(Be Creative) Protocol

Jordan Marling	- A00845629
Damien Sathanielle	- A00851340
Cory Thomas	- A00817721
Joshua Campbell	- A00815859

Table of Contents

Introduction

This application will send files between two wireless modems. The application will be using the (Be Creative) Protocol to send data. This will mean this program will be able to send and recieve files from other programs using this protocol too. The packets used in this program will be checked with CRC 16 and will be fully event driven.

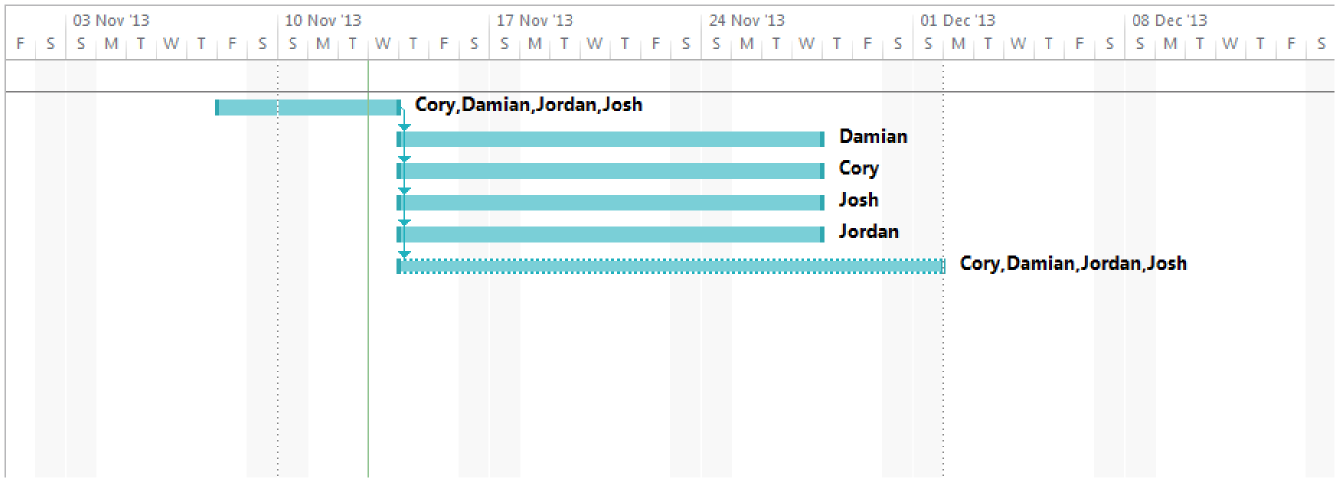
Statistics will be stored as packets are sent/recieved.

The statistics are:

- Protocol efficiency
- Effective bits per second
- Number of packets sent/recieved
- Number of ACKs sent/recieved
- Number of NAKs sent/recieved
- Number of ENQs sent/recieved
- Bits sent/recieved
- Bit error rate
- Response time (lag)
- Total file transfer time
- Number of timeouts
- Average amount of padding in each packet
- Packets sent/recieved per second

