„Zeszyty Naukowe DWSPiT.
Studia z Nauk Technicznych"
2014 (3), s. 81–98.

Władimir Dragajew

Właściwości programowania wC++ bezprzewodowego kanału transmisji danych

Streszczenie: Rozważano problem bezprzewodowego kanału komunikacyjnego do sterowania robotami transportowymi. Przeprowadzone testy kanału komunikacyjnego pozwoliły na zastosowanie tego kanału do sterowania innymi ruchomymi obiektami.

Słowa kluczowe: robot, komunikacja, bezprzewodowa, sterowanie.

Accomplished and researched computer system of wireless data transfer which may be applied to remotely control mobile robots

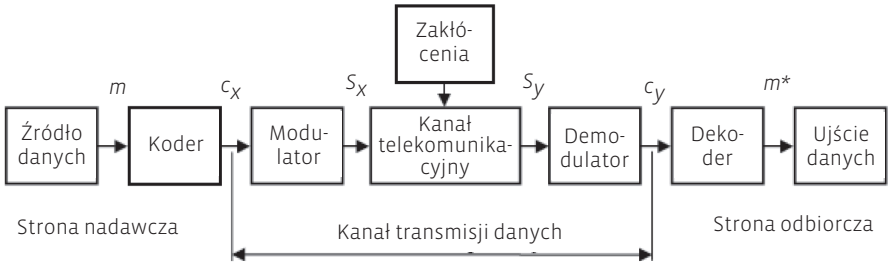
Summary: The article considers the issues of wireless communication channel construction for the purposes of transportation robots management. The test conducted on the computer communication channel allows for its application also for remote control of other moving objects.

Keywords: robots, communication, wireless, management.

1.Wstęp

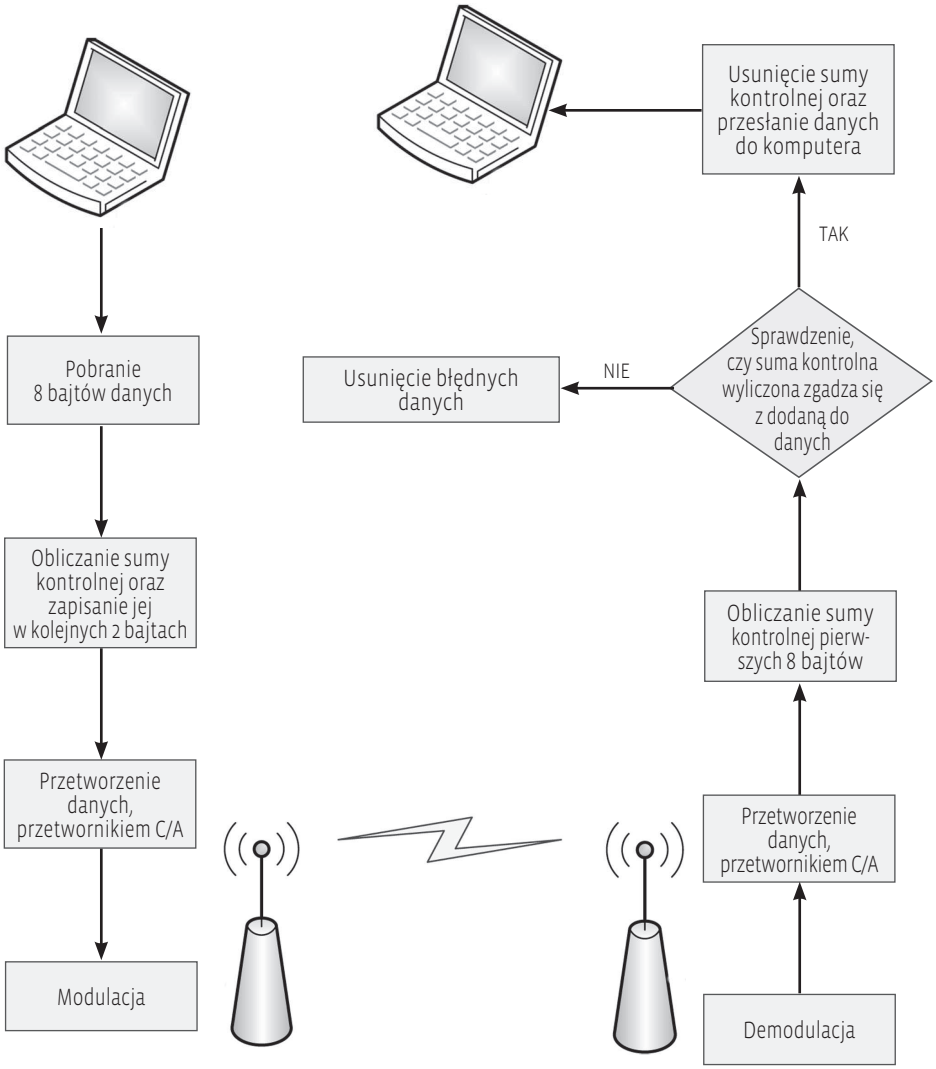
Transmisja danych w systemach informatycznych jest podatna na różnego rodzaju błędy, przed których konsekwencjami musi zostać zabezpieczona. Zabezpieczenie informacji odbywa się przez kodowanie korekcyjne, dzięki czemu zwiększa się niezawodność systemów informatycznych. Na sygnał w kanale telekomunikacyjnym działają różne zakłócenia, które powodują przekłamania w transmisji. W systemach transmisji danych wymaga się stopy błędów rzędu 10^{-6}–10^{-9}.

Rysunek 1. przedstawia sposób transmisji danych z wykorzystaniem kodu korekcyjnego. Do standardowego systemu transmisyjnego został dodany koder oraz dekoder. Koder realizuje proces kodowania i jest on umieszczony między źródłem danych a łączem transmisji danych. Zadaniem kodera jest przekształcenie ciągu wiadomości m w ciąg kodów c_x, który następnie jest poddawany procesowi modulacji.



