УДК 004.421: 004.773

д.т.н., доц. **Гунченко Ю.О.** (ОНУ імені І.І. Мечникова)
Ємельянов П.С. (ОНУ імені І.І. Мечникова)
Малахов В.Є. (ОНУ імені І.І. Мечникова)
Щербакова Т.О. (ОНУ імені І.І. Мечникова)

КОНЦЕПЦІЯ ПОБУДОВИ КОМПЛЕКСУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ДОСЛІДЖЕННЯ ЗАВАДОСТІЙКИХ КОДІВ

У статті запропоновано концепцію побудови програмного комплексу для дослідження завадостійких кодів з наявністю і без наявності помилок у повідомленні. Розглянуто необхідні модулі комплексу програмного забезпечення, які повинні здійснювати кодування, декодування повідомлень, внесення помилок, модулі зберігання, обробки та виводу даних, наведено умови і алгоритми їх функціонування. Показано доцільність використання OLAP-систем для зберігання і обробки отриманих даних. Наведено результати функціонування комплексу програмного забезпечення, створеного згідно концепції, що пропонується, на прикладі дослідження характеристик чотирьох кодів двох різних типів.

Ключові слова: завада, перешкода, завадостійкий код, програмне забезпечення, алгоритм побудови програмного модуля, база даних, OLAP-система.

Вступ. Перешкоди та завади є небажаними факторами деструктивного характеру у різних технічних системах, оскільки заважають нормальному функціонуванню таких систем. Вони примушують творців обчислювальних систем йти на певні міри, які призвані мінімізувати вплив завад на конкретну систему.

Маючи різну природу своєї появи, завади можуть змінювати дані, які передаються між різними обчислювальними системами, або у рамках однієї системи.

Завадостійкістю називається властивість системи максимально ефективно виконувати свої функції при наявності завад і перешкод. Завадостійкість оцінюють інтенсивністю завад, при яких порушення функцій системи (пристрою) ще не перевищує допустимих меж. Чим сильніша завада, при який система продовжує правильно й ефективно функціонувати, тим вище вважається її завадостійкість.

Огляд завадостійких пристроїв. Існує широкий спектр обладнання, пристроїв і устаткування, де протидія перешкодам і завадам є важливою умовою для їх правильного функціонування. Це можуть бути персональні комп'ютери, системи навігації, супутниковий зв'язок, балістичні ракети.

Особливо часто поняття «завадостійкість» застосовують для характеристики пристроїв передачі інформації (наприклад, лінії зв'язку) або пристроїв спостереження (наприклад, радіолокаційні станції). Для них у більшості випадків може бути визначено поняття «сигнал», і оцінка завадостійкості може проводитися шляхом розгляду співвідношення між завадою і сигналом, при якому забезпечується необхідна для нормального функціонування якість. Наприклад, у радіолокації – співвідношення сигналу к заваді, при якому забезпечується задана достовірність виявлення (вірогідність правильного виявлення при визначеної ймовірності помилкової тривоги).

Побудова оптимальних пристроїв, які реалізують необхідний рівень завадостійкості, зазвичай дуже складно, а їх неминучі технічні недосконалості не дозволяють досягти цей рівень у повному обсязі. Тому частіше задовольняються тими пристроями, які при найбільший їх простоті забезпечують добре приближення до оптимального рівня.

Існує багато способів зменшення впливу завад на данні, що передаються або зберігаються. Найбільш частіше використовують засоби завадостійкого кодування на основі певних завадостійких кодів.

Ціллю роботи є розробка концепції і принципів побудови програмного забезпечення для дослідження методів кодування та декодування інформації при наявності і відсутності помилок. Такий програмний комплекс дозволить проводити віртуальні дослідження систем, що розробляються, без їх безпосереднього натурного макетування або створення опитних зразків.

Вимоги до комплексу. Комплекс програмного забезпечення повинен служити основою для аналізу кодів, а саме таких характеристик коду, як час кодування, розмір кодового слова і довжина вихідної символьної послідовності. Слід відмітити, що даний комплекс призначено для порівняння різних кодів між собою.

