Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Кваліфікаційна наукова
праця на правах рукопису

ЄМЕЛЬЯНОВ ІГОР ВАЛЕРІЙОВИЧ

УДК 658:512.011:681.326:519.713

ДИСЕРТАЦІЯ

МОДЕЛІ ТА МЕТОДИ КУБІТНОГО ТЕСТУВАННЯ ЦИФРОВИХ ПРИСТРОЇВ НА ОСНОВІ MEMORY-DRIVEN СТРУКТУР ДАНИХ

05.13.05 – комп'ютерні системи та компоненти
Подається на здобуття наукового ступеня кандидата технічних наук

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело
_____ І.В. Ємельянов

Науковий керівник Хаханов Володимир Іванович,
доктор технічних наук, професор

Цей примірник дисертаційної роботи ідентичний за змістом
з іншими, поданими до спеціалізованої вченої ради Д 64.052.01

Учений секретар спеціалізованої
вченої ради Д 64.052.01 Є. І. Литвинова

Харків - 2018

АНОТАЦІЯ

Ємельянов Ігор Валерійович. Моделі та методи кубітного тестування цифрових пристроїв на основі memory-driven структур даних. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня кандидата технічних наук (доктора філософії) за спеціальністю 05.13.05 «комп'ютерні системи та компоненти». – Харківський національний університет радіоелектронікі, Міністерство освіти і науки України, Харків, 2018.

Дисертаційна робота спрямована на хмарну реалізацію інноваційної технології проектування і тестування цифрових систем і компонентів на основі використання кубітних структур даних, орієнтованих на паралельне виконання процедур синтезу та аналізу. У роботі вирішено науково-практичну задачу квантового проектування, моделювання і тестування цифрових пристроїв і компонентів на основі використання memory-driven кубітних структур даних, вільних від застосування логіки.

Сутність дослідження полягає в розробці хмарних сервісів, моделей і методів квантового синтезу та аналізу memory-driven кубітних моделей цифрових пристроїв і компонентів для істотного підвищення швидкодії засобів проектування, тестування, моделювання та діагностування несправностей.

Мета дослідження – розробка квантових методів паралельного синтезу та аналізу цифрових пристроїв і компонентів для істотного підвищення швидкодії програмних хмарних сервісів і зменшення часу проектування програмно-апаратних компьютингових систем за рахунок збільшення пам'яті для зберігання кубітних структур даних.

Основні результати:

1) Вперше запропоновано модель метричної взаємодії класичного та квантового комп'ютингу, яка характеризується взаємно-однозначною

відповідністю за параметрами паралелізму, суперпозиційності та переплутування в обох видах обчислень, що дає можливість реалізувати квантовий комп'ютинг в класичному виконанні за рахунок збільшення пам'яті.

Модель містить всі метричні компоненти, необхідні для реалізації комп'ютингу: 1) пам'ять; 2) структури адресовних даних; 3) операції; 4) алгоритми; 5) технології; 6) енерговитрати; 7) швидкодія; 8) температурні умови; 9) клас вирішуваних задач.

Представлено memory-driven інноваційну архітектуру квантового комп'ютингу, яка визначається можливістю усунення логіки, пов'язаної з суперпозицією і переплутуванням станів на основі використання характеристичного рівняння, що реалізує транзакції запису-зчитування на структурі електронів. Виключення логічних операцій з квантового комп'ютингу дозволяє істотно спростити архітектуру до рівня структури пам'яті на електронах для виконання транзакцій між ними за допомогою квантів або фотонів. Показано формальну відмінність квантового комп'ютингу від класичного, що полягає в можливості паралельного і одночасного виконання логічних операцій над множинами.

2) Удосконалено метод невизначених коефіцієнтів для мінімізації булевих функцій, який відрізняється від класичного унітарним кодуванням даних для паралельного виконання логічних операцій в цілях отримання двох векторів, відповідних мінімальній диз'юнктивній і кон'юнктивній нормальним формам. Метод дає можливість істотно підвищити швидкодію за рахунок надлишкової пам'яті.

Обчислювальна складність отримання компактного квантового покриття для мінімізації булевих функцій істотно менше в порівнянні з базовим методом невизначених коефіцієнтів, що використовує таблицю істинності спеціальної форми. Заміна попередньої обробки таблиці істинності на унарне кодування станів дозволяє зменшити обчислювальну складність методу мінімізації булевих функцій за рахунок використання трьох векторних пара-

лельних операцій (диз'юнкція нульових рядків, диз'юнкція одиничних рядків і кон'юнкція одиничного результуючого вектора з інверсією нульового). Витрати пам'яті для зберігання структур даних визначаються розмірністю таблиці, необхідної для суперпозиційного отримання двох векторів квантового покриття, де елементи таблиці представлені унарними кодами станів.

3) Удосконалено кубітний метод пошуку дефектів, який відрізняється від існуючого унітарним кодуванням таблиці дефектів, що перевіряються, для паралельного виконання операцій; це дає можливість привести обчислення до логічної різниці двох векторів, відповідних одиничному і нульовому значенням станів-реакцій виходів цифрового пристрою при виконанні тест-експерименту.

Пошук дефектів здійснюється шляхом теоретико-множинної різниці двох векторів, відповідних одиничному і нульовому значенням станів виходів, як реакцій виходів, що спостерігаються, на вхідний тест перевірки несправностей. Структури даних представлені таблицею несправностей на декартовому добутку тестових наборів і множині ліній об'єкта діагностування, де кожна комірка являє собою два біти: перший з них ідентифікує константну несправність нуля (10), що перевіряється, а другий – константну несправність одиниці (01).

Суперпозиція несправностей (дві одиниці на одній лінії-комірці) дає можливість істотно зменшити структури даних для зберігання інформації в цілях подальшого пошуку дефектів при виконанні діагностичного експерименту в режимі online.

4) Отримав подальший розвиток квантовий метод синтезу тестів для логічних функциональностей за рахунок використання булевих похідних за змінними на кубітних структурах даних, що дає можливість підвищити швидкодію методу шляхом паралельного виконання логічних операцій.

Метод базується на використанні регістрових паралельних операцій над апаратно-орієнтованими структурами даних, які являють собою матрицю

кубітних похідних для black-box функціональності. Основна ідея отримання квазіоптимального тесту полягає у визначенні мінімальної кількості стовпців в нульовій та одиничній підмножині матриці кубітних похідних, які своїми одиничними координатами покриють всі рядки або змінні розглянутої функціональності. При цьому, якщо черговий стовпець не додає перевіряльних властивостей раніше доданим до тесту векторам, то він виключається зі списку.

Метод синтезу тестів на основі використання кубітних похідних дозволяє згенерувати вхідні набори, які перевіряють всі поодинокі константні несправності вхідних і внутрішніх ліній. Однак для синтезу мінімального тесту необхідно використовувати структуру цифрового пристрою.

5) Отримав подальший розвиток квантовий метод моделювання справної поведінки за рахунок memory-driven реалізації кубітних структур даних, що дає можливість використовувати транзакційні адресно-орієнтовані процедури аналізу цифрових пристроїв, що виключають логічні операції.

Одним з можливих варіантів інноваційної форми опису цифрових логічних схем є суперпозиційна структура кубітних векторів, де результат аналізу визначається суперпозицією кубітних векторів, що створюють вектор моделювання. Алгоритм аналізу містить одну процедуру – зсув векторів вгору-вниз щодо вектора моделювання. Віртуальний зсув технологічно просто реалізується шляхом обчислення адрес комірок кубітних векторів, з яких зчитується інформація і заноситься в вектор моделювання справної поведінки. Структурний взаємозв'язок кубітних векторів-примітивів здійснюється за допомогою нумерації логічних змінних: вхідних, внутрішніх і вихідних. Змінні формують своїми двійковими станами адреси комірок кубітних векторів для обчислення реакції цифрового пристрою на вхідний вплив.

Використано характеристичне рівняння для моделювання цифрового пристрою, яке оперує обчисленням адрес для запису-зчитування даних, що створює прості і швидкодіючі транзакції між вектором моделювання і кубіт-

ними покриттями. Для визначення двійкового значення логічної змінної або лінії необхідно сформувати адресу комірки кубітного покриття шляхом конкатенації двійкових станів вектора моделювання, де адреси комірок вектора задаються номерами-ідентифікаторами вхідних змінних. Характеристичне рівняння безпосередньо впливає на швидкодію квантового методу моделювання, яка залежить від операцій конкатенації, зчитування і запису бітів, кількості кубітних покриттів в цифровій схемі або логічних примітивів, а також довжини тесту. Показано, що для обробки логічного елемента будь-якої функціональної складності необхідні всього три транзакційних операції.

6) Розроблено хмарні сервіси синтезу та аналізу кубітних моделей цифрових пристроїв і компонентів, які протестовані на різних прикладах комбінаційних схем і пройшли вичерпну апробацію моделей і методів при вивченні курсів «Квантові обчислення», «Схемотехнічне проектування та моделювання НВІС», «Проектування та тестування цифрових пристроїв на ПЛІС», «Комп'ютерна логіка». Середовище проектування: SWIFT, C++, Verilog, Jawa Script, Python 2.7 і платформи: Microsoft Windows, X Window і Macintosh OS X, Google Cloud Platform, Daemon Docker Engine.