Rys. 1. System transmisyjny z kodem korekcyjnym

Kanały transmisji danych tworzy się na łączach telekomunikacyjnych, do których dodaje się modulator i demodulator [L1]. Modulator zajmuje się przekształceniem sygnału cyfrowego c_x w sygnały S_X, przystosowując go do parametrów odpowiednich dla danego kanału telekomunikacyjnego pod względem pasma i amplitudy. Sygnał wyjściowy kanału transmisyjnego S_Y zostaje poddany procesowi demodulacji, w czasie której następuje zamiana sygnału analogowego na postać cyfrową. Dekoder układu transmisyjnego zajmuje się korekcją wszystkich napotkanych błędów. W tym celu dekoder wykorzystuje określoną metodę korekcji, która zależy od charakterystyki kanału transmisyjnego.



Rys. 2. Schemat przebiegu transmisji od nadajnika do odbiornika

Podczas projektowania pary kodera i dekodera głównym problemem jest osiągnięcie wymaganej stopy błędów transmisji oraz uzyskanie możliwie jak najszybszej prędkości przesyłu danych.

2. Projekt stanowiska laboratoryjnego do bezprzewodowej transmisji danych

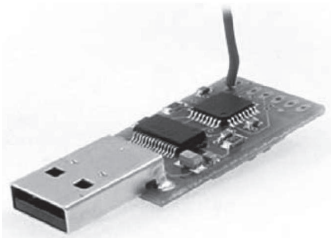
Celem stworzenia stanowiska było : 1) badanie transmisji danych na częstotliwości 868 MHz, 2) sprawdzenie przesłanych danych pod kątem poprawności oraz napisania oprogramowania obsługującego urządzenia nadawczo-odbiorcze.

Stanowisko zostało wyposażone w następujące elementy (rys. 2.):

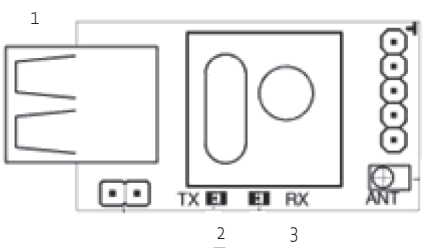
a) dwa urządzenia nadawczo-odbiorcze,
b) dwa laptopy z zainstalowanym programem.

3. Urządzenie nadawczo-odbiorcze

Urządzenie stworzone na potrzeby badawcze umożliwia bezprzewodowe przesyłanie danych z maksymalną prędkością do 44000 b/s. Urządzenie wyposażone jest w złącze USB, za pomocą którego odbywa się komunikacja z komputerem. Wykorzystywana częstotliwość do transmisji wynosi 868 MHz. Urządzenie zasilane jest bezpośrednio z portu USB (rys. 3).

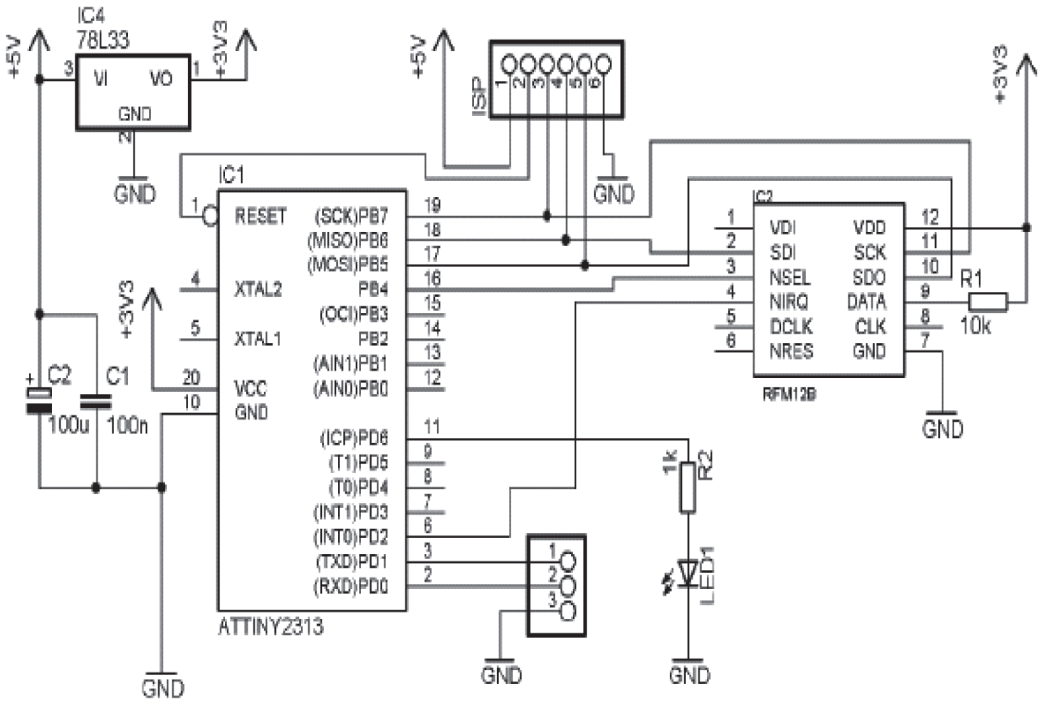


Rys. 3. Urządzenie nadawczo-odbiorcze



1. USB typu B
2. TX – dioda sygnalizująca nadawanie danych
3. RX – dioda sygnalizująca odbieranie danych
4. Antena

Rys. 4. Schemat wyprowadzeń



Rys. 5. Schemat urządzenia

4. Oprogramowanie

Do poprawnego działania aplikacji wymagana jest instalacja sterownika FT232R, który jest zamieszczony na stronie internetowej o adresie: **http://www.ftdichip.com/Drivers/VCP.htm** [L4]. Zadaniem tego sterownika jest pośrednictwo przy przesyłaniu danych pomiędzy aplikacją a portem USB.

Po uruchomieniu aplikacji należy wybrać tryb pracy (rys. 5). Do wyboru są dwa tryby:

- Odbiornik.
- Nadajnik.

W załączniku zamieszczono kod programu.

5. Badanie transmisji danych

Badania zostały przeprowadzone w dwóch środowiskach: na otwartym płaskim terenie oraz w zakładzie mechanicznym. Przetestowano przekazywanie danych z prędkościami transmisji: 44 000 b/s, 28 000 b/s, 22 000 b/s, 16 000 b/s oraz odległościami między nadajnikiem oraz odbiornikiem: 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 m.