До його функційних можливостей повинно входити порівняння різних кодів в однакових умовах. Такий підхід дозволяє отримати адекватну і коректну інформацію о різнице між можливостями кодів, в залежності від їх властивостей.

Даний комплекс пропонується реалізовувати не у вигляді монолітної системи, а у вигляді декількох дискретних модулів, які зв'язані між собою, де кожний модуль буде виконувати своє призначення. Взаємодія між модулями повинна бути частково автоматичною, а частково підтримуватися оператором. Таке рішення дозволяє зробити програмний комплекс більш гнучким, оскільки подібна реалізація сприяє швидкому корегуванню при зміні функціоналу той або іншої частини комплексу. Таким чином, впровадження змін спричинить відносно менші витрати, ніж впровадження змін в єдину монолітну систему.

Алгоритм функціонування. Опишемо життєвий цикл повідомлення, яке оброблюється програмним комплексом, у вигляді алгоритму:

1) Повідомлення у символьному вигляді подається в модуль, який здійснює кодування (модуль «КОДЕР»).

2) Модуль «КОДЕР» створює бінарне представлення символів повідомлення на основі їх порядкового номеру в таблиці ASCII.

3) Модуль «КОДЕР» виконує відповідні потрібному коду (наприклад, коду Геммінга) алгоритмічні операції й записує дані о довжині слова, що кодується, і часу кодування у лог-файл.

4) «КОДЕР» посилає отриману закодовану бінарну послідовність модулю «ДЕКОДЕР» через канал зв'язку.

5) В залежності від необхідності, спрацьовує блок генерації помилок.

6) Модуль «ДЕКОДЕР» оброблює отримане повідомлення за необхідним алгоритмом и заносить до лог-файлу дані, аналогічні даним сторони, що кодує.

7) На основі лог-файлів модулів «КОДЕР» і «ДЕКОДЕР» реалізується запис у базу даних.

8) База даних оброблюється за допомогою функціональних можливостей OLAP-аналізатора.

9) Реалізується графічне представлення отриманих даних.

На рис. 1 представлена схема, яка демонструє взаємодію модулів.

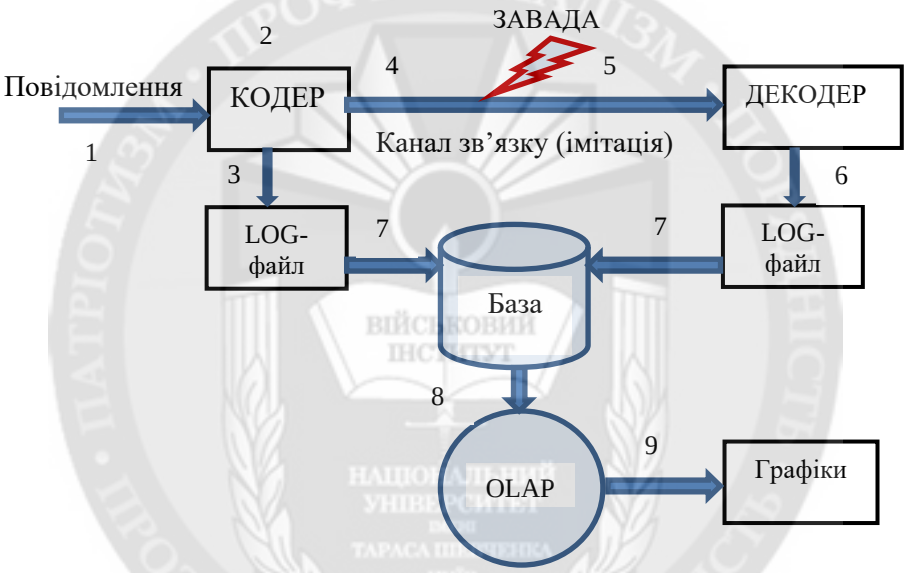

Рис. 1. Функціональна схема програмного комплексу аналізу кодів

Проектування модуля «КОДЕР». Модуль кодування повідомлення, за своєму призначенню повинен виконувати наступні функції:

- переклад символьної послідовності у бінарну послідовність, яка складається з символів **1** и **0**;
- кодування бінарної послідовності по відповідному алгоритму і передача закодованої послідовності через канал зв'язку, що імітується, а саме до сторони, що декодує;
- вивід даних о послідовності, що кодується, і часу, витраченому на процес кодування даної послідовності.