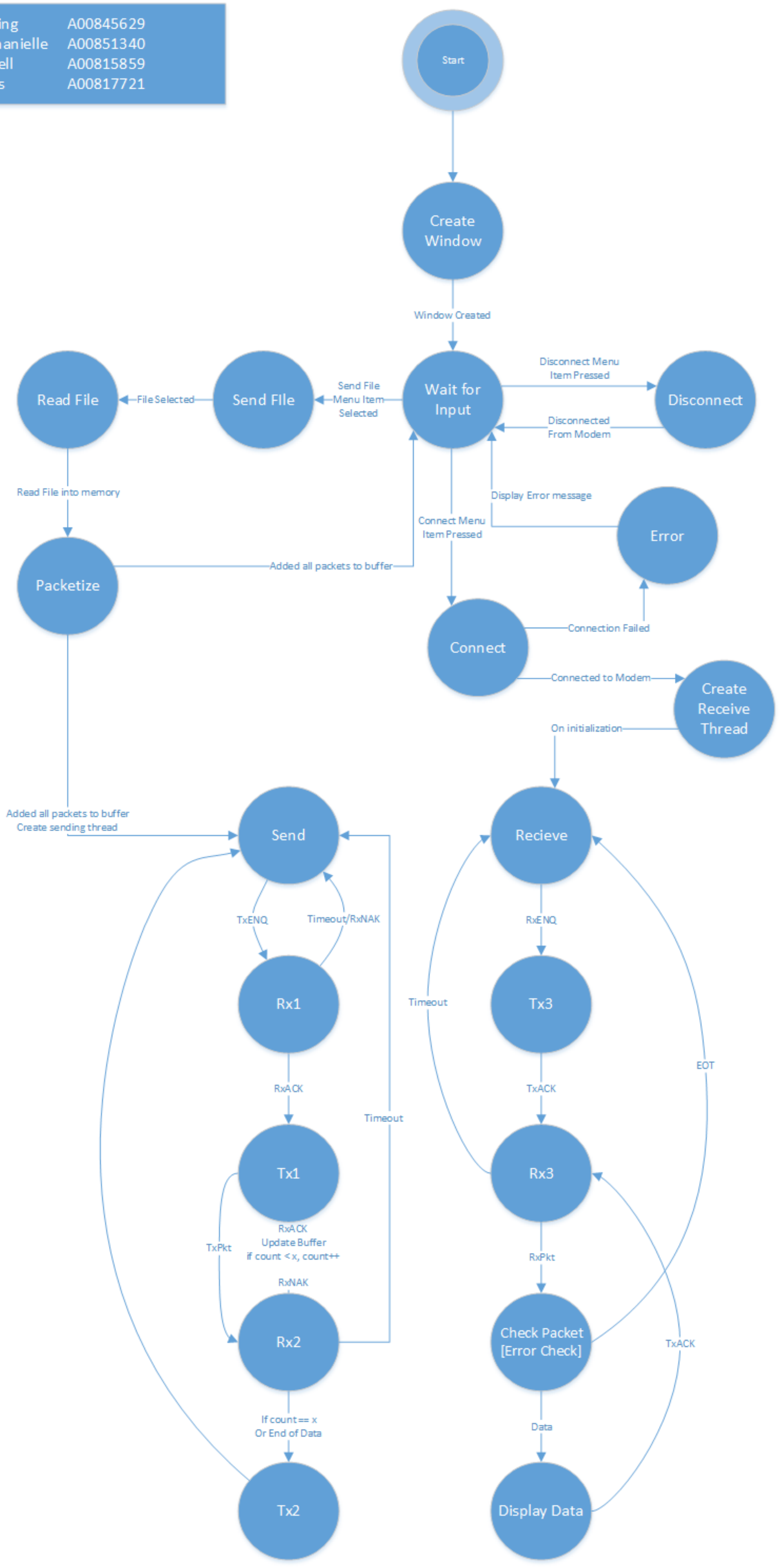
Design

Gantt Chart



State Transition Diagram

Jordan Marling	A00845629
Damien Sathanielle	A00851340
Josh Campbell	A00815859
Cory Thomas	A00817721



Start
This state exists when the application is initially started. It immediately goes into the “Create Window” state.

Create Window
This state creates a win32 window. If it is successful it waits for user input.

Wait For Input
This state waits for the user to either send a file, change COM port settings, disconnect or connect.

Send File
This state opens a file specified by a user and passes it into the Read File state.

ReadFile
This state reads the file into data chunks and passes each data chunk into the Packetize state to be packetized.

Packetize
This state adds a SYN char and a DC1/DC2 char to the beginning of the data chunk. It calcualtes the CRC value and appends this as a 2 character value at the end of the packet. When the packet is created it is put into a buffer. If the program isnt already sending packets, a Send thread is created to send the packets until all packets are sent successfully.

Disconnect
This state disconnects from the COM port.

Connect
This state establishes a connection with the COM port with the desired settings. If there is an error, it goes into the Error state. If it is successful, it creates a recieve thread.

Error
This state is triggered by a connection error. It displays an error message and then goes into the Wait for Input state.

Create Receive Thread
This state is created after a successful connection to the COM port and listens for packets.

Send
In the send state, there will be packets to send. This state will send an ENQ packet to get the line and move over to Rx1 state.

Recieve
In the reiceve state, the application will wait for an ENQ packet, if there is one then the program will move into the Tx3 state.

RX1
The first stage of the Transmit state where the wireless device will wait to receive an acknowledgement to begin transmitting data to another device. If there is a timeout, the application will return to an Idle state. If there is a bad packet or error check fails a NAK packet will be sent and the program will return to the Idle state. An ACK packet will be sent if error checking passes and we recieve an ENQ packet.

TX1
After having received the acknowledgement, the device will begin to transmit data over the wireless network. This state transmits the packet of data to the other devices. When it does this the state moves into Rx2.

RX2
The device will receive acknowledgement. If the count is less than the number of packets that are to be sent, an RxAck will be sent to TX1 to tell the transmitter to keep sending packets. If the count is equal to the total that is to be sent or there is an end of data control character, the state will move to the TX2 state.

TX2
Final state for the transmitter. This state will be received only when the total number of packets has been received or that the TxEoD control character has been received signalling end of data. This state transmits an EOT packet to the other device and goes to the Send state.

TX3
The first state of the receive mode. Upon receiving the RxEnq, it will signal the wireless device to begin receive mode and notifies the transmitter that it can begin to accept data. An ACK packet is sent to the other device and the state transfers into Rx3.

RX3
After having received acknowledgement to receive data, the receiver will begin to receive information from the Transmitter. If this state times out, the state goes back into Recieve. If a data packet is recieved the application goes into the Check Packet [Error Cbeck] state.

Check Packet/ErrorCheck

Uses a form of CRC-32 to check for errors in the received frame. If the data is an EOT packet, go back into the Recieve state. If the packet is a data packet, go into the Display Data state. If the packet has an error in it, a NAK packet will be sent and go back into the Rx3 state.

Display Data
This state will have received the frame, displays the data contents, and will send out an ACK packet to accept more packets, transfering into the Recieve state.

}

Pseudo-Code

```
WaitForInput() {
    Display Menu (ChooseComPort, OpenConnection, CloseConnection, OpenFile)
}

Connect() {
    Open connection to wireless transmitter/receiver
    if connection is valid and File is open
        send message that the connection was successfully established
        create reading thread
    else
        throw failure to open connection exception
}

Disconnect() {
    if connection exists
        Disconnect from wireless transmitter/receiver
        display message saying successfully disconnected
}

SendFile() {
    FILE = openFile for reading
    if FILE exists and Connection is established
        SendFile()
    else
        throw file not found exception
}

DisplayData() {
    calculate different statistics
    display statistics to screen
    if checkPacketType equals datapacket
        display packet data to FileContent section of application
        write packet data to file
    receive next packet
}


Send() {
  create semaphore
  loop and check the buffer for packets
  if there are packets
      send ENQ
    Rx1()
}
```

```
}

Rx1() {
   while packets sent is less than 5 and buffer is not empty
         if timed out, go back to Send
      transmit packet with Tx1()
      increase packets sent and remove packet from buffer
   sendControlChar(EOT)
   release semaphore
}

Tx1() {
         send packet
}

Rx2() {
   if we have sent less than 5 packets, go back to Tx1() and send more.
   if we have timed out, go back to Send and ENQ the line.
}

Tx2() {
   send EOT
   go back to Send
}

Receive() {
   Wait for other side to ENQ
   Tx3()
}

Tx3() {
   Ack the ENQ
   Rx3()
}

Rx3() {
   If Recieve packet, CheckPacket()
   If timeout, go to Recieve()
}

ReadFile() {
         Loop through file contents
                  Every 1020 characters Packetize() data.

         If less than 1020 characters in buffer
                  Pad packet with NUL characters
                  Packetize() data
}
```

