

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 introductions:

On this project we have three types of PCs. first PC operate with windows operating system , second PC and third PC operate with Linux operating system. For testing the software for protocol converter First we need to execute serial RX.C on PC3, protocol converter on PC2 and then executed UDPFileTx.java on PC1. When executing the Java file on PC1, the file is sent over the Ethernet link to PC2. Then PC2 catches the Ethernet frames, converts them into serial packets and transmits them to PC3. PC3 displays the file which was sent from pc1.

The following figure explain the three PCs connected together , PC1 and PC2 connected over Ethernet link , PC2 and PC3 connected over serial link. Implementation screen for each device also appeared on the figure.



Figure 4.1: implementation of protocol converter

4.2 System Assumption:

4.2.1:program1:UDPFileTx

```
Public static int buffer size=100;
```

```
Public static int server port=999;
```

```
Public static int client port=666;
```

4.3Result and Discussion:

4.3.1: Program one: UDPFileTx

This program takes a filename as input, generate the filename packets and Data packets and sends them over the Ethernet link . effectively using aprotocol similar to the UDP protocol for file transfer.

The figure below explains implementation screen for java program which uses to encapsulate file in Ethernet frame to send it over Ethernet link .

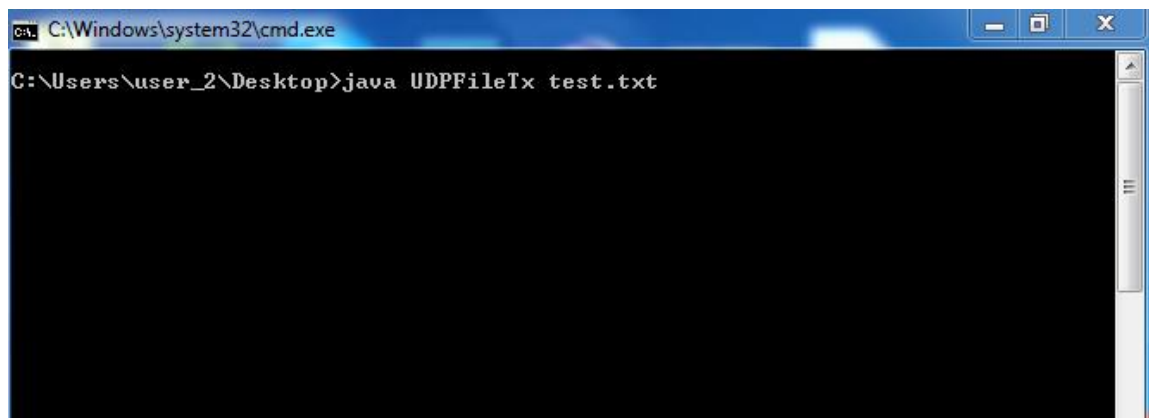
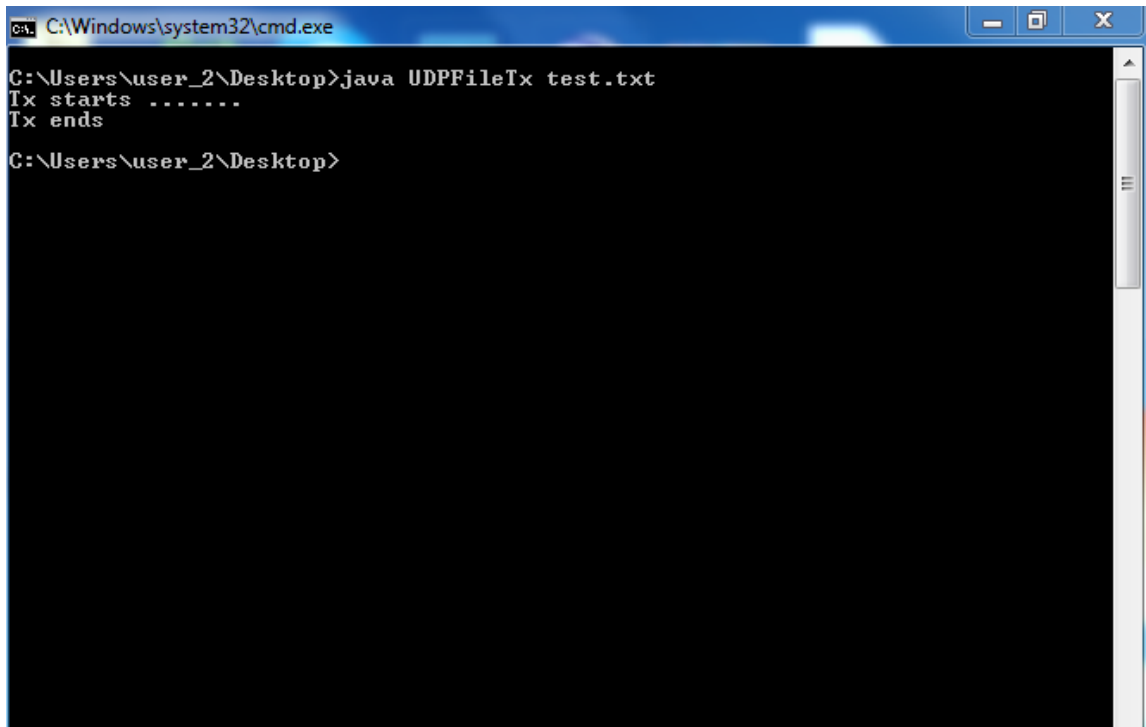