У реальних умовах, операції кодування і декодування найчастіше реалізується за допомогою апаратних засобів. У програмному комплексі аналіз різних типів кодів, що використовуються для завадостійкого кодування буде досліджується на програмному рівні, що імітує апаратну частку.

Передбачається, що три вищенаведені функціональні можливості даного модуля будуть реалізовані за допомогою механізму використання функцій, а саме написання дискретних, логічне цілісних ділянок програмного коду, які згодом можна багаторазово викликати, передаючи їм на обробку необхідні дані, або всіляко редагувати і змінювати.

Такий метод дозволить виконати у вигляді програмного забезпечення весь спектр функціональних можливостей даного модуля.

Проектування модуля генерації помилок. В умовах програмного комплексу модуль генерації помилок є необхідним елементом для реалізації аналізу кодів. При передачі інформації в реальних умовах через реальний канал зв'язку, інформація може змінюватися під впливом завад, що, в свою чергу, може спричинити зміни символів в послідовності, що передається. Оскільки даний комплекс не передбачає інформаційного обміну між сторонами, що кодують і декодують, за допомогою каналу зв'язку, то закодована послідовність позбавлена ризику бути зміненою.

Це добре при апаратному кодуванні, однак, оскільки даний програмний комплекс розробляється для аналізу і порівняння кодів і їх можливостей, то більш широкий спектр інформації для аналізу можна отримати, маючи дані про те, як коди виправляють помилки.

Виходячи з вищевказаного, з'являється необхідність генерації помилок для перевірки часу роботи алгоритму при їх виправленні. Генератором цих помилок і буде служити модуль генерації помилок. Обробка переданого закодованого повідомлення буде здійснюватися блоком генерації помилок за наступним алгоритмом:

- закодована послідовність надходить від модуля «КОДЕР» на блок генерації помилок;
- блок генерації помилок змінює інформаційний біт закодованого повідомлення;
- блок генерації помилок передає закодовану бінарну послідовність наступному модулю «ДЕКОДЕР»

Даний алгоритм досить простий, однак слід відзначити декілька моментів. Використання блоку генерації помилок буде застосовуватися до закодованої послідовності строго в необхідних випадках. Тобто передбачається, що кожна закодована послідовність буде передаватися спочатку без помилки, а потім з помилкою в інформаційному біті, що згенеровано у результаті роботи модуля генерації помилок. Згодом, результати роботи коду при декодуванні закодованої бінарної послідовності без помилки і послідовності із помилкою, що згенеровано, будуть фіксуватися в лог-файлі при роботі модуля «КОДЕР».

Проектування модуля «ДЕКОДЕР». Модуль декодування закодованої бінарної послідовності є модулем, який отримує дані або від модуля «КОДЕР», або від модуля генерації помилок.

У зв'язку зі своїм призначенням модуль «ДЕКОДЕР» повинен виконувати наступні функції:

- одержувати на вхід закодовану бінарну послідовність від блоку генерації помилок або безпосередньо від модуля «ДЕКОДЕР» минаючи блок генерації помилок;
- реалізовувати декодування отриманої закодованої бінарної послідовності, що складається з символів 1 і 0, відповідно до необхідного алгоритму декодування;
- подавати на вихід декодовану послідовність символів і заносити необхідні дані в лог-файл.

Слід зазначити, що алгоритм декодування, повинен відповідати алгоритму кодування не тільки за своїм типом (наприклад, Код Геммінга або код CRC), але і за ключовими характеристиками, таким як довжина кодового слова і кількість контрольних символів. Дані, що заносяться в лог-файл модулем «ДЕКОДЕР» повністю є такими самими, що заносяться модулем «КОДЕР» в свій лог-файл, а саме – довжина слова і час декодування. Згодом ці дані

будуть необхідні для аналізу та порівняння ефективності тих чи інших кодів з різними характеристиками.

