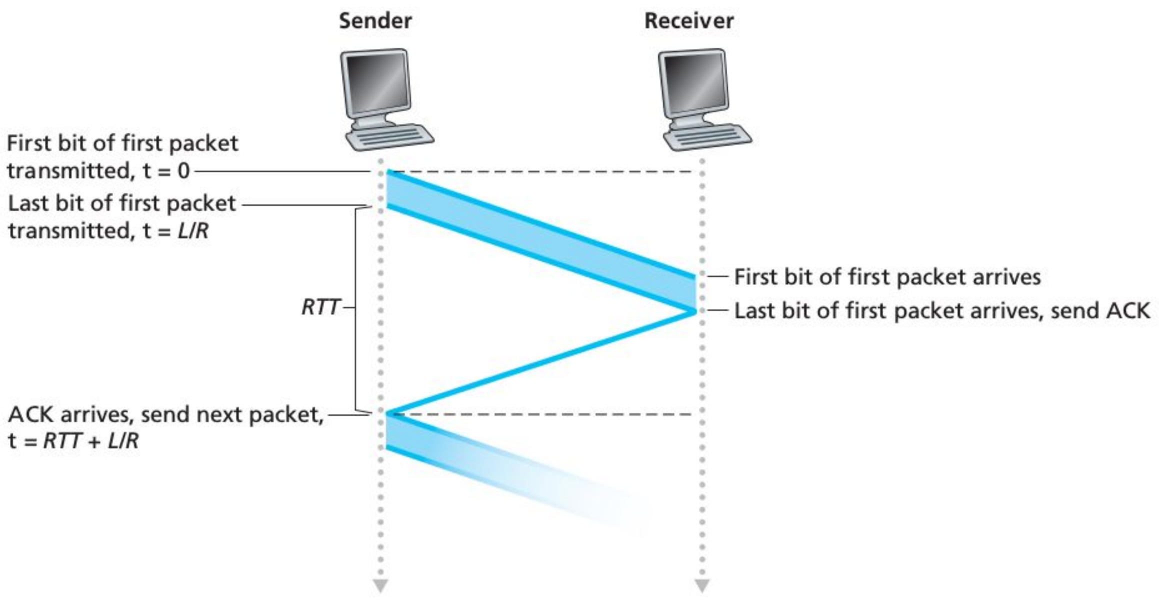
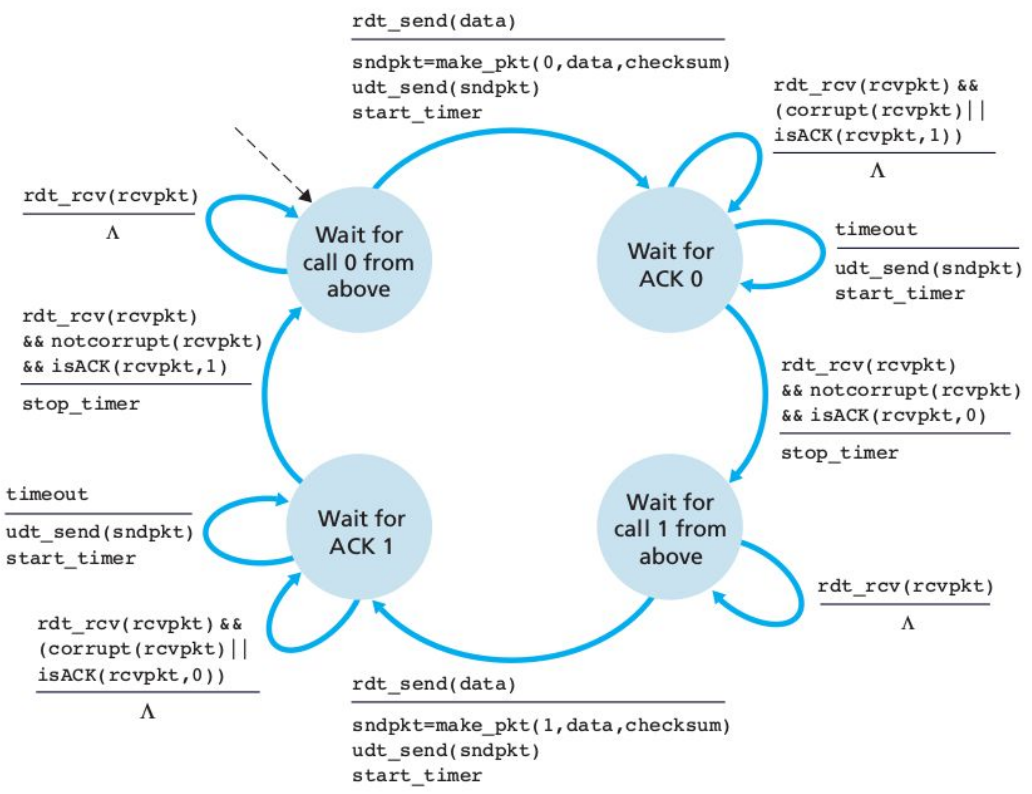
Reliable Data Delivery over UDP

Task1: Stop and Wait Protocol

1. Implement Stop and Wait ARQ mechanism as an application layer protocol in C/C++. Use UDP as an underlying transport protocol for sending the application layer messages.
    1.1. Create appropriate headers for your ARQ protocol such as checksum, Data/ACK, segment size, sequence number etc. You can design the header based on your protocol design and can append before the data. It is a good practice to keep your header fixed size and put into the packet in binary format. You can assume the received packet on either side to be valid/not corrupt by keeping the checksum value dummy.
    1.2. Split the data received from the higher layer protocol into fixed size segments (except the last one) and add the appropriate header before sending.
    1.3. Use timer (fixed time) to resend the packet if the ACK is not received within specific time.
    1.4. Recommended to use the FSM discussed in the class for the implementation.