Figure 4.2: before send message

On PC1 first created file which contain of text data to be send on notepad and saved it in desktop , then java program implemented by opened implementation screen "command line" in windows by using the next instruction "java UDPFILETX test.txt".



```
C:\Windows\system32\cmd.exe
C:\Users\user_2\Desktop>java UDPFileTx test.txt
Tx starts .....
Tx ends
C:\Users\user_2\Desktop>
```

Figure4.3: after send message

4.3.2: Program two: protocol converter

This program captured the Ethernet packets , decoded the filename and files' contents, put them into serial communication packets as per the format discussed in chapter3 and then send these packets over serial link.

After serial port opened the program is been ready to execute and terminal screen will be as shown in figure below .

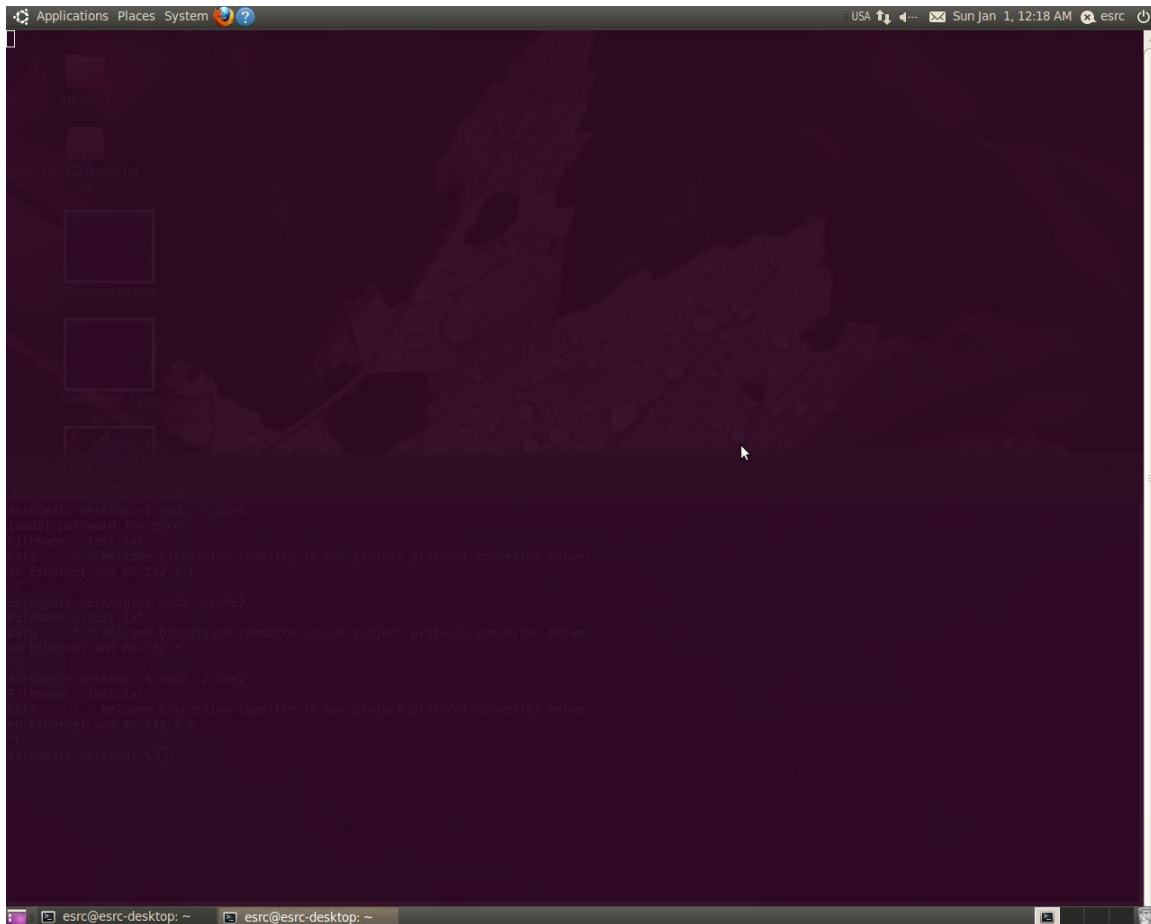


Figure4.5: after open serial port

Before run protocol converter there is no data will be send so the terminal screen is shown as in figure below:

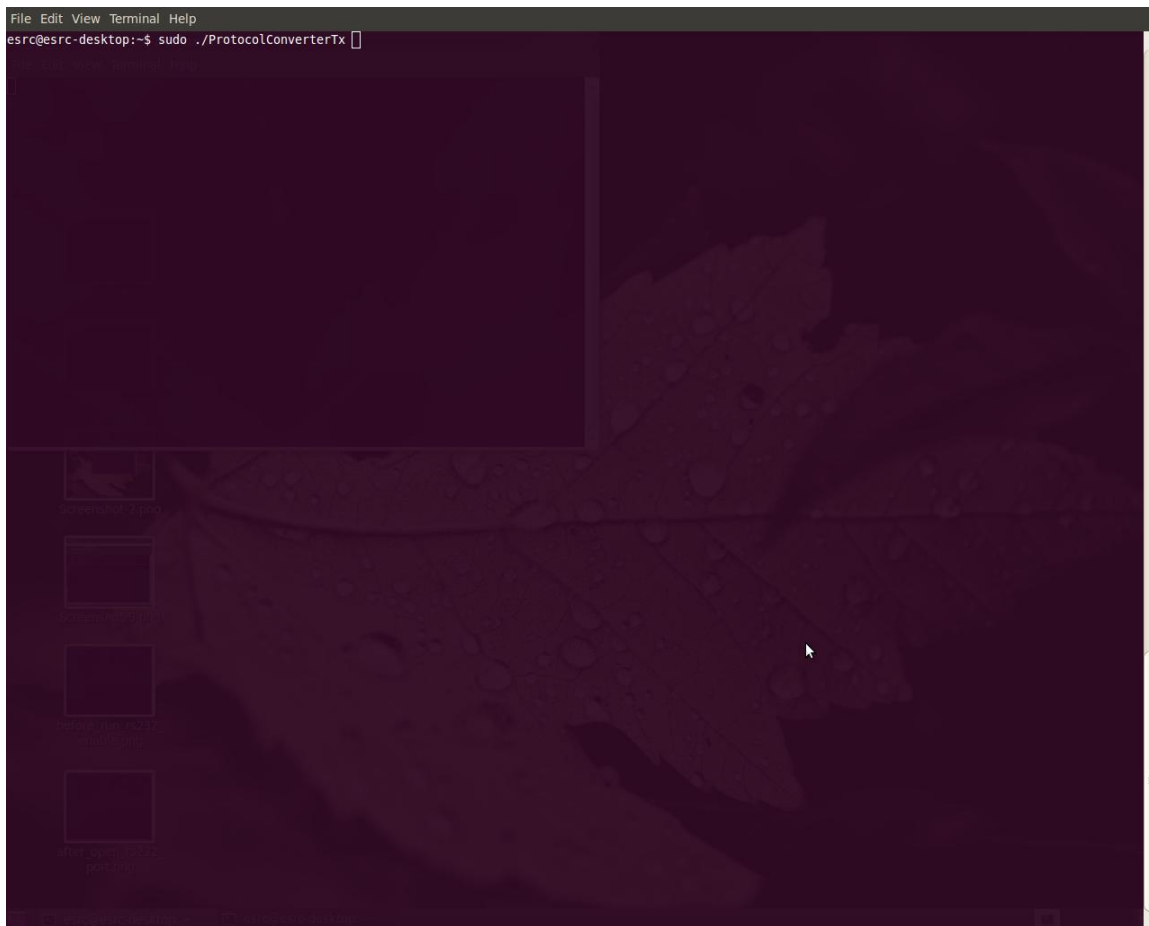


Figure4.6: before run protocol converter

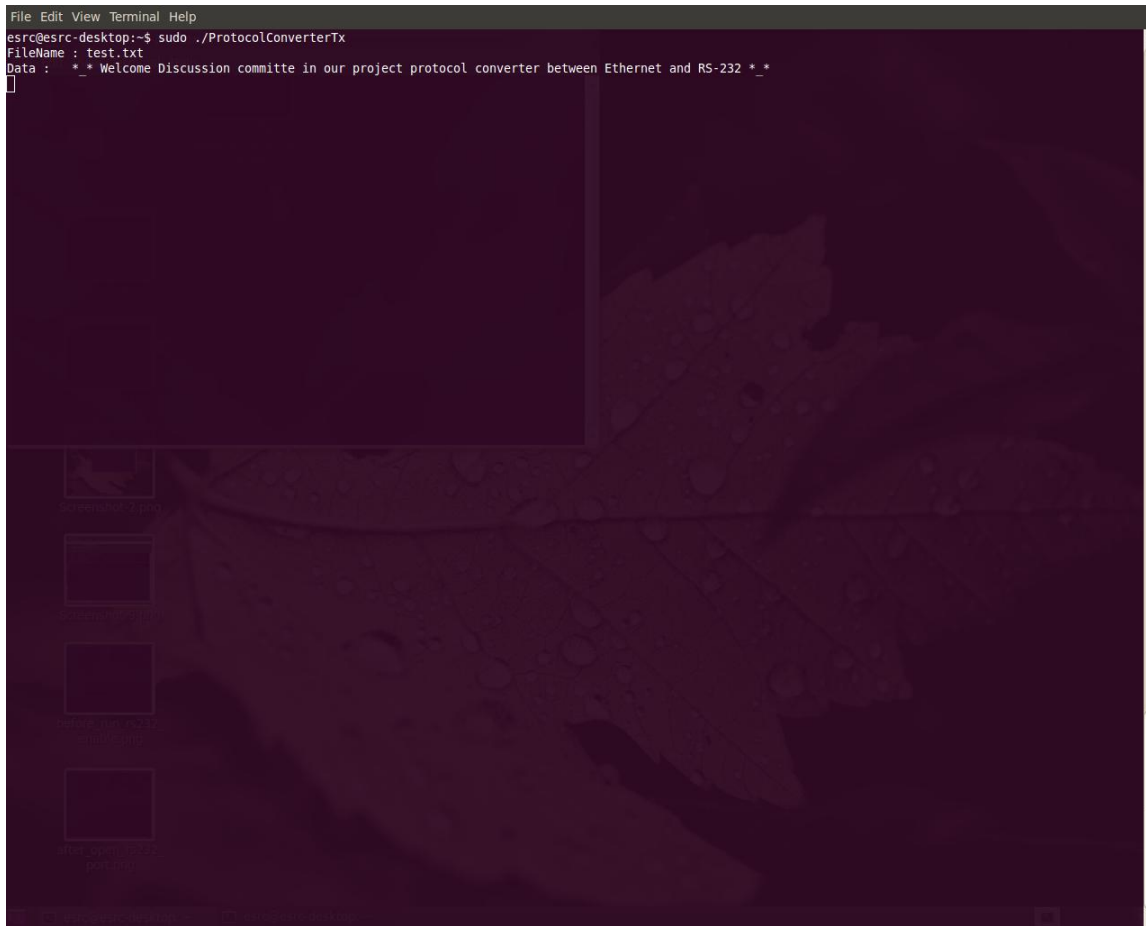


Figure4.7: after run protocol converter

On PC2 c program implemented by using Linux