

Санкт-Петербургский Политехнический Университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Телекоммуникационные технологии

Отчет по лабораторной работе №8
Модель телекоммуникационного канала

Работу
выполнил:
Шустенков О.А.
Группа: 33501/1
Преподаватель:
Богач Н.В.

Санкт-Петербург
2018

Содержание

1. Цель работы	2
2. Постановка задачи	2
3. Теоретическая информация	2
3.1. Прием и передача сигналов	2
4. Ход работы	3
5. Выводы	5

1. Цель работы

Создать модель телекоммуникационного канала.

2. Постановка задачи

Пакетный сигнал длительностью 200 мкс состоит из 64 бит полезной информации и 8 нулевых tail-бит. В нулевом 16-битном слове пакета передается ID, в первом - период излучения в мс, во втором – сквозной номер пакета, в третьем - контрольная сумма (CRC-16). На передающей стороне пакет сформированный таким образом проходит следующие этапы обработки:

1. Помехоустойчивое кодирование сверточным кодом с образующими полиномами 753, 561(octal) и кодовым ограничением 9. На выходе кодера количество бит становится равным 144.
2. Перемежение бит. Количество бит на этом этапе остается неизменным.
3. Модуляция символов. На этом этапе пакет из 144 полученных с выхода перемежителя бит разбивается на 24 символа из 6 бит. Генерируется таблица функций Уолша длиной 64 бита. Каждый 6-битный символ заменяется последовательностью Уолша, номер которой равен значению данных 6-ти бит. Т.о. на выходе модулятора получается $24 * 64 = 1536$ знаковых символов.
4. Прямое расширение спектра. Полученная последовательность из 1536 символов периодически умножается с учетом знака на ПСП длиной 511 символов. Далее к началу сформированного символьного пакета прикрепляется немодулированная ПСП. Т.о. символьная длина становится равной 1747. Далее полученные символы модулируются методом BPSK.

Задача: по имеющейся записи сигнала из эфира и коду модели передатчика создать модель приемника, в которой найти позицию начала пакета и, выполнив операции демодуляции, деперемежения и декодирования, получить передаваемые параметры: ID, период, и номер пакета. Известно, что ID = 4, период 100 мс, номер пакета 373. Запись сделана с передискретизацией 2, т.е. одному BPSK символу соответствуют 2 лежащих друг за другом отсчета в файле. Запись сделана на нулевой частоте и представляет из себя последовательность 32-х битных комплексных отсчетов, где младшие 16 бит вещественная часть, старшие 16 бит – мнимая часть.

3. Теоретическая информация

3.1. Прием и передача сигналов

Приемник и передающее "устройство" выполняет последовательность обратимых операций над пакетом обмена данными. В канале передачи информации действуют шумы. При неизвестных параметрах шума на приемнике выполняется синхронизация записи сигнала по известной опорной псевдослучайной последовательности (ПСП).

При демодуляции и одновременном сужении спектра принятого сигнала также используется корреляционный метод - обратное быстрое преобразование Уолша-Адамара. В обоих случаях - при синхронизации и при сужении спектра - определяется максимальный по абсолютному значению элемент строки матрицы результатов, который указывает на

начало пакета (при синхронизации) или на бинарный номер строки матрицы Уолша (при сужении спектра и демодуляции).

4. Ход работы

Код программы представлен ниже.