Sender
Receiver
First bit of first packet transmitted, t = 0
Last bit of first packet transmitted, t = L/R
RTT
ACK arrives, send next packet, t = RTT + L/R
First bit of first packet arrives
Last bit of first packet arrives, send ACK

a. Stop-and-wait operation



rdt_send(data)
sndpkt=make_pkt(0,data,checksum)
udt_send(sndpkt)
start_timer
rdt_rcv(rcvpkt) && (corrupt(rcvpkt)|| isACK(rcvpkt,1))
Λ
rdt_rcv(rcvpkt)
Λ
Wait for call 0 from above
Wait for ACK 0
timeout
udt_send(sndpkt)
start_timer
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt,1)
stop_timer
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt,0)
stop_timer
timeout
udt_send(sndpkt)
start_timer
Wait for ACK 1
Wait for call 1 from above
rdt_rcv(rcvpkt)
Λ
rdt_rcv(rcvpkt) && (corrupt(rcvpkt)|| isACK(rcvpkt,0))
Λ
rdt_send(data)
sndpkt=make_pkt(1,data,checksum)
udt_send(sndpkt)
start_timer

Figure 3.15 ♦ rdt3.0 sender

2. Implement a simple file transfer protocol using the developed ARQ protocol in Step 1.
   2.1. Get the file name from the user and transfer the file using the ARQ mechanism developed in Step 1.
   2.2. Study the effect of a few random delay values by varying the average delay below and above the timeout value and loss rate (0% to 10% in steps of 2%) on the time taken to transfer the file (of reasonable size). You can create artificial delay and losses using the tc command in linux.

★ Command to add a delay of 97ms on eth0 network interface;

```
# tc qdisc add dev eth0 root netem delay 97ms
```

To verify the command set the rule use;

```
# tc -s qdisc
```

To remove the delay on eth0 interface use;

```
# tc qdisc del dev eth0 root netem
```

To add a random loss on network Interface eth0 use;

```
# tc qdisc change dev eth0 root netem loss 0.1%
```

Task2: Sliding Window Protocol (Go-Back-N)

1. Implement an application layer reliable file transfer protocol using Sliding Window Protocol (Go-Back-N) in C/C++. Use underlying UDP protocol for sending the application layer messages. You can assume the received packet on either side to be valid/not corrupt by keeping the checksum value dummy.

   1.1. Use the following headers for you protocol implementation,

```
packetType::InitialRequest
{       packetType,
        sequenceNum,
        ackNum,
        packetChecksum,
}
//*payload of this packet should contain the file name that the
user/client wants to fetch from the server.

packetType::InitialResponse
{       packetType,
        sequenceNum,
        file size,
        window size,
        packetChecksum,
}
//*server sends this information to the client and after receiving
the ACK from the client, server starts sending the requested file.


packetType::DataPack
{       packetType
        sequenceNum,
        ackNum,
        packetChecksum,
        payloadLength,
}
//*payload of this packet contains the file content.

packetType::Ack
{       packetType,
        ExpectedSeqNum,
        ackNum,
        packetChecksum,
}
//*ACK packet sent by the client
```