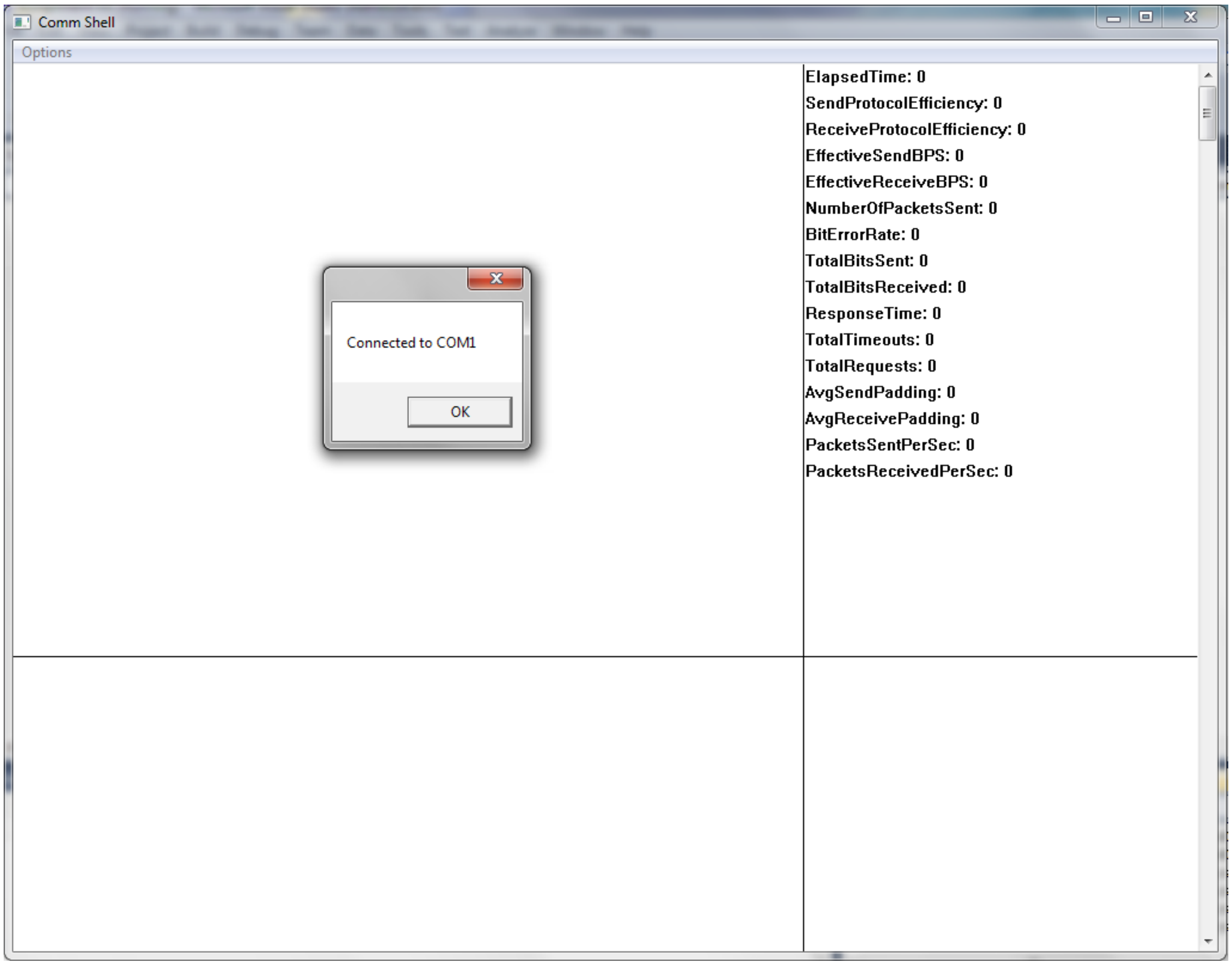
```
Packetize() {
        Create character buffer of 1024 length.
        Put SYN into first character.
        Alternate DC1 and DC2 into the packet.
        Calculate the CRC value of the data
        Append CRC value to the last 2 characters in the buffer.
        Put packet in buffer.
        Create Send thread.
}

CheckPacket() {
        Check if first character is SYN
        If second character is DC1/DC2 and different from the last packet.
                Get CRC value of 1020 characters of data
                Compare the calculated CRC value with the last two characters
                                                        (appended CRC characters)
                DisplayData()
        If packet is EOT
                Receive()
}
```

Testing

Test #	Description	Tools Used	Expected Result	Pass/Fail
1	Connect to COM port	becreative	Com port opened.	Pass. See Fig. 1
2	Received ENQ from other computer	becreative Visual Studio	Received the ENQ	Pass See Fig. 2
3	Received ACK from other computer	becreative Visual Studio	Received the ACK	Pass See Fig. 3
4	Received NAK from other computer	becreative Visual Studio	Received the NAK	Pass See Fig. 4
5	File sent is the same as the File received.	becreative Notepad	The files are the same	Pass See Fig. 5 & 6
6	Data is packetized properly	becreative GHex	The data is put into a packet	Pass See Fig. 7 & 8