В рамках дослідження створені такі засоби: графічний інтерфейс, зручний для ручного проектування кубітних моделей цифрових пристроїв і компонентів, який дає можливість в режимі online здійснювати корекцію помилок; структури даних для кубітного опису цифрових пристроїв і компонентів, які відрізняються компактністю і високим паралелізмом їх обробки; інфраструктура для проектування і тестування цифрових пристроїв і компонентів з функціями зберігання, видалення та корекції даних, що дає можливість здійснювати одночасне створення цифрових схем декількома проектувальниками; програмні модулі для квантового моделювання цифрових пристроїв і компонентів в режимах ручного і автоматичного введення вхідних тестових послідовностей, що дає можливість наочного навчання студентів методам синтезу та аналізу; програмні засоби для синтезу таблиць несправностей на

основі дедуктивного методу квантового моделювання дефектів шляхом використання кубітних структур даних, які забезпечують пошук поодиноких і кратних константних дефектів в режимі реального часу.

Ключові слова: кубіт, кубітне покриття, квантове проектування, моделювання і тестування, верифікація, діагностування, memory-driven кубітні структури даних, цифрові системи на кристалах, квантові обчислення.

Список публікацій здобувача

в яких опубліковані основні наукові результати дисертації:

1. Hahanov V. Cyber Physical Computing for IoT-driven Services [4. Hahanov I., Iemelianov I., Liubarskyi M., Hahanov V. Qubit Description of the Functions and Structures for Service Computing Synthesis; 5. Hahanov V., Bani Amer T., Iemelianov I., Liubarskyi M. Quantum Computing for Test Synthesis; 6. Hahanov I., Bani Amer T., Iemelianov I., Liubarskyi M., Hahanov V. QuaSim – Cloud Service for Quantum Circuits Simulation.] – New York. – Springer. – 2018.– С. 73-143.
2. Hahanov V. Matrix-Model for Diagnosing SoC HDL-Code / V. Hahanov, E. Litvinova, V. Obrizan, I. Yemelyanov // Radioelektroniks and informatics.– 2013.– №1.– P. 12-19. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).
3. Хаханов В.И. Процессорные структуры для анализа Big Data / В.И. Хаханов, Е.И. Литвинова, С.В. Чумаченко, И.В. Емельянов, Т. Bani Amer // Радіоелектронні і комп'ютерні системи. – 2016.– № 6 (80).– С. 163-175. (Входить до міжнародних бібліометричних і наукометричних баз даних: наукової електронної бібліотеки eLIBRARY.RU (Російська Федерація); Index Copernicus (Польща); INSPEC IDEAS (Institution of Engineering and Technology, Великобританія); CiteFactor; Academic Keys; Infobase Index; Google Scholar).

4. Bani Amer T. Кубитная форма описания вычислительных структур / T. Bani Amer, С.В. Чумаченко, И.В. Емельянов // Радиоэлектроника и информатика. – 2016. – № 1.– С.47-52. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

5. Хаханов В.И. Синтез Q-тестов по кубитному описанию функциональностей / В.И. Хаханов, T. Bani Amer, И.В. Емельянов, М.М. Любарский // Радиоэлектроника и информатика.– 2016.– № 2(72).– С. 38-48. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

6. Хаханов В.И. Кубитный метод дедуктивного анализа неисправностей для логических схем / В.И. Хаханов, И.В. Емельянов, М.М. Любарский, С.В. Чумаченко, Е.И. Литвинова, Т. Бани Амер // Электронное моделирование.– 2017.– Том 39, № 6.– С. 59-92 [Входить до міжнародних наукометричних баз Cambridge Scientific Abstracts, Computer and Information Systems Abstracts, INIS Collection, Inspec, ВИНИТИ РАН].

7. Хмарний сервіс для тестування і верифікації систем на кристалах / Є.І. Литвинова, І.В. Ємельянов, І.В. Хаханов // Радиоэлектроника и информатика.– 2017.– №3.– С. 60-69. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

8. Емельянов И.В. Квантовые модели и облачные сервисы для анализа и диагностирования логических схем / И.В. Емельянов, М.М. Любарский, В.И. Хаханов // Радиоэлектроника и информатика. – 2017. – № 4.– С.28-45. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR)

9. Хаханов В.И. Квантовый метод синтеза тестов на основе кубитных структур данных / В.И. Хаханов, И.В. Емельянов, М.М. Любарский, С.В. Чумаченко, Е.И. Литвинова // Электронное моделирование.– 2018.– Том 40, № 1.– С.

63-80 [Входить до міжнародних наукометричних баз Cambridge Scientific Abstracts, Computer and Information Systems Abstracts, INIS Collection, Inspec, ВИНИТИ РАН].

10. Hahanov V. Cloud-driven Cyber Managing Resources / V. Hahanov, S. Chumachenko, E. Litvinova, O. Mishchenko, I. Yemelyanov, Bani Amer Tamer // Australian Journal of Scientific Reseach.– № 1(5).– 2014.– С. 202-215.

11. Hahanov V. Quantum Data Structures for SoC Component Testing / V. Hahanov, W. Gharibi, S. Chumachenko, E. Litvinova, I. Iemelianov, M. Liubarskyi // International Journal of Design, Analysis & Tools for Integrated Circuits & Systems.– Oct. 2017.– vol. 6, iss. 1.– P. 23. (Входить до міжнародної наукометричної бази EBSCO Information Services).

12. Хаханов В.И. Gartner 2017 топ-технологии: их анализ и применение / В.И. Хаханов, А.С. Мищенко, И.В. Емельянов, М.М. Любарский, Т.И. Соклакова, В.Г. Абдулаев // Paradigmata poznání. Vědecko vydavatelské centrum «Sociosféra-CZ», s.r.o., Praha, Česká republika. – 2017. – №4. – P. 33-62. (The journal is indexed by Electronic Research Library, Russia; Research Bible, China; Scientific Indexing Services, USA; Cite Factor, Canada; General Impact Factor, India; Scientific Journal Impact Factor, India; CrossRef, USA; ORCID, USA).

13. Bani Amer T. Компьютинговые модели облачных сервисов / T. Bani Amer, В.И. Хаханов, И.В. Емельянов, М. Любарский // АСУ и приборы автоматики.– 2015.– Вип. 173.– С.48-57. (Входить до міжнародних наукометричних баз Google Scholar, Cyberleninka).

14. Bani Amer T. Синтез и анализ кубитных моделей цифровых систем / T. Bani Amer, И.В. Хаханов, Е.И. Литвинова, И.В. Емельянов // АСУ и приборы автоматики.– 2016.– Вып. 174.– С. 24-41. (Входить до міжнародних наукометричних баз Google Scholar, Cyberleninka).

які засвідчують апробацію матеріалів дисертації:

15. Emelyanov I. Qubit Modeling Digital Systems / I. Emelyanov, I. Hahanova, T. Bani Amer // Proc. of IEEE East-West Design and Test Symposium. – Kiev, 26-29

September.– 2014.– P. 246-248. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

16. Hahanova A. Metric for Analyzing Big Data / A. Hahanova, Yu. Hahanova, I. Yemelyanov, V. Obrizan, D. Krulevska, M. Skorobogatiy // Матеріали XIII Міжнародної науково-технічної конференції CADSM 2015 «Досвід розробки та застосування приладо-технологічних САПР в мікроелектроніці».– 24-27 лютого, 2015.– Львів – Поляна.– С.81-83. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

17. Hahanov V. «Quantum» diagnosis and simulation of SOC / V. Hahanov, I. Yemelyanov, V. Obrizan, I. Hahanov // Proc. of XIth International Conference "Perspective Technologies And Methods In Mems Design".– Lviv-Polyana.– 2-6 September, 2015.– P.58-60. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

18. Hahanov V. «Quantum» Processor for Digital Systems Analysis / V. Hahanov, W. Gharibi, I. Iemelianov, D. Shcherbin // Proceedings of IEEE East-West Design & Test Symposium (EWDTS-2015).– 2015.– Batumi, Georgia.– P. 104-110. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

19. Soklakova T. Technological culture of Big Data / T. Soklakova, I. Iemelianov, T. Bani Amer, I. Hahanov // Матеріали XIII Міжнародної конференції TCSET 2016.– 23-26 лютого, 2016.– Львів – Славське.– С.549-554. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

20. Hahanov I. QuaSim – Cloud Service for Digital Circuits Simulation / I. Hahanov, W. Gharibi, I. Iemelianov, T. Bani Amer // Proceedings of IEEE East-West Design & Test Symposium.– 2016.– Yerevan, Armenia.– P. 363-370. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

21. Hahanov I. Deductive qubit fault simulation / I. Hahanov, I. Iemelianov, T. Bani Amer, D. Timofieiev, S. Chumachenko, V. Hahanov, L. Larchenko // Матеріали XIV Міжнародної науково-технічної конференції CADSM 2017 «Досвід розробки та застосування приладо-технологічних САПР в мікроелек-

троніці».– 21-25 лютого, 2017.– Львів – Поляна.– С.256-259. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

22. Hahanov V. Qubit test synthesis for the black box functionalities / V. Hahanov, E. Litvinova, S. Chumachenko, I. Iemelianov, M. Liubarskyi // Proc. of 5th Prague Embedded Systems Workshop.– June 29-30, 2017.– Roztoky u Prahy, Czech Republic.– P.45-51.