Проектування бази даних и модуля запису до неї. Одна з ключових частин програмного комплексу це база даних, яка міститиме інформацію з лог-файлів модулів «КОДЕР» і «ДЕКОДЕР». В один модуль разом із базою даних повинне входити і програмне забезпечення, що реалізує заповнення бази даних на основі лог-файлів.

Насамперед необхідно визначитися зі структурою бази даних, а саме з її таблицями і полями в цих таблицях. База даних складається з п'яти таблиць. Це таблиці (наприклад) **Sequenses**, **CodeTypes**, **Encodes**, **Decodes** і **DecodesEr**, призначені для запису в них даних послідовностей для кодування, алгоритмах і результатів кодування, декодування і декодування з помилкою, відповідно. Саме в ці таблиці повинна заноситися необхідна для аналізу інформація.

Структуру бази даних, представлено на рис. 2.

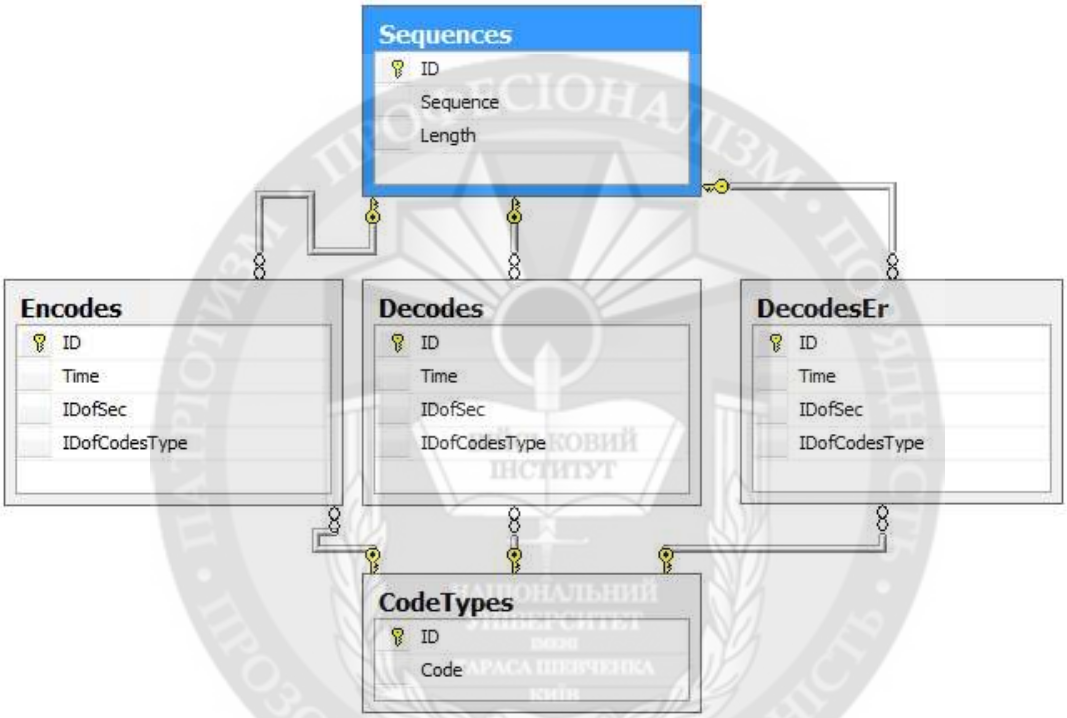

Рис. 2. Структура бази даних

Розглянемо таблиці **Sequenses** (табл. 1), **CodeTypes** (табл. 2), **Encodes** (табл. 3), **Decodes** (табл. 4) и **DecodesEr** (табл. 5) з їх полями, типами даних і опис полій.

