Санкт-Петербургский государственный технический университет
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе №8
Модель телекоммуникационного канала

Телекоммуникационные технологии

Кузьмин Н.А. 33501/1
Богач Н.В.

Санкт-Петербург 2018

Содержание

1 Цель

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 сто- роне 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 неизмен- ным. 3. Модуляция символов. На этом этапе пакет из 144 полученных с выхода перемежителя бит разбивается на 24 символа из 6 бит. Ге- нерируется таблица функций Уолша длиной 64 бита. Каждый 6- битный символ заменяется последовательностью Уолша, номер ко- торой 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 символьно- го пакета прикрепляется немодулированная ПСП. Т.о. символьная длина становится равной 1747. Далее полученные символы моду- лируются методом BPSK .

3 Ход работы

3.1 Чтение сообщения

```
%read file
fid=fopen('test.sig', 'rb');
message = fread(fid, 'int16')';
fclose(fid);
```

Для начала читаем принимаемое сообщение из файла. Далее, зная, что данные сгенерированы с передискретизацией, равной двум (что означает, что одному BPSK символу соответствуют два расположенных друг за другом отчета), считываем каждое второе значение.

```
l = length(message)/2;
IQ = zeros(1, l);
k = 1;
for(i=1:2:length(message))
    IQ(k) = message(i);
    IQ(k) = IQ(k) + 1j*(message(i+1));
```

```
    k = k + 1;
end
```

3.2 Демодуляция

Производим демодуляцию сигнала.

```
demodulated_signal = pskdemod(IQ, 2);
```

3.3 Модуляция символов

С помощью корреляции определяем положение синхропосылки PRS и восстанавливаем последовательность до добавления синхропосылки.

```
N = 2^nextpow2(length(demodulated_signal));
signalSpectr = fft(demodulated_signal, N);
syncSpectr = fft(PRS', N);

mult = signalSpectr.*conj(syncSpectr);
result = ifft(mult, N);
n = find(result >= 200);
signal_to_modulate2 = IQdemod((length(PRS)+n(1)):length(IQdemod));
signal_to_modulate1 = signal_to_modulate2.*[PRS' PRS' PRS' PRS(1:3)'];
```

Теперь нужно получить последовательность до применения функции Уолша. В вектор `walsh_row` записываем индексы из матрицы Уолша, а затем переписываем номера индексов как двоичную последовательность в матрицу `sig_matrix`.

```
signal1 = reshape(signal_to_modulate1, [64 24])';
N = 64;
hadamardMatrix = hadamard(N);

walsh_row = zeros(1,24);
for i=1:24
    ind=ismember(hadamardMatrix, signal1(i,:), 'rows');
    walsh_row(i) = find(ind == 1);
end
walsh_row = (walsh_row == 1)';
sig_matrix = de2bi(walsh_row, 6, 'left-msb');
```

3.4 Перемежение бит

Теперь необходимо выполнить перемежение бит.

```
sig = reshape(sig_matrix', [1 144]);
convcode(int16(interleaver+1)) = sig;
```

3.5 Декодирование

Осталось раскодировать посылку, закодированную сверточным кодом с образующими полиномами 753, 561(octal) и кодовым ограничением 9

```
trellis = poly2trellis(9, [753 561]);
packet = vitdec(convcode, trellis, 1, 'trunc', 'hard');

packetR = [packet(1:16); packet(17:32); packet(33:48)];
code = bi2de(packetR, 'left-msb');
```

После всех операций можно убедиться, что принятое сообщение совпало с переданным.

```
Исходное сообщение: 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 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 0
Принятое сообщение: 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 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 0
```

4 Вывод

В итоге была получена модель приемника, выполняющая демодуляцию, перемежение и декодирование принятого сигнала. Полученный сигнал полностью соответствует переданному сообщению.