实验五：802.11a 通信系统仿真

学号：5112119045
班级：F1121102
姓名：刘斌

一、实验目的：

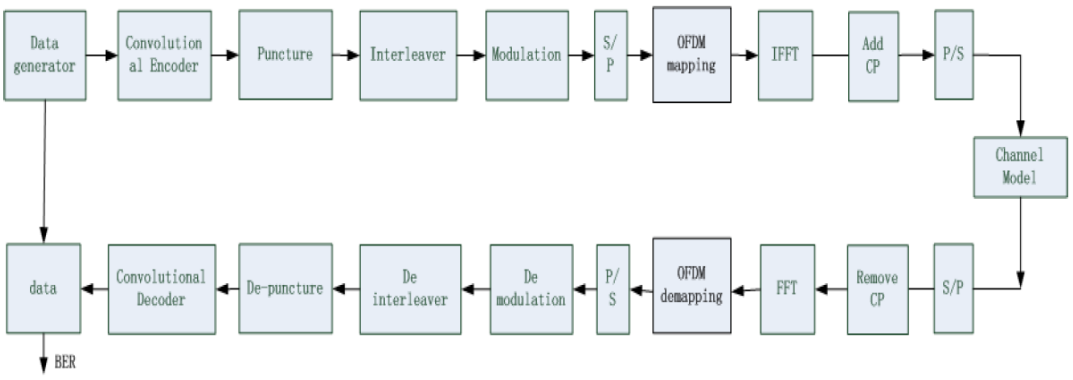
学习 IEEE802.11a 无线通信标准，理解 802.11a 的数据帧结构，实现(puncture)以及交织(interleaver)模块，在前几次实验（调制解调，OFDM，信道编解码）的基础上，对 802.11a 系统进行仿真。

二、实验要求：

(1)调制方式：BPSK,QPSK,16QAM
(2)信道编码/解调方式：Convolutional encode/Viterbi Decode
(3)码率：1/2,2/3,3/4
(4)信道：AWGN
(5)理解 IEEE 802.11a 系统的帧结构
(6)理解 OFDM 发射机和接收机的基带模块及其功能

三、实验原理：

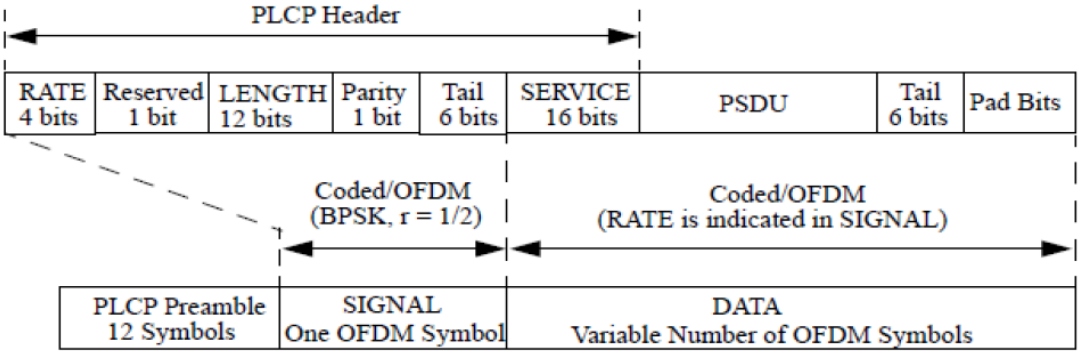
基于 802.11a 协议的系统框图



由图可知：IEEE802.11a 系统的结构包括数据产生->（2,1,7）卷积编码->puncture(凿孔)->交织->调制->串并转换->OFDM 映射->IFFT->添加循环前缀保护间隔->并串转换->AWGN 信道->串并转换->去除循环前缀保护间隔->FFT->OFDM 解映射->并串转换->解调->结交织->解凿孔->Viterbi decode->计算 BER.

1.数据产生

IEEE 802.11a 系统的帧结构如下图所示：



（1）PLCP Preamble 不加， SIGNAL 域不加， 只包含上述数据单元真结构的 DATA 域；

（2）在 DATA 域中的 pilot(导频)信号，可以用“0”，代替协议中的规定，此外，OFDM 和 Viterbi 的配置沿用“LAB3 基于 802.11a 的 OFDM 通信系统仿真”和“LAB4 卷积编解码”。

DATA 区数据产生：

对给定的 LENGTH，产生该长度的有效数据，填进 DATA 域中 PSDU 区中。对 SERVICE 区和 TAIL 区，进行填 0 处理。

为了保证 OFDM 调制的时候每个 OFDM symbol 的长度相同，对于达不到要求的 length 对应的数据需进行 padding 垫零处理。

Padding 的具体算法如下：

NSYM = ceil ((16 + 8 * LENGTH + 6)/NDBPS);