23. Hahanov V. Quantum Sequencer for the Minimal Test Synthesis of Black-box Functionality / V. Hahanov, S. Chumachenko, I. Hahanova, I. Iemelianov, I. Hahanov // Proc. of IEEE East-West Design and Test Symposium.– Novi Sad.– October, 2017.– P.445-450. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

24. Hahanov V. Quantum Data Structures for SoC Component Testing / V. Hahanov, W. Gharibi, K. L. Man, S. Chumachenko, E. Litvinova, I. Iemelianov, M. Liubarskyi // The International Conference on Recent Advancements in Computing, IoT and Computer Engineering Technology.– The Tamkang University.– Taipei, Taiwan.– 2017.– 7 p.

25. Емельянов И.В. Models of Quantum Sequential Primitives / И.В. Емельянов, Д.И. Тимофеев, Б.Д. Ларченко // Материалы XXI Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– 25-27 апреля, 2017.– Ч. 5.– С.70-71.

які додатково відображають наукові результати дисертації:

26. Емельянов И. Телеметрический модуль «SherLock» для управления мобильными объектами / И. Емельянов, А. Котляров // Материалы XVII Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– Ч. 5. – 22-24 апреля, 2013.– С. 64-65.

27. Bani Amer T. Кибер-компьютинг – новый бренд IoT-рынка / T. Bani Amer, И. Емельянов // Материалы XX Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– 2016.– Ч. 5.– С.36-37.

ABSTRACT

Igor Valerievich Iemelianov. Models and methods for qubit testing digital devices based on memory-driven data structures. – Qualification scientific work. Manuscript.

PhD thesis (candidate degree of technical sciences) in speciality 05.13.05 – Computer Systems and Components. - Kharkiv National University of Radio Electronics, Ministry of Education and Science of Ukraine, Kharkiv, 2018.

The thesis is aimed at the cloud implementation of innovative technology for designing and testing digital systems and components based on the use of qubit data structures focused on parallel execution of synthesis and analysis procedures. The scientific and practical problem of quantum design, simulation and testing digital devices and components based on the use of memory-driven logic-free qubit data structures.

The essence of the research is the development of cloud services, models and methods for quantum synthesis and analysis of memory-driven qubit models of digital devices and components to significantly improve the performance of tools for designing, testing, simulation and diagnosis.

The goal of the investigation is to develop quantum methods for parallel synthesis and analysis of digital devices and components to significantly improve the performance of cloud software services and reduce the design time of software and hardware computing systems through increasing memory for storing qubit data structures.

The main results are the following:

1) The new model of the metric interaction of classical and quantum computing is defined, which is characterized by a one-to-one correspondence of the parameters of parallelism, superposition and entanglement in both types of computing;

this makes it possible to realize quantum computing in the classical execution through increasing the memory.

The model includes all the metric components necessary for the realization of computing: 1) memory; 2) structures of addressable data; 3) operations; 4) algorithms; 5) technologies; 6) energy costs; 7) performance; 8) temperature conditions; 9) tasks.

The memory-driven, innovative architecture of quantum computing is presented, which is determined by the opportunity to eliminate logic associated with superposition and entanglement of states based on the use of the characteristic equation that defines read-write transactions in the structure of electrons. Eliminating logical operations in quantum computing makes it possible to significantly simplify the architecture to the level of the memory structure on electrons to perform transactions between them using quanta or photons. The formal difference between quantum computing and classical one is shown, which consists in the opportunity of parallel and simultaneous execution of logical operations on sets.

2) The method of undetermined coefficients for minimizing Boolean functions is improved, which differs from the classical one by unitary coding of data for parallel execution of logical operations in order to obtain two vectors corresponding to the minimal disjunctive and conjunctive normal forms. The method makes it possible to significantly improve performance through the use of redundant memory.

The computational complexity of obtaining a compact quantum coverage for minimization of Boolean functions is significantly smaller in comparison with the base method of undetermined coefficients that uses a special truth table form. Replacing preprocessing a truth table with unary state coding can reduce the computational complexity of the method for minimizing Boolean functions by using three vector parallel operations (a disjunction of zero rows, disjunction of unit rows and a conjunction of a unit result vector and negated zero vector). The memory costs for storing data structures are defined by the dimension of the table necessary for super-

positioning two vectors of quantum coverage, where the table cells are represented by unary codes of states.

3) The qubit method of fault detection is improved, which differs from the existing one by unitary encoding of the fault detection table for parallel execution of operations, which makes it possible to reduce the calculations to the logical difference of two vectors corresponding to the unit and zero values of states-responses of the digital device outputs during the test experiment.

Fault detection is carried out by using the set-theoretic difference of two vectors corresponding to the unit and zero values of the states of outputs, as the responses of the observed outputs to the input test. The data structures are represented by a fault detection table on the Cartesian product of the test patterns and a set of lines of the object under diagnosis, where each cell is two bits: the first one identifies the verified zero constant fault (10), and the second one identifies a unit constant fault (01).

The superposition of faults (two units in a single line-cell) makes it possible to substantially minimize data structures for storing information for the purpose of subsequent fault detection when performing an online diagnostic experiment.

4) The quantum method for test synthesis of logical functionalities has been further developed through the use of Boolean derivatives with respect to variables on qubit data structures, which makes it possible to increase the performance of the method through the parallel execution of logical operations.

The method is based on the use of register parallel operations on hardware-focused data structures, which represent a matrix of qubit derivatives for black-box functionality. The basic idea of generating a quasi-optimal test is to determine the minimum number of columns in a zero and a unit subset of the matrix of the qubit derivatives, which will cover all the rows or variables of the functionality by their unit coordinates . In this case, if the next column does not add checking properties to the vectors previously included in the test, then it is excluded from the list.

The method for test synthesis is based on the use of qubit derivatives allows generating input patterns, detecting all single constant faults of input and internal lines. However, for the synthesis of the minimum test, it is necessary to use the structure of the digital device.

5) The quantum method for fault-free simulation is further developed through the memory-driven implementation of qubit data structures, which makes it possible to use transactional address-focused procedures for analyzing digital devices free of logical operations.

One of the possible variants of the innovative form of describing digital logic circuits is the superposition structure of qubit vectors, where the result of analysis is determined by the superposition of the qubit vectors defining the simulation vector. The analysis algorithm contains one procedure - shifting the vectors up and down relative to the simulation vector. The virtual shift is technologically simple realized by calculating the addresses of the cells of the qubit vectors from which the information is read and entered into the fault-free simulation vector. The structural interrelation of qubit vector-primitives is realized by means of numbering of logical variables: input, internal and output. The binary states of the variables define addresses of the cells of the qubit vectors to calculate the response of the digital device on the input signals.

The characteristic equation is used for simulation of a digital device, which operates by calculating addresses for writing-reading data; this creates simple and high-speed transactions between the simulation vector and qubit coverage. To determine the binary value of a logical variable or a line, it is necessary to generate the cell address of the qubit coverage by concatenating the binary states of the simulation vector, where the addresses of the vector cells are defined by the numbers-identifiers of the input variables. The characteristic equation directly affects the speed of the quantum modeling method, which depends on the operations of concatenation, reading and writing bits, the number of qubit coverages in the digital circuit or logical primitives, and the length of the test. It is shown that only three transac-

tional operations are required to process a logical element of any functional complexity.

6) Cloud services for the synthesis and analysis of qubit models of digital devices and components have been developed, which have been tested on various examples of combinational circuits; models and methods have been thoroughly tested in the course "Design and diagnosis of digital systems on FPGA." Design environment includes the following tools: SWIFT, C ++, Verilog, Jawa Script, Python 2.7 and platforms: Microsoft Windows, X Window and Macintosh OS X, Google Cloud Platform, Daemon Docker Engine.

Within the framework of the investigation, the following tools have been developed: a graphical interface, convenient for manual design of qubit models of digital devices and components, which enables online correction of errors; data structures for the qubit description of digital devices and components, which differ by compactness and high parallelism in their processing; infrastructure for designing and testing digital devices and components with functional components for storing, deleting and correcting data, which makes it possible to simultaneously develop digital circuits by several designers; software modules for quantum simulation of digital devices and components in the modes of manual and automatic input of test patterns, which allows to visualize learning methods of synthesis and analysis for students; software for the synthesis of fault detection tables based on the deductive method of quantum defect simulation by using qubit data structures, which provide detection of single and multiple constant faults in real time.

Key words: qubit, qubit coverage, quantum design, simulation and test, verification, diagnosis, memory-driven qubit data structures, digital system-on-chip, quantum computing.

Список публікацій здобувача

в яких опубліковані основні наукові результати дисертації:

1. Hahanov V. Cyber Physical Computing for IoT-driven Services [4. Hahanov I., Iemelianov I., Liubarskyi M., Hahanov V. Qubit Description of the Functions and Structures for Service Computing Synthesis; 5. Hahanov V., Bani Amer T., Iemelianov I., Liubarskyi M. Quantum Computing for Test Synthesis; 6. Hahanov I., Bani Amer T., Iemelianov I., Liubarskyi M., Hahanov V. QuaSim – Cloud Service for Quantum Circuits Simulation.] – New York. – Springer. – 2018.– С. 73-143.