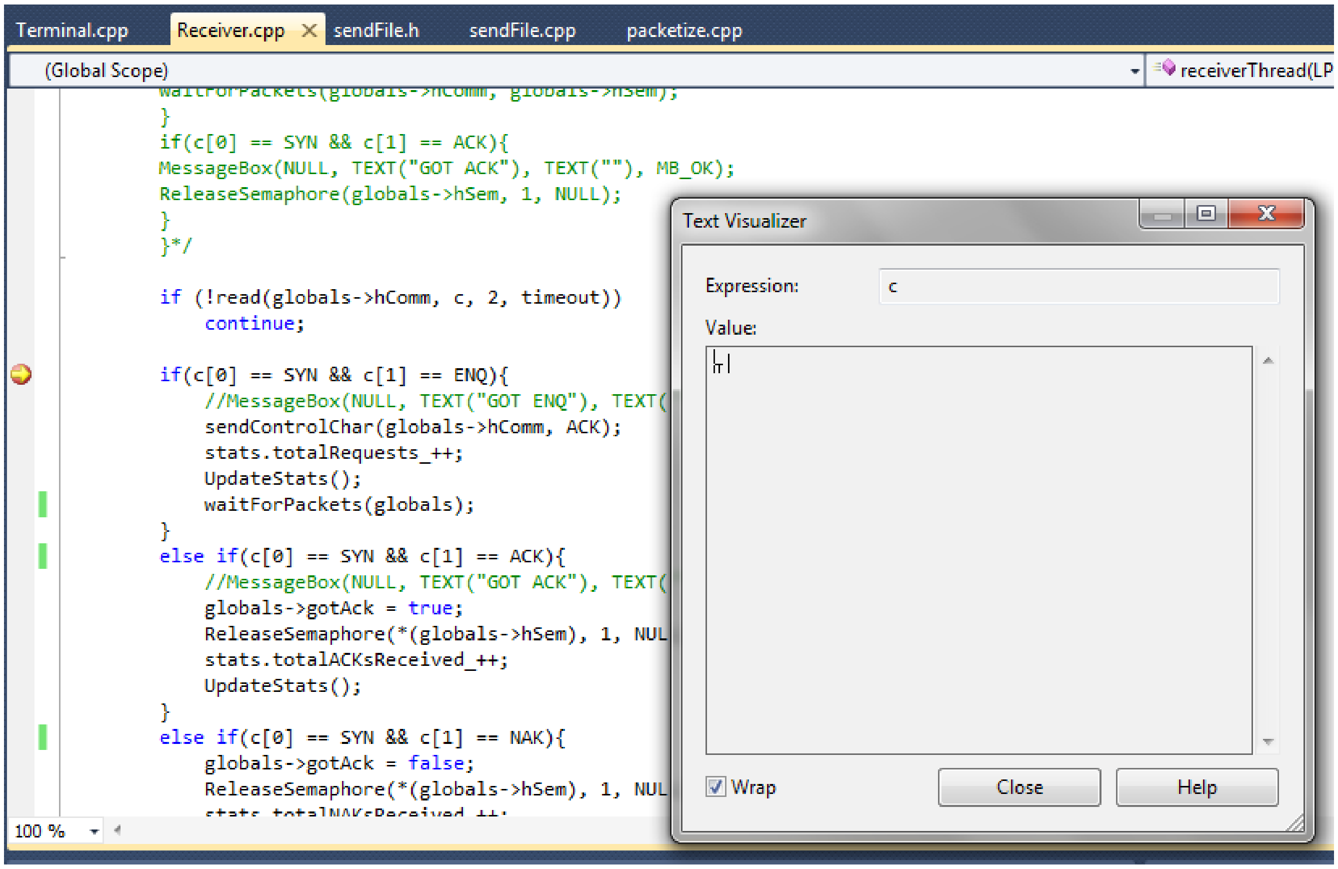
Fig. 1



Connected to COM1

```
ElapsedTime: 0
SendProtocolEfficiency: 0
ReceiveProtocolEfficiency: 0
EffectiveSendBPS: 0
EffectiveReceiveBPS: 0
NumberOfPacketsSent: 0
BitErrorRate: 0
TotalBitsSent: 0
TotalBitsReceived: 0
ResponseTime: 0
TotalTimeouts: 0
TotalRequests: 0
AvgSendPadding: 0
AvgReceivePadding: 0
PacketsSentPerSec: 0
PacketsReceivedPerSec: 0
```