Badanie transmisji danych z prędkością 16 000 b/s przeprowadzono na odległościach 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 m. Zaobserwowano, iż 100% poprawności transmisji odbywa się w przedziale od 1 do 10 m.

Przy transmisji danych na odległość większą niż 10 m zaobserwowano spadek poprawności przesyłu. Poprawność w przedziale od 10 do 35 m waha od 100 do 80%. Badanie transmisji danych z prędkością 22 000 b/s przeprowadza się na odległościach 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 m. 100-proc. poprawność transmisji odbywa się w przedziale od 1 do 35 m. Nieznaczny 3-proc. spadek poprawności odebranych danych obserwujemy przy odległości 45 m. Przy odległości 50 m następuje spadek o 46%.

Badanie transmisji danych z prędkością 28 000 b/s przeprowadza się na odległościach 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 m. W przedziale od 1 do 20 m transmisja odbywała się bez zakłóceń ze 100-proc. poprawnością. Od 20 do 50 m zaobserwowano powolny spadek poprawności przesyłu o 36%. Powyżej 50 m następuje spadek poprawności do 0%.

Badanie transmisji danych z prędkością 44 000 b/s przeprowadza się na odległościach 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 m. Dla tej prędkości bezbłędna 100-proc. skuteczność transmisji występuje w przedziale od 1 do 15 m. Powyżej odległości 15 transmisja staje się niestabilna, waha się, a odchyłki wynoszą 3%.

Przy odległości przekraczającej 35 m następuje duży spadek przesyłanych informacji aż do 55 m, gdzie zachodzi całkowite zerwanie łączności.

Kolejną część badań przeprowadzono w zakładzie mechanicznym. Wyniki dla poszczególnych prędkości są następujące:

Badania transmisji danych z prędkością 16 000 b/s przeprowadza się na odległościach 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 m. W tym terenie łączność była możliwa od 1 do 35 m w tym do 10 m przekaz w 100% okazał się prawidłowy. Powyżej 10 m aż do końca granicy łączności poprawność bardzo mocno się waha i ma tendencje spadkowe.

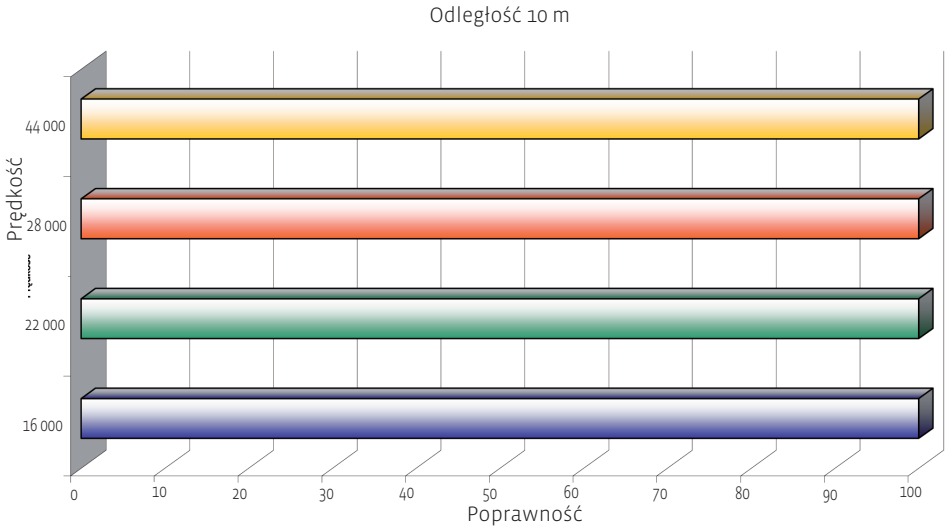
Badania transmisji danych z prędkością 22 000 b/s przeprowadza się na odległościach 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 m. Przy tej prędkości transmisji zaobserwowano nieznaczne wydłużenie granicy łączności, wynosi ona 45 m. W pełni poprawna transmisja odbywa się od 1 do 15 m. Od 15 do 30 m występuje delikatny spadek poprawności przesyłanych danych i wynosi on 3%.

Powyżej 30 m poprawność transmisji gwałtownie spada i urywa się przy 45 m. Badania transmisji danych z prędkością 22 000 b/s przeprowadza się na odległościach 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 m. Prędkość ta w tych warunkach gwarantuje niezakłóconą transmisję od 1 do 10 m. W przedziale od 15 do 25 m obserwowany jest powolny spadek poprawności ze 100 do 95%. Natomiast od 25 do 35 m następuje ogromne obniżenie poprawności danych aż o 81%. Powyżej 35 m transmisja nie spada tak gwałtownie, ale i tak jest znaczna, dążąca do 0% poprawności danych. Badania transmisji danych z prędkością 44 000 b/s przeprowadzona się na odległościach 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 m.

Zaobserwowano, że 100% transmisji odbywa się w przedziale od 1 do 10 m. Przy transmisji danych na odległość większą niż 10 m zaobserwowano znaczny spadek poprawności przesyłu. Poprawność w przedziale od 10 do 20 m spada o 4%, natomiast od 20 do 35 m poprawność spada o 83%. Transmisja zostaje całkowicie zerwana na 45 m.

6. Porównanie poprawności przesyłanych danych w zależności od prędkości transmisji oraz wybranych odległości nadajnik – odbiornik

Z przeprowadzonych badań wynika, że transmisja w odległościach od 1 do 10 m, na otwartym terenie przebiega bez zakłóceń (rys. 6).