NDATA = NSYM * NDBPS;

NPAD = NDATA - (16 + 8 * LENGTH + 6);

其中，NDBPS 为一个 OFDM symbol 有效的数据个数，NSYM 为 OFDM symbol 的个数，NDATA 为 DATA 区比特数，NPAD 为垫零的个数。Ceil()为取大于该数的最小的整数。

2.凿孔与填回数据

在 IEEE 802.11a 协议中，根据 coding rate，可能会有两种不同深度的凿孔算法。

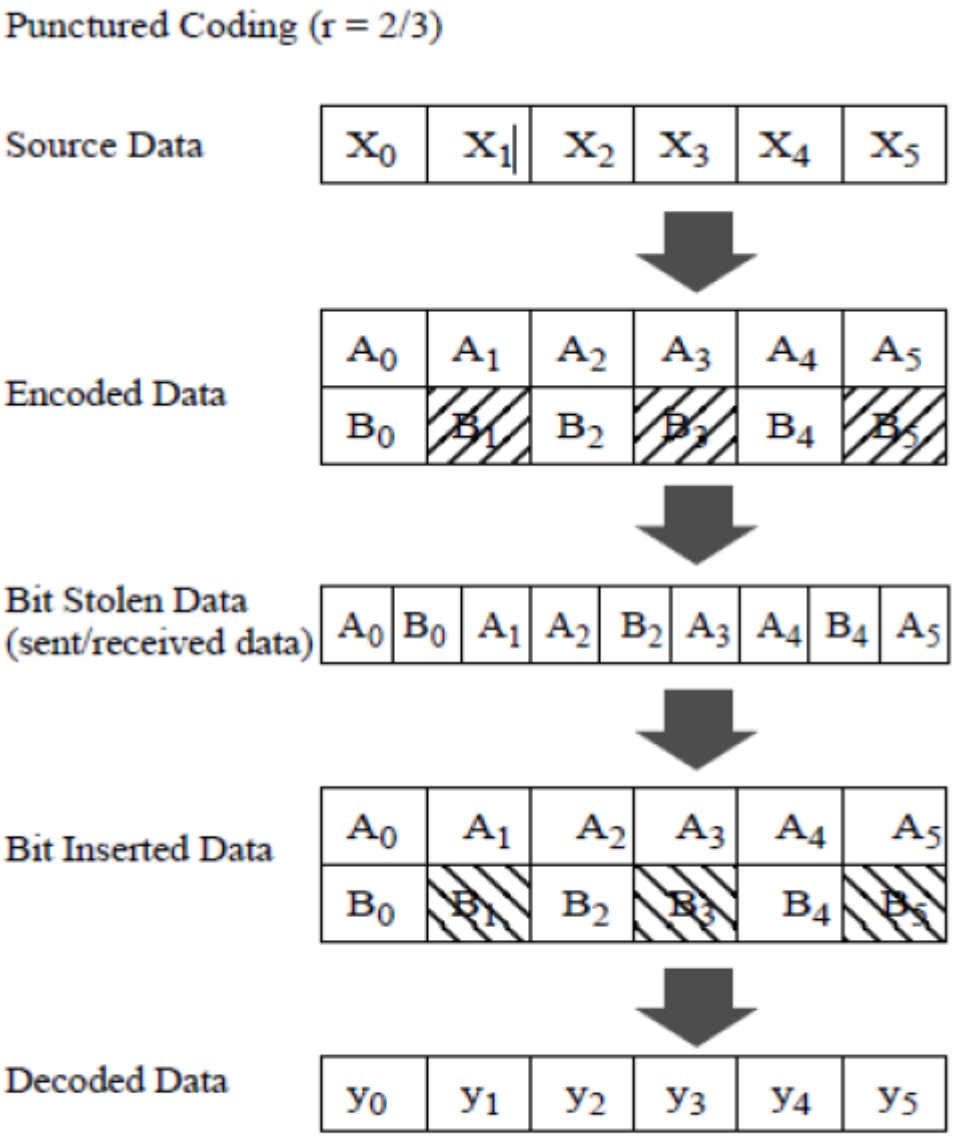
当 R(coding rate)=1/2 时，不进行凿孔处理；

当 R(coding rate)= 3/4 时，凿孔处理算法图如下所示:

由图我们可以看出，对编码后的两路数据，第一路去掉第三的倍数个数据，第二路去掉标号除三余数为 2 的数据。

对凿孔后的数据进行恢复的算法为在去掉数据的相应部分补加上一个垃圾数据。即在标号除 4 余 3 的数据后加上两个垃圾数据。

当 R(coding rate)= 2/3 时，凿孔处理算法图如下所示:



Coding Rate= 1/2 * 4/3

由图我们可以看出，对编码后的两路数据，第一路不作变化，第二路去掉标号为 2 的倍数的数据。

对凿孔后的数据进行恢复的算法为在去掉数据的相应部分补加上一个垃圾数据。即在标号为 3 的倍数的数据后加上一个垃圾数据。

3. 交织与解交织

交织的作用简单点说就是变突发错误为随机错误，减少纠错的集中性和设备成本。其算法分为两个位置交换步骤，对数据进行打散。具体实现如下：

设 k 为原数据的下标，

第一步产生数据的下标：i = (NCBPS/16) (k mod 16) + floor(k/16) k = 0,1,…,NCBPS - 1、

第二步产生数据的下标：j = s × floor(i/s) + (i + NCBPS - floor(16 × i/NCBPS)) mod s i = 0,1,… NCBPS - 1

其中，s = max(NBPSC/2,1)。

由于解交织为交织的逆过程，因此该过程也分为两步：

设 j 为接受数据的下标，k 为原数据的下标，

第一步产生数据的下标：i=s × floor(j/s) + (j + floor(16 × j/NCBPS)) mod s j = 0,1,… NCBPS - 1

第二步产生数据的下标：k = 16 × i - (NCBPS - 1)floor(16 × i/NCBPS) i = 0,1,… NCBPS - 1

经过这两步，我们可得原始数据。

4. 维特比译码

本实验是基于前面的所有的实验,直接使用 Lab4 中编写的 Vitdecode 代码。当码率为 3/4 和 2/3 时，需要执行凿孔和解凿孔操作，而 Lab4 中的译码函数不需要识别解凿孔的位置，因此在对之前的代码不需要进行改进。译码的关键代码如下：

解凿孔：

```
function output=depuncture(input,r)

%****************** variables ************************
% r: coding ratio
% *****************************************************

n=length(input);

switch r
    case 1/2
        output=ones(1,n)*2;
        output=input;
    case 3/4
        output=ones(1,n*3/2);
        output=output*2;
        output(1:6:n*3/2-5)=input(1:4:n-3);
        output(2:6:n*3/2-4)=input(2:4:n-2);
        output(3:6:n*3/2-3)=input(3:4:n-1);
        output(6:6:n*3/2)=input(4:4:n);
    case 2/3
        output=ones(1,n*4/3);
        output=output*2;
        output(1:4:n*4/3-3)=input(1:3:n-2);
        output(2:4:n*4/3-2)=input(2:3:n-1);
        output(3:4:n*4/3-1)=input(3:3:n);
end

end

  %******************** end of file ***************************
```

Viterbi 译码：

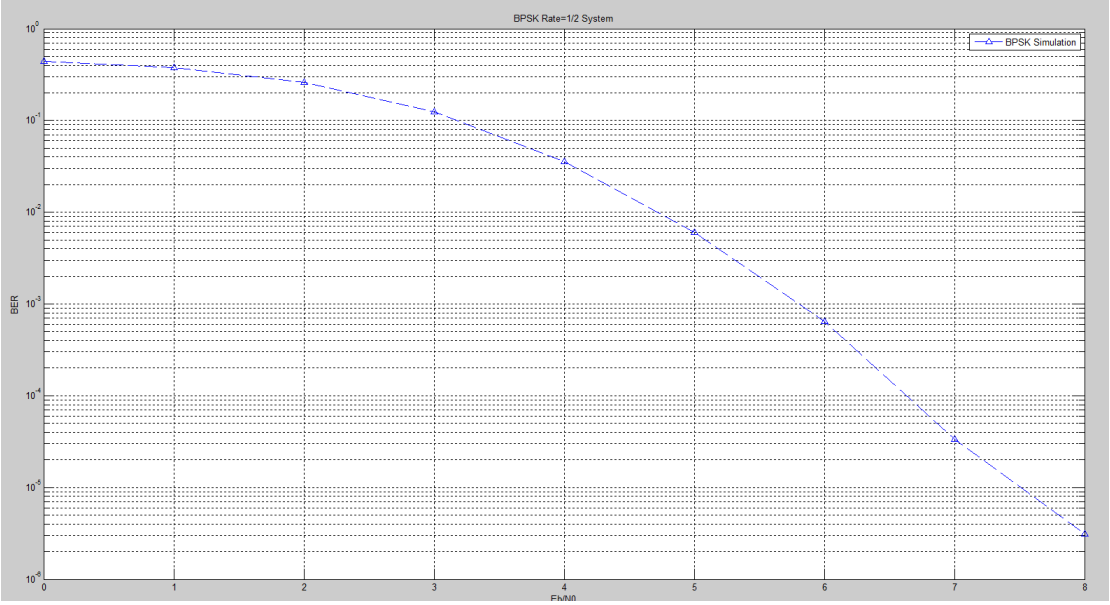
```
for (int j = 0; j < 2; j++)
{
    if (receive[j] != expect_out[st0][j])    bm[0][0] += 1;      //状态
    st0 输入 0 时的 bm0 值
    if (receive[j] != expect_out[st0][j + 2])bm[0][1] += 1;      //状态
    st0 输入 1 时的 bm1 值

    if (receive[j] != expect_out[st1][j])    bm[1][0] += 1;      //状态
    st1 输入 0 时的 bm0 值
    if (receive[j] != expect_out[st1][j + 2])bm[1][1] += 1;      //状态
    st1 输入 1 时的 bm1 值
 }
```

