НИУ ИТМО

Факультет программной инженерии и компьютерной техники

Отчет по Лабораторной работе №2    
по курсу “Информатика”

"Синтез помехоустойчивого кода"    
Вариант №53

Выполнила:

Студентка группы P3110

Бегинина Анастасия Алексеевна   
Преподаватели:

Балакшин Павел Валерьевич  
Рудникова Тамара Владимировна

Санкт-Петербург

2021

Оглавление

[Задания 3](#_Toc84977913)

[Основные этапы вычисления 4](#_Toc84977914)

[Вывод 6](#_Toc84977915)

[Список литературы 7](#_Toc84977916)

[Приложения 8](#_Toc84977917)

# Задания

1. Определить свой вариант задания с помощью номера в ISU (он же номер студенческого билета). Вариантом является комбинация 3-й и 5-й цифр. Т. е. если номер в ISU = 12**3**4**5**6, то вариант = 35.
2. На основании номера варианта задания выбрать набор из 4 полученных сообщений в виде последовательности 7-символьного кода.
3. Построить схему декодирования классического кода Хэмминга (7;4), которую представить в отчёте в виде изображения.
4. Показать, исходя из выбранных вариантов сообщений (по 4 у каждого – часть No1 в варианте), имеются ли в принятом сообщении ошибки, и если имеются, то какие. **Подробно прокомментировать** и записать правильное сообщение.
5. На основании номера варианта задания выбрать 1 полученное сообщение в виде последовательности 11-символьного кода.
6. Построить схему декодирования классического кода Хэмминга (15;11), которую представить в отчёте в виде изображения.
7. Показать, исходя из выбранного варианта сообщений (по 1 у каждого – часть No2 в варианте), имеются ли в принятом сообщении ошибки, и если имеются, то какие. **Подробно прокомментировать** и записать правильное сообщение.
8. Сложить номера всех 5 вариантов заданий. **Умножить полученное число на 4**. Принять данное число как число информационных разрядов в передаваемом сообщении. Вычислить для данного числа минимальное число проверочных разрядов и коэффициент избыточности.
9. Необязательное задания для получения оценки «5» (позволяет набрать от 86 до 100 процентов от максимального числа баллов БаРС за данную лабораторную). Написать программу на любом языке программирования, которая на вход из командной строки получает набор из 7 цифр «0» и «1», записанных подряд, анализирует это сообщение на основе классического кода Хэмминга (7,4), а затем выдает правильное сообщение (только информационные биты) и указывает бит с ошибкой при его наличии.

# Основные этапы вычисления

1. Номер ИСУ 335038. Вариант **53**

|  |  |  |  |
| --- | --- | --- | --- |
| 38 | 70 | 102 | 22 |

1. (см. Приложение 1)
2. Сообщение 38–**1010010**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| степень | r1 | r2 | i1 | r3 | i2 | i3 | i4 | S |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 2 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 4 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |

Вычислим синдром последовательности по формулам:

S1 = R1 ⊕ I1 ⊕ I2 ⊕ I4

S2 = R2 ⊕ I1 ⊕ I3 ⊕ I4

S3 = R3 ⊕ I2 ⊕ I3 ⊕ I4

Получили синдром последовательности S = (0, 0, 1). Так как синдром ≠ (0, 0, 0), то в последовательности произошли искажения.  
 reverse (001) = 100\_2 = 4\_10. Значит ошибка в четвертом бите (r3). Инвертируем этот бит, полученное сообщение: **1011010**

2) Сообщение 70–**1111100**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| степень | r1 | r2 | i1 | r3 | i2 | i3 | i4 | S |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 2 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 4 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

Вычислим синдром последовательности по формулам:

S1 = R1 ⊕ I1 ⊕ I2 ⊕ I4

S2 = R2 ⊕ I1 ⊕ I3 ⊕ I4

S3 = R3 ⊕ I2 ⊕ I3 ⊕ I4

Получили синдром последовательности S = (1, 0, 0). Так как синдром ≠ (0, 0, 0), то в последовательности произошли искажения.  
 reverse (100) = 001\_2 = 1\_10. Значит ошибка в первом бите (r1). Инвертируем этот бит, полученное сообщение: **0111100**