operating system and opened two terminal in Linux system .On first terminal written the instruction "sudo screen /dev/ttys0" to open serial port . On second terminal implemented protocol converter program by written the instruction "sudo ./code1" .

4.3.3: Program three: serialRx

This program received the serial communication data and assembles the file.

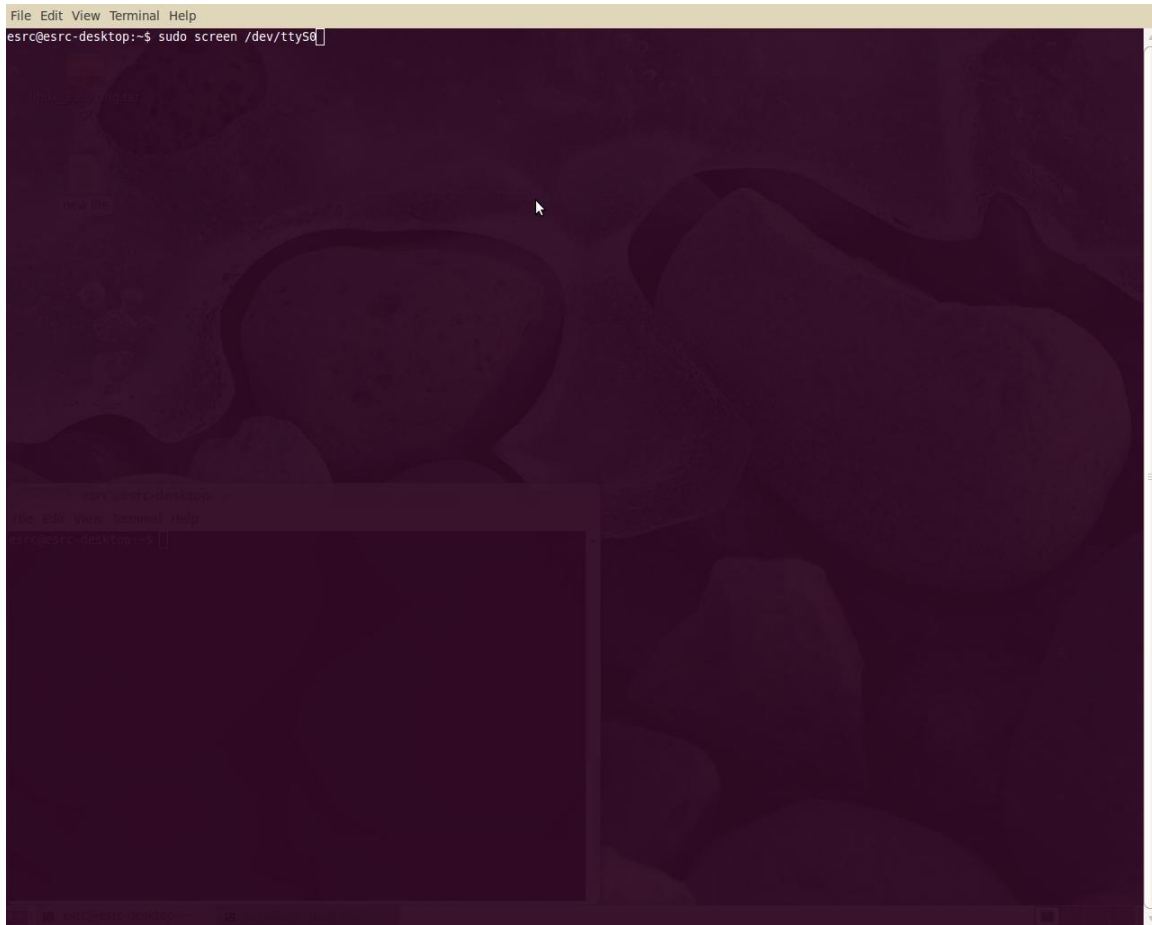


Figure4.8: before open serial port.

