ułamkowe części minuty
5	Półkula N/S	N	N – północna, S – południowa
6	Długość geograficzna	12100.1485	dddmm.mmmm – stopnie, minuty, ułamkowe części minuty
7	Półkula E/W	E	E – wschodnia, W – zachodnia
8	Prędkość podróżna	000.0	prędkość względem Ziemi w milach na godzinę (mph)
9	Kurs rzeczywisty	000.0	kurs względem Ziemi w stopniach
10	Data	210802	ddmmyy – dzień, miesiąc, rok
11	Deklinacja magnetyczna	003.3	lokalna odchyłka kierunku północy magnetycznej od rzeczywistej
12	Kierunek deklinacji	W	E – wschodni, W – zachodni
13	Suma kontrolna	*76	XOR wszystkich bajtów pomiędzy '\$' a '*'

Program wraz z objaśnieniem algorytmu programu w komentarzach:

```
1. #include <conio.h> // getch
2. #include <iostream> // string
3. #include <string.h> // strtok, strcmp
4. #include <stdio.h> // printf
5. #include <stdlib.h> // strtod
6.
7. using namespace std;
8.
9. // klasa Wspolrzedna pozwala na proste wyświetlanie dwóch postaci współrzędnych
   geograficznych: stopnie, minuty, sekundy oraz stopnie dziesiętne ze znakiem.
10. // posiada składowe reprezentacji znaku współrzędnej w dwóch formach: +/- oraz N/S, W/E.
11. // obliczanie składowych odbywa się w dwóch krokach: poprzez konstruktor, który przyjmuje
   jako argument
12. // wartość współrzędnej w formacie SSMM.MMMM (S - stopnie, M - minuty kątowe) a następnie
   podanie odpowiedniej półkuli jako argument metody dodajZnak().
13. // Argumentem konstruktora jest tablica znaków char[], argumentem metody jest pojedynczy znak
   char.
14. class Wspolrzedne {
15. public:
16.     string znakPlusMinus; string znakTekstowy; // w printf() należy dopisać
       .c_str() - zamiana na char[]
17.     float wspolrzednaStopnieDziesietne;
18.     int stopnie; int minuty; int sekundy;
19.
20.     Wspolrzedne(char asdf[]) {
21.         float wspolrzedna = strtod(asdf, NULL);
22.         stopnie = ((int) wspolrzedna)/100;
23.         minuty = ((int) wspolrzedna) % 100;
24.         sekundy = (int)((wspolrzedna - (int)wspolrzedna)*60);
25.         wspolrzednaStopnieDziesietne = ((sekundy/60.0) + minuty)/60.0 + stopnie;
26.     }
27.
28.     void dodajZnak(char znak_) {
29.         char znakiMinusujace[] = "SE";
30.         if ( (znak_ == znakiMinusujace[0]) || (znak_ == znakiMinusujace[1]) )
31.             znakPlusMinus = "-";
32.         else znakPlusMinus = "+";
33.         znakTekstowy = znak_;
34.     }
35. };
36.
37. char RMCstr[85] = "$GPRMC,135618.00,A,5215.2979,N,02054.1103,E,123.0,000.0,180308,0
   2.5,E,A*08"; // badana ramka
38. char *p; // pomocniczy wskaznik
39. char oczekiwanyRodzajPakietu[] = "$GPRMC"; // typ ramki, który chcemy badać
40.
41. int main () {
42.     p = strtok (RMCstr,","); if (p) printf("pakiet= %s\n",p); // wyświetlanie typu
       ramki
43.
44.     if (strcmp(p, oczekiwanyRodzajPakietu) == 0) { // test, czy badamy właściwa
       ramke
45.         p = strtok (NULL,","); if (p) printf("czas UTM = %s\n",p); // wyświetlanie
       czasu wysłania ramki
46.         p = strtok (NULL,","); if (p) printf("dane = %s\n\n",p); // wyświetlanie
       typu danych
```

