C code for stop and wait protocol :

Program:

```
#include <conio.h>
#include <dos.h>
#include <stdio.h>
#include <stdlib.h>

#define TIMEOUT 5
#define MAX_SEQ 1
#define TOT_PACKETS 8
#define inc(k) if(k<MAX_SEQ) k++; else k=0;
typedef struct
{
 int data;
}packet;
typedef struct
{
 int kind;
 int seq;
 int ack;
 packet info;
 int err;
}frame;
frame DATA;
typedef enum{frame_arrival,err,timeout,no_event} event_type;
```

```
void from_network_layer(packet *);
void to_network_layer(packet *);
void to_physical_layer(frame *);
void from_physical_layer(frame *);
void wait_for_event_sender(event_type *);
void wait_for_event_reciever(event_type *);
void reciever();
void sender();

int i=1;        //Data to be sent by sender
char turn;      //r , s
int DISCONNECT=0;
/*_____________________________________________________________________________*/
void main()
{
 clrscr();
 randomize();
 while(!DISCONNECT)
 {
   sender();
   delay(400);
   reciever();
 }
 getch();
}
/*_____________________________________________________________________________*/
void sender()
{
 static int frame_to_send=0;
```

```
static frame s;
packet buffer;
event_type event;
static int flag=0;

if(flag==0)
{
 from_network_layer(&buffer);
 s.info = buffer;
 s.seq = frame_to_send;
 printf("SENDER :  Info = %d    Seq No = %d     ",s.info,s.seq);
 turn = 'r';
 to_physical_layer(&s);
 flag = 1;
}
wait_for_event_sender(&event);
if(turn=='s')
 {
  if(event==frame_arrival)
   {
     from_network_layer(&buffer);
     inc(frame_to_send);
     s.info = buffer;
     s.seq = frame_to_send;
     printf("SENDER :  Info = %d    Seq No = %d     ",s.info,s.seq);
     turn = 'r';
     to_physical_layer(&s);
   }
  if(event==timeout)
```

```
   {
     printf("SENDER : Resending Frame            ");
     turn = 'r';
     to_physical_layer(&s);
   }
  }
}
/*_____________________________________________________________________________*/
void reciever()
{
 static int frame_expected=0;
 frame r,s;
 event_type event;

 wait_for_event_reciever(&event);
 if(turn=='r')
 {
  if(event==frame_arrival)
  {
    from_physical_layer(&r);
    if(r.seq==frame_expected)
    {
      to_network_layer(&r.info);
      inc(frame_expected);
    }
    else
      printf("RECIEVER : Acknowledgement Resent\n");

    turn = 's';
```

```
    to_physical_layer(&s);
   }
  if(event==err)
   {
    printf("RECIEVER : Garbled Frame\n");
    turn = 's';    //if frame not recieved
   }              //sender shold send it again
 }
}
/*_____________________________________________________________________________*/
void from_network_layer(packet *buffer)
{
  (*buffer).data = i;
  i++;
}
/*_____________________________________________________________________________*/
void to_physical_layer(frame *s)
{                 // 0 means error
 s->err = random(4);  //non zero means no error
 DATA = *s;          //probability of error = 1/4
}
/*_____________________________________________________________________________*/
void to_network_layer(packet *buffer)
{
 printf("RECIEVER :Packet %d recieved , Ack Sent\n",(*buffer).data);
 if(i>TOT_PACKETS)         //if all packets recieved then disconnect
  {
   DISCONNECT = 1;
   printf("\nDISCONNECTED");
```

```
  }
}
/*_____________________________________________________________________________*/
void from_physical_layer(frame *buffer)
{
 *buffer = DATA;
}
/*_____________________________________________________________________________*/
void wait_for_event_sender(event_type * e)
{
 static int timer=0;

 if(turn=='s')
  {
   timer++;
   if(timer==TIMEOUT)
    {
     *e = timeout;
     printf("SENDER : Ack not recieved=> TIMEOUT\n");
     timer = 0;
     return;
    }
   if(DATA.err==0)
     *e = err;
   else
    {
     timer = 0;
     *e = frame_arrival;
    }
```

```
 }
}
/*_____*/
void wait_for_event_reciever(event_type * e)
{
 if(turn=='r')
  {
   if(DATA.err==0)
     *e = err;
   else
     *e = frame_arrival;
  }
}
```

Output:

```
SENDER :  Info = 1    Seq No = 0     RECIEVER :Packet 1 recieved , Ack Sent
SENDER : Ack not recieved=> TIMEOUT
SENDER : Resending Frame            RECIEVER : Acknowledgement Resent
SENDER :  Info = 2    Seq No = 1     RECIEVER :Packet 2 recieved , Ack Sent
SENDER :  Info = 3    Seq No = 0     RECIEVER :Packet 3 recieved , Ack Sent
SENDER : Ack not recieved=> TIMEOUT
SENDER : Resending Frame            RECIEVER : Acknowledgement Resent
SENDER :  Info = 4    Seq No = 1     RECIEVER :Packet 4 recieved , Ack Sent
SENDER :  Info = 5    Seq No = 0     RECIEVER :Packet 5 recieved , Ack Sent
SENDER :  Info = 6    Seq No = 1     RECIEVER :Packet 6 recieved , Ack Sent
SENDER :  Info = 7    Seq No = 0     RECIEVER :Packet 7 recieved , Ack Sent
SENDER :  Info = 8    Seq No = 1     RECIEVER : Garbled Frame
SENDER : Ack not recieved=> TIMEOUT
SENDER : Resending Frame            RECIEVER :Packet 8 recieved , Ack Sent
```