2. Hahanov V. Matrix-Model for Diagnosing SoC HDL-Code / V. Hahanov, E. Litvinova, V. Obrizan, I. Yemelyanov // Radioelektroniks and informatics.– 2013.– №1.– P. 12-19. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

3. Хаханов В.И. Процессорные структуры для анализа Big Data / В.И. Хаханов, Е.И. Литвинова, С.В. Чумаченко, И.В. Емельянов, T. Bani Amer // Радіоелектронні і комп'ютерні системи. – 2016.– № 6 (80).– С. 163-175. (Входить до міжнародних бібліометричних і наукометричних баз даних: наукової електронної бібліотеки eLIBRARY.RU (Російська Федерація); Index Copernicus (Польща); INSPEC IDEAS (Institution of Engineering and Technology, Великобританія); CiteFactor; Academic Keys; Infobase Index; Google Scholar).

4. Bani Amer T. Кубитная форма описания вычислительных структур / T. Bani Amer, С.В. Чумаченко, И.В. Емельянов // Радиоэлектроника и информатика. – 2016. – № 1.– С.47-52. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

5. Хаханов В.И. Синтез Q-тестов по кубитному описанию функциональностей / В.И. Хаханов, T. Bani Amer, И.В. Емельянов, М.М. Любарский // Радиоэлектроника и информатика.– 2016.– № 2(72).– С. 38-48. (Входить до

міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

6. Хаханов В.И. Кубитный метод дедуктивного анализа неисправностей для логических схем / В.И. Хаханов, И.В. Емельянов, М.М. Любарский, С.В. Чумаченко, Е.И. Литвинова, Т. Бани Амер // Электронное моделирование.– 2017.– Том 39, № 6.– С. 59-92 [Входить до міжнародних наукометричних баз Cambridge Scientific Abstracts, Computer and Information Systems Abstracts, INIS Collection, Inspec, ВИНИТИ РАН].

7. Хмарний сервіс для тестування і верифікації систем на кристалах / Є.І. Литвинова, І.В. Ємельянов, І.В. Хаханов // Радиоэлектроника и информатика.– 2017.– №3.– С. 60-69. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

8. Емельянов И.В. Квантовые модели и облачные сервисы для анализа и диагностирования логических схем / И.В. Емельянов, М.М. Любарский, В.И. Хаханов // Радиоэлектроника и информатика. – 2017. – № 4.– С.28-45. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR)

9. Хаханов В.И. Квантовый метод синтеза тестов на основе кубитных структур данных / В.И. Хаханов, И.В. Емельянов, М.М. Любарский, С.В. Чумаченко, Е.И. Литвинова // Электронное моделирование.– 2018.– Том 40, № 1.– С. 63-80 [Входить до міжнародних наукометричних баз Cambridge Scientific Abstracts, Computer and Information Systems Abstracts, INIS Collection, Inspec, ВИНИТИ РАН].

10. Hahanov V. Cloud-driven Cyber Managing Resources / V. Hahanov, S. Chumachenko, E. Litvinova, O. Mishchenko, I. Yemelyanov, Bani Amer Tamer // Australian Journal of Scientific Reseach.– № 1(5).– 2014.– С. 202-215.

11. Hahanov V. Quantum Data Structures for SoC Component Testing / V. Hahanov, W. Gharibi, S. Chumachenko, E. Litvinova, I. Iemelianov, M. Liubarskyi // International Journal of Design, Analysis & Tools for Integrated Circuits & Systems.– Oct. 2017.– vol. 6, iss. 1.– P. 23. (Входить до міжнародної наукометричної бази EBSCO Information Services).
12. Хаханов В.И. Gartner 2017 топ-технологии: их анализ и применение / В.И. Хаханов, А.С. Мищенко, И.В. Емельянов, М.М. Любарский, Т.И. Соклакова, В.Г. Абдулаев // Paradigmata poznání. Vědecko vydavatelské centrum «Sociosféra-CZ», s.r.o., Praha, Česká republika. – 2017. – №4. – P. 33-62. (The journal is indexed by Electronic Research Library, Russia; Research Bible, China; Scientific Indexing Services, USA; Cite Factor, Canada; General Impact Factor, India; Scientific Journal Impact Factor, India; CrossRef, USA; ORCID, USA).
13. Bani Amer T. Компьютинговые модели облачных сервисов / T. Bani Amer, В.И. Хаханов, И.В. Емельянов, М. Любарский // АСУ и приборы автоматики.– 2015.– Вип. 173.– С.48-57. (Входить до міжнародних наукометричних баз Google Scholar, Cyberleninka).
14. Bani Amer T. Синтез и анализ кубитных моделей цифровых систем / T. Bani Amer, И.В. Хаханов, Е.И. Литвинова, И.В. Емельянов // АСУ и приборы автоматики.– 2016.– Вып. 174.– С. 24-41. (Входить до міжнародних наукометричних баз Google Scholar, Cyberleninka).

які засвідчують апробацію матеріалів дисертації:

15. Emelyanov I. Qubit Modeling Digital Systems / I. Emelyanov, I. Hahanova, T. Bani Amer // Proc. of IEEE East-West Design and Test Symposium. – Kiev, 26-29 September.– 2014.– P. 246-248. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).
16. Hahanova A. Metric for Analyzing Big Data / A. Hahanova, Yu. Hahanova, I. Yemelyanov, V. Obrizan, D. Krulevska, M. Skorobogatiy // Матеріали XIII Міжнародної науково-технічної конференції CADSM 2015 «Досвід розробки

та застосування приладо-технологічних САПР в мікроелектроніці».– 24-27 лютого, 2015.– Львів – Поляна.– С.81-83. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

17. Hahanov V. «Quantum» diagnosis and simulation of SOC / V. Hahanov, I. Yemelyanov, V. Obrizan, I. Hahanov // Proc. of XIth International Conference "Perspective Technologies And Methods In Mems Design".– Lviv-Polyana.– 2-6 September, 2015.– P.58-60. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

18. Hahanov V. «Quantum» Processor for Digital Systems Analysis / V. Hahanov, W. Gharibi, I. Iemelianov, D. Shcherbin // Proceedings of IEEE East-West Design & Test Symposium (EWDTS-2015).– 2015.– Batumi, Georgia.– P. 104-110. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

19. Soklakova T. Technological culture of Big Data / T. Soklakova, I. Iemelianov, T. Bani Amer, I. Hahanov // Матеріали XIII Міжнародної конференції TCSET 2016.– 23-26 лютого, 2016.– Львів – Славське.– С.549-554. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

20. Hahanov I. QuaSim – Cloud Service for Digital Circuits Simulation / I. Hahanov, W. Gharibi, I. Iemelianov, T. Bani Amer // Proceedings of IEEE East-West Design & Test Symposium.– 2016.– Yerevan, Armenia.– P. 363-370. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

21. Hahanov I. Deductive qubit fault simulation / I. Hahanov, I. Iemelianov, T. Bani Amer, D. Timofieiev, S. Chumachenko, V. Hahanov, L. Larchenko // Матеріали XIV Міжнародної науково-технічної конференції CADSM 2017 «Досвід розробки та застосування приладо-технологічних САПР в мікроелектроніці».– 21-25 лютого, 2017.– Львів – Поляна.– С.256-259. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

22. Hahanov V. Qubit test synthesis for the black box functionalities / V. Hahanov, E. Litvinova, S. Chumachenko, I. Iemelianov, M. Liubarskyi // Proc. of 5th Prague

Embedded Systems Workshop.– June 29-30, 2017.– Roztoky u Prahy, Czech Republic.– P.45-51.

23. Hahanov V. Quantum Sequencer for the Minimal Test Synthesis of Black-box Functionality / V. Hahanov, S. Chumachenko, I. Hahanova, I. Iemelianov, I. Hahanov // Proc. of IEEE East-West Design and Test Symposium.– Novi Sad.– October, 2017.– P.445-450. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

24. Hahanov V. Quantum Data Structures for SoC Component Testing / V. Hahanov, W. Gharibi, K. L. Man, S. Chumachenko, E. Litvinova, I. Iemelianov, M. Liubarskyi // The International Conference on Recent Advancements in Computing, IoT and Computer Engineering Technology.– The Tamkang University.– Taipei, Taiwan.– 2017.– 7 p.

25. Емельянов И.В. Models of Quantum Sequential Primitives / И.В. Емельянов, Д.И. Тимофеев, Б.Д. Ларченко // Материалы XXI Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– 25-27 апреля, 2017.– Ч. 5.– С.70-71.

які додатково відображають наукові результати дисертації:

26. Емельянов И. Телеметрический модуль «SherLock» для управления мобильными объектами / И. Емельянов, А. Котляров // Материалы XVII Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– Ч. 5. – 22-24 апреля, 2013.– С. 64-65.

27. Bani Amer T. Кибер-компьютинг – новый бренд IoT-рынка / T. Bani Amer, И. Емельянов // Материалы XX Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– 2016.– Ч. 5.– С.36-37.

