

题目：模拟总线型以太网数据帧发送过程

姓名:李奇

学号:2019302863

班号:10011904

时间:2021.10.13

计算机学院

题目:

### 模拟总线型以太网数据帧发送过程

#### 1、目的

以太网是当前应用最广泛的局域网类型。学习以太网内容对深入掌握局域网知识是非常重要的。本课程设计的主要目的是通过模拟以太网帧的发送过程，熟悉以太网的基本工作原理，特别是 CSMA/CD 介质访问控制方法

#### 2、要求

(1) 在一台计算机上模拟总线型以太网数据帧发送过程，总线上连接的计算机个数为 2 个，支持 CSMA\CD 协议（二进制指数退避算法）

(2) 用两个线程 a 和 b 模拟以太网上的两台主机。用一个双字类型变量 Bus 模拟总线。

(3) 两个子线程向总线发送自己的数据。数据用该线程的线程号进行模拟，发送数据用线程号的 Bus 的“或”进行模拟。每台主机须向总线上成功发送 5 次数据，如果其中某次数据发送失败，则该线程结束。

(4) 发送流程须遵循 CSMA/CD 方法，随机延迟算法中的冲突窗口值取 0.005。数据发送成功 (Bus==ID) 后，报告“xxx send success”；产生冲突 (Bus!=ID) 后，报告“xxx send collision”；数据发送失败（冲突计数器值为 0）后，报告“xxx send failure”。其中，xxx 为其线程的线程号。在主机发送成功

次数增加后，报告已发送成功的次数。

### 3、相关知识

以太网的核心技术是随机争用型介质访问方法，即带有冲突检测的载波侦听多路访问（CSMA/CD）方法。

#### 1. 以太网的帧的发送流程

（1）载波侦听过程。以太网中每个节点利用总线发送数据，总线是每个结点共享的公共传输介质。所以结点在发送一个帧前，必须侦听总线是否空闲，由于以太网的数据采用曼彻斯特编码方式，所以可以通过判断总线电平是否跳变来确定总线是否空闲。若总线空闲，就可启动发送，否则继续侦听。

（2）冲突检测。在数据发送过程中，可能会产生冲突（冲突是指总线上同时出现两个或两个以上的发送信号，他们叠加后的信号波形与任何发送结点输出的信号波形不相同。因为可能有多个主机都在侦听总线，当它们侦听到总线空闲时，都会往总线上发送数据）。所以在发送数据的过程中，也应该进行冲突检测，只要发现冲突就应该停止发送数据。

（3）随机延迟后重发。在检测到冲突、停止发送后，结点进行随机延迟重发。若重发 16 次后还没成功，则宣告发送失败，取消该帧的发送。随机延迟的计算方法一般采用截至二进制指数后退算法。该算法可表示为： $t = 2^k * R * a$ ，其中  $t$  为结点重新发送需要的后退延迟时间， $a$  为冲突窗口值（冲突窗口为总线最大长度和电磁波在介质中的传播速度比值的 2 倍）， $R$  为随机数， $k$  的取值为  $k = \min(n, 10)$ ， $n$  为该帧已发送的次数。

### 4、实现原理及流程图

原理：

发送方：

某站点需要发送数据帧，首先侦听信道：

- (1) 如果信道空闲，站点立即发送数据帧；
- (2) 在发送数据帧过程中，边发送边冲突检测；
- (3) 如果信道忙，继续侦听直到信道变为空闲，再发送数据帧