Also on PC3 when program executed without opened serial port the data cannot be send so first we need to execute the instruction on other terminal and wait for program execution .

figure below explain terminal screen which used to opened serial port on PC3 and an instruction written over it .

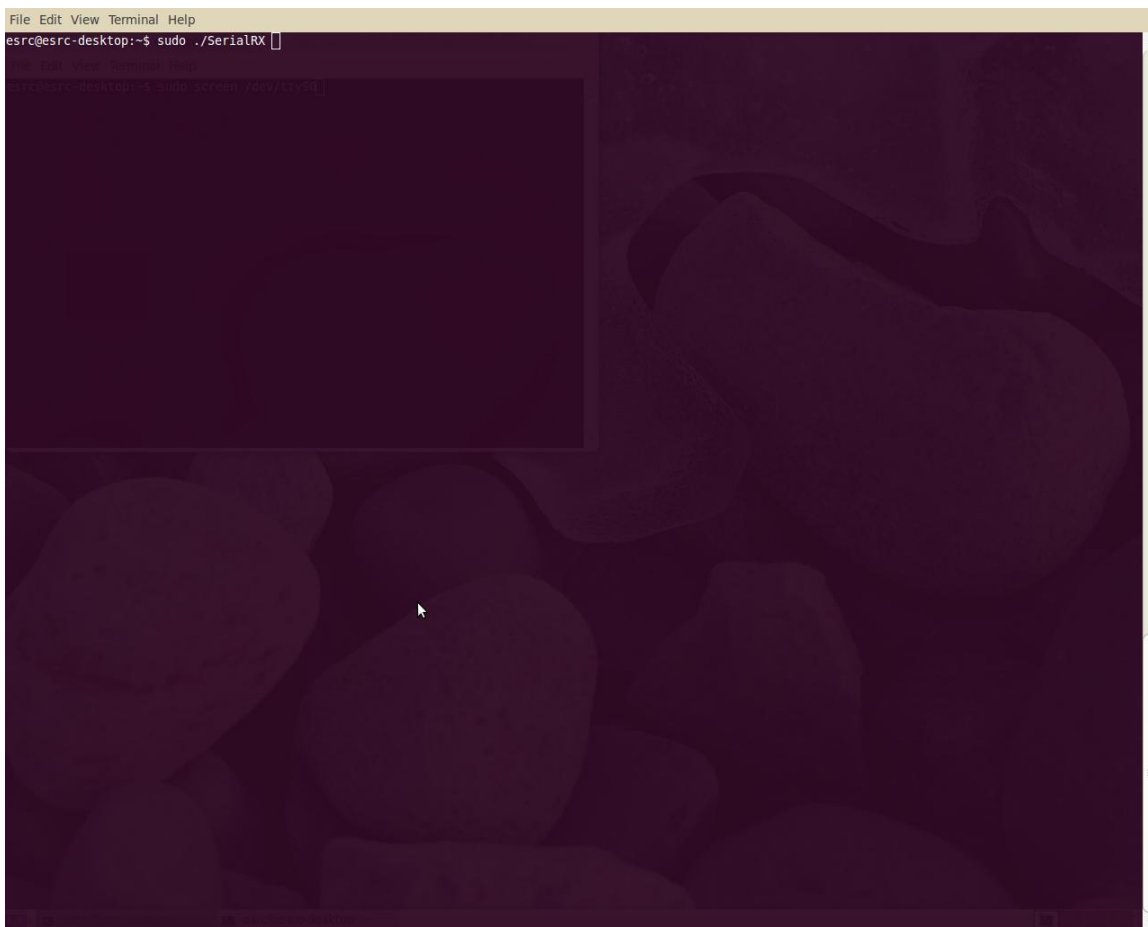


Figure4.9: before run Serial RX.