Таблиця 1 – схема таблиці Sequences

№	**Ім'я поля**	**Тип даних**	**Описання**
1.	ID **<pk>**	int	Поле унікального ідентифікатора
2.	Sequence	nvarchar(1100)	Поле, яке містить текст послідовності, що кодується
3.	Length	int	Поле, яке містить довжину послідовності, що кодується

Таблиця 2 – схема таблиці CodeTypes

№	Ім’я поля	Тип даних	Описання
1.	ID <**pk**>	int	Поле унікального ідентифікатора
2.	Code	nvarchar(100)	Поле, яке містить назву типу алгоритму

Таблиця 3 – схема таблиці Encodes

№	Ім’я поля	Тип даних	Описання
1.	ID <**pk**>	int	Поле унікального ідентифікатора
2.	Time	float	Поле, яке містить значення часу, що витрачено на кодування послідовності
3.	IDofSec	int	Поле зовнішнього ключа. Служить для зв’язку часу кодування и послідовності, що кодується
4.	IDofCodesType	int	Поле зовнішнього ключа. Служить для зв’язку часу кодування і типу алгоритму кодування

Таблиця 4 – схема таблиці Decodes

№	Ім’я поля	Тип даних	Описання
1.	ID <**pk**>	int	Поле унікального ідентифікатора
2.	Time	float	Поле, яке містить значення часу, витраченого на декодування послідовності
3.	IDofSec	int	Поле зовнішнього ключа. Служить для зв’язку часу декодування і послідовності, що декодується
4.	IDofCodesType	int	Поле зовнішнього ключа. Служить для зв’язку часу декодування і типу алгоритму декодування

Таблиця 5 – схема таблиці DecodesEr

№	Ім’я поля	Тип даних	Описання
1.	ID <**pk**>	int	Поле унікального ідентифікатора
2.	Time	float	Поле, яке містить значення часу, що витрачено на декодування послідовності
3.	IDofSec	int	Поле зовнішнього ключа. Служить для зв’язку часу декодування з помилкою і послідовності, що декодується
4.	IDofCodesType	int	Поле зовнішнього ключа. Служить для зв’язку часу декодування з помилкою і типу алгоритму декодування

Також, необхідно розписати функціональні можливості програми, призначеної для заповнення бази даних наявною інформацією. Зробимо це у вигляді алгоритму:

- програма повинна зчитувати дані з лог-файлів;
- наводити дані, що зчитано, у відповідний для запису в базу даних формат;
- ініціювати підключення до бази даних;
- здійснювати запис у відповідні поля таблиць.

Такий алгоритм, дозволить максимально надійно розподілити дані з лог-файлів у відповідні табличні поля. Також, використання об'єктно-орієнтованої мови в даному модулі,

дозволить швидко і на відносно високому рівні здійснити запис необхідних даних в базу даних.

Проектування засобів OLAP. OLAP-системи дозволяють працювати з даними, а саме представляти їх в агрегованому вигляді, для подальшої зручності роботи з ними. Також, OLAP-системи допомагають прийняти необхідне рішення, яке повинно бути прийнято в майбутньому, ґрунтуючись на вже наявних статистичних даних. Найчастіше OLAP-системи не є самостійними рішеннями, а інтегруються в інші системи, наприклад, в системи підтримки прийняття рішень. [11]

Як сховище даних для OLAP-систем можуть використовуватися джерела даних різного типу. Для програмного модулю, що пропонується буде використовуватися реляційна база даних MsSQL.

Проектування засобів візуального представлення даних. Заключним етапом у процесі проектування даного програмного комплексу є питання візуального представлення даних.

Даний етап найбільш простий з точки зору реалізації, так як доцільне використання готового програмного забезпечення.

Частину, що демонструє роботу різних типів кодів, раціонально реалізувати у вигляді графіків на основі отриманих даних.

Дані, які будуть представлені у вигляді графіків, будуть отримані з попереднього етапу, де реалізується OLAP-компонент даного комплексу, який, в свою чергу, буде займатися обробкою інформації з бази даних.

Практична реалізація концепції. Згідно з описаними вище вимогами був розроблений програмний комплекс для аналізу та тестуванню завадостійкого кодування.