对于解凿孔操作之后的译码，比较的时候，若接收到的数据为 2，则表示该位有一位不同，则分支度量加 1.（由于凿孔，导致纠错性能降低）

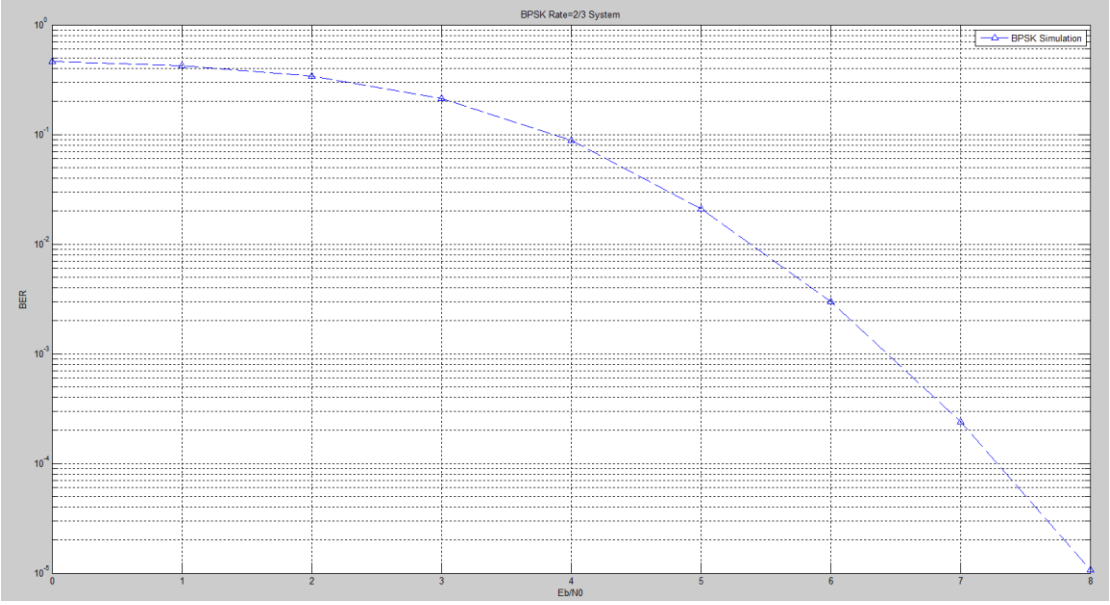
四、实验仿真

1.BPSK

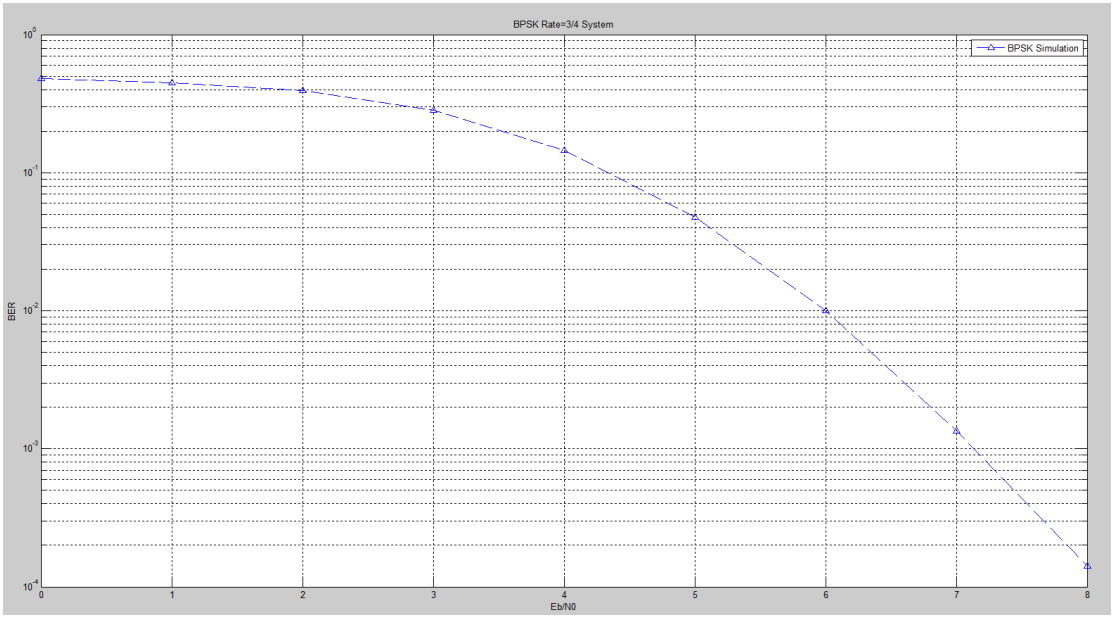
(1)Code Rate=1/2



(2)Code Rate=2/3

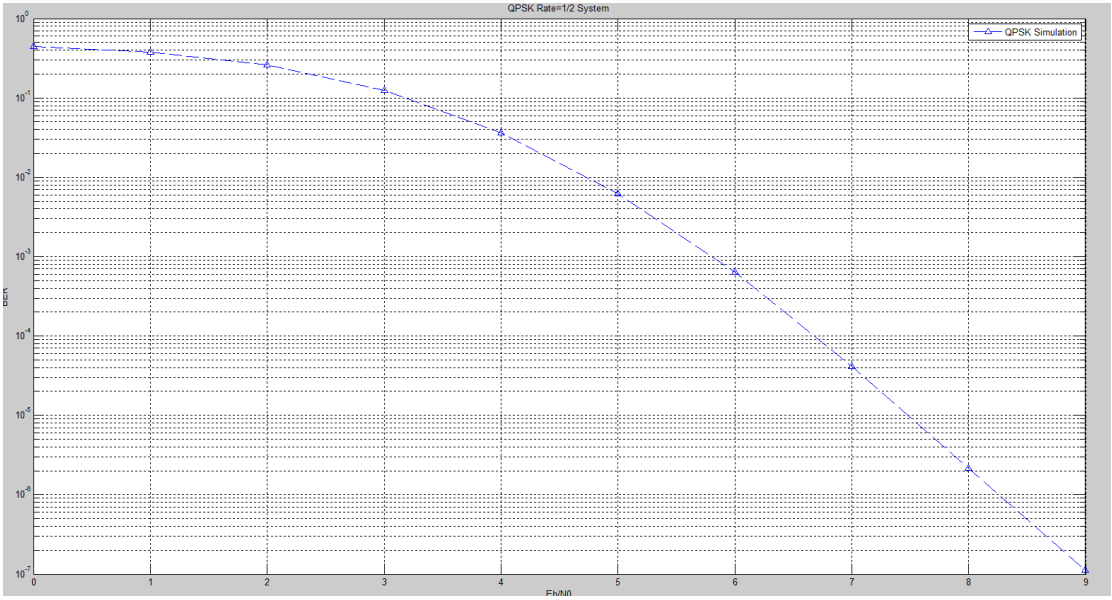


(3)Code Rate=3/4

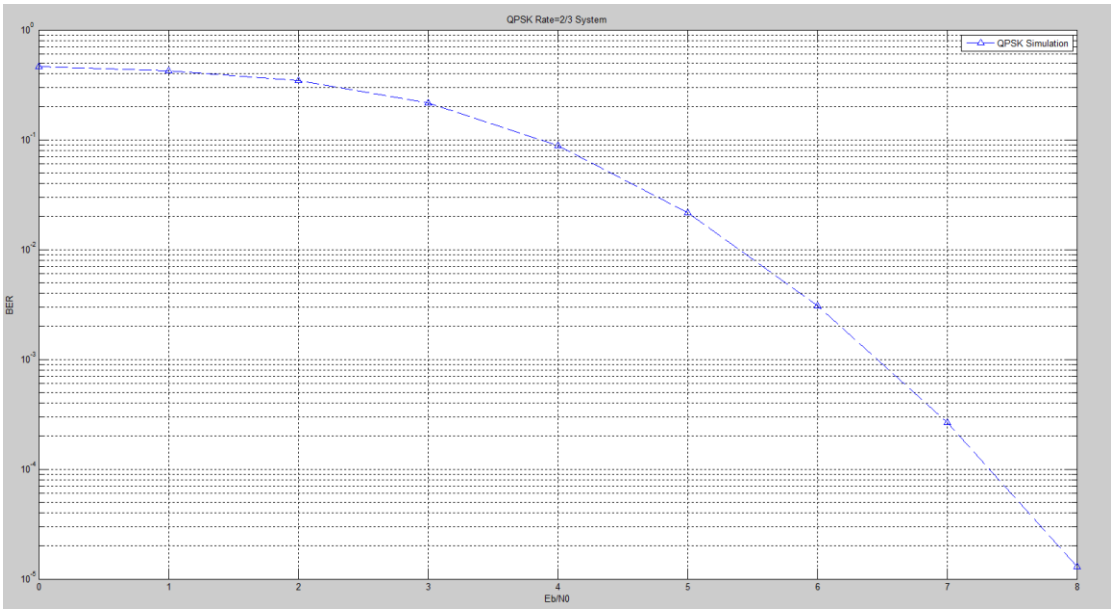


2.QPSK