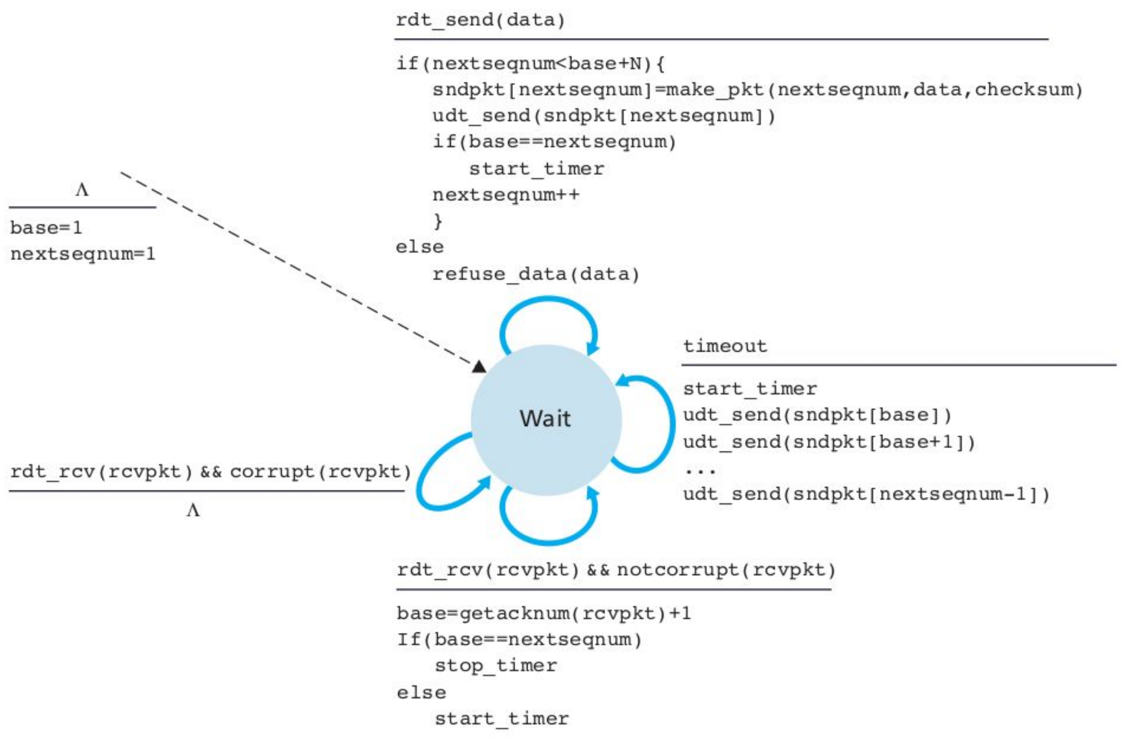


Figure 3.20 ♦ Extended FSM description of GBN sender

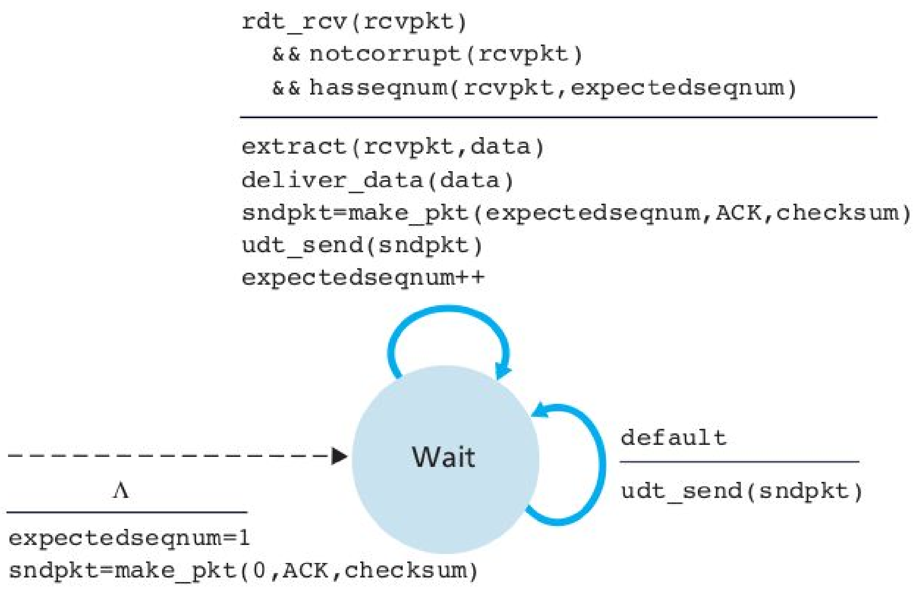


Figure 3.21 ♦ Extended FSM description of GBN receiver

*Image source - "Computer Networking : A Top-Down Approach, 6th edition"

1.2. Implemented protocol should be able to handle all the states and conditions mentioned in above FSM figures for sender and receiver.
1.3. Split the file into fixed size segments (except the last one) and append the header before sending.
1.4. To handle the case of a packet loss or time out implement the go-back-n mechanism.

2. Test your protocol by transfering a large file using the developed file transfer protocol developed in Task 1.
    2.1. Get the file name from the user and transfer the file using the Go-back-N ARQ mechanism developed in Step 1.
    2.2. Study the effect of a few random delay values by varying the average delay below and above the timeout value and loss rate (0% to 10% in steps of 2%) on the time taken to transfer the file for a set of window sizes. You can create artificial delay and losses using the tc command in linux.

Submission

Add appropriate comments to your code in the programs to make it readable. Prepare a single zip file with name format **<assignment4_roll no>.zip** including the sub folders for Task1 and Task2, containing their source files and respective readme of your programs. Submit it to google classroom in the posted assignment section.

PLAGIARISM STATEMENT <Include it in your readme>

I certify that this assignment/report is my own work, based on my personal study and/or research and that I have acknowledged all material and sources used in its preparation, whether they be books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. I also certify that this assignment/report has not previously been submitted for assessment in any other course, except where specific permission has been granted from all course instructors involved, or at any other time in this course, and that I have not copied in part or whole or otherwise plagiarised the work of other students and/or persons. I pledge to uphold the principles of honesty and responsibility at CSE@IITH. In addition, I understand my responsibility to report honour violations by other students if I become aware of it.

Name of the student
Roll No