Програмний комплекс реалізовано за допомогою продуктів однієї екосистеми – Microsoft. Це дозволило спростити і зробити більш ефективним взаємодію між модулями, оскільки продукти, які використовувалися для розробки модулів, мають добре налагоджену взаємодію. Мінусом такого підходу є необхідність наявності навичок, що дозволяють працювати з продуктами даної екосистеми.

Розробка модуля «КОДЕР» проведена за допомогою середовища розробки Visual Studio 2012 від компанії Microsoft на мові C++. Мова C++ часто використовується для програмування електронних пристроїв. Модуль генерації помилок реалізовано за допомогою того ж середовища розробки програмного забезпечення, що і модуль «КОДЕР» - Visual Studio. Це дозволило зменшити складності, що з'являються при налагодженні взаємодії модулів програмного комплексу. Для реалізації бази обрано СУБД Microsoft SQL Server 2012. Даний продукт добре інтегрований з використовуваним середовищем розробки Visual Studio 2012, а також володіє потужними засобами для роботи з базою даних.

Для роботи з OLAP використаємо службу SQL Server Microsoft Analysis Services. Вона добре інтегрована з наявною в проекті базою даних, а також дозволяє робити досить гнучкі конфігураційні налаштування для аналізу даних, і має можливості для їх зручної візуалізації.

Для візуалізації даних було вирішено використовувати програмний продукт Microsoft Office 2010, а саме Excel. Причини використання цього програмного забезпечення, обумовлюються хорошою інтеграцією продуктів Excel і SQL Server Analyses Services. Дані з OLAP-системи можна легко передавати в таблиці Excel, на підставі яких згодом, можна будувати інформативні графіки.

Такий підхід дозволив найменш витратним шляхом реалізувати програмний комплекс, без негативного впливу на його якість.

Результати функціонування і перевірка адекватності. Для перевірки адекватності запропонованого підходу до побудови програмного комплексу, з його допомогою було

проведено дослідження і аналіз двох типів кодів, що широко використовуються – це Код Геммінга і Інверсний код. Характеристики аналізованих кодів наведені в таблиці 6.

Таблиця 6

Характеристики аналізованих кодів

Тип коду	**Код Геммінга**	**Код Геммінга**	**Інверсний код**	**Інверсний код**
Кількість інформаційних бітів	8	16	8	16
Загальна кількість бітів	12	21	16	32
Кількість контрольних бітів	4	5	8	16
Специфіка роботи з помилками	Виправляє 1 помилку	Виправляє 1 помилку	Перевіряє наявність помилок	Перевіряє наявність помилок
Позначення	HEM(8,12,4)	HEM(16,21,5)	INV(8,16,8)	INV(16,32,16)

Таким чином, в порівнянні бере участь чотири коду двох типів. Також, Код Геммінга має здатність виправляти одну помилку, а інверсний код діє за принципом перевірки коректності отриманих даних.

Загальний аналіз коду, який використовується далі, являє собою порівняння часових характеристик кожного коду за такими характеристиками, як час кодування інформаційної символьної послідовності, час декодування закодованої послідовності і час декодування символьної послідовності з помилкою.

Результатом аналізу стали дані, наведенні у графічному вигляді на рис. 3 та рис. 4.