Fig. 2



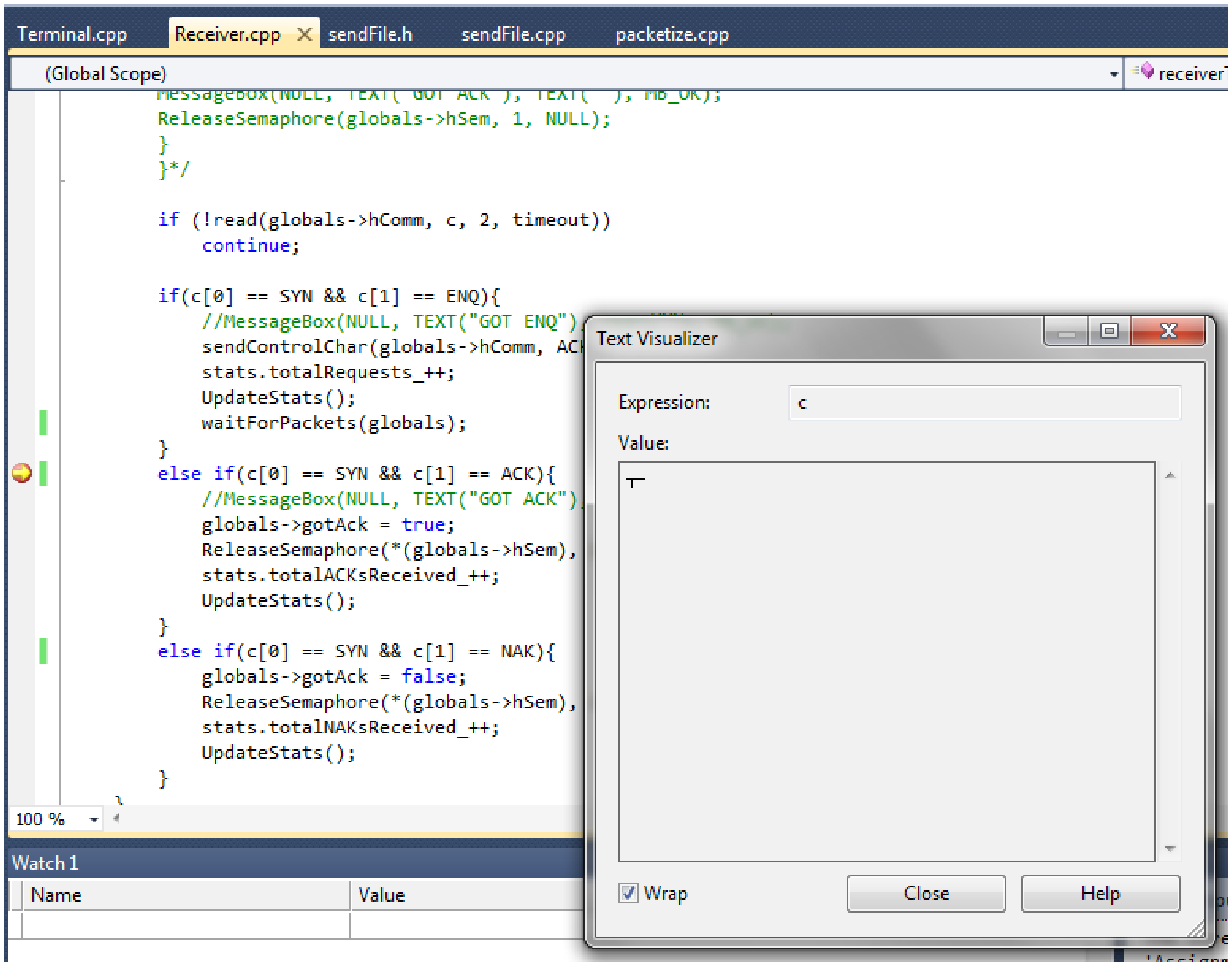
```
            waitForPackets(globals->hComm, globals->hSem);
        }
        if(c[0] == SYN && c[1] == ACK){
        MessageBox(NULL, TEXT("GOT ACK"), TEXT(""), MB_OK);
        ReleaseSemaphore(globals->hSem, 1, NULL);
        }
        }*/

        if (!read(globals->hComm, c, 2, timeout))
            continue;

        if(c[0] == SYN && c[1] == ENQ){
            //MessageBox(NULL, TEXT("GOT ENQ"), TEXT(
            sendControlChar(globals->hComm, ACK);
            stats.totalRequests_++;
            UpdateStats();
            waitForPackets(globals);
        }
        else if(c[0] == SYN && c[1] == ACK){
            //MessageBox(NULL, TEXT("GOT ACK"), TEXT(
            globals->gotAck = true;
            ReleaseSemaphore(*(globals->hSem), 1, NUL
            stats.totalACKsReceived_++;
            UpdateStats();
        }
        else if(c[0] == SYN && c[1] == NAK){
            globals->gotAck = false;
            ReleaseSemaphore(*(globals->hSem), 1, NUL
            stats.totalNAKsReceived_++;
```

Characters are 0x16 and 0x05

Fig. 3



```
        MessageBox(NULL, TEXT("GOT ACK"), TEXT(""), MB_OK);
        ReleaseSemaphore(globals->hSem, 1, NULL);
        }
        }*/

        if (!read(globals->hComm, c, 2, timeout))
            continue;

        if(c[0] == SYN && c[1] == ENQ){
            //MessageBox(NULL, TEXT("GOT ENQ")
            sendControlChar(globals->hComm, ACK
            stats.totalRequests_++;
            UpdateStats();
            waitForPackets(globals);
        }
        else if(c[0] == SYN && c[1] == ACK){
            //MessageBox(NULL, TEXT("GOT ACK")
            globals->gotAck = true;
            ReleaseSemaphore(*(globals->hSem),
            stats.totalACKsReceived_++;
            UpdateStats();
        }
        else if(c[0] == SYN && c[1] == NAK){
            globals->gotAck = false;
            ReleaseSemaphore(*(globals->hSem),
            stats.totalNAKsReceived_++;
            UpdateStats();
        }
```