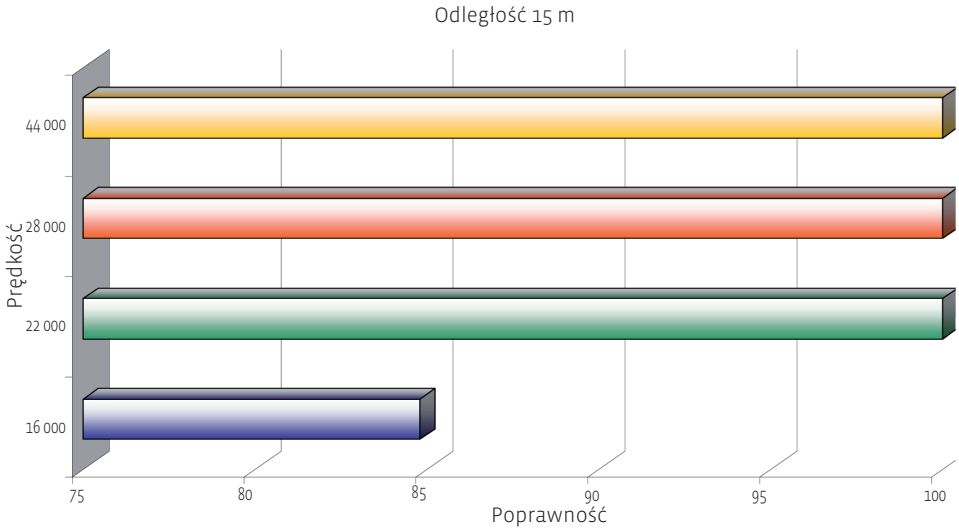


Rys. 6. Porównanie poprawności przesyłu danych dla 4 różnych prędkości

Sytuacja zmienia się powyżej 15 m w zależności od prędkości transmisji, jak widać na wykresie zamieszczonym na nastepnej stronie (rys. 7).

Dla tej odległości najbardziej efektywne są prędkości transmisji 22 000, 28 000, 44 000 b/s. W przypadku prędkości 16 000 b/s zaobserwowano spadek poprawności przesyłanych danych o 16%.

Dla 20 m i prędkości 16 000 b/s odnotowano poprawę o 14%. Prędkość 22 000, 28 000b/s dalej cechuje się 100-proc. poprawnością transmisji.



Rys. 7. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 15 m

Przy kolejnej badanej odległości 25 m dla prędkości 16 000 b/s zaobserwowano postępujący spadek procentowej wartości poprawnych danych. W przypadku prędkości 44 000 b/s i 28 000 b/s nastąpił niewielki spadek poprawności 2% dla 44 000b/s i 1% dla 28 000 b/s. Prędkość 22 000 b/s nadal jest w 100% poprawna.

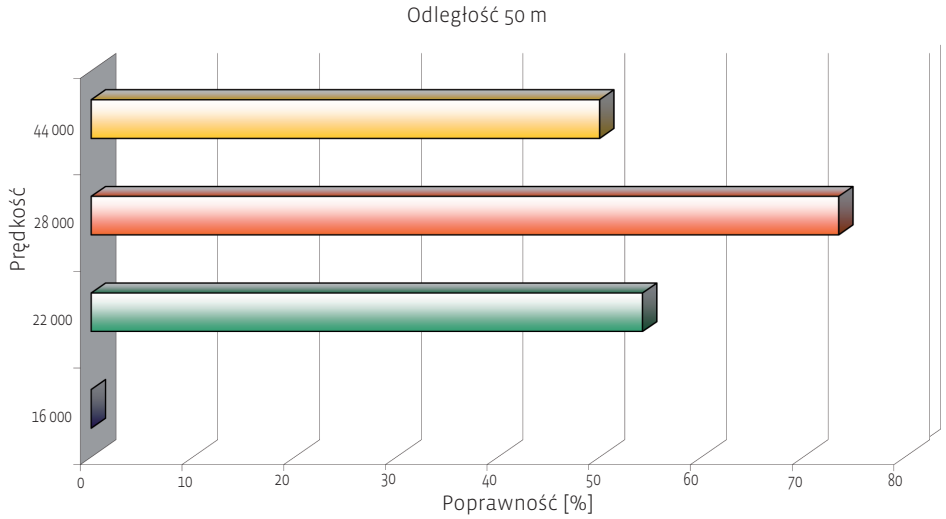
Analiza 30 m ukazuje dalszy spadek poprawności dla prędkości 16 000 b/s transmitowanych danych i wynosi ona 75% dla 22 000 b/s poprawność danych wynosi 100% , a dla prędkości 28 000 b/s wynosi 98,3%, natomiast poprawność dla prędkości 44 000 b/s wynosi 99,74%.

Na 35 m odnotowano poprawność dla poszczególnych prędkości : 16 000 b/s – 80%, 22 000 b/s – 100%, 28 000 b/s – 98,3%, 44 000 b/s – 99,74%.

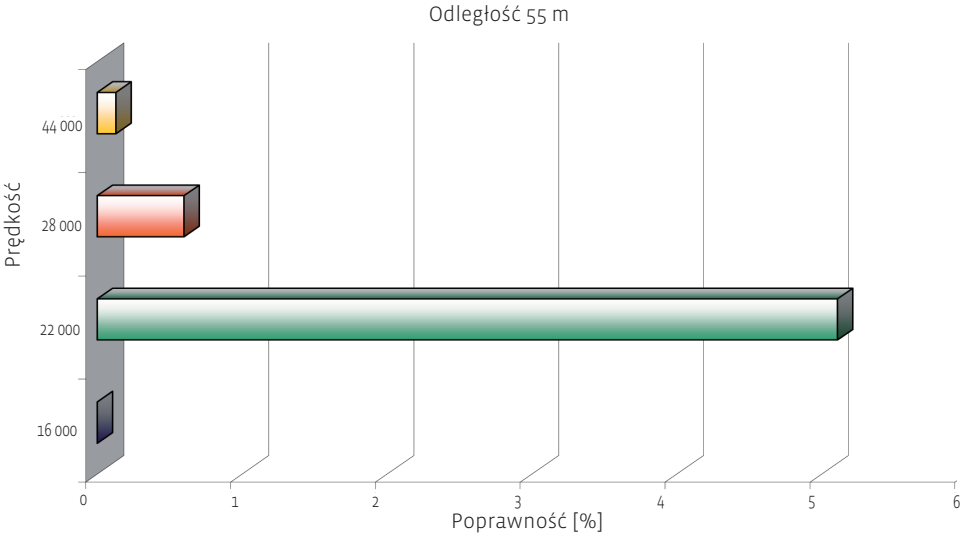
Natomiast na 40 m odnotowano poprawność dla poszczególnych prędkości : 16 000 b/s – 80%, 22 000 b/s – 100%, 28 000 b/s – 98,3%, 44 000 b/s – 99,74%.

Badanie 45 m wykazało bardzo duży spadek poprawności z 80% do 0,5%. W przypadku prędkości 22 000 b/s wynosi ona aż 96,4%, przy prędkości 28 000 b/s poprawność spadła do 75,4%, natomiast przy prędkości 44 000 b/s wynosi 76%.

W kolejnych dwóch badanych odległościach – 50 oraz 55 m transmisja z prędkością 16 000 b/s była niemożliwa. W przypadku prędkości 22 000 b/s z 54% spada do 5% oraz prędkość 28 000 b/s z 73,52% spada do 0,6%, a dla prędkości 44 000 b/s z 50% spada do 0,19% (patrz rys. 8 oraz 9).

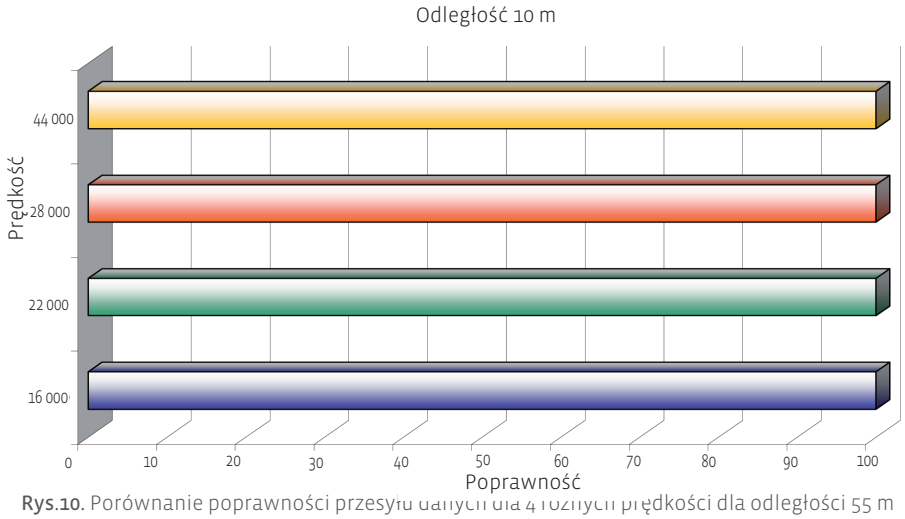


Rys. 8. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 50 m



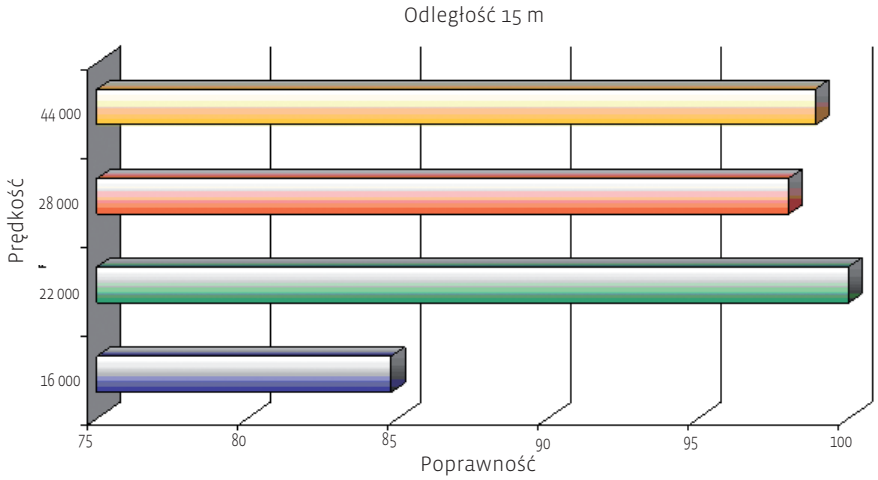
Rys. 9. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 55 m

W drugiej części badań zostały porównane prędkości w terenie zabudowanym. Podobnie jak w terenie otwartym, transmisja odbywała się bez przeszkód i ze 100-proc. poprawnością od 1 do 10 m (rys. 10).



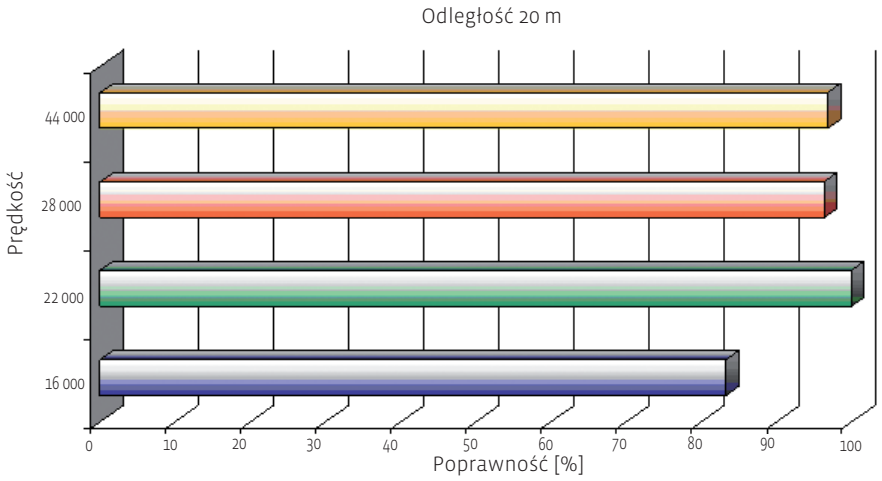
Rys.10. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 55 m

Analiza 15 m wykazała, że jedynie dla prędkości 22 000 b/s dane były przesyłane ze 100-proc. poprawnością. Najgorzej w transmisji wypadła prędkość 16 000 b/s, gdzie odnotowano 16-proc. spadek, natomiast w przypadku prędkości 44 000 b/s spadek wyniósł jedynie 1,11%, a w przypadku prędkości 28 000 b/s spadek wyniósł 3,7% (rys. 11).



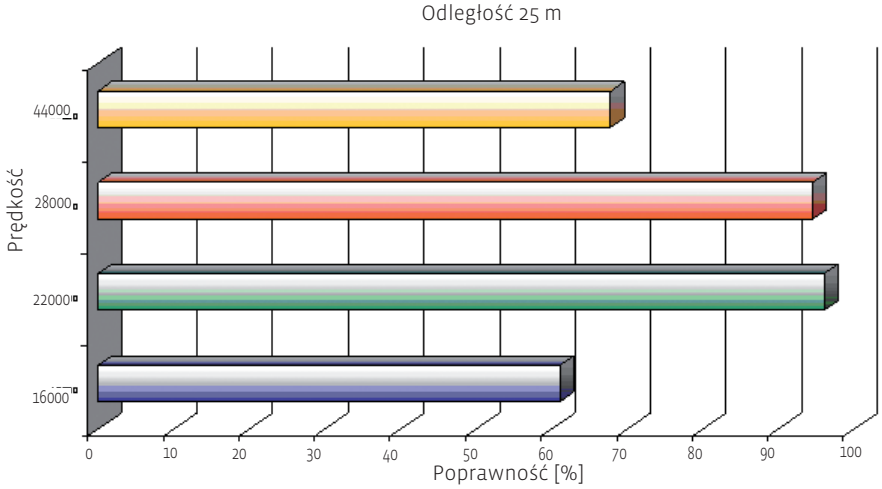
Rys. 11. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 15 m

Na 20 m odnotowano poprawność dla poszczególnych prędkości (rys. 12) : 16 000 b/s – 83,24%, 22 000 b/s – 99,94%, 28 000 b/s – 96,3%, 44 000 b/s – 96,77%.



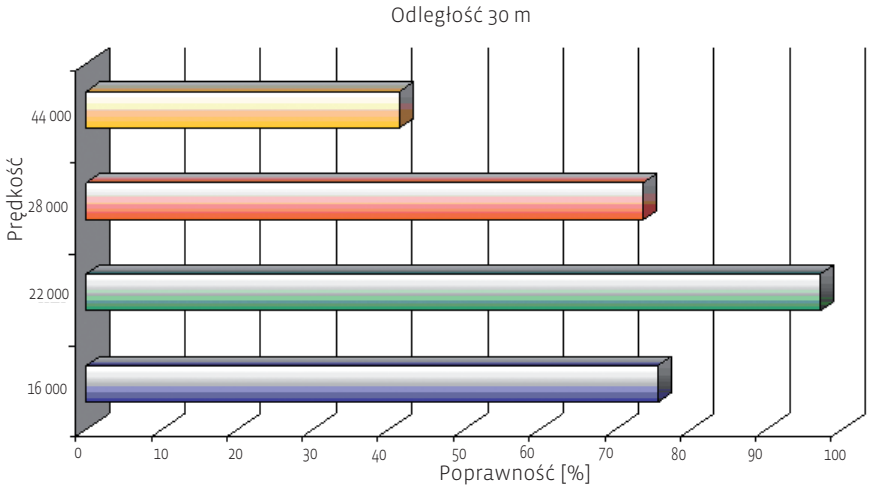
Rys. 12. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 20 m.

Badanie 25 m wykazało (rys. 13) spadek poprawności prędkośsi 16 000 b/s z 83,24% do 61,11%. W przypadku prędkości 22 000 b/s wynosi ona aż 96,09%, przy prędkości 28 000 b/s poprawność spadła o 3,85%, natomiast przy prędkości 44 000 b/s wynosi ona 67,8.



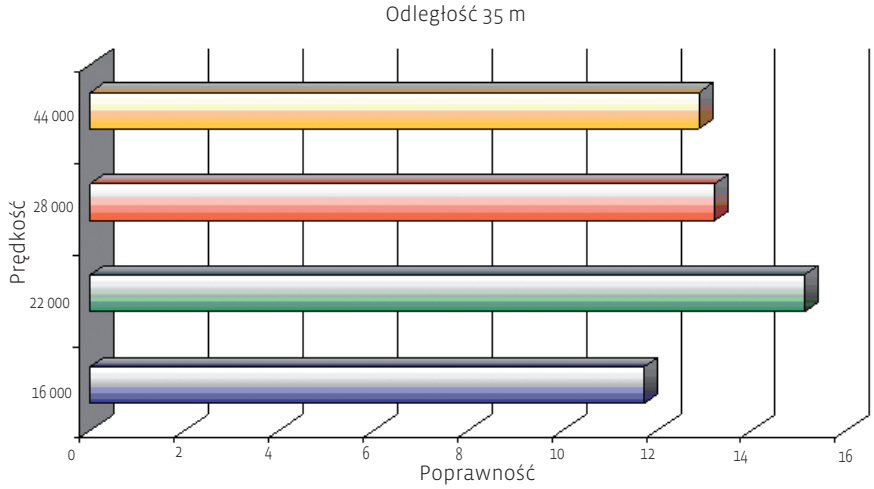
Rys. 13. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 25 m

Na 30 m badanej transmisji (rys. 14) zaobserwowano wzrost poprawności dla prędkości 16 000 b/s wynoszącej 14,51%, natomiast w przypadku prędkości 22 000 b/s odnotowano niewielką poprawę o 1,1% względem wcześniejszego odczytu. Prędkość 28 000 b/s ma tendencje spadkowe i dla tej odległości poprawność wynosi 73,76%. Podczas transmisja przy prędkości 44 000 b/s zaobserwowano duży spadek aż o 26,28%.

