Michał Muzyk 218332 Data: 30.10.2016  
Jakub Jaskóła 218349

Grupa: Wtorek TN 13:15-16:15

Prowadzący: Dr inż. Jan Nikodem

**Urządzenia peryferyjne**

**GPS**

**1. Zadania do wykonania**

Celem ćwiczenia było napisanie aplikacji graficznej, która umożliwi komunikacje między komputerem a urządzeniem GPS korzystającymi przy połączeniu z technologii Bluetooth. Aplikacja ta miała być w stanie otworzyć wirtualny port szeregowy pomiędzy urządzeniami i odbierać oraz wyświetlać przesyłane dane z odbiornika GPS.

**2. Wstęp teoretyczny**

GPS (Global Positioning System) to amerykański system nawigacji satelitarnej obejmujący swym zasięgiem całą kulę ziemską. Służy m.in. do wyznaczania pozycji geograficznej (długość, szerokość geograficzną oraz wysokość elipsoidalną). System jest ogólnodostępny, z jego usług może korzystać każdy - wystarczy tylko posiadać odpowiedni odbiornik GPS. Do określenia pozycji w trójwymiarowej przestrzeni i czasu systemu konieczny jest jednoczesny odbiór sygnału z przynajmniej czterech satelitów. Odbiornik użytkownika oblicza trzy pseudoodległości do satelitów oraz odchyłki czasu (różnicy między tanim i niedostatecznie dokładnym wzorcem kwarcowym zainstalowanym na odbiorniku i precyzyjnym zegarem atomowym na satelicie). Dokładne współrzędne satelity są transmitowane w depeszy nawigacyjnej. W projekcie wykorzystywaliśmy funkcje znajdujące się w plikach nagłówkowych:

* tchar.h – biblioteka umożliwiająca konwersję ciągów znaków
* windows.h – biblioteka zawierająca funkcje potrzebne do otwarcia portu szeregowego

**3.Opisany kod programu**

#include <tchar.h>

#include <iostream>

#include <windows.h>   
using namespace std;

BOOL modyfikacja(HANDLE hComPort);

HANDLE hSerial;

int \_tmain(int argc, TCHAR \*argv[], TCHAR \*envp[])

{

DWORD cBytes\_out, cBytes\_in;

DWORD dwMask;

char cBuffer\_out[] = "";

TCHAR cBuffer\_in[80];

cout << "Port szeregowy otwarty, nacisnij ctrl+c aby przerwac" << endl;

**// otwarcie portu szeregowego COM2 dla odczytu i zapisu**

hSerial = CreateFile(\_T("COM2:"), GENERIC\_READ | GENERIC\_WRITE, 0, NULL,

OPEN\_EXISTING, 0, NULL);

**// sprawdzenie wartosci uchwytu**

if (hSerial == INVALID\_HANDLE\_VALUE) {

cout << "Blad otwarcia" << endl;

system("pause");

return 0;

}

**// modyfikacja parametrów portu szeregowego**

if (!modyfikacja(hSerial)) {

cout << "Bledy konfiguracji" << endl;

system("pause");

return 0;

}

**// zapis do portu szeregowego**

if (!WriteFile(hSerial, cBuffer\_out, strlen(cBuffer\_out), &cBytes\_out, NULL)) {

cout << "Blad zapisu" << endl;

system("pause");

return 0;

}

SetCommMask(hSerial, EV\_RXCHAR | EV\_ERR);

cBuffer\_in[0] = 0;

**// dopóki ctrl+c nie jest wciśnięty, rób!**

while (cBuffer\_in[0] != 0x03) {

WaitCommEvent(hSerial, &dwMask, 0);

cBytes\_in = 1;

while (cBytes\_in != 0) {

**// czyta dane z urządzenia i wyświetla**

if (ReadFile(hSerial, cBuffer\_in, 79, &cBytes\_in, NULL)) {

if (cBytes\_in == 0) break;

printf("%s", cBuffer\_in, cBytes\_in);

if (!WriteFile(hSerial, cBuffer\_in, cBytes\_in, &cBytes\_out, NULL)) {

cout << "Blad zapisu" << endl;

system("pause");

return 0;

}

}

}

}

**//zamknięcie uchwytu**

CloseHandle(hSerial);

return 1;

}

BOOL modyfikacja(HANDLE hComPort)

{

COMMTIMEOUTS ctos;

DCB PortDCB; **//struktura kontrolująca ustawienia portu szeregowego**

PortDCB.DCBlength = sizeof(DCB);

**// ustaw standardowe ustawienia dla portu szeregowego**

GetCommState(hSerial, &PortDCB);

**// zmiana ustawien portu szeregowego**

PortDCB.BaudRate = 9600; **// szybkosc transmisji**

PortDCB.fBinary = TRUE;  **// tryb binarny**

PortDCB.fParity = TRUE; **// wlacz sprawdzanie parzystosci**

PortDCB.ByteSize = 8;  **// liczba bitów**

PortDCB.Parity = NOPARITY;  **// parzystosc bitow (bez parzystości)**

PortDCB.StopBits = ONESTOPBIT; **// bity stopu**

if (!SetCommState(hSerial, &PortDCB)) {

cout << "Blad konfiguracji portu szeregowego" << endl;

system("pause");

return false;

}

**// ustaw timeouty odczytu**

ctos.ReadIntervalTimeout = MAXDWORD; **//maksymalne dopuszczalne przerwy pomiędzy bajtami**

ctos.ReadTotalTimeoutMultiplier = MAXDWORD; **//mnożnik do wyliczenia okresu timeoutu**

ctos.ReadTotalTimeoutConstant = 1; **//stała służąca do wyliczenia całego okresu timeotu**

if (!SetCommTimeouts(hSerial, &ctos)) {

cout << "Blad konfiguracji portu szeregowego" << endl;

system("pause");

return false;

}

return true;

}  
  
**4. Wnioski**

Zrealizowanie powyższego ćwiczenia pozwoliło nam na zapoznanie się z obsługą urządzeń używających technologii Bluetooth oraz z konfiguracją portu szeregowego. Poznaliśmy biblioteki, które pomogły nam w napisaniu programu komunikującego się z poprzez port szeregowy. W trakcie zajęć udało nam się zrealizować zadanie i po uruchomieniu programu mogliśmy odczytać dane z odbiornika GPS w postaci protokołu NMEA.