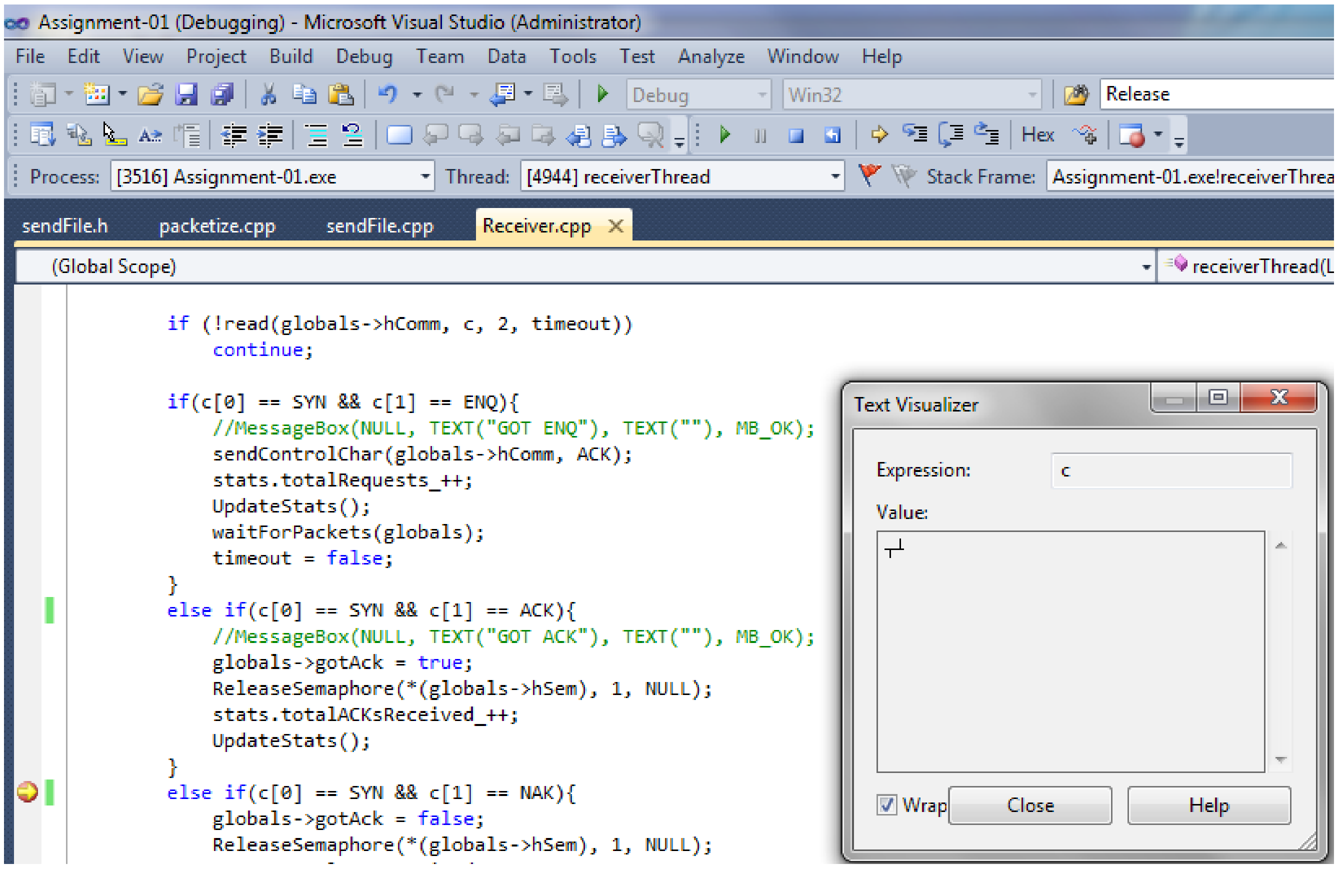
Expression: c

Value:

```
┬
```

Characters are 0x16 and 0x06

Fig. 4

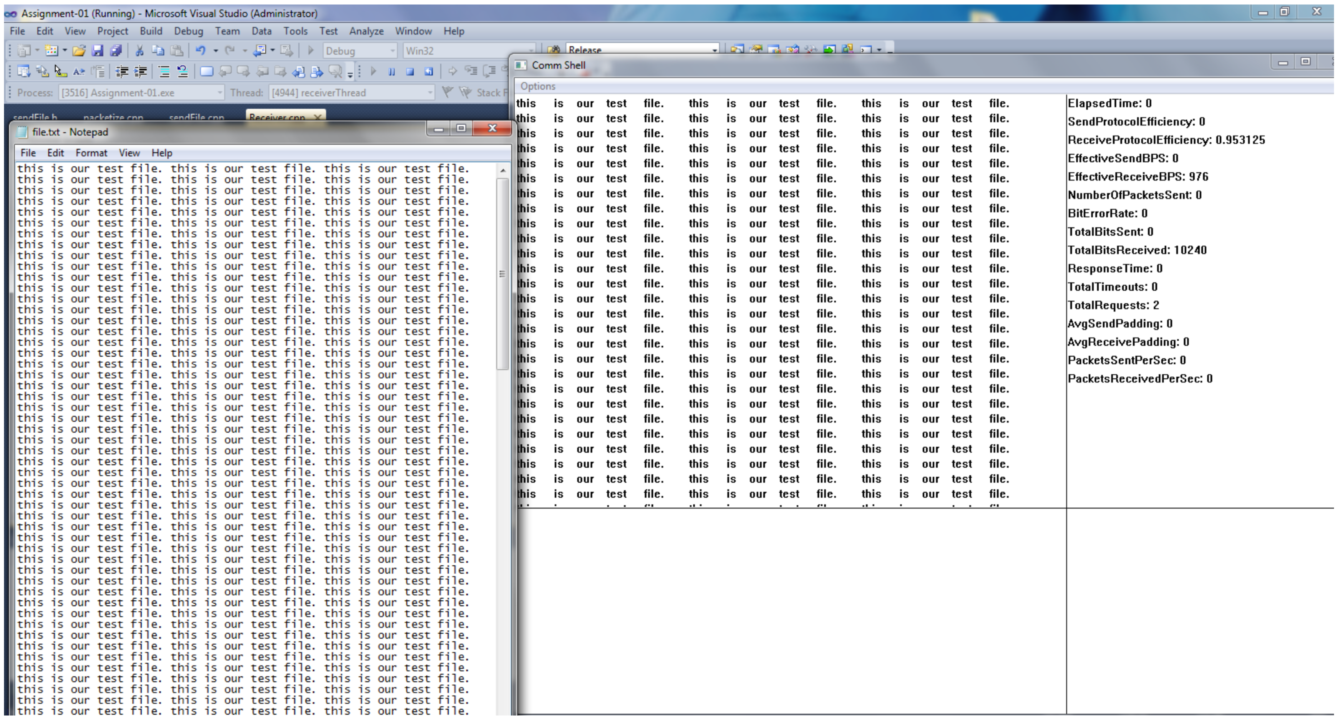

Assignment-01 (Debugging) - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team Data Tools Test Analyze Window Help
Debug
Win32
Release
Hex
Process: [3516] Assignment-01.exe
Thread: [4944] receiverThread
Stack Frame: Assignment-01.exe!receiverThrea
sendFile.h
packetize.cpp
sendFile.cpp
Receiver.cpp
(Global Scope)
receiverThread(L
Text Visualizer
Expression: c
Value:
Wrap
Close
Help

```
        if (!read(globals->hComm, c, 2, timeout))
            continue;

        if(c[0] == SYN && c[1] == ENQ){
            //MessageBox(NULL, TEXT("GOT ENQ"), TEXT(""), MB_OK);
            sendControlChar(globals->hComm, ACK);
            stats.totalRequests_++;
            UpdateStats();
            waitForPackets(globals);
            timeout = false;
        }
        else if(c[0] == SYN && c[1] == ACK){
            //MessageBox(NULL, TEXT("GOT ACK"), TEXT(""), MB_OK);
            globals->gotAck = true;
            ReleaseSemaphore(*(globals->hSem), 1, NULL);
            stats.totalACKsReceived_++;
            UpdateStats();
        }
        else if(c[0] == SYN && c[1] == NAK){
            globals->gotAck = false;
            ReleaseSemaphore(*(globals->hSem), 1, NULL);
```

