实验 3 参考资料

RDT 编程接口（API）

为方便同学们实验，我们对原始的 UDP 发送接收程序进行了若干调整，并提供相关的文件读写及包封装的接口，体现在如下几个函数中：

”net_exp.h”头文件中：
给出了 linux 网络编程的各引用库：

```
#include <sys/socket.h>
#include <sys/time.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <unistd.h>
#include <string.h>
#include <fcntl.h>
#include <time.h>
```

给出了进行 RDT 通信的相关宏定义，包括 UDP 的地址、端口、一些数据包相关的参数及等待时间等：

```
#define     RDT_SERVER_ADDRESS      "127.0.0.1"      //RDT服务器端IP
#define  RDT_RECV_PORT          8003              //RDT接收端端口号
#define     RDT_SEND_PORT          8004              //RDT发送端端口号
#define     RDT_BEGIN_SEQ          1                 //RDT数据包初始序列号，（假设数据包序列号不循环）

#define     RDT_PKT_LOSS_RATE      10                //不可靠数据传输层的丢包率
#define     RDT_TIME_OUT           5000              //数据包超时时限
#define     RDT_HEADER_LEN             (4 + 4)           //RDT头标长度
#define  RDT_DATA_LEN           1000              //RDT中数据域长度
#define     RDT_PKT_LEN            ( RDT_DATA_LEN + RDT_HEADER_LEN )  //RDT中数据包长度

//RDT包类型
#define     RDT_CTRL_BEGN          0    //初始包
#define     RDT_CTRL_DATA          1     //数据包
```

```
#define  RDT_CTRL_ACK        2   //ACK包
#define      RDT_CTRL_NACK       3   //NACK包
#define      RDT_CTRL_END        4   //结束包
```

给出了功能接口：

int pack_rdt_pkt(char *data_buf, char *rdt_pkt, int data_len, int seq_num, int flag);

功能：用于向空包rdt_pkt中写入：data_len字节长度的data_buff数据块，包序号seq_num（int是4个字节，在发送文件小于10M的情况下，足够保证包序号从头到尾都不重复，这里是简化的处理方法），以及相关的包头标识flag（这里flag只反映出RDT_CTRL_BEGN RDT_CTRL_DATA RDT_CTRL_ACK RDT_CTRL_NACK RDT_CTRL_END这几个包属性，同学们要注意ACK和NACK这两个最重要的属性！）。

返回：封装完毕的包长度。

int unpack_rdt_pkt(char *data_buf, char *rdt_pkt, int pkt_len, int *seq_num, int *flag);

功能：用于从总长为pkt_len字节的包中读取：data_buff数据块，包序号seq_num，以及相关的包头标识flag。

返回：包中的数据（载荷）长度。

void udt_sendto(int sock_fd, char *pkt, int pkt_len, int flags, struct sockaddr *recv_addr, int addr_len);

功能：模拟不可靠信道的丢包现象，以百分之 RDT_PKT_LOSS_RATE 的概率（默认 5%）发送失败，并给出提示信息。

返回：无。

"rdt_pkt_util.c"中给出了上述三个功能接口的实现细节，有兴趣的同学可以自己看一下。

************************重点********************************

"rdt_gbn_receiver.c"是 GBN 协议接收端的主程序

GBN 实验中主要是对其中的核心函数填空（选择重传实验，则需要修改）：

`int receive_file(char *save_file_name, int sock_fd );`

功能：以可靠通信的方式接收文件，接收到时回复 ACK。

其它的发送端的完整功能请看程序本身注释。

重点是对函数中的主循环进行合理修改和添加

****