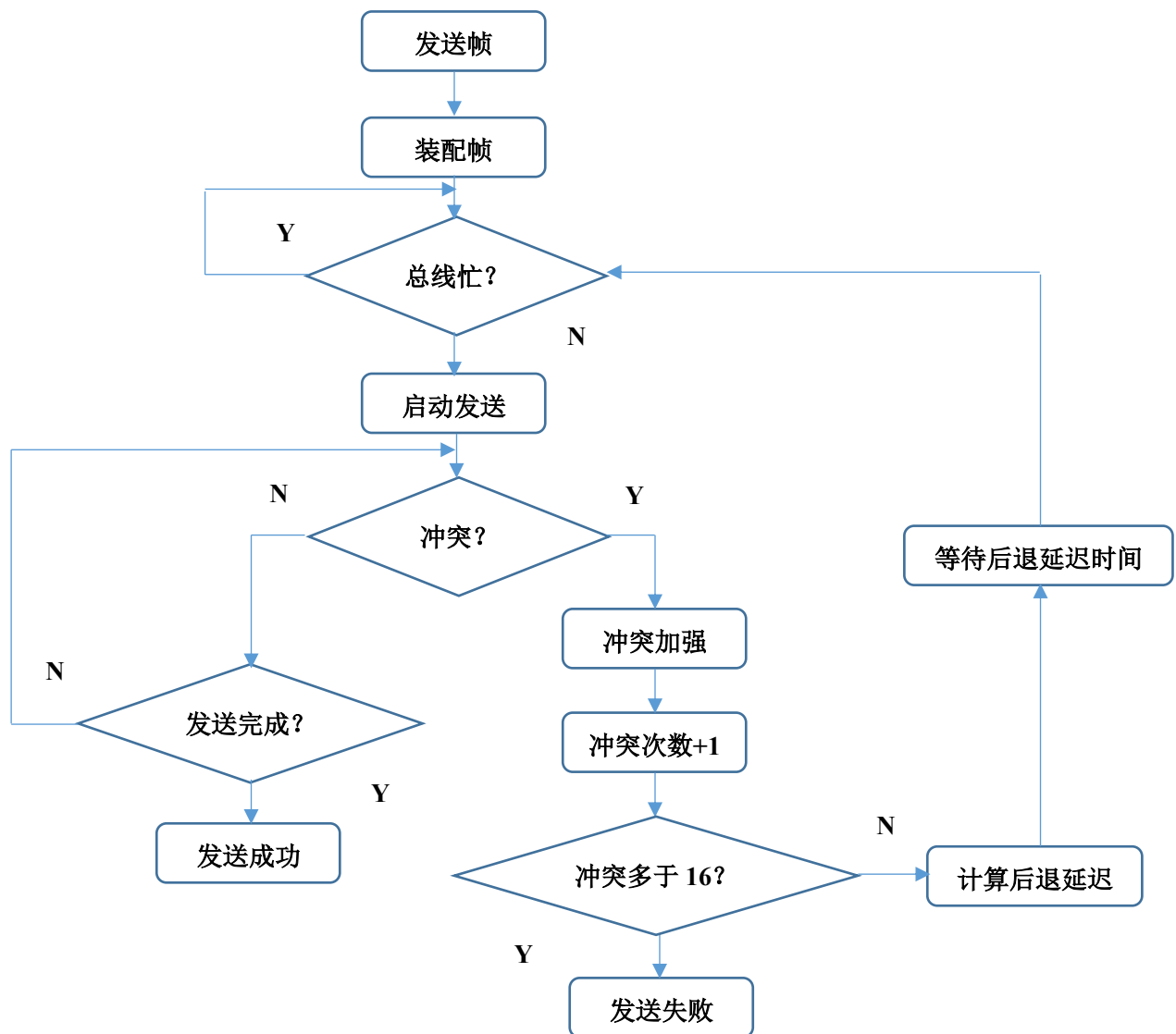
如果再发送过程中检测到冲突，则：

- (1) 立即停止发送该数据帧；
- (2) 给总线上发送一串阻塞加强信号，告诉其他站点总线发生冲突；
- (3) 等待一段随机时间（利用二进制指数退避算法），再重新争用总线，重复上面步骤，并重发该数据帧。

接收方：

- (1) 检查是否发生冲突，若发生冲突，则丢弃该帧；若没有冲突，进入下一步。
- (2) 检查该帧的目的地址是否可以接收该帧，若可以接收，则进入下一步。
- (3) 检查 CRC 校验和 LLC 数据长度。若都正确，接收该帧，否则丢弃。

流程图:

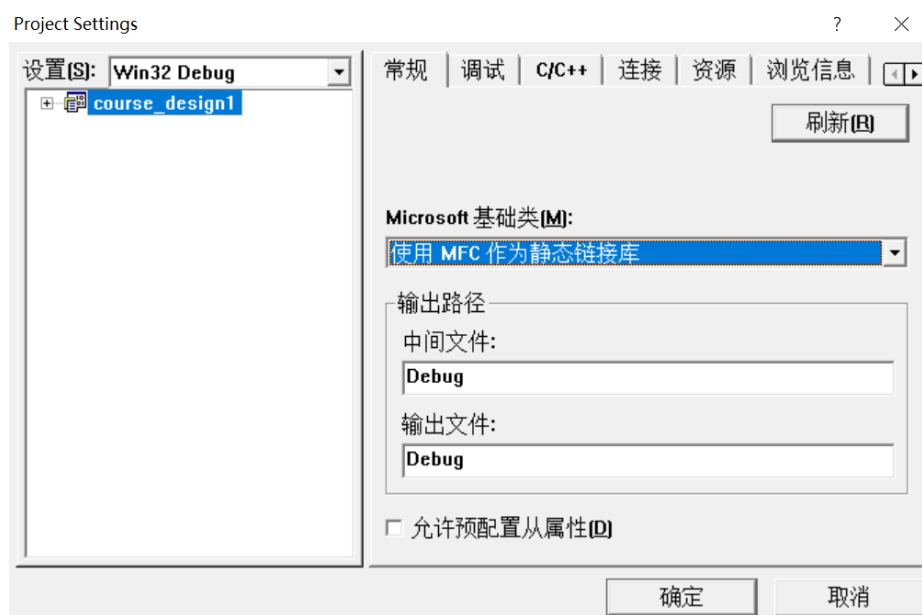


## 5、程序代码(以附件形式, 编程环境:VC++6.0)

见附件

## 6、运行结果与分析

修改 MFC 为静态链接库:



主要代码:

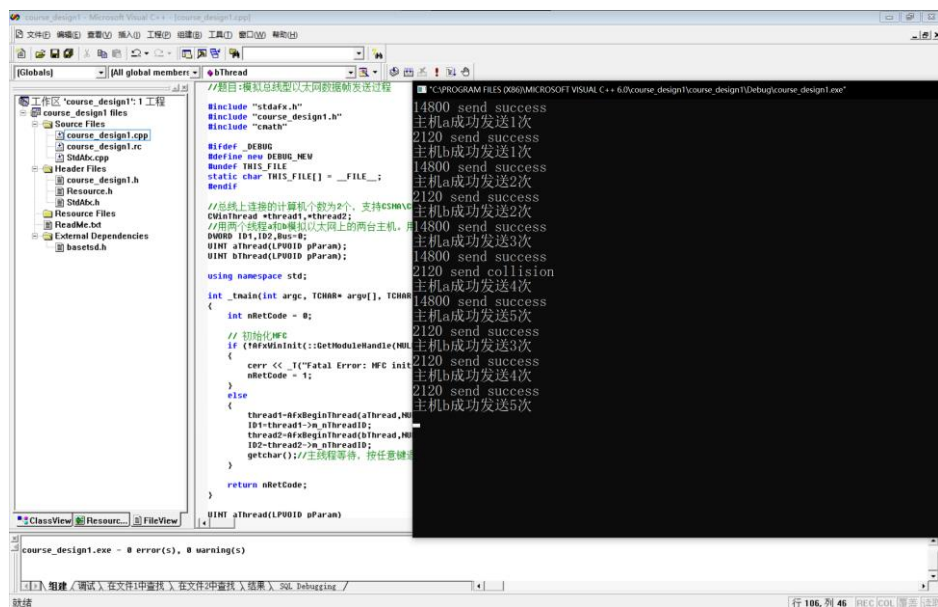
```
int _tmain(int argc, TCHAR* argv[], TCHAR* envp[])
{
    int nRetCode = 0;

    // 初始化MFC
    if (!AfxWinInit(::GetModuleHandle(NULL), NULL, ::GetCommandLine(), 0))
    {
        cerr << _T("Fatal Error: MFC initialization failed") << endl;
        nRetCode = 1;
    }
    else
    {
        thread1=AfxBeginThread(aThread,NULL);//启动线程a
        ID1=thread1->m_nThreadID;
        thread2=AfxBeginThread(bThread,NULL);//启动线程b
        ID2=thread2->m_nThreadID;
        getchar();//主线程等待, 按任意键退出程序
    }

    return nRetCode;
}
```

```
UINT aThread(LPVOID pParam)
{
    int i=0;//发送成功次数
    int CollisionCounter=16;//冲突计数器初始值为16
    //随机延迟算法中的冲突窗口值取0.005
    double collisionWindow=0.005;
    int randNum=rand()%3;
    Loop:if(Bus==0)//总线空闲
    {
        Bus=Bus|ID1;//模拟发送包
        Sleep(2);
        if(Bus==ID1)//若判断为无冲突
        {
            //数据发送成功 (Bus==ID), 报告"xxx send success"
            printf("%d send success\n",ID1);
            Bus=0;//内存清零
            CollisionCounter=16;//复原冲突计数器
            Sleep(rand()%10);//经过一个随即延时
            i++;
            //在主机发送成功次数增加后, 报告已发送成功的次数
            printf("主机a成功发送%d次\n",i);
            if(i<5)//每台主机须向总线上成功发送5次数据
                goto Loop;
        }
        else
        {
            //产生冲突 (Bus!=ID), 报告"xxx send collision"
            printf("%d send collision\n",ID1);
            Bus=0;
            CollisionCounter--;//冲突次数+1
            if(CollisionCounter>0)
            {
                //延迟重发, 延迟算法采用截断二进制指数退避算法
                Sleep(randNum*(int)pow(2,(CollisionCounter>10)?10:CollisionCounter)*collisionWindow);
                goto Loop;
            }
            else
            //数据发送失败 (冲突计数器值为0) 后, 报告"xxx send failure"
            printf("%d send failure\n",ID1);
        }
    }
    else
        goto Loop;
    return 0;
}
```

运行结果:



通过本次课程设计, 我充分理解了 CSMA/CD 协议, 在发送帧的时候首先要进行监听, 看看总线是否空闲, 若空闲, 也不能立即发送数据, 因为可能有

另一方正在发送数据二没有监听到，所以此时应该进行冲突检测，如果冲突检测结果也是无冲突，这时才可以发送数据，然而在发送数据的过程中，依然要保持监听，边听边发，一旦有冲突，立刻停止发送。相关的问题只有通过亲自动手实践才能真正地充分理解，本次课程设计提升了我对相关知识的领会，帮助我与其他知识融会贯通。

### 参考文献

- (1) 姚焱等；计算机网络原理实验分析与实践；北京：电子工业出版社，2020.6
- (2) 谢希仁；计算机网络；北京：电子工业出版社，2017.1
- (3) 蔡皖东；计算机网络；北京：清华大学出版社，2015.7