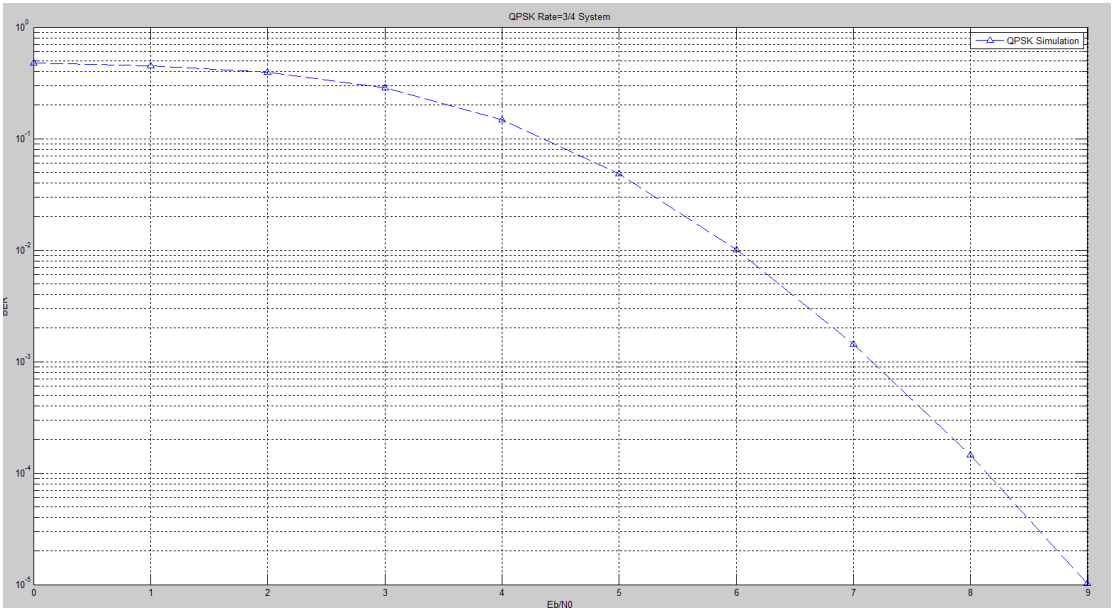
(1)Code Rate=1/2



(2)Code Rate=2/3

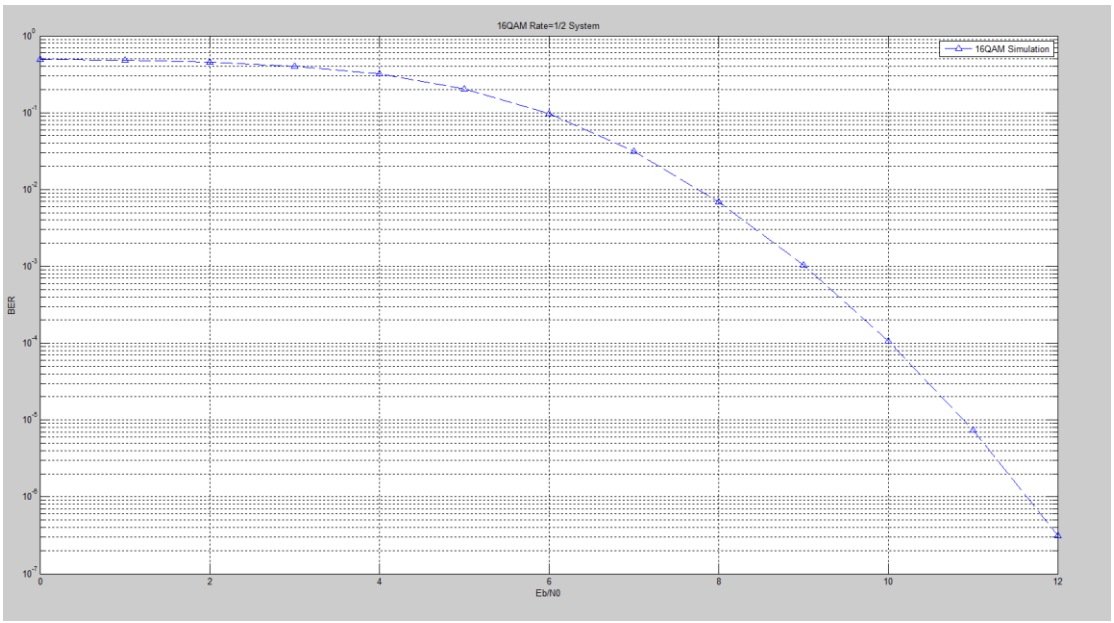


(3)Code Rate=3/4

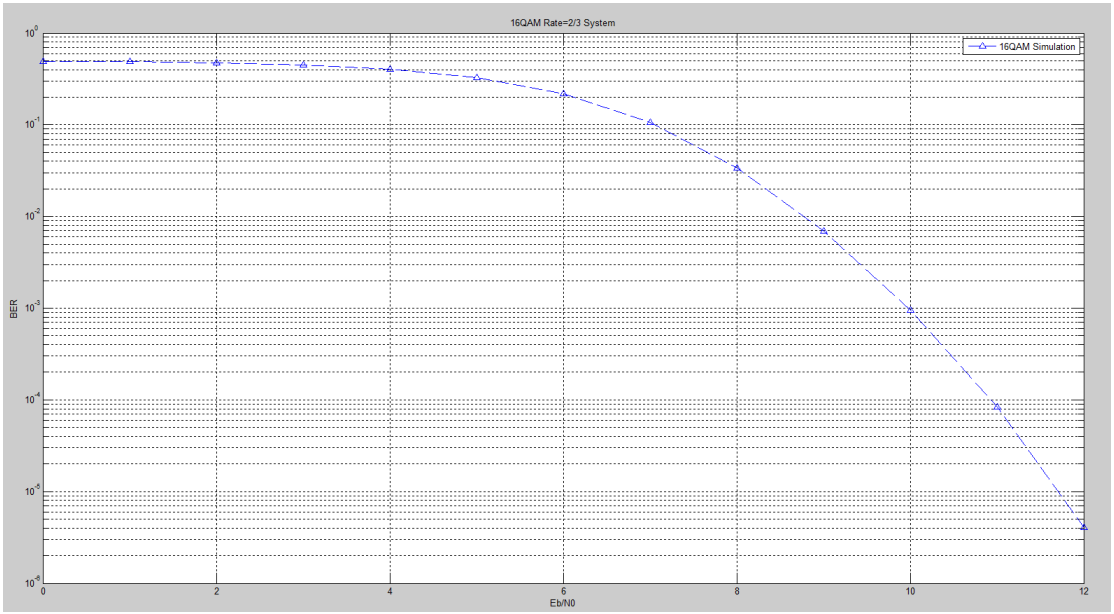


3.16QAM

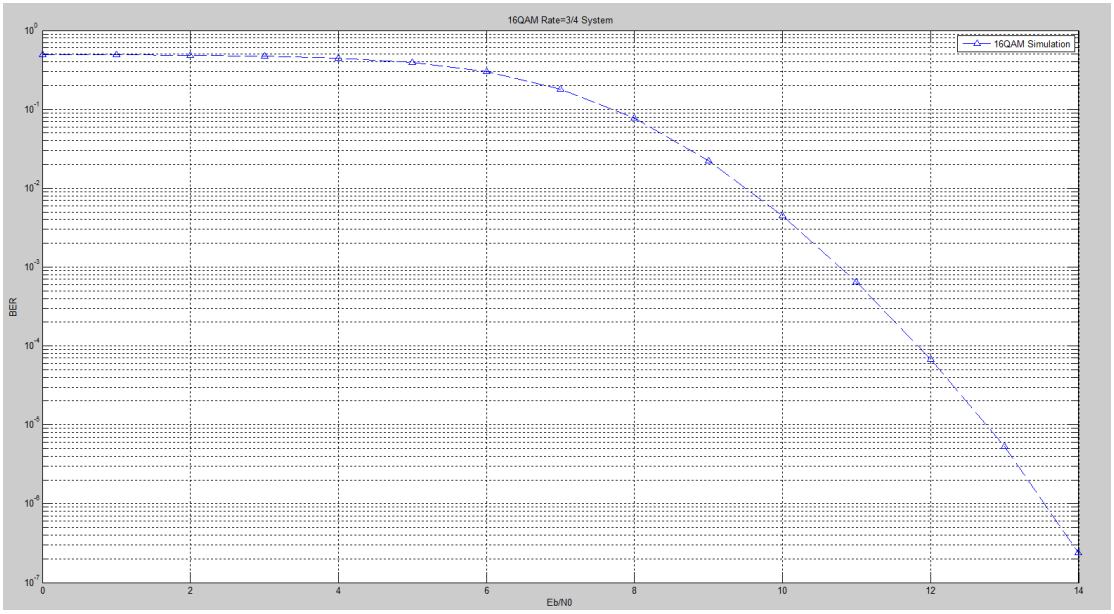
(1) Code Rate=1/2



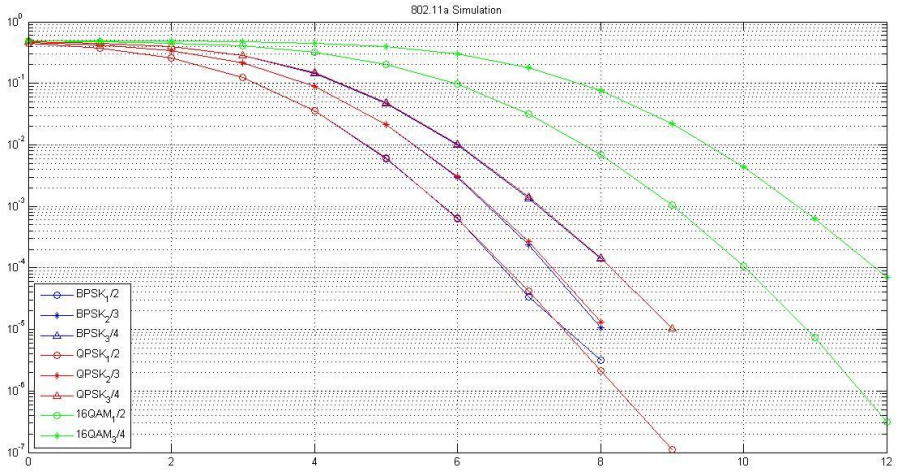
(2) Code Rate=2/3



(3) Code Rate=3/4



4.所有的结果：



五、思考题

1.码率为 1/2 的卷积码编码比特通过 puncture 得到 2/3, 3/4 码率的卷积码编码比特，为何通过 puncture 方式得到的 BER 性能与不采用 puncture 得到的 BER 性能差别不大？

答：因为采用卷积编码之后，会产生用于的纠错的冗余比特。经 puncture 之后，只是去除了部分的冗余比特，在纠错能力上有减弱，但是在满足香农容限的条件下，在译码时，插入无用的信息比特，仍能实现较好的译码，其 BER 性能较未凿孔的时候相差不大，但是实际上其纠错性能降低了。因此，插入的无用信息越多，在相同的调制方式以及 OFDM 系统下，其码率越大，BER 性能越差。码率为 1/2 的 BER 性能强于码率为 2/3 的系统，后者又优于码率为 3/4 的 BER 性能。

2.Bit Interleaver 中，为什么交织深度需要和调制方式（QPSK、16-QAM、64QAM）有关？

答：因为交织是为了将连续的突发错误打散成随机的错误进行传输，再通过纠正随机错误的编码或者其他的技术消除随机差错。交织深度越深，其纠正连续突发错误的能力越强。调制方式不一样，产生的误码率也不一样，对于误码率较高的调制方式，为了使误码率满足系统的要求，因此要采用较深的交织编码。同时，交织深度越深，交织编码所需的时间就越长，因此使得系统的时间性能会变差。交织编码以系统的时间性能为代价。

六、实验心得：

最后一次实验，通过将之前的三次的实验整合起来，实现了对 IEEE 802.11a 通信系统的模拟，使得对于通信系统的整体架构有了一定的认识。要实现一个完整的通信系统，对于现在的我来说，还是较为困难。尤其是 Viterbi 译码的编写以及最后的凿孔，解凿孔，交织，解交织的理解，是实现整个 802.11a 系统的最难的部分。理解都很困难，要写出来就更难了。但是，难也有难的好处，对于整个通信系统的理解将更加深刻。

通过一次次的 VLSI 课程设计的实验，自己的能力得到了提升。尤其是逻辑思考能力，代码编写能力得到了锻炼提升。对使用 Matlab 语言和 C++语言以及 Verilog 编写程序，自己现在相对成熟了很多。

在此，感谢老师的耐心认真的课堂讲授，以及助教耐心的帮助与解答困惑，在此一并致谢！