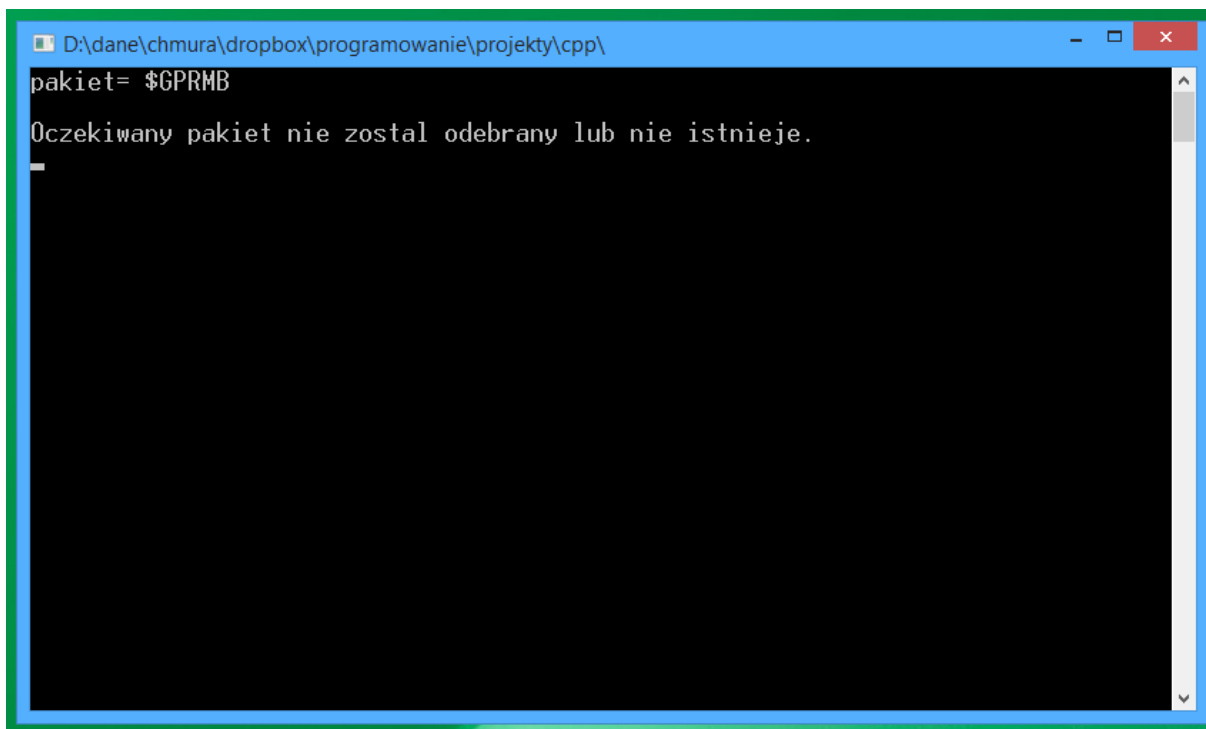
```

47.
48.     p = strtok (NULL, ","); if (p) printf("szerokosc geograficzna = %s\n", p);
    // wyświetlanie surowej szerokości geograficznej
49.     Wspolrzedne szerokosc(p); // tworzenie nowego obiektu typu Wspolrzedna za
    pomocą konstruktora
50.     p = strtok (NULL, ","); if (p) printf("polkula = %s\n",p); // wyswietlanie
    polkuli
51.     szerokosc.dodajZnak(*p); // definiowanie znaku wspolrzednej w obiekcie
    szerokosc
52.     printf("%ist. %i\' %i\' %s\n", szerokosc.stopnie, szerokosc.minuty,
    szerokosc.sekundy, szerokosc.znakTekstowy.c_str()); // wyświetlanie sformatowanej
    szerokości geograficznej
53.     printf("%s%.5fst. \n\n", szerokosc.znakPlusMinus.c_str(),
    szerokosc.wspolrzednaStopnieDziesietne); // wyświetlanie sformatowanej szerokości
    geograficznej
54.
55.     p = strtok (NULL, ","); if (p) printf("dlugosc geograficzna = %s\n", p); //
    wyświetlanie surowej dlugosci geograficznej
56.
57.     Wspolrzedne dlugosc(p); // tworzenie nowego obiektu typu Wspolrzedna za
    pomocą konstruktora
58.     p = strtok (NULL, ","); if (p) printf("polkula = %s\n",p); // wyswietlanie
    polkuli
59.     dlugosc.dodajZnak(*p); // definiowanie znaku wspolrzednej w obiekcie
    dlugosc
60.     printf("%ist. %i\' %i\' %s\n", dlugosc.stopnie, dlugosc.minuty,
    dlugosc.sekundy, dlugosc.znakTekstowy.c_str()); // wyświetlanie sformatowanej
    dlugosci geograficznej
61.
62.     printf("%s%.5fst. \n\n", dlugosc.znakPlusMinus.c_str(),
    dlugosc.wspolrzednaStopnieDziesietne); // wyświetlanie sformatowanej dlugosci
    geograficznej
63.
64.     p = strtok (NULL, ","); if (p) printf("predkosc [wezly] = %s, [m/s] = %.1f,
    [km/h] = %.1f\n\n",p, 1.852*strtof(p, NULL),0.514444*strtof(p, NULL)); //
    wyswietlanie predkosci w trzech roznym jednostkach
65.
66.     p = strtok (NULL, ","); if (p) printf("%s\n",p); // kat przemieszczenia
67.     p = strtok (NULL, ","); if (p) printf("%s\n",p); // data
68.     p = strtok (NULL, ","); if (p) printf("%s\n",p); // zmiany magnetyczne
69.     p = strtok (NULL, ","); if (p) printf("%s\n\n",p); // kierunek zmian
    magnetycznych
70.
71.     p = strtok (NULL, ","); if (p) printf("suma kontrolna: %s\n",p); //
    wyswietlanie sumy kontrolnej
72.     }
73.     else printf("\nOczekiwany pakiet nie zostal odebrany lub nie istnieje.\n"); //
    komunikat bledu
74.     getch(); // oczekiwanie na dowolny klawisz
75. }