ЗМІСТ

ВСТУП

Актуальність дослідження. Майбутнє людства відчиняється п'ятьма ключами: 1) Цифровий світ. 2) Нові матеріали. 3) Генна інженерія. 4) Мутація природи. 5) Сонячна енергетика. Цифровий світ передбачає точну цифрову ідентифікацію всіх кіберфізичних процесів і явищ в часі та просторі. Метою створення цифрового світу є розумний кіберфізичний і соціальний комп'ютинг вичерпного цифрового моніторингу та метричного інтелектуального керування всіма процесами та явищами для підвищення якості життя людей і збереження екології планети. Існуючі обчислювальні потужності всієї планети не здатні задовільно вирішити згадану проблему через низьку продуктивність глобального комп'ютингу і високу енерговитратність існуючих класичних і квантових комп'ютерів. Це дає підстави для пошуку альтернативних технологій реалізації компьютингових інноваційних архітектур. Можливим вирішенням проблеми може бути квантовий memory-driven і logic-free комп'ютинг, що виключає дорогі, енерговитратні, логіко-подібні (or, not) операції суперпозиції і переплутування. Далі наведено аксіоми, які необхідно враховувати при створенні інноваційного комп'ютингу: 1) Квантовий комп'ютинг ізоморфний класичному: структури даних і операції (qubit, superposition, entangelment) квантового комп'ютингу мають взаємнооднозначну відповідність аналогічним компонентам класичного (bits, or, not). 2) Квантова невизначеність в комп'ютингу має строгий детермінізм, що виключає «ймовірність, як фіговий листок на голому тілі нашого невігластва». Альберт Ейнштейн. 3) Будь-які процеси і явища квантового комп'ютингу можна реалізувати (моделювати) на класичному. 4) Будь-які процеси і явища класичного комп'ютингу можна реалізувати (моделювати) на квантовому обчислювачі. 5) Затвердження. Memory-driven і logic-free архітектура

комп'ютингу є більш технологічною і менш енерговитратною в класичному і квантовому форматах реалізації.

Інтегрально квантовий комп'ютинг відрізняється від класичного паралельним вирішенням комбінаторних задач, що забезпечує високу швидкодію, за рахунок одночасного подання в квантовій точці субатомного масштабу кінцевої множини дискретних станів. Класичний комп'ютер для цього використовує нанотехнологічні розміри регістрової пам'яті кремнієвого кристалу, які в кілька порядків перевершують розміри субатомних частинок. Однак для фотонного керування субатомними частинками необхідні близькі до абсолютного нуля температури надпровідності, які забезпечуються кількістю енергії, яка на кілька порядків вище енергоспоживання класичного комп'ютера. Природно, що всі завдання паралельного квантового комп'ютингу можна виконувати на класичному обчислювачі за рахунок унітарного кодування станів, яке вимагає експоненціального підвищення витрат пам'яті для зберігання даних. Очевидно, що шляхи розвитку класичного і квантового комп'ютингу перетинаються в точці адитивного синтезу або вирощування субатомних, вільних від логіки, структур пам'яті, керованих фотонами, які матимуть добову циклічну сонячну енергозалежність, подібно рослинним організмам.

Квантовий комп'ютинг є паралельна комбінаторика, де неоднорідні підмножини даних обробляються за один такт. Класичний комп'ютинг є паралельна регуляторика, де регулярні адресовні дані обробляються за один такт. Тільки поєднання двох видів комп'ютингу надасть людству необхідні обчислювальні потужності для створення глобального інтелекту. Теоретичні обмеження паралелізму квантового комп'ютингу пов'язані з кількістю можливих комбінацій на множині з n-примітивів, що дорівнює 2**n. Практичні обмеження паралелізму класичного комп'ютингу пов'язані з кількістю n розрядів регістрової пам'яті, на якій виконуються паралельні арифметико-логічні

операції. Теоретична основа квантового комп'ютингу – множини, а класичного – байти (впорядковані структури адресовних даних).

Тема дисертаційної роботи націлена на хмарну реалізацію інноваційної технології проектування і тестування цифрових систем і компонентів на основі використання кубітних структур даних, орієнтованих на паралельне виконання процедур синтезу та аналізу.

Проблеми (квантового) проектування, моделювання, тестування, діагностування та відновлення працездатності цифрових систем знаходять відображення в публікаціях вчених: Y. Zorian, J. Bergeron, Z. Navabi, A. Jerraya, D.B. Armstrong, P. Prinetto, J. Abraham, H. Fujiwara, T. Nishida, X. Wang, Peter Mueller, Анатолій Петренко, Раймунд Убар, Andre Ivanov, Олексій Романкевич, Дмитро Сперанський, Анжела Матросова, Павло Пархоменко, John Paul Roth, Вазген Мелікян, Самвел Шукурян, Володимир Тарасенко, Михайло Коровай, Олександр Палагін, Володимир Опанасенко, В'ячеслав Харченко, Леонід Дербунович, В'ячеслав Ярмолік, Рімантас Шейнаускас, Ніна Євтушенко, Роман Базилевич, Геннадій Кривуля.

Зв'язок роботи з науковими програмами, держбюджетними темами. Розробка теми дисертації здійснювалася відповідно до планів держбюджетних НДР та договорів, виконуваних на кафедрі АПОТ Харківського національного університету радіоелектроніки в період з 2011 року, в тому числі: 1) Договір про дружбу і співробітництво між ХНУРЕ та корпорацією «Aldec Inc.» (USA) № 04 від 01.11.2011. 2) Фундаментальна НДР «Персональний віртуальний кіберкомп'ютер та інфраструктура аналізу кіберпростору», №258 (2012-2014). 3) Держбюджетна НДР «Кіберфізична система – «Розумне хмарне управління транспортом», № 0115U-000712 (2015-2017); 4) «Розумний кібер університет – Cloud-Mobile сервіси управління науково-освітніми процесами», № 0117U-002524 (2017-2019).

Автор дисертаційної роботи брав участь при виконанні зазначених договорів і програм, як розробник, менеджер і програміст інфраструктури те-

стування цифрових систем для моделювання і діагностування обчислювальних пристроїв. Автор також брав участь в SWIFT-кодуванні програмних модулів системи верифікації і моделювання на основі IEEE стандартів, інтегрованих з сервісами компанії Aldec.

Дисертація присвячена вирішенню науково-практичної задачі квантового проектування, моделювання і тестування цифрових пристроїв і компонентів на основі використання memory-driven кубітних структур даних, вільних від застосування логіки.

Сутність дослідження полягає в розробці хмарних сервісів, моделей і методів квантового синтезу та аналізу memory-driven кубітних моделей цифрових пристроїв і компонентів для істотного підвищення швидкодії засобів проектування, тестування, моделювання та діагностування несправностей.

Об'єкт дослідження – технології квантового паралельного комп'ютингу на класичних обчислювальних пристроях, що використовують memory-driven кубітні структури даних без застосування логічних елементів.

Предмет дослідження – моделі, методи, алгоритми та засоби квантового синтезу та аналізу цифрових пристроїв і компонентів на основі використання кубітних memory-driven структур даних без застосування логічних елементів.

Мета дослідження – розробка квантових методів паралельного синтезу та аналізу цифрових пристроїв і компонентів для істотного підвищення швидкодії програмних хмарних сервісів і зменшення часу проектування програмно-апаратних комп'ютингових систем за рахунок збільшення пам'яті для зберігання кубітних структур даних.

Задачі дослідження:

1) Визначити модель метричної взаємодії класичного та квантового комп'ютингу за параметрами паралелізму, суперпозиційності та переплутування для реалізації квантового комп'ютингу в класичному виконанні за рахунок збільшення пам'яті.

2) Удосконалити метод невизначених коефіцієнтів для мінімізації булевих функцій шляхом паралельного виконання логічних операцій в цілях отримання двох векторів, відповідних мінімальній диз'юнктивній і кон'юнктивній нормальним формам.

3) Удосконалити кубітний метод пошуку дефектів для паралельного виконання операцій над двома групами векторів таблиці несправностей цифрового пристрою, отриманих після виконання діагностичного експерименту.

4) Реалізувати подальший розвиток квантового методу синтезу тестів для логічних функціональностей на основі використання булевих похідних за кубітними структурами даних для підвищення швидкодії за рахунок паралельного виконання логічних операцій.

5) Реалізувати подальший розвиток квантового методу моделювання справної поведінки на основі memory-driven кубітних структур даних шляхом використання транзакційних адресно-орієнтованих процедур аналізу цифрових пристроїв, які виключають логічні операції.

6) Розробити хмарні сервіси синтезу та аналізу кубітних моделей цифрових пристроїв і компонентів, верифікувати їх на різних прикладах цифрових схем і впровадити їх в навчальний процес. Середовище проектування: SWIFT, C++, Verilog, Jawa Script, Google Cloud Platform.

Методи дослідження – квантовий комп'ютинг, прикладна теорія цифрових автоматів, архітектури комп'ютерів, булева алгебра, теорія множин, теорія графів, квантово-кубітні методи обчислень і структури даних – для побудови моделей цифрових пристроїв; векторно-логічний аналіз, теорія алгоритмів, методи, засоби, мови проектування і моделювання цифрових систем – для синтезу та аналізу; методи і критерії якості створення обчислювальних проектів – для оцінювання тестопридатності цифрових виробів; засоби синтезу схем і аналізу кубітних покриттів – для тестування програмно-апаратних компонентів інфраструктури хмарних сервісів.

Наукова новизна:

1) Вперше запропоновано модель метричної взаємодії класичного та квантового комп'ютингу, яка характеризується взаємно-однозначною відповідністю за параметрами паралелізму, суперпозиційності та переплутування в обох видах обчислень, що дає можливість реалізувати квантовий комп'ютинг в класичному виконанні за рахунок збільшення пам'яті.

2) Удосконалено метод невизначених коефіцієнтів для мінімізації булевих функцій, який відрізняється від класичного унітарним кодуванням даних для паралельного виконання логічних операцій в цілях отримання двох векторів, відповідних мінімальній диз'юнктивній і кон'юнктивній нормальним формам.