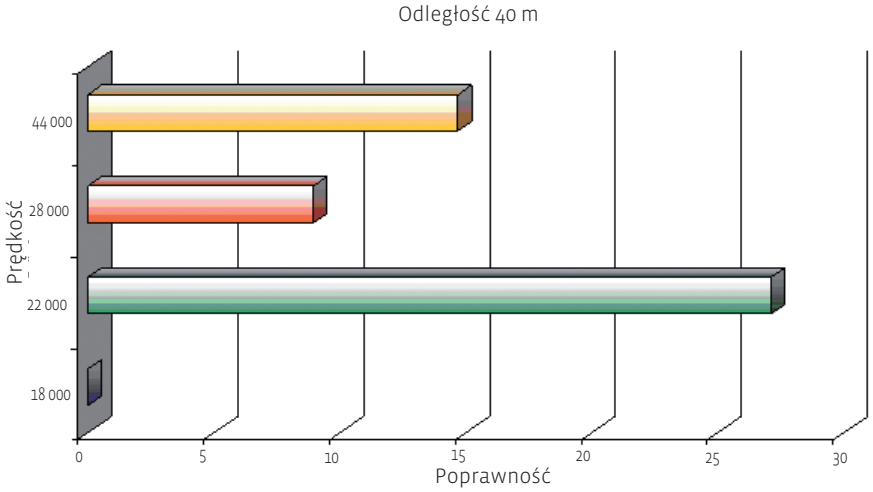


Rys. 14. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 30 m

W kolejnych dwóch badanych odległościach – 35 oraz 40 m transmisja z prędkością 16 000 b/s spadła z 11,75% do 0%. W przypadku prędkości 22 000 b/s z 15,12% wzrosła do wartości 27,15%, prędkość 28 000 b/s z 13,22% spadła do 8,97%, a dla prędkości 44 000 b/s poprawność nieznacznie wzrosła o 1,75%. Ilustrują to rysunki 15 i 16.



Rys. 15. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 35 m



Rys. 16. Porównanie poprawności przesyłu danych dla 4 różnych prędkości dla odległości 40 m

7. Wnioski

1. W zakresie odległości od 1 do 10 m zgodność danych jest 100-procentowa oraz niezależna od prędkości i środowiska. Na większych odległościach odnotowano postępujący spadek poprawności przesyłanych danych.
2. Badania wykazały, że najbardziej efektywną prędkością dla skonstruowanego urządzenia jest prędkość 22 000 b/s.
3. W przypadku aglomeracji miejskich poprawność transmisji, bez względu na jej prędkość, się pogarszała. Spowodowane było to dużą ilością zakłóceń, jaka występuje w tym środowisku.

Zaprojektowane urządzenie może znaleźć swoje zastosowanie w telemetrii i automatyce oraz w pozostałych dziedzinach, gdzie transmisja przewodowa jest niemożliwa, a ilość przesyłanych danych jest niewielka.