```

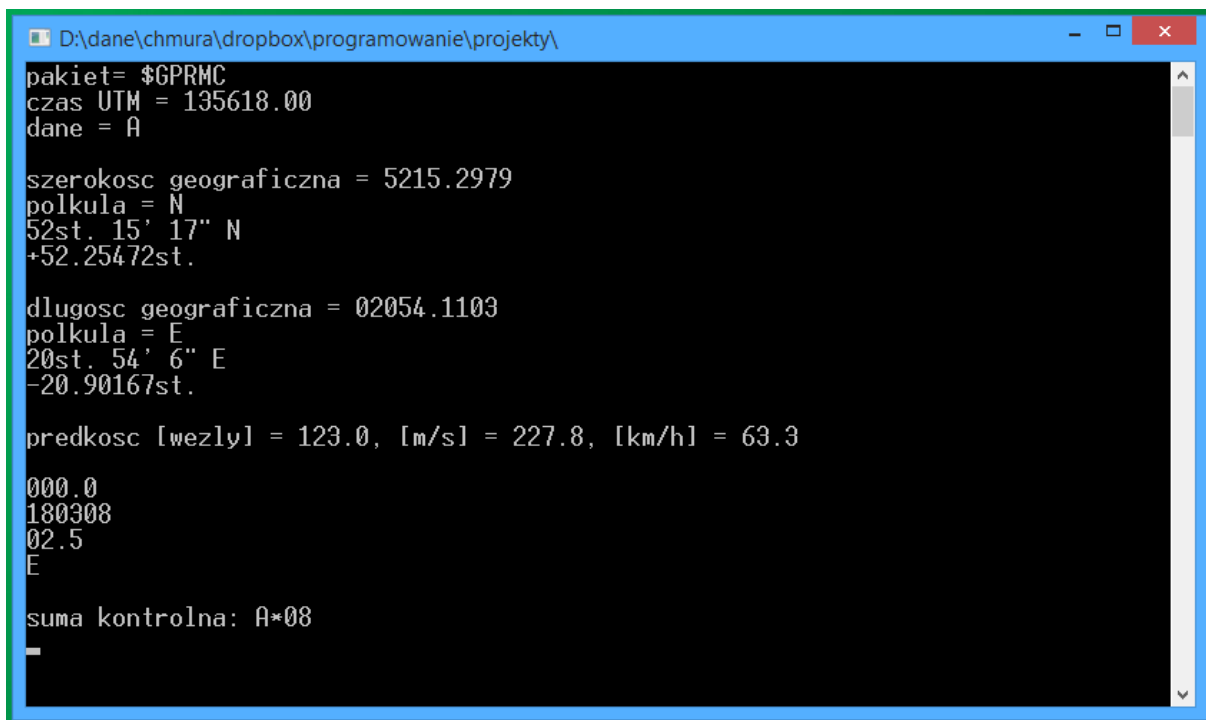
Program rozpoznaje pakiet, który został wskazany przez prowadzącego do odczytu i po jego zdekodowaniu wyświetla odczytane dane w formie podanej przez prowadzącego.

W przypadku podania pakietu o innym identyfikatorze program wyświetli komunikat, że oczekiwany pakiet nie został odebrany lub nie istnieje.



```
D:\dane\chmura\dropbox\programowanie\projekty\cpp\
pakiet= $GPRMB
Oczekiwany pakiet nie został odebrany lub nie istnieje.
```

Rysunek 1 Widok Programu



```
D:\dane\chmura\dropbox\programowanie\projekty\
pakiet= $GPRMC
czas UTM = 195618.00
dane = A

szerokosc geograficzna = 5215.2979
polkula = N
52st. 15' 17" N
+52.25472st.

dlugosc geograficzna = 02054.1103
polkula = E
20st. 54' 6" E
-20.90167st.

predkosc [wezly] = 123.0, [m/s] = 227.8, [km/h] = 63.3

000.0
180308
02.5
E

suma kontrolna: A*08
```

Rysunek 2 Widok Programu

Wnioski

Zadanie zostało zrealizowane, napisany program w języku C++ rozpoznaje pakiet danych i dekoduje wiadomość RMC protokołu NMEA. Dane zostają przedstawione w formie podanej przez prowadzącego.