On pc3 also c program implemented by opened Linux operating system by opened two terminal .on first terminal written "sudo screen /dev/ttyS0" On second terminal written " sudo ./code2".

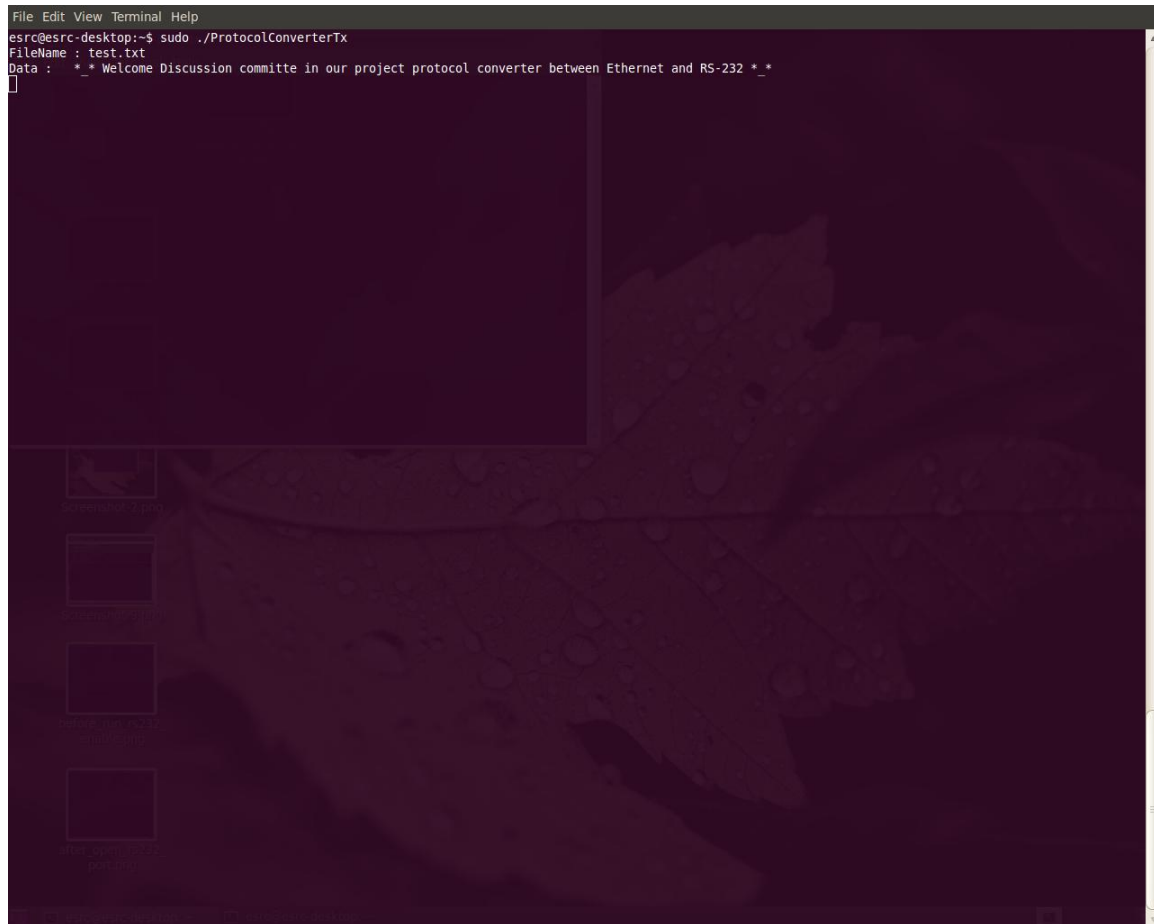


Figure4.10: after run Serial RX.

For testing the software first executed SerialRx.c on PC3, Protocolconverter.c on PC2 and then executed UDPFILETX.java on PC1.