Для начала была взята псевдослучайная последовательность (ПСП):

```

1 PRS=[1; 1; 1; 1; 1; 1; -1; -1; -1; -1; 1; 1; 1; 1; -1; 1; 1; 1; -1;
  ↪ -1; -1; -1; 1; -1; 1; 1; -1; -1; 1; 1; -1; 1;
2      1; -1; 1; 1; 1; 1; -1; 1; -1; -1; -1; -1; 1; 1; 1; -1; -1; 1;
  ↪ 1; -1; -1; -1; -1; 1; -1; -1; 1; -1; -1; -1; 1; -1;
3      1; -1; 1; 1; 1; -1; 1; -1; 1; 1; 1; 1; -1; -1; 1; -1; -1; 1;
  ↪ -1; 1; 1; 1; -1; -1; 1; 1; 1; -1; -1; -1; -1; -1;
4      -1; 1; 1; 1; -1; 1; 1; 1; -1; 1; -1; -1; 1; 1; 1; 1; -1; 1;
  ↪ -1; 1; -1; -1; 1; -1; 1; -1; -1; -1; -1; -1; -1; 1;
5      -1; 1; -1; 1; -1; 1; -1; 1; -1; 1; 1; 1; -1; 1; -1; 1; -1;
  ↪ 1; -1; -1; -1; -1; -1; 1; 1; -1; 1; 1; 1; -1; 1;
6      1; -1; 1; 1; -1; 1; -1; 1; -1; 1; -1; -1; -1; -1; 1; -1; 1; 1;
  ↪ 1; -1; 1; 1; 1; 1; 1; -1; -1; -1; 1; 1; 1; 1;
7      -1; -1; 1; 1; -1; 1; -1; -1; 1; 1; -1; 1; -1; 1; 1; 1; -1; -1;
  ↪ -1; 1; 1; -1; 1; -1; -1; 1; -1; 1; -1; 1; 1; 1;
8      1; 1; 1; -1; 1; -1; -1; 1; -1; 1; 1; -1; -1; -1; 1; -1; 1; -1;
  ↪ -1; 1; 1; -1; -1; -1; 1; 1; -1; -1; -1; -1; -1;
9      -1; 1; 1; -1; -1; 1; 1; -1; -1; 1; -1; 1; -1; 1; 1; -1; -1; 1;
  ↪ -1; -1; 1; 1; 1; 1; 1; -1; 1; 1; -1; 1; -1;
10     -1; 1; -1; -1; 1; -1; -1; 1; 1; -1; 1; 1; 1; 1; 1; 1; -1; -1;
  ↪ 1; -1; 1; 1; -1; 1; -1; 1; -1; -1; -1; 1; -1;
11     1; -1; -1; -1; 1; -1; -1; 1; 1; 1; -1; 1; 1; -1; -1; 1; -1; 1;
  ↪ 1; 1; 1; -1; 1; 1; -1; -1; -1; -1; 1; 1; -1; 1;
12     -1; 1; -1; 1; -1; -1; 1; 1; 1; -1; -1; 1; -1; -1; -1; -1; 1; 1;
  ↪ -1; -1; -1; 1; -1; -1; -1; 1; -1; -1; -1; -1; -1;
13     -1; -1; -1; 1; -1; -1; -1; 1; -1; -1; -1; 1; 1; -1; -1; 1; -1; -1;
  ↪ -1; 1; 1; 1; -1; 1; -1; 1; -1; 1; 1; -1; 1; 1;
14     -1; -1; -1; 1; 1; 1; -1; -1; -1; 1; -1; -1; 1; -1; 1; -1; 1; -1;
  ↪ -1; -1; 1; 1; -1; 1; 1; -1; -1; 1; 1; 1; 1; 1;
15     -1; -1; 1; 1; 1; 1; -1; -1; -1; 1; -1; 1; 1; -1; 1; 1; 1; -1;
  ↪ -1; 1; -1; 1; -1; -1; 1; -1; -1; -1; -1; -1; 1; -1;
16     -1; 1; 1; -1; -1; 1; 1; 1; -1; 1; -1; -1; -1; 1; 1; 1; 1; 1;
  ↪ -1; 1; 1; 1; 1; -1; -1; -1; -1; -1; 1; 1; 1 1 ];

```

Затем взята последовательность перемежения:

```

1 interleaver = [0; 133; 122; 111; 100; 89; 78; 67; 56; 45; 34; 23; 12; 1; 134;
  ↪ 123;
2 112; 101; 90; 79; 68; 57; 46; 35; 24; 13; 2; 135; 124; 113; 102; 91;
3 80; 69; 58; 47; 36; 25; 14; 3; 136; 125; 114; 103; 92; 81; 70; 59;
4 48; 37; 26; 15; 4; 137; 126; 115; 104; 93; 82; 71; 60; 49; 38; 27;
5 16; 5; 138; 127; 116; 105; 94; 83; 72; 61; 50; 39; 28; 17; 6; 139;
6 128; 117; 106; 95; 84; 73; 62; 51; 40; 29; 18; 7; 140; 129; 118; 107;
7 96; 85; 74; 63; 52; 41; 30; 19; 8; 141; 130; 119; 108; 97; 86; 75;
8 64; 53; 42; 31; 20; 9; 142; 131; 120; 109; 98; 87; 76; 65; 54; 43;
9 32; 21; 10; 143; 132; 121; 110; 99; 88; 77; 66; 55; 44; 33; 22; 11];

