第一章：

10. 使用层次协议的两个理由：

① 降低网络设计的复杂性。将一个大的设计问题进行分解，各层之间的实现、分工更为明确 ② 对于新协议或新实现来说，它所要做的仅仅是向紧邻的上层提供与旧协议或旧实现完全相同的服务。

缺点：为了便于设计与维护，使用了层次结构，但其对性能也会降低。相比于未分层的设计，分层设计的性能可能会差一些。

11. SP公司总裁俞和顾啤酒酒造总裁作为最高管理者，还对经济方面问题进行了交流；同样两个公司之间也有同级交流现象。

而在TCP协议中，物理通信不是每层都有，仅仅在最底层。

19. 2009年有6亿台，

则2010年有 6亿×(1+1)=10(亿台)

至2020年，则11年，到今年年底，则过了12年，即翻8次，即 $6\times 2^8 = 6\times 256 = 1536$(亿)

若全球80亿人，一人有 $\frac{1536}{80} = 19.2$(台)，算上电视、手表、耳机等，平均一个19.2台显夸张，但也不过分。

37.

```c
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
#include "assert.h"

char* attach_header(char *buffer, char *header, int *length) {
    int len = strlen(buffer);
    int hlen = strlen(header);
    *length = len + hlen;
    char *buffer2 = (char*)malloc(*length + 1);
    memcpy(buffer2, header, hlen);
    memcpy(buffer2 + hlen, buffer, len);
    buffer2[*length] = '\0';
```

```
    return buffer2;
}

void physical_layer_send(char *buffer, int len) {
    char header[] = "physical_layer_header;";
    assert(strlen(buffer) == len);
    int length = 0;
    char *buffer2 = attach_header(buffer, header, &length);
    printf("%s\n", buffer2);
    free(buffer2);
}

void datalink_layer_send(char *buffer, int len) {
    char header[] = "datalink_layer_header;";
    assert(strlen(buffer) == len);
    int length = 0;
    char *buffer2 = attach_header(buffer, header, &length);
    printf("%s\n", buffer2);
    physical_layer_send(buffer2, length);
    free(buffer2);
}

void network_layer_send(char *buffer, int len) {
    char header[] = "network_layer_header;";
    assert(strlen(buffer) == len);
    int length = 0;
    char *buffer2 = attach_header(buffer, header, &length);
    printf("%s\n", buffer2);
    datalink_layer_send(buffer2, length);
    free(buffer2);
}

void transport_layer_send(char *buffer, int len) {
    char header[] = "transport_layer_header;";
    assert(strlen(buffer) == len);
    int length = 0;
    char *buffer2 = attach_header(buffer, header, &length);
    printf("%s\n", buffer2);
    network_layer_send(buffer2, length);
    free(buffer2);
}
```

```c
void session_layer_send(char *buffer, int len) {
    char header[] = "session_layer_header;";
    assert(strlen(buffer) == len);
    int length = 0;
    char *buffer2 = attach_header(buffer, header, &length);
    printf("%s\n", buffer2);
    transport_layer_send(buffer2, length);
    free(buffer2);
}

void presentation_layer_send(char *buffer, int len) {
    char header[] = "presentation_layer_header;";
    assert(strlen(buffer) == len);
    int length = 0;
    char *buffer2 = attach_header(buffer, header, &length);
    printf("%s\n", buffer2);
    session_layer_send(buffer2, length);
    free(buffer2);
}

void application_layer_send(char *buffer, int len) {
    static char header[] = "application_layer_header;";
    assert(strlen(buffer) == len);
    int length = 0;
    char *buffer2 = attach_header(buffer, header, &length);
    printf("%s\n", buffer2);
    presentation_layer_send(buffer2, length);
    free(buffer2);
}

void application_routine() {
    char message[] = "This is a test message!";
    application_layer_send(message, strlen(message));
}

int main(int argc, char *argv[]) {
    application_routine();
    return 0;
}
```

第二章：

7.

$$\text{带宽} = \frac{\text{光速}}{\text{波长}} \times \frac{\text{频谱长度}}{\text{波长}}$$

$$= \frac{3\times10^8}{10^{-6}} \times \frac{10^{-7}}{10^{-6}} = 3\times10^{13}(\text{Hz})$$

$$= 3\times10^{4}(\text{GHz})$$

所以其带宽为 3×10^4 GHz.

30.

解调器只接受调制的正弦波并产生数字信号，而编码器接受任意模拟信号从转换为数字信号进行传输。两者所处理的对象的范围不同。

35.

VT1.5的传输速率：8000 frame/s × 3列 × 9行 × 8 bits = 1.728 (Mbps)

VT2.0的传输速率：8000 frames/s × 4列 × 9行 × 8 bits = 2.304 (Mbps)

VT3.0的传输速率：8000 frames/s × 6列 × 9行 × 8 bits = 3.456 (Mbps)

VT6.0的传输速率：8000 frames/s × 12列 × 9行 × 8 bits = 6.912 (Mbps)

(a) DS-1服务(1.544Mbps)，皆可满足，但VT1.5更适合.

(b) 欧洲的CEPT-1服务(2.048 Mbps)，VT2.0，VT3.0，VT6.0皆可，但VT2.0更合适.

(c) DS-2服务(6.312Mbps)，VT6.0适合.

36.