Załącznik. Kod programu Transmiter

```
#include <vcl.h>
#pragma hdrstop
#include „Unit1.h”
#pragma package(smart_init)
#pragma resource „*.dfm”
TForm1 *Form1;
HANDLE hCom;
void TForm1::otworz_rs232(String com){
BOOL fSuccess;
```

```
DWORD RS_ile;
bool wykonaj=true;
hCom = CreateFile(com.c_str(), GENERIC_READ | GENERIC_WRITE,
    0,
    NULL,
    OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);

if (hCom == INVALID_HANDLE_VALUE) {
    wykonaj=false;
    ShowMessage(„Błędny port COM, Błąd numer:" + IntToStr(GetLastError()));}
if(wykonaj) {
    DCB dcb;
    COMMTIMEOUTS czasy;
if (!GetCommState(hCom, &dcb))  {
    ShowMessage(„GetCommState failed with error „ + IntToStr(GetLastError()));}
dcb.BaudRate=CBR_56000;
dcb.ByteSize = 8;
dcb.Parity = NOPARITY;
dcb.StopBits = TWOSTOPBITS
  if (!SetCommState(hCom, &dcb)) {
    ShowMessage(„SetCommState failed with error „ + IntToStr(GetLastError()));}
    czasy.WriteTotalTimeoutConstant=200;
    czasy.WriteTotalTimeoutMultiplier=100;
    czasy.ReadTotalTimeoutConstant=0;
    czasy.ReadTotalTimeoutMultiplier=0;
    SetCommTimeouts(hCom,&czasy); }}
void __fastcall TForm1::BtWczytajClick(TObject *Sender) {
  if(DialPlik->Execute()){
    nazwa_pliku=DialPlik->FileName.c_str();
    txtInfo->Text=nazwa_pliku;
    BtWyslij->Visible=true;
    FILE * plik;
    plik = fopen( nazwa_pliku.c_str(), „rb");
      while((fgetc(plik))!=EOF)
      wielkosc_pliku++;
    fseek(plik,0,0);
    fclose(plik);
    txtWielkosc->Visible=true;
    LWielkoscPliku->Visible=true;
```

```
        txtWielkosc->Text=IntToStr(wielkosc_pliku);
        wielkosc_pliku==0; }
      else{
        txtWielkosc->Visible=false;
        LWielkoscPliku->Visible=false;}
    BtNasluch->Visible=false; }
    void __fastcall TForm1::BtWyslijClick(TObject *Sender) {
      DWORD RS_buf;
      DWORD RS_ile;
      FILE * plik;
      plik = fopen( nazwa_pliku.c_str(), "rb");
      PasekPostepu->Visible=true;
      PasekPostepu->Min=0;
      PasekPostepu->Max=wielkosc_pliku;
      LProgresInfo->Caption ="Wysyłam plik ...";
      LProgresInfo->Visible=true;
      otworz_rs232(nr_com);
      while((RS_buf=fgetc(plik))!=EOF) {
        WriteFile( hCom, &RS_buf, 1, &RS_ile, 0);
        PasekPostepu->StepBy(1); }
      fclose(plik);
      CloseHandle(hCom);
      PasekPostepu->Position =0;
      PasekPostepu->Visible=false;
      LProgresInfo->Visible=false;
      ShowMessage(„Plik został wysłany!"); }

    void __fastcall TForm1::BtUstawClick(TObject *Sender) {
      BtUstaw->Enabled=false;
      BtZapisz->Visible=true;
      LPredkosc->Visible=true;
      LPort->Visible=true;
      cmbPorty->Visible=true;
      cmbPredkosc->ItemIndex=0;
      cmbPredkosc->Visible=true;
      BtAnuluj->Visible=true;
      cmbPorty->Clear();
      for(int i=1; i<10; i++)
        cmbPorty->Items->Add(„COM"+IntToStr(i));
```

```
  cmbPorty-> ItemIndex=0; }
void __fastcall TForm1::BtZapiszClick(TObject *Sender) {
   BOOL wynik;
   DWORD RS_ile,RS_buf;
   char command[]="\x43\x78\x1E\x08\x00";
   String odp="";
   int predkosc;
   nr_com=cmbPorty->Text;
   predkosc =StrToInt(cmbPredkosc->Text.SubString(0,2));

  switch( predkosc ) {
     case 44: command[4]= ,\x2C'; break;
     case 28: command[4]= ,\x1C'; break;
     case 22: command[4]= ,\x16'; break;
     case 16: command[4]= ,\x10'; break;
     case 12: command[4]= ,\x0C'; break; }
  otworz_rs232(nr_com);
  for(int x=0; x<5; x++) {
     WriteFile( hCom, &command[x], 1, &RS_ile, 0); }
  RS_buf=0;
  for(int x=0; x<2; x++){
     wynik=ReadFile( hCom, &RS_buf, 1, &RS_ile, 0);
     if (wynik)
     odp=odp + IntToStr(RS_buf);
     if( odp=="7975" )
     ShowMessage("Zmiany zostały zapisane!");
     else
     ShowMessage(„Błąd zapisu!");
  CloseHandle(hCom);
  BtUstaw->Enabled=true;
  BtZapisz->Visible=false;
  LPredkosc->Visible=false;
  LPort->Visible=false;
  cmbPorty->Visible=false;
  cmbPredkosc->Visible=false; }
void __fastcall TForm1::Inicjalizacja_zmiennych(TObject *Sender) {
nr_com="COM1";}
void __fastcall TForm1::BtOdbiorClick(TObject *Sender) {
  BtZapisz->Visible=false;
```

```
  LPredkosc->Visible=false;
  LPort->Visible=false;
  cmbPorty->Visible=false;
  cmbPredkosc->Visible=false;
  BtWczytaj->Visible=false;
  LWielkoscPliku->Visible=true;
  LWielkoscPliku->Caption="Ilosc bajtow :";
  txtWielkosc->Visible=true;
  BtMiejsce->Visible=true;
  BtUstaw->Visible=true;
  BtNasluch->Visible=true;
  BtZgodnosc->Visible=true;
  BtWyslij->Visible=false; }

void __fastcall TForm1::BtMiejsceClick(TObject *Sender) {
if(DialSavePlik->Execute()){
  nazwa_pliku=DialSavePlik->FileName;
  txtInfo->Text = nazwa_pliku; }}
void __fastcall TForm1::BtNasluchClick(TObject *Sender) {
  BOOL wynik;
  DWORD RS_ile,RS_buf=0;
  FILE * plik;
  int i=1;
  plik=fopen(nazwa_pliku.c_str(),"wb");
  otworz_rs232(nr_com);
  if(wielkosc_pliku==0)
    if(txtWielkosc->Text=="")
    ShowMessage("Wpisz ilosc bajtów do odczytu!");
    else
    wielkosc_pliku = StrToInt (txtWielkosc->Text);
  PasekPostepu->Visible=true;
  PasekPostepu->Min=0;
  PasekPostepu->Max=wielkosc_pliku;

  while(wielkosc_pliku>i){
    RS_buf=0;
    wynik = ReadFile( hCom, &RS_buf, 1, &RS_ile, 0);
    i++;
    fputc(RS_buf,plik);
```

```
    PasekPostepu->StepBy(1); }
  fclose(plik);
  PasekPostepu->Visible=false; }
void __fastcall TForm1::BtNadawajClick(TObject *Sender) {
  BtZapisz->Visible=false;
  BtUstaw->Visible=true;
  BtWczytaj->Visible=true;
  BtMiejsce->Visible=false;
  BtZgodnosc->Visible=false;
  BtNasluch->Visible=false;}
void __fastcall TForm1::wyswietl_nasuch(TObject *Sender) {
  if( txtWielkosc->Text == „”)
  BtNasluch->Visible=false;
  else
    BtNasluch->Visible=true; }
void __fastcall TForm1::BtZgodnoscClick(TObject *Sender)
{
  if( nazwa_pliku ==””){
  ShowMessage(“Wybierz plik do testu”);}
  else  {
  FILE * plik;
  FILE * plik2;
  bool wykonaj=true;
  int znak=0,znak2=0;
  int i=0,bledy=0;
  float procent=0.00;
  if((plik = fopen(„c:\\oryginal.txt”, „rb”))==NULL)  {
    ShowMessage(“Plik nie istnieje”);
    wykonaj=false; }

  if((plik2=fopen(nazwa_pliku.c_str(),”rb”))==NULL) {
  ShowMessage(“Brakuje oryginalnego pliku do porównania”);
  wykonaj=false; }
  if(wykonaj) {
  while((znak=fgetc(plik))!=EOF){
  if(i==0)
  znak2=fgetc(plik2);
    if(znak==znak2)
    znak2=fgetc(plik2);
```

```
        else
        bledy++;
    i++; }
    procent= (i-bledy)*100;
    if(procent==0){
    ShowMessage(FloatToStr(procent) + „% zgodnosci.”); }
    else{
    procent=(float)procent/i;
    ShowMessage(FloatToStr(procent) + „% zgodnosci.”); }
    }}}
  void __fastcall TForm1::BtAnulujClick(TObject *Sender)
  {   BtUstaw->Enabled=true;
    BtZapisz->Visible=false;
    LPredkosc->Visible=false;
    LPort->Visible=false;
    cmbPorty->Visible=false;
    cmbPredkosc->Visible=false;
    BtAnuluj->Visible=false; }
```

Literatura

1. http://www.ftdichip.com.
2. http://www.kt.agh.edu.pl.
3. Dragajew W., Komputerowy system wymiany informacji dla robotów transportowych w elastycznym systemie automatyki transportowej. Metody i systemy komputerowe w automatyce i elektrotechnice. Monografia, Częstochowa 2005.
4. Mochnacki W. Kody korekcyjne i kryptografia, Wrocław 2000.