```

Необходимо получить сигнал (test.sig), который заранее был содан с помощью *transmittertract.m*:

```

1 file=fopen('C:\Users\shus\Desktop\8\test.sig', 'r');
2 IQ_record = fread(file, 'int16');
3 fclose(file);

```

По заданию передискретизация равна 2, отсчеты дублируются подряд (вещественная часть - нечетные числа, комплексная часть - четные)

```
1 IQ_record= IQ_record(81:end)';
2 imag_=_IQ_record(2:2:end);
3 real_=_IQ_record(1:2:end);
```

Затем возвращаемся в комплексную форму:

```
1 IQ_record = complex(real,imag);
2
3 IQ_record = IQ_record(1:2:end);
4 size(IQ_record);
```

Производим демодуляцию сигнала:

```
1 IQ=pskdemod(IQ_record,2)
```

Преобразование униполярной формы в биполярную:

```
1 for u=1:length(IQ)
2     if (IQ(u)==0)
3         IQ(u)=-1;
4     else IQ(u)=1;
5     end;
6 end;
7
8 demod_sig2=IQ(length(PRS)+1:end);
9 demod_sig=demod_sig2./[PRS' _PRS' PRS' _PRS(1:3)'];
```

Затем используем готовый блок для генерации матрицы:

```
1 % Walsh matrix generation by Hadamard matrix index rearrangement
2 N=64;
3 hadamardMatrix=hadamard(N);
4
5 HadIdx = 0:N-1; % Hadamard index
6 M = log2(N)+1;
7
8 binHadIdx = fliplr(dec2bin(HadIdx,M))-'0'; % Bit reversing of the binary index
9 binSeqIdx = zeros(N,M-1); % Pre-allocate memory
10 for k = M-1:2
11     % Binary sequency index
12     binSeqIdx(:,k) = xor(binHadIdx(:,k),binHadIdx(:,k-1));
13 end
14 SeqIdx = binSeqIdx*pow2((M-1:-1:0)'); % Binary to integer sequency index
15 walshMatrix=_hadamardMatrix(SeqIdx+1,:); % 1-based indexing
16
17 _signal2=reshape(demod_sig,[64_24]);
```

Получаем значения 6-ти битных символов:

```
1 for j=1:size(signal2,1)
2     for i=1:length(walshMatrix)
3         if(walshMatrix(i,:)==signal2(j,:))
4             Walsh_row_num(j)=i;
5         end
6     end
7 end
```

Из 10-го числа в 2ый код (reshape - преобразование размеров массив; de2bi - преобразование чисел в векторы цифр):

```

1  for i=1:1:24
2      line(i,1:6)= de2bi( Walsh_row_num(i)-1,6);
3      line(i,1:6)=line(i,end:-1:1);
4  end;
5
6  signal=reshape(line',[1_144]);
7
8
9  for _i=1:1:144
10     sig2(interleaver(i)+1)=signal(i);
11 end

```

Функция poly2trellis принимает на входе полиномиальное описание сверточного кода и возвращает структуру, содержащую соответствующую таблицу переходов.

Функция vitdec производит декодирование вектора sig2 с помощью алгоритма Витерби.

Функция biterr вычисляет числа ошибочных бит и вероятности ошибки на бит.

```

1  tr1=poly2trellis(9,[753 561]);
2  tblen=9;
3  decoded=vitdec(sig2,tr1,9,'cont','hard');
4  message=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0
    ↪ 0 0 0 1 0 1 1 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];
5  [decoded,message] = biterr(decoded(tblen+1:end),message(1:end-tblen))

```

Исходное и декодированное сообщения совпадают:

decoded =

0

message =

0

Рис. 4.0.1. Сравнение

5. Выводы

В данной работе был реализован алгоритм для приемника сигналов. Проведено тестирование в среде Matlab (успешно).