对于计算用户带宽，就要除去每一帧用于传输系统管理信息的前三列，除去SPE第一列的路径开销。

所以OC-12C用户数据占有 $12\times90-12\times3-1=1043$ (列)

用户带宽：$8000\ frame/s \times 1043(列) \times 9(行) \times 8(bits) = 600.768(Mbps)$

∴一个OC-12C连接的用户可用带宽为 600.768 Mbps.

44.

$\overline{A}=(+1+1+1-1-1+1-1-1)$

$\overline{B}=(+1+1-1+1-1-1-1+1)$

$\overline{C}=(+1-1+1-1-1-1+1+1)$

$S=\overline{A}+\overline{B}+\overline{C}=(+3+1+1+1-3-1-1+1)$

结果码片序列为 (+3 +1 +1 +1 -3 -1 -1 +1)

第三章

1.

随机变量X代表尝试次数，

$$\text{发}P = 0.8^{10} = 0.107$$

则 $X \sim G(0.107)$

又 $\because EX = \frac{1}{p} = \frac{1}{0.107} = 9.3$

$\therefore$ 平均需要9.3次才能使帧完成.

6.

经过比特填充后：01111101111110011111010

17. (1) 帧：1001110|

生成多项式：1001

1000|011 ÷(2) 1001 余101

则实际传输位串为：10011101101

(2) 当左边第二个比特变反，即 10111101101，则 10111101101 ÷(2) 1001 余101，不为0，能被探测.

(3) 当全变为0时，余数为0，探测不出来.

18.

(a) 能。CRC能检测所有的一位差错

(b) 能。k最大取偶数值不超过1055，能.

(c) 否。偶数个比特误不一定能被检测出来.

(d) 能。奇数个错误(独立)都能被检验出来.

(e) 能。小于32位的突发错误能被检测出来.

(f) 否。大于32位的突发比特误不一定能被检测出来.

30.

每个报文平均被传输两次。

① 当A向B发送一个帧，当该帧到达B时，B无向A发送帧

② 此时过了一会（10毫秒），A超时，重发该帧。

③ B接收到A重发的帧，且B认为A发生错了.

④ B返回一个捎带ACK的NAK给A.

⑤ 最终，A第一次发送的帧得知已接收.

通过该过程可看出该帧被发送了两次

第四章

6.

(a) 信号传播速度：$3\times10^8\times0.82=2.46\times10^8\ (m/s)$

竞争时间槽长度：$2\times\frac{2\times10^3}{2.46\times10^8}=1.626\times10^{-5}=16.26\ (\mu sec)$

即 16.26 微秒.

(b) 信号传播速度：$3\times10^8\times0.65=1.95\times10^8\ (m/s)$

竞争时间槽长度：$2\times\frac{4\times10^3}{1.95\times10^8}=410.26\ (\mu sec)$

即 410.26 微秒.

7.

当所有站都想发送数据时，从最到达队头准备发送数据时间：

$$(N-1)d+N\times1=(N-1)d+N$$

最坏等 $((N-1)d+N)$ 个时间单位

8. 在继二进制倒计数协议中，如果一个编号高的站和一个编号低的站同时发送数据包，且编号高的站源源不断地发送数据包，则编号低的站会放弃每轮竞争.

15.

(1) 发送时检测信道占用等待时间：

$$2\times\frac{1000\ m}{200\ m/\mu sec}=5(\mu sec)\times2=10(\mu sec)$$

(2) 第一位数据到达接收方 $\frac{1000\ m}{200\ m/sec\ \mu sec}=5(\mu sec)$

(3) 数据发送：$\frac{256bit}{10Mbps} = 25.6(usec)$

(4) 发送ACK侦听时间：$2\times\frac{1000m}{200m/usec} = 10(usec)$

(5) 发送第一位数据ACK：$\frac{1000m}{200m/usec} = 5(usec)$

(6) 发送ACK：$\frac{32bit}{10Mbps} = 3.2(usec)$

总计时：$10+5+25.6+10+5+3.2 = 58.8(usec)$

有效数据位：$256-32 = 224(bits)$

则有效数据率：$\frac{224bits}{58.8usec} = 3.81(Mbps)$

16.

第i次冲突发生的概率 $p = \frac{1}{2^{i-1}}$

则第K次成功，说明前K-1次失败

~~则 $P(K) = \prod_{i=1}^{K-1} 2^{-(i-1)}$~~ 则 $P(K) = (1-2^{(1-K)})\prod_{i=1}^{K} 2^{(1-i)}$

$EP(K) = \sum_{K=0}^{+\infty} K(1-2^{(1-K)})\prod_{i=1}^{K} 2^{(1-i)}$

答：第K轮结束竞争的概率为 $(1-2^{(1-K)})\prod_{i=1}^{K} 2^{(1-i)}$

竞争周期的平均次数为 $\sum_{K=0}^{+\infty} K(1-2^{(1-K)})\prod_{i=1}^{K} 2^{(1-i)}$

25.

~~一秒内：传输 $\frac{11\times10^{6}(bits)}{64\times32(bits)}$ 帧~~

一帧具有 $64 Bytes \times 8 = 512 (bits)$

一帧被损坏的概率为 $1-(1-10^{-7})^{512} = 5\times10^{-5}$

平均每秒传播：$\frac{11\times10^6\ bits}{64\times8\ bits}=21484$（帧）

平均损坏：$21484\times5\times10^{-5}=1.0742$（帧）

所以平均每秒有1.7042帧被损坏.