Рис. 3. Результати дослідження коду Геммінга

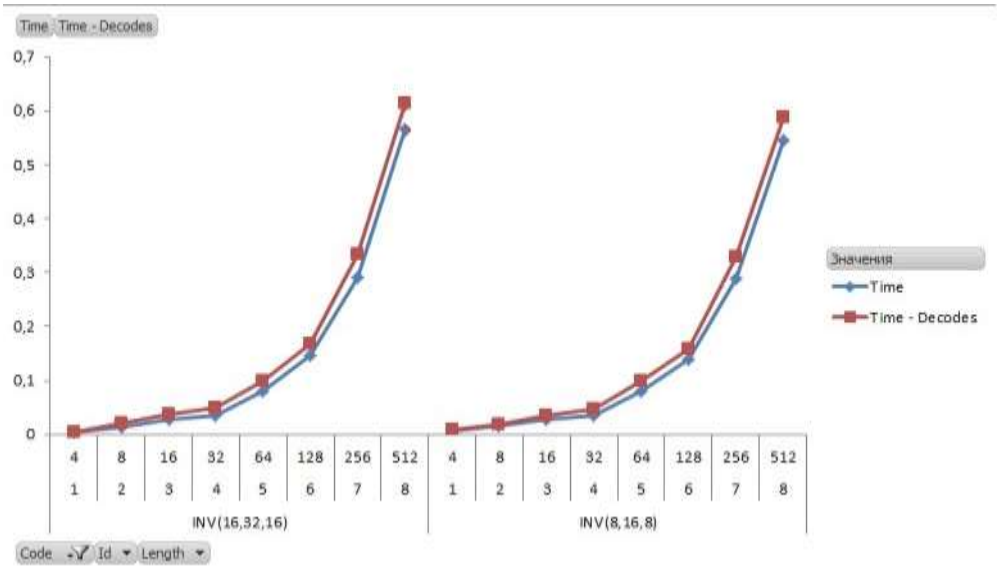

Рис. 4. Результати дослідження Інверсного коду

Порівняємо чотири коди двох типів, а саме код Геммінга і Інверсний код за такими параметрами, як час кодування, час декодування і час декодування з помилкою. Ці операції здійснювалися над вісьмома інформаційними символьними послідовностями різної довжини – від 2 до 512 бітів.

В результаті роботи програмного комплексу, було показано переваги і недоліки кожного з чотирьох кодів, наприклад код INV (16,32,16) за кількістю інформаційних і контрольних біт в два рази перевищує аналогічні параметри коду INV (8,16,8). Однак виконує необхідні операції майже за ті ж часові проміжки, що і код INV (8,16,8). Цим обумовлюється недоцільність використання даного типу кодів, в порівнянні з кодом Геммінга. Винятком може слугувати ситуація, при якій є необхідними специфічні можливості інверсний коду.

Також було визначено, що найменше часу на роботу витрачає код Геммінга HEM (16,21,5). Цей код в даній ситуації можна назвати найбільш прийнятним з точки зору часу виконання операцій кодування і декодування. Варто також відзначити, що код HEM (16,21,5) є версією коду HEM (8,12,4), але з чисельне збільшеними параметрами. Як показали результати дослідження даних кодів, код HEM (16,21,5) працює набагато швидше коду HEM (8,12,4). Це дозволяє вважати, що при збільшенні характеристик коду цього типу, зменшиться час кодування. Інакше кажучи, зі збільшенням числа інформаційних і контрольних біт, зросте і продуктивність даного коду.

Порівняння декількох завадостійких кодів за допомогою пропонованого програмного комплексу показало його адекватність. За допомогою комплексу були визначені переваги та недоліки кожного коду, а також їх відмінності.

Висновки і перспективи подальших досліджень. У даній роботі запропоновано концепцію побудови і реалізовано програмний комплекс для тестування і аналізу завадостійких кодів. У комплекс увійшли чотири компоненти, а також були задіяні ще два сторонніх компонента у вигляді програмних пакетів.

Цей програмний комплекс дозволив провести тестове порівняння чотирьох кодів двох типів, а саме двох реалізацій коду Геммінга і двох реалізацій Інверсного коду. В ході тестування і подальшого аналізу були виявлені переваги і недоліки досліджуваних кодів. Перевагою такого підходу є той факт, що аналіз проводився на основі реальних даних, отриманих в результаті роботи кожного коду.

Також, слід зазначити, що даний комплекс покликаний лише емулювати реальні процеси кодування і передачі даних, оскільки операції кодування і декодування реалізовані на програмному рівні. Існує ймовірність, що апаратна реалізація операцій кодування і декодування може відрізнятися від реалізації, представленої в даній роботі.

Однак при схожих підходах в реалізації як програмної, так і апаратної, слід очікувати схожий характер отриманих залежностей.