Characters are 0x16 and 0x15

Fig. 5


Assignment-01 (Running) - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team Data Tools Test Analyze Window Help
Debug
Win32
Release
Process: [3516] Assignment-01.exe
Thread: [4944] receiverThread
sendFile.h
packetize.cpp
sendFile.cpp
Receiver.cpp
file.txt - Notepad
File Edit Format View Help
Comm Shell
Options

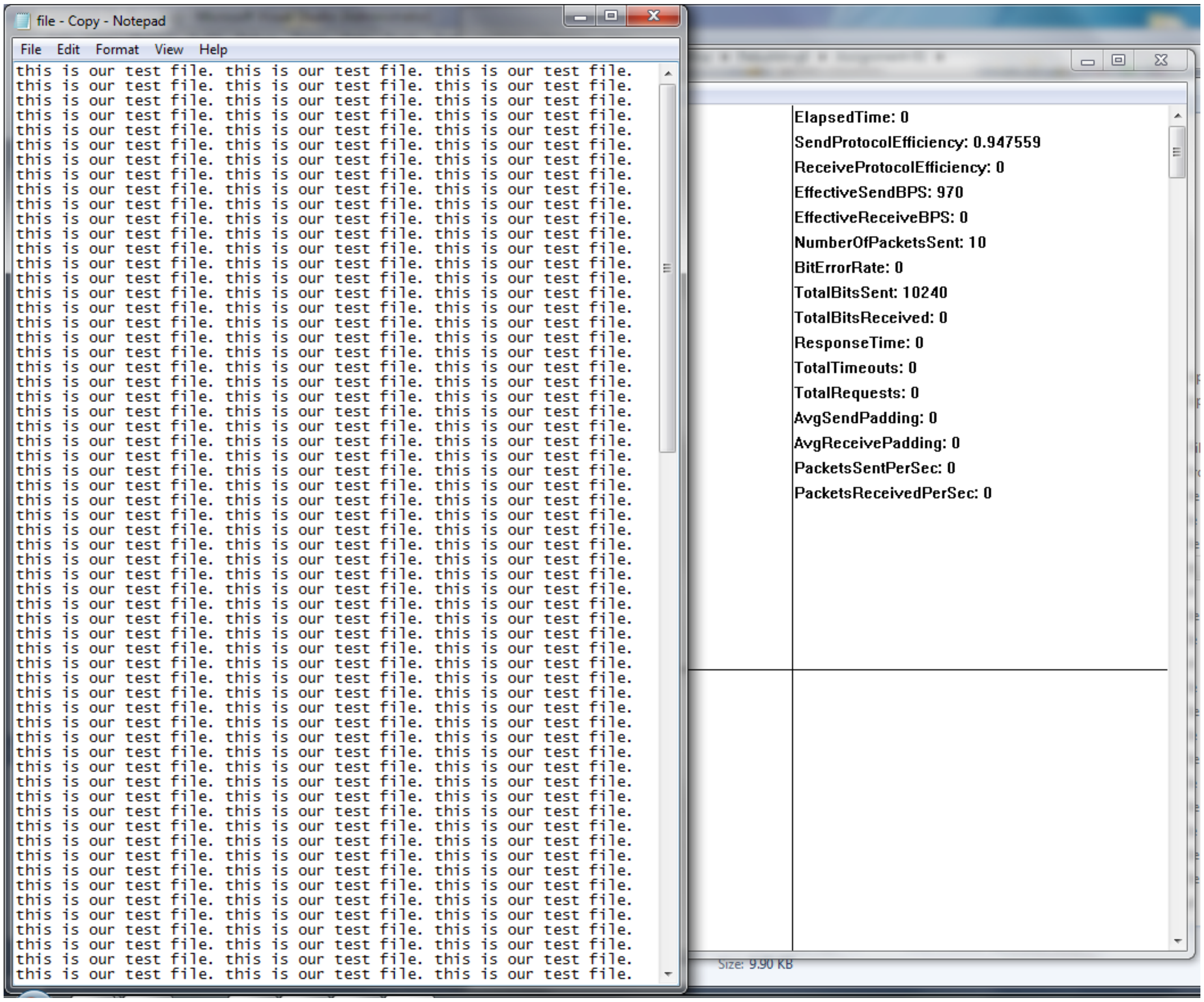
```
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file. this is our test file.
```

```
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
this   is   our   test   file.    this   is   our   test   file.    this   is   our   test   file.
```

```
ElapsedTime: 0
SendProtocolEfficiency: 0
ReceiveProtocolEfficiency: 0.953125
EffectiveSendBPS: 0
EffectiveReceiveBPS: 976
NumberOfPacketsSent: 0
BitErrorRate: 0
TotalBitsSent: 0
TotalBitsReceived: 10240
ResponseTime: 0
TotalTimeouts: 0
TotalRequests: 2
AvgSendPadding: 0
AvgReceivePadding: 0
PacketsSentPerSec: 0
PacketsReceivedPerSec: 0
```

File received.

Fig. 6

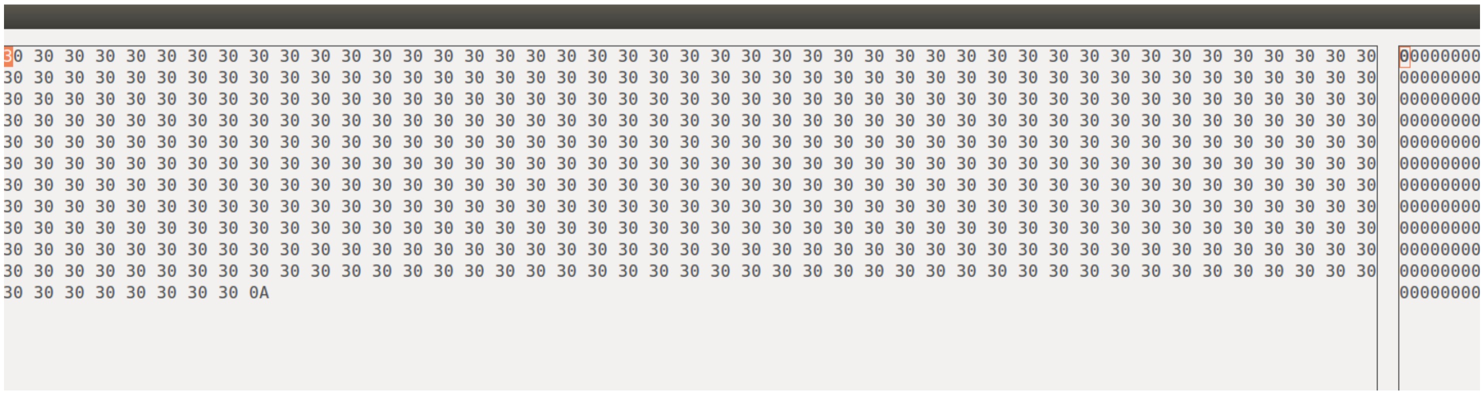


```
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
this is our test file. this is our test file. this is our test file.
```

```
ElapsedTime: 0
SendProtocolEfficiency: 0.947559
ReceiveProtocolEfficiency: 0
EffectiveSendBPS: 970
EffectiveReceiveBPS: 0
NumberOfPacketsSent: 10
BitErrorRate: 0
TotalBitsSent: 10240
TotalBitsReceived: 0
ResponseTime: 0
TotalTimeouts: 0
TotalRequests: 0
AvgSendPadding: 0
AvgReceivePadding: 0
PacketsSentPerSec: 0
PacketsReceivedPerSec: 0
```

File sent.

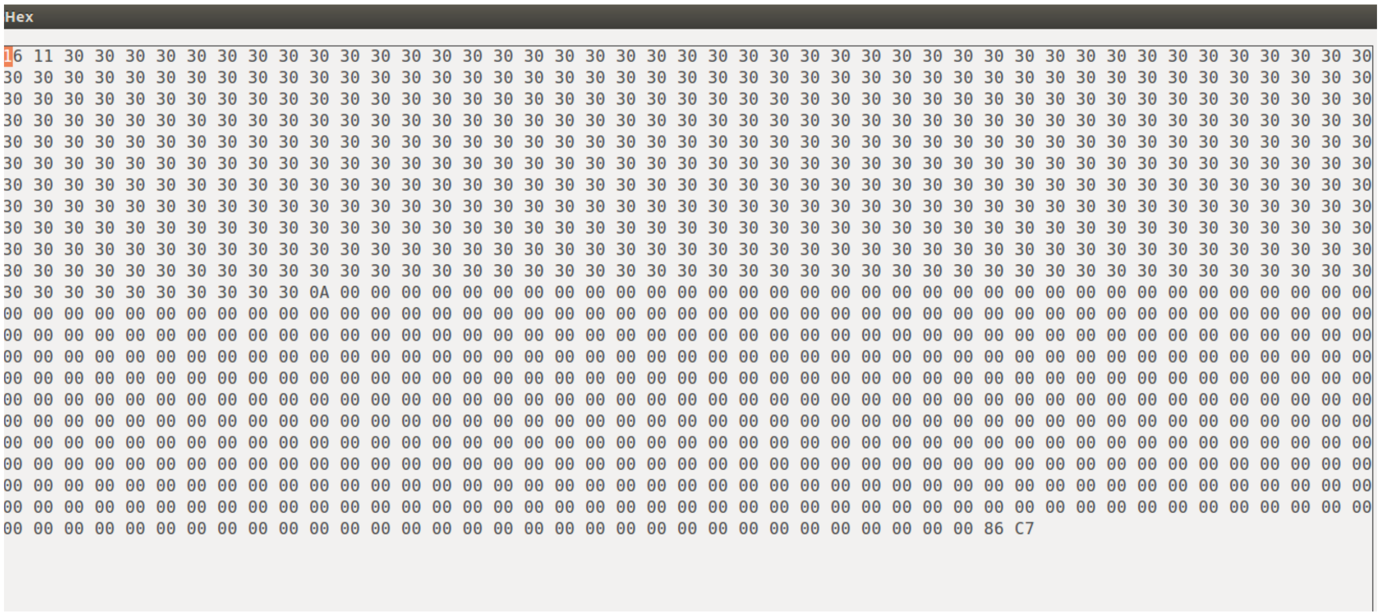
Fig. 7



```
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30  00000000
30 30 30 30 30 30 30 30 0A                                                                                                                 00000000
```

Plain text data

Fig. 8


Hex

```
16 11 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
30 30 30 30 30 30 30 30 30 30 0A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 86 C7
```

Packetized data

Conclusion

This application will send files between two wireless modems. The application will be using the (Be Creative) Protocol to send data. This will mean this program will be able to send and recieve files from other programs using this protocol too. The packets used in this program will be checked with CRC 16 and will be fully event driven.