НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Дисциплина

Лабораторная работа № 2

Выполнил студент

Круглов Егор Ильич

Группа № P3124

Преподаватель: Болдырева Елена Александровна

г. Санкт-Петербург

2022

**Оглавление**

[Вариант 2](#_Toc115619090)

[Задание: 2](#_Toc115619091)

[Доп. задание: 2](#_Toc115619092)

[Отчет: 3](#_Toc115619093)

[Вывод: 7](#_Toc115619094)

[Список литературы: 7](#_Toc115619095)

Вариант**: 72**

# Задание:

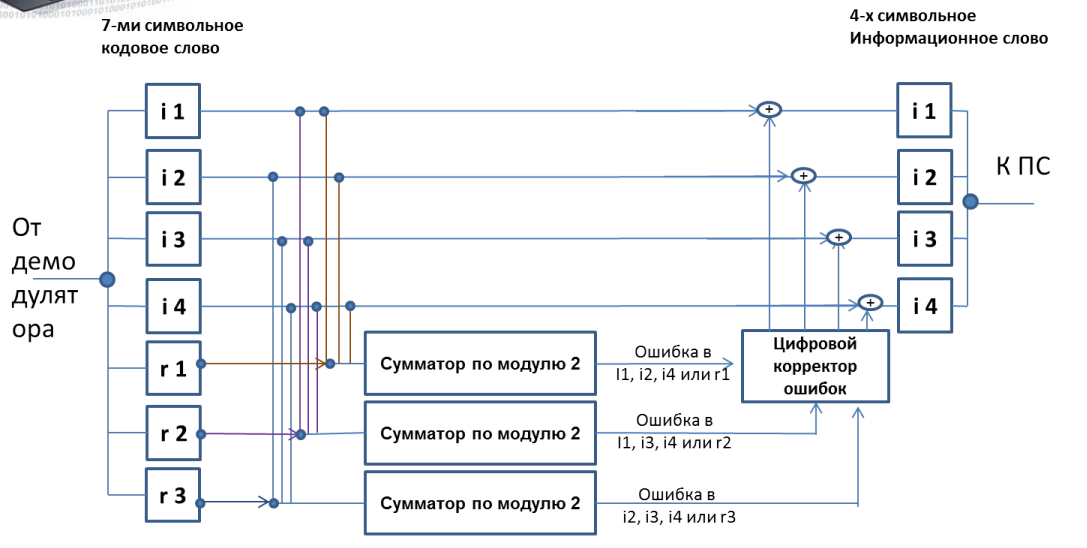
Познакомиться с принципом помехоустойчивого кода на примере классического кода Хэмминга. На практике проверить, имеются ли в переданном сообщении ошибки, и если да, то какие.

# Доп. задание:

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.

# Отчет:

1. Схема декодирования классического кода Хэмминга (7;4).



1. Проверим, есть ли ошибки в первом сообщении. Возьмём сообщение под номером 54:



Для этого посчитаем s1, s2, s3:

s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 1 ⊕ 0 ⊕ 0 ⊕ 1 = 0.

S2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 1 ⊕ 0 ⊕ 1 ⊕ 1 = 1.

S3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 1 ⊕ 0 ⊕ 1 ⊕ 1 = 1.

Получилось, что S = s3s2s1 = 1102 = 610, значит ошибка во 6-й ячейке, то есть в i3. Инвертируем значение в i3, получим запись вида 11010012.

1. По аналогии проверим сообщение 91:



Посчитаем s1, s2, s3:

s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 0 ⊕ 1 ⊕ 1 ⊕ 0 = 0.

S2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 1 ⊕ 1 ⊕ 1 ⊕ 0 = 1.

S3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 1 ⊕ 1 ⊕ 1 ⊕ 0 = 1.

S = 1102 = 610. Ошибка в 6-й ячейке, то есть в i3. Правильная запись 01111002.

1. По аналогии проверим сообщение 16.



Посчитаем s1, s2, s3:

s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 0 ⊕ 0 ⊕ 0 ⊕ 1 = 1.

S2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 0 ⊕ 0 ⊕ 0 ⊕ 1 = 1.

S3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 1 ⊕ 0 ⊕ 0 ⊕ 1 = 0.

S = 0112 = 310. Ошибка в 3-й ячейке, то есть в i1. Правильная запись 00110012.