3) Сообщение 102–**0100111**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| степень | r1 | r2 | i1 | r3 | i2 | i3 | i4 | S |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 2 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |

Получили синдром последовательности S = (0, 1, 1). Так как синдром ≠ (0, 0, 0), то в последовательности произошли искажения.  
 reverse (011) = 110\_2 = 6\_10. Значит ошибка в шестом бите (i3). Инвертируем этот бит, полученное сообщение: **0100101**

4)  Сообщение 22–**1000001**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| степень | r1 | r2 | i1 | r3 | i2 | i3 | i4 | S |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 2 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 4 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

Получили синдром последовательности S = (0, 1, 1). Так как синдром ≠ (0, 0, 0), то в последовательности произошли искажения.  
 reverse (011) = 110\_2 = 6\_10. Значит ошибка в шестом бите (i3). Инвертируем этот бит, полученное сообщение: **1000011**

1. 0 1 0 0 0 1 1 0 1 0 1 0 0 1 1
2. (см. Приложение 2)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| степень | r1 | r2 | i1 | r3 | i2 | i3 | i4 | r4 | i5 | i6 | i7 | i8 | i9 | i10 | i11 | S |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 2 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 4 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 8 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |

Определяем синдром последовательности:

S1 = R1 ⊕ I1 ⊕ I2 ⊕ I4 ⊕ I5 ⊕ I7 ⊕ I9 ⊕ I11

S2 = R2 ⊕ I1 ⊕ I3 ⊕ I4 ⊕ I6 ⊕ I7 ⊕ I10 ⊕ I11

S3 = R3 ⊕ I2 ⊕ I3 ⊕ I4 ⊕ I8 ⊕ I9 ⊕ I10 ⊕ I11

S4 = R4 ⊕ I5 ⊕ I6 ⊕ I7 ⊕ I8 ⊕ I9 ⊕ I10 ⊕ I11

Получили синдром последовательности S = (0, 0, 0, 0). Так как синдром = (0, 0, 0, 0), то в последовательности не произошло искажения.  
Полученное сообщение: **010001101010011**

1. 1) (38 + 70 + 102 + 22 + 53) \* 4 = 1140 – число инф. разрядов

2) 2r ≥ r + i + 1 => min r = 11 – вычисляем число пров. разрядов

3) r / n = r / (r + i) Откуда r/n ~ **0.00955691**

9.

def hamming(message):

bits = list(message)

bits = [int(i) for i in bits]

s1 = bits[0] ^ bits[2] ^ bits [4] ^ bits[6]

s2 = bits[1] ^ bits[2] ^ bits [5] ^ bits[6]

s3 = bits[3] ^ bits[4] ^ bits [5] ^ bits[6]

ans = ""

if [s1, s2, s3] != [0, 0, 0]:

res = ''

mistake = int(str(s3) + str(s2) + str(s1), 2) - 1 #mistake position

for i in range(len(bits)):

if i != mistake:

res += str(bits[i])

else:

res += str((bits[i] + 1) % 2) #reverse operation

for i in range(len(res)):

if i not in [0, 1, 3]:

ans += res[i]

return "There was a mistake in {} bit".format(mistake + 1) + "\nCorrect information bits are {}".format(ans)

return res

else:

for i in range(len(message)):

if i not in [0, 1, 3]:

ans += res[i]

return "The message is correct. Information bits are {}".format(ans)

print(hamming("1010010"))

# Вывод

В ходе выполнения лабораторной работы я научилась работать со схемами Хэмминга (7;4) и (15;11). Я научилась определять биты информации переданные ошибочно и исправлять их на верные.

# Список литературы

1. Код Хэмминга // Wikipedia.com: интернет-энциклопедия, 2001 г. URL: https://ru.wikipedia.org/wiki/Код\_Хэмминга (дата обращения: 13.10.2021).
2. Создание диаграмм // app.diagrams.net: создание блок-схем и диаграмм. 2012. URL: https://app.diagrams.net (дата обращения: 13.10.2021).

# Приложения

Приложение 1.

Diagram, schematic

Description automatically generated

Приложение 2.

Diagram, schematic

Description automatically generated