Крім цього, результати, отримані в даній роботі на підставі використання програмного комплексу для тестування і аналізу кодів, є відносними, оскільки були отримані в результаті роботи конкретного комп'ютера з конкретними технічними характеристиками. Тому справедливо стверджувати, що час, витрачений аналогічними кодами на операції кодування і декодування, може змінюватися в одну або іншу сторону, в залежності від технічних характеристик використовуваної платформи.

Одним з основних переваг концепції програмного комплексу є її універсальність з точки зору використання інших кодів. Це дозволяє тестувати, аналізувати і порівнювати різні реалізації різних завадостійких кодів в однакових умовах. Досягається такий результат за рахунок специфіки виклику модулів функцій, що кодують і декодують: механізм виклику функції майже ніяк не пов'язаний з реалізацією самої функції і дозволяє подавати одну й ту ж послідовність на вхід безлічі різних функцій.

Для подальших досліджень планується вдосконалити раніше використані алгоритми кодування і декодування, а також налагодити автоматизовану міжкомпонентну взаємодію, що потребує якомога меншого втручання користувача.

ЛІТЕРАТУРА:

1. Линейные блочные коды [Електронний ресурс] – Режим доступу: http://sernam.ru/book_tec.php?id=80.

2. Коды с обнаружением ошибок [Електронний ресурс] – Режим доступу: http://informkod.narod.ru/5_1item.htm.

3. Кутузаки А.С., Гунченко Ю.А. Анализ проблем разбиения на кадры данных // Тезисы доклада Десятой всеукраинской конференции студентов и молодых специалистов «ИНФОРМАТИКА, ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ». – Одесса: - 2013. – с. 193 – 195.

REFERENCES:

1. Lineynie blochnie kodi [Electronic resource] - Access mode: http://sernam.ru/book_tec.php?id=80.

2. Kodi s obnarujeniem oshibok [Electronic resource] - Access mode: http://informkod.narod.ru/5_1item.htm.

3. Kutuzaki A.S., Gunchenko Y.A. Analiz problem razbieniya na kadri dannih // Abstracts of the Tenth All-Ukrainian conference of students and young professionals "INFORMATICS, INFORMATION SYSTEMS AND TECHNOLOGY". – Odessa: - 2013. – p. 193 – 195.

Рецензент: д.т.н., проф. Лєнков С.В., начальник науково-дослідного центру Військового інституту Київського національного університету імені Тараса Шевченка

д.т.н., доц. Гунченко Ю.А., Емельянов П.С., Малахов В.С., Щербакова Т.А.

КОНЦЕПЦИЯ ПОСТРОЕНИЯ КОМПЛЕКСА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ИССЛЕДОВАНИЯ ПОМЕХОУСТОЙЧИВЫХ КОДОВ

В статье предложена концепция построения программного комплекса для исследования помехоустойчивых кодов с наличием и без наличия ошибок в сообщении. Рассмотрены необходимые модули комплекса программного обеспечения, которые должны осуществлять кодирование, декодирование сообщений, внесение ошибок, модули хранения, обработки и вывода

данных, приведены условия и алгоритмы их функционирования. Показана целесообразность использования OLAP-систем для хранения и обработки полученных данных. Приведены результаты функционирования комплекса программного обеспечения, созданного согласно предложенной концепции на примере исследования характеристик четырех кодов двух различных типов.

Ключевые слова: помеха, помехоустойчивый код, программное обеспечение, алгоритм построения программного модуля, база данных, OLAP-система.

GunchenkoYu.A., Emelyanov P.S., Shcerbakova T.A.

CONCEPT OF THE CREATING OF THE SOFTWARE COMPLEX FOR THE NOISE-IMMUNITY CODES STUDY

The paper proposes a concept of the creating of the software complex for study of noise immunity codes with/without errors in the message. The software complex modules required for coding/decoding messages and error entering, the modules for data storage, processing and output are considered, their functioning conditions and algorithms are adduced. The advisability of using the OLAP-system for obtained data storage and processing is shown. The results of the software complex functioning which is created according to offered concept are adduced on study example of the features of the four codes of two various types.

Keywords: noise, noise-immunity code, software, software module creating algorithm, database, OLAP-system