1. По аналогии проверим сообщение 51:



Посчитаем s1, s2, s3:

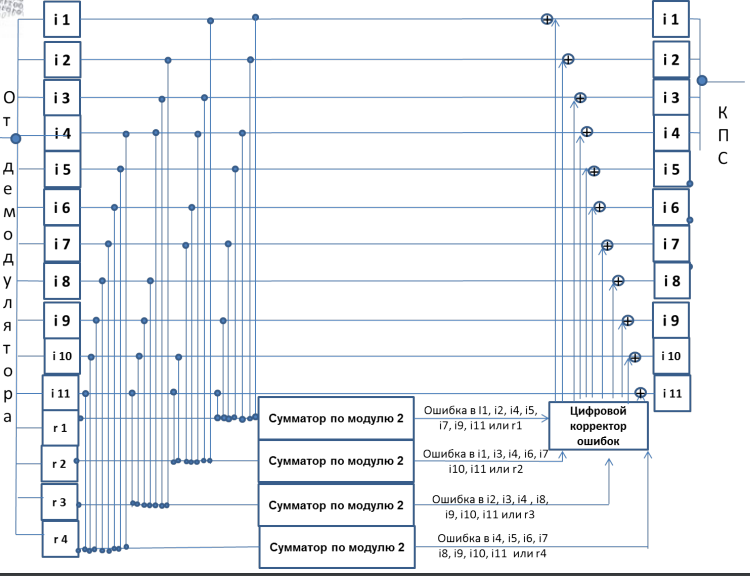
s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 1 ⊕ 1 ⊕ 0 ⊕ 1 = 1.

S2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 0 ⊕ 1 ⊕ 1 ⊕ 1 = 1.

S3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 0 ⊕ 0 ⊕ 1 ⊕ 1 = 0.

S = 0112 = 310. Ошибка в 3-й ячейке, то есть в i1. Правильная запись 10000112.

1. Схема декодирования классического кода Хэмминга (15;11).



1. Проверим есть ли в сообщении 71 ошибки.



Посчитаем s1, s2, s3, s4:

s1 = r1 ⊕ i1 ⊕ i2 ⊕ i4 ⊕ i5 ⊕ i7 ⊕ i9 ⊕ i11 = 0 ⊕ 1 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 1 ⊕ 1 ⊕ 0 = 1.

S2 = r2 ⊕ i1 ⊕ i3 ⊕ i4 ⊕ i6 ⊕ i7 ⊕ i10 ⊕ i11 = 0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 1 ⊕ 0 ⊕ 0 = 1.

S3 = r3 ⊕ i2 ⊕ i3 ⊕ i4 ⊕ i8 ⊕ i9 ⊕ i10 ⊕ i11 = 1 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 = 1.

S4 = r4 ⊕ i5 ⊕ i6 ⊕ i7 ⊕ i8 ⊕ i9 ⊕ i10 ⊕ i11 = 0 ⊕ 1 ⊕ 1 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 = 0.

Получилось, что S = s4s3s2s1 = 01112 = 710. Значит, ошибка в 7-й ячейке, то есть в i4. Инвертируем i4 и получим верную запись 0011101011101002.

1. Сложим все проверенные числа и умножим их на 4: (54+91+16+51+71)\*4 = 283 \* 4 = 1132. Примем это число за количество информационных разрядов в передаваемом сообщении и определим минимальное число проверочных разрядов и коэффициент избыточности. Для нахождения r воспользуемся формулой:

В нашем случае i = 1132, r будем подбирать. Начнём с 11, так как 11 – это минимальная степень двойки, необходимая для покрытия i = 1132. Проверим:

211 ? 1132 + 11 + 1

2048 ? 1144

2048 ≥ 1144

Значит, 11 – это минимальное число проверочных разрядов. Определим коэффициент избыточности по формуле:

Таким образом, минимальное число проверочных разрядов = 11, а коэффициент избыточности ≈ 0.0096.

1. Написал программу на языке программирования Python для дешифровки сообщений, закодированных в классическом коде Хэмминга (7;4). Прикрепляю код (<https://github.com/KruglovEgor/Hamming_74_on_python>):
2. code = input('Here: ')
3. code = '0'\*(**7**-len(code)) + code
4. s1, s2, s3 = sum([int(code[**0**]), int(code[**2**]),
5. int(code[**4**]), int(code[-**1**])]) % **2**, \
6. sum([int(code[**1**]), int(code[**2**]), int(code[-**2**]), \
7. int(code[-**1**])]) % **2**, \
8. sum([int(code[-**4**]), int(code[-**3**]), int(code[-**2**]), \
9. int(code[-**1**])]) % **2**
10. error = int(str(s3)+str(s2)+str(s1), **2**)
11. **if** error == **0**:
12. print('Everything is correct!')
13. print(code)
14. **else**:
15. code = code[:error-**1**] + str((int(code[error-**1**])\
16. + **1**) % **2**) + code[error:]
17. print('There is error in', error, 'position.
18. Right one is: ')
19. print(code)

# Вывод:

Познакомился с помехоустойчивым кодом на примере классического кода Хэмминга. Вручную расшифровал 5 сообщений, проверил их на наличие ошибок. Написал программу на Python для проверки сообщений, полученных в классическом коде Хэмминга (7;4).

# Список литературы:

1. Балакшин П.В., Соснин В.В., Машина Е.А. Информатика. – СПб: Университет ИТМО, 2020. – 122 с.
2. Изучаем Python, 3‑е издание – Пер. с англ. – СПб.: Символ‑Плюс, 2009. – 848 с., ил.