3) Удосконалено кубітний метод пошуку дефектів, який відрізняється від існуючого унітарним кодуванням таблиці дефектів, що перевіряються, для паралельного виконання операцій; це дає можливість привести обчислення до логічної різниці двох векторів, відповідних одиничному і нульовому значенням станів-реакцій виходів цифрового пристрою при виконанні тест-експерименту.

4) Отримав подальший розвиток квантовий метод синтезу тестів для логічних функціональностей за рахунок використання булевих похідних за змінними на кубітних структурах даних, що дає можливість підвищити швидкодію методу шляхом паралельного виконання логічних операцій.

5) Отримав подальший розвиток квантовий метод моделювання справної поведінки за рахунок memory-driven реалізації кубітних структур даних, що дає можливість використовувати транзакційні адресно-орієнтовані процедури аналізу цифрових пристроїв, що виключають логічні операції.

Практична значущість отриманих результатів:

6) Розроблено хмарні сервіси синтезу та аналізу кубітних моделей цифрових пристроїв і компонентів, які протестовані на різних прикладах комбінаційних схем і пройшли вичерпну апробацію моделей і методів при

вивченні курсів «Квантові обчислення», «Схемотехнічне проектування та моделювання НВІС», «Проектування та тестування цифрових пристроїв на ПЛІС», «Комп'ютерна логіка». Середовище проектування: SWIFT, C++, Verilog, Jawa Script, Python 2.7 і платформи: Microsoft Windows, X Window і Macintosh OS X, Google Cloud Platform, Daemon Docker Engine.

Отримані в процесі виконання досліджень наукові висновки і практичні результати є достовірними, що підтверджується достатньою кількістю проведених експериментів, тестуванням і моделюванням реальних і тестових функціональних модулів з відкритих бібліотек компаній і конференцій. Практична значущість наукових досліджень підтверджується інтеграцією програмних додатків аналізу і синтезу з сервісами проектування і верифікації компанії Aldec. Результати дисертації в складі моделей, методів та інфраструктури впроваджено в навчальний процес Харківського національного університету радіоелектроніки (акт про впровадження від 04.01.2018); в науково-дослідну і виробничу діяльність компанії Aldec, USA (довідка про впровадження від 09.01.2018).

Особистий внесок здобувача. Всі наукові і практичні результати отримані автором особисто. У роботах, опублікованих зі співавторами, здобувачеві належать: [1] – квантовий метод синтезу тестів, кубітний метод дедуктивного моделювання, хмарний сервіс аналізу цифрових пристроїв; [2] – матрична модель для паралельного діагностування несправностей цифрових пристроїв, кубітний метод пошуку дефектів; [3] – аналіз великих даних для векторно-кубітного моделювання функціональних цифрових схем, квантовий метод моделювання справної поведінки; [4] – кубітні форми опису обчислювальних структур, квантовий метод моделювання справної поведінки; [5] – квантовий метод синтезу Q-тестів за допомогою кубітного опису функціональностей; [6] – кубітний векторний метод дедуктивного аналізу несправностей для логічних схем, квантовий метод синтезу тестів для логічних функціональностей; [7] – хмарні сервіси на основі квантових моделей і ме-

тодів тестування, моделювання та діагностування цифрових пристроїв; [8] – квантовий метод моделювання цифрових пристроїв у формі хмарного сервісу, метод невизначених коефіцієнтів для мінімізації булевих функцій, кубітний метод пошуку дефектів; [9] – визначення класичного та квантового комп'ютингу, метод синтезу тестів для логічних функціональностей за рахунок використання булевих похідних за змінними на кубітних структурах даних, метод отримання квазіоптимального тесту, алгоритм і структура секвенсора; [10] – хмарна інфраструктура квантово-векторного моделювання цифрових пристроїв на основі кубітного опису функцій примітивів, квантовий метод моделювання справної поведінки; [11] – квантові структури даних для тестування цифрових логічних компонентів; [12] – комп'ютингові та квантові технології синтезу та аналізу цифрових пристроїв, модель метричної взаємодії класичного та квантового комп'ютингу; [13] – комп'ютингові моделі хмарних сервісів для тестування цифрових систем; [14] – синтез і аналіз кубітних моделей цифрових систем, квантовий метод синтезу тестів для логічних функціональностей; [15] – синтез моделей і кубітне моделювання цифрових систем; [16] – метрика для аналізу даних при діагностуванні цифрових виробів; [17] – квантове діагностування та моделювання цифрових систем на кристалах, кубітний метод пошуку дефектів; [18] – квантовий процесор для аналізу цифрових компонентів; [19] – технологічна культура великих даних для діагностування дефектів; [20] –квантовий метод моделювання справної поведінки, хмарні сервіси синтезу та аналізу кубітних моделей цифрових пристроїв і компонентів; [21] – дедуктивне моделювання несправностей на основі використання кубітних структур даних; [22] – кубітний метод синтезу тестів для примітивних функціональностей, модель метричної взаємодії класичного та квантового комп'ютингу; [23] – квантовий процесор для мінімізації тестів логічних схем; [24] – квантові методи аналізу і синтезу цифрових систем на кристалах; [25] – квантові моделі послідовностних

примітивів; [26] – хмарна інфраструктура для реалізації мікросервісів; [27] – технології кіберкомп'ютингу, як нового бренду IoT-ринку.

Апробація результатів дисертації. Результати роботи були представлені та обговорені на таких конференціях: 1-4) IEEE East-West Design and Test Symposium 2014 (Ukraine), 2015 (Georgia), 2016 (Armenia); 2017 (Serbia); 5-7) XX Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI сторіччі» 2013, 2016 2017, Україна; 8) the 11-th IEEE International Conference TCSET 2016 Slavsk, Ukraine; 9) XI International Conference "Perspective Technologies and Methods in MEMS Design", Lviv-Polyana, 2015; 10-11) 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2015 2017, Lvov, Ukraine; 12) 5th Prague Embedded Systems Workshop 2017 Roztoky u Prahy, Czech Republic; 13) the 4th International Microelectronics Congress 2017, Taiwan.

Публікації. Результати наукових досліджень опубліковано в 27 друкованих працях: 1 монографії, 10 статтях у наукових фахових виданнях України (з них 10 статей входять до міжнародних наукометричних баз), 3 статтях у міжнародних наукових журналах за кордоном, а також 13 міжнародних наукових конференціях (з них 10 за кордоном і 9 входять до наукометричної бази Scopus).

Структура дисертації представлена 197 сторінками (з них 127 сторінок основного тексту) і містить: 4 розділи, 48 рисунків, список джерел з 187 найменувань (на 21 с.), 4 додатки (на 24 с.).

РОЗДІЛ 1

ХМАРНЕ ТЕСТУВАННЯ І МОДЕЛЮВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ

Пропонуються моделі, методи, алгоритми та структури хмарно-орієнтованого квантового комп'ютингу для тестування і моделювання цифрових систем в цілях надання відкритих освітніх сервісів студентам і фахівцям. Дається короткий аналітичний огляд технологій (квантового) проектування, тестування, моделювання справної поведінки і несправностей цифрових систем і схем. Робиться висновок про фінансову та освітню спроможність хмарного комп'ютингу для проектування і тестування кіберфізичної продукції, online доступного для кожної людини в будь-якій точці земної кулі. Ставляться науково-практичні задачі дослідження, присвяченого хмарній реалізації квантового комп'ютингу для синтезу тестів, моделювання справної поведінки і несправностей функціональних компонентів SoC.

Мета дослідження в рамках розділу – обґрунтування актуальності виконання паралельних квантових методів тестування та верифікації цифрових систем для їх подальшої релізації як хмарних сервісів, доступних для вивчення в часі і просторі кожному студенту і фахівцеві.

Задачі дослідження в рамках розділу: 1) Хмарні технології кіберфізичного комп'ютингу для проектування і тестування цифрових систем. 2) Традиційні методи синтезу тестів і моделювання цифрових компонентів обчислювальних пристроїв. 3) Постановка мети і задач дослідження, спрямованих на кіберфізичне поєднання хмарних сервісів квантового комп'ютингу з реальними об'єктами для кубітного проектування, тестування, верифікації та діагностування цифрових виробів.

1.1 Вступ

Модель метричної взаємодії квантового та класичного комп'ютингу на основі аналізу їх взаємно-однозначної відповідності представлена на рис. 1.1. Тут розглядаються всі метричні компоненти, необхідні для реалізації комп'ютингу: 1) пам'ять; 2) структури адресовних даних; 3) операції; 4) алгоритми; 5) технології; 6) енерговитрати; 7) швидкодія; 8) температурні умови; 9) клас розв'язуваних задач.

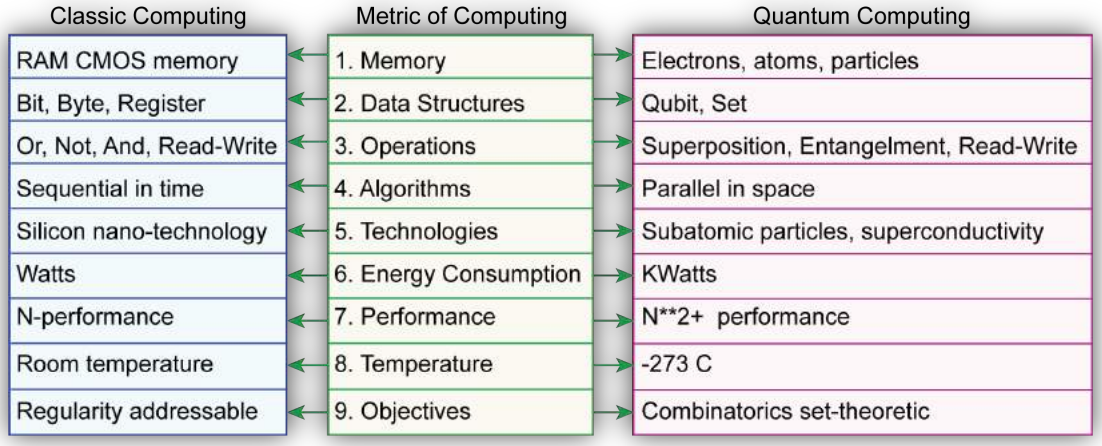


Рис. 1.1 – Метричне порівняння класичного та квантового комп'ютингу

Алгоритм Дойча для розпізнавання булевої функції однієї змінної [Richard J. LiptonandKenneth W. Regan. Quantum Algorithms via Linear Algebra. MIT Press eBook. 2014] описано нижче. Задано чотири функції однієї змінної f(X)={0, 1, x, notx}. Щоб розрізнити функції f(X)={0,x} та f(X)={notx, 1}, класичному алгоритму необхідне одне вимірювання на тестовому наборі X=0, що ілюструється таким перетворенням таблиць істинності для чотирьох примітивних функцій:

X	Y_0	X	$\bar{X}$	Y_1
0	0	0	1	1
1	0	1	0	1

$\xrightarrow{x=\{0,1\}}$

X	$Y_0 \cup X$	$\bar{X} \cup Y_1$
0	0	1
1	x	x

Щоб розрізнити функції f(X)={0,1} та f(X)={x, notx}, класичному алгоритму необхідні два вимірювання у двох часових фреймах (X1=0, X2=1),

оскільки попарна мінімізіція стовпців {0,1} та {x, notx} шляхом їх об'єднання для примітивних функцій неможлива:

$$\begin{array}{|l} X_1 = 0 \rightarrow Y = 0 \\ X_2 = 1 \rightarrow Y = 0 \end{array} \rightarrow Y_0;$$

$$\begin{array}{|l} X_1 = 0 \rightarrow Y = 1 \\ X_2 = 1 \rightarrow Y = 1 \end{array} \rightarrow Y_1.$$

Для квантового алгоритму досить одного вимірювання. Щоб мати можливість об'єднати зазначені пари функцій, що не можна зробити в позиційній системі кодування, необхідно представити вхідні значення (0=10, 1=01), а також коди-кубіти вихідних значень в унітарному форматі:

X	Y_0	Y_x	$Y_{\bar{x}}$	Y_1
0	0	0	1	1
1	0	1	0	1

00=1000
01=0100
10=0010
11=0001 →

X	Y_0	Y_x	$Y_{\bar{x}}$	Y_1
0 = 10	0	0	1	1
1 = 01	0	1	0	1
11	1	0	0	0
11	0	1	0	0
11	0	0	1	0
11	0	0	0	1

=

=

X	$Y_0 \cup Y_1$	$Y_x \cup Y_{\bar{x}}$
11	1	0
11	0	1
11	0	1
11	1	0

1∪4
2∪3 →

X	$Y_0 \cup Y_1$	$Y_x \cup Y_{\bar{x}}$
11	1	0
11	0	1

Таким чином, найпростіші перетворення табличного завдання функцій шляхом суперпозиції (об'єднання) вектор-стовпців, представлених в унітарному коді, з подальшим об'єднанням попарно однакових рядків (1 і 4; 2 і 3) дають можливість отримати компактне представлення двох таблиць істинності:

X	$Y_0 \cup Y_1$	$Y_x \cup Y_{\bar{x}}$
11	1	0
11	0	1

Інноваційна цінність такої таблиці полягає в очевидному розрізненні двох станів (стовпців) за одне формальне вимірювання на інтегрально записаному однорядковому тесті, завдяки унітарному кодуванню вхідних і вихідних станів таблиць істинності. Унітарний код хороший тим, що він здатний мінімізувати або суперпозиціонувати, або виконувати паралельно в часі будь-

які поєднання, в даному випадку: примітивних функцій. Все сказане відноситься і до квантових обчислень, які не є "чудовим" інструментом у порівнянні з теоретичними можливостями класичного комп'ютингу. Очевидно, що, будь-які квантові алгоритми реалізуються в класичному виконанні за рахунок збільшення пам'яті для унітарного представлення примітивних алфавітів, моделей, функцій і структур даних. У технічній діагностиці чудові властивості паралелізму при унітарному кодуванні символів багатозначного алфавіту продемонстрував у 1981 році Олександр Герцович Біргер, на конференції в Каунаському Політехнічному Інституті. Таким чином, існує взаємно-однозначна відповідність між теоретичною продуктивністю квантового та класичного комп'ютингу, з якого випливає практичний висновок: використання квантових структур даних і алгоритмів дає можливість класичному комп'ютингу підвищити швидкодію розв'язання практичних задач за рахунок збільшення апаратних витрат. Іншими словами, що можна зробити паралельно на квантовому комп'ютері, реалізується і на класичному з можливим збільшенням пам'яті для зберігання даних в унітарному форматі.

Алгоритм Шора використовується для розкладання натурального числа на прості множники. Класичні алгоритми вирішують цю задачу повним перебором, причому з ростом кількості знаків в розкладаному натуральному числі обчислювальна складність алгоритму зростає експоненціально: кожен знак збільшує час отримання результату в 10 разів. Квантовий алгоритм має поліноміальну залежність часу від кількості знаків у розкладаному числі. Повний перебір в класичному комп'ютингу можна зменшити до полінома обчислювальної складності шляхом паралельного розв'язання задачі покриття на основі апаратної реалізації Хассе-процесора [1].

Кібер-фізичний комп'ютинг є теорією і практикою точного хмарного керування віртуальними, соціальними і фізичними процесами і явищами на основі використання великих даних, метричного онлайн-моніторингу цифрового стану процесу або явища з метою поліпшення якості життя людини і збере-

ження екології планети. Існують певні тенденції в світі, які забезпечують технологічну основу для створення кіберсоціального комп'ютингу, як частини кіберфізичного комп'ютингу, в рамках технологічного укладу Internet of Things. Компанія Gartner Inc., яка пророкує глобальну технологічну кібермоду, в 2017 році додала вісім нових трендів в свій бренд Hype Emerging Technologies Cycle (рис. 1.2): 5G, Artificial General Intelligence, Deep Learning, Deep Reinforcement Learning, Digital Twin, Edge Computing, Serverless PaaS, and Cognitive Computing [2,3].

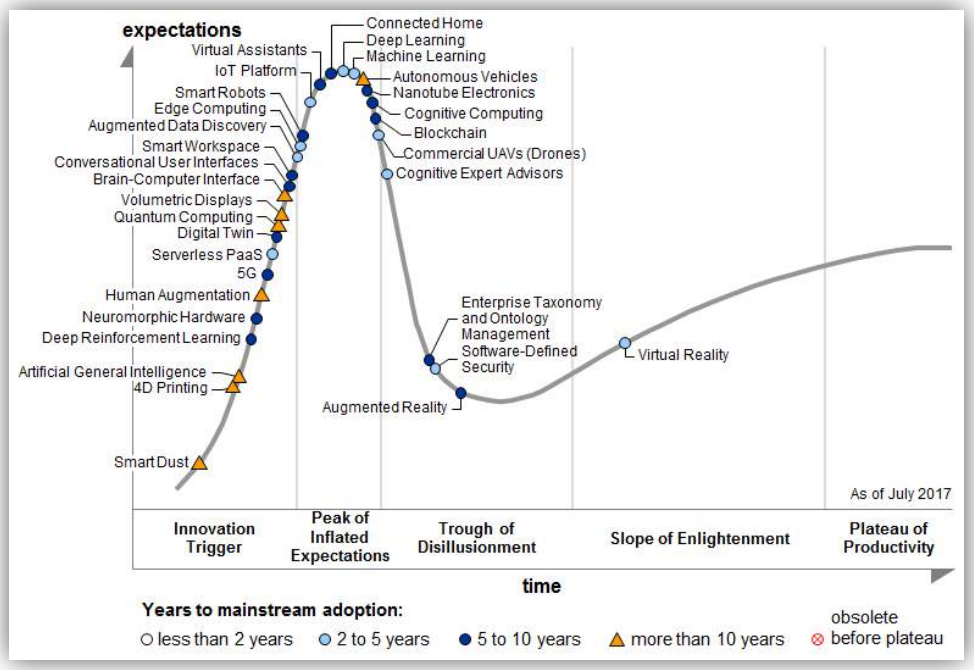


Рис. 1.2 – Gartner's Hype Cycle for Emerging Technologies

Edge computing є технологією підвищення продуктивності хмарних сервісів шляхом виконання локальних обчислень за місцем мобільного користувача. Digital Twin створює кібер-образи фізичних процесів і явищ. Як в дзеркалі, якщо немає відображення оцифрованої компанії (університету) в кіберпросторі, то її немає в фізичному просторі. Serverless PaaS – безсерверна архітектура для організації хмарних обчислювальних процесів на основі платформи як послуги (Platform as a Service). Економіка хмарної платформи є явно виграшною в порівнянні з серверною підтримкою діяльності компаній. Тому

весь малий і середній бізнес протягом двох років перейде на хмарні інфраструктури та сервіси.

Як слід розуміти фази Gartner циклу? 1) Innovation Trigger – запуск інновації, де потенційно цікаві для ринку проривні технології з ще недоведеною комерційної спроможністю, йдуть на зміну існуючим кіберфізичним укладам. 2) Peak of Inflated Expectations – пік роздутих ринкових очікувань, де своєчасна реклама створює успішні прецеденти створення інноваційних технологій на тлі безлічі невдач. 3) Trough of Disillusionment – прихід розчарування, коли інтерес до технологій згасає, експерименти не підтверджують очікувану ринкову привабливість, окремі розробники покращують свою продукцію і отримують інвестиції. 4) Slope of Enlightenment – схил прозріння, коли з'являються приклади технологій, які дають користь підприємству, знаходяться фінанси для пілотних проектів. 5) Plateau of Productivity – площина стійкого підвищення продуктивності, коли створювані технології, товари та послуги знаходять свого споживача на міжнародному ринку.

1.2 Три головні напрями кіберкультури

Hype-cycle 2017 формує кіберкультуру планети на наступні 5-10+ років шляхом експертного аналізу більш, ніж 1800 можливих технологій, що виконується провідними дослідницькими та консалтинговими компаніями. Список з 33+2 топ-технологій Gartner-таблиці створює технологічну кібер-культуру, структурно представлену на рис. 1.3, а також конкурентні переваги для суб'єктів ринку науки, освіти, індустрії та транспорту.

Перші три місця в Gartner-топ-циклі закріплено за такими стратегічними напрямами: Artifical Intelligence Everywhere, Transparently Immersive Experiences і Digital Platforms.

1) Artifical Intelligence Everywhere. Штучний інтелект стає найбільш дизрапторною технологією в наступні 10 років завдяки наявності обчислювальних потужностей, нескінченних обсягів великих даних і досягнень в ре-

алізації нейронних мереж для адаптації до нових ситуацій, з якими ніхто і ніколи не стикався раніше. Підприємства, які зацікавлені у використанні штучного інтелекту, розглядають корисними для себе наступні технології: Deep Learning, Deep Reinforcement Learning, Artificial General Intelligence, Autonomous Vehicles, Cognitive Computing, Commercial UAVs (Drones), Conversational User Interfaces, Enterprise Taxonomy and Ontology Management, Machine Learning, Smart Dust, Smart Robots and Smart Workspace. Таким чином, Artificial General Intelligence в наступні 10 років буде проникати в усі сфери людської діяльності, як технологічна послуга, занурена в кіберфізичний простір, у тому числі 30 відсотків високотехнологічних і транспортних компаній.

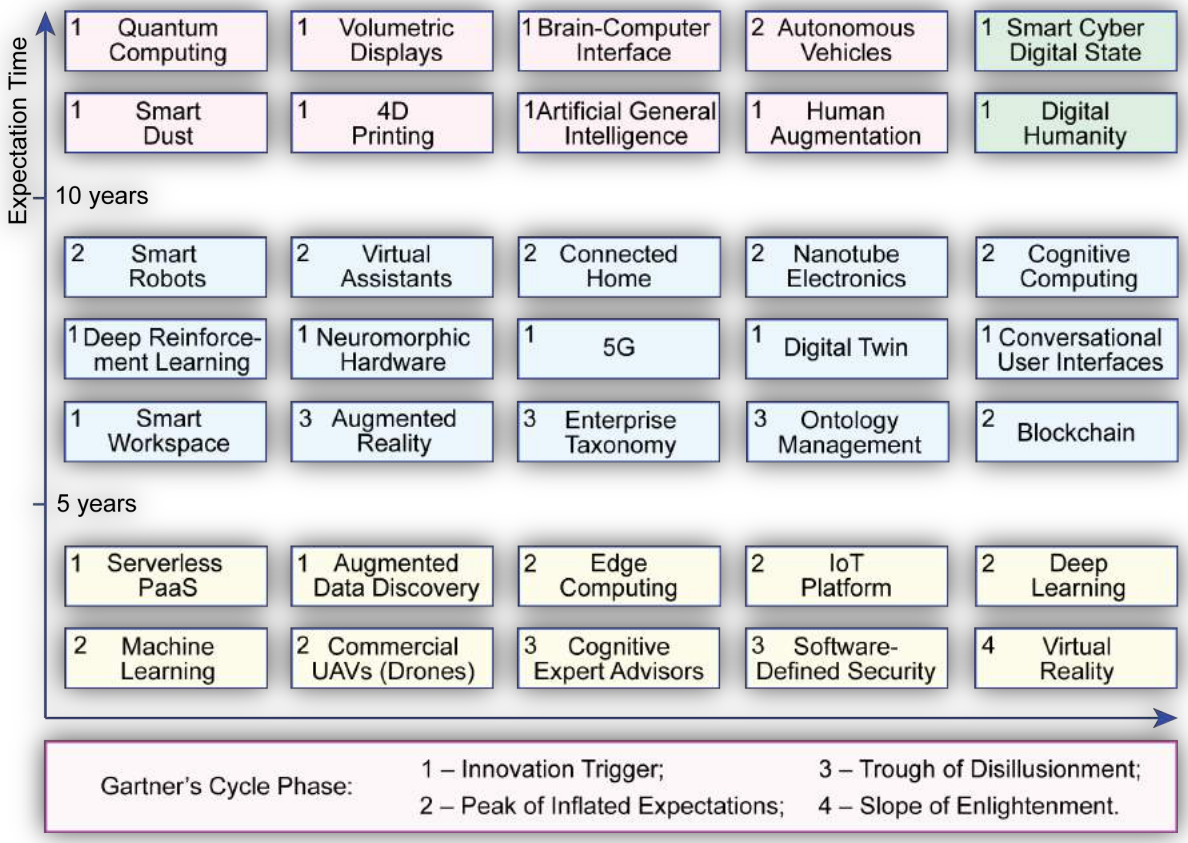


Рис. 1.3 – Gartner's Table for Emerging Technologies

Smart Workspace – розумне робоче місце означає бути під'єднаним до інфраструктури вирішення виробничих проблем в просторі і часі за форматом 24/7. При цьому використовуються віртуальні приватні мережі, метрика

вимірювання потенціалу і результатів діяльності, наявність певної кіберкультури і вибір найбільш зручних місць для ведення бізнесу. Висока самомотивація до успішного і результативного виконання завдання обумовлює використання динамічно змінюваного кіберфізичного робочого простору для творчості, інваріантного до офісу, дому, транспорту, місць відпочинку і спорту.

2) Transparently Immersive Experiences. Технології досвіду прозорого занурення стають все більш орієнтованими на людину і забезпечують: 1) прозорість відносин між людьми, бізнесом і речами; 2) гнучкість і адаптивність зв'язків між робочим місцем, будинком, підприємством та іншими людьми. Gartner-Inc. також передбачає впровадження в практику наступних, очікуваних усіма, критичних технологій [6-12]: Autonomous Vehicles, Brain-Computer Interfaces, Smart Dust, 4D Printing, Augmented Reality(AR), Connected Home, Human Augmentation, Nanotube Electronics, Virtual Reality (VR), Volumetric Displays. Інтеграція кібер-технологій спрямована на забезпечення якості життя людини шляхом створення: розумного робочого простору (smart workspace), під’єднаного дому (connected home), доповненої реальності (augmented reality), віртуальної реальності (virtual reality), мозг-комп'ютер интерфейсу (the growing brain-computer interface). Так наприклад, Human Augmentation технологія спрямована на розширення або доповнення людських можливостей з метою поліпшення здоров'я і якості життя за рахунок гармонійного використання когнітивних і біотехнічних покращень, як частин людського тіла. Volumetric Displays, як об'ємні дисплеї, візуалізують об'єкти за допомогою 3D активних елементів-вокселей (voxels) в трьох вимірах зі сферичним кутом огляду в 360 градусів, де зображення явища змінюється при переміщенні глядача. Технологія 4D Printing є інновацією 3D-друку, де конструктивні матеріали можуть трансформуватися після виробництва виробу з метою адаптації продукту до потреб людини і до навколишнього середовища.

3) Digital Platforms. Ключові платформи технологічної культури формуються компонентами [6-12]: 5G, Digital Twin, Edge Computing, Blockchain, IoT,

Neuromorphic Hardware, Quantum Computing, Serverless PaaS і Software-Defined Security. Такі технології, як Quantum Computing і Blockchain, будуть створювати найбільш непередбачувані і дизрапторні прориви для людини в найближчі 5-10 років. Neuromorphic Hardware розглядається як майбутнє штучного інтелекту, спрямоване на створення нейроморфного компьютингового чіпа, здатного замінити хмарні обчислювальні потужності Apple Siri Data Center при вирішенні складних задач Machine Learning (Chris Eliasmith, a theoretical neuroscientist and co-CEO of Canadian AI startup Applied Brain Research) [4]. Інакше, всередині iPhone з'явиться цифровий мозок у формі нейроморфного IP-core, здатний оперативно і на місці вирішувати всі завдання взаємодії гаджета з зовнішнім світом в реальному часі. Нейроморфний універсальний чіп IBM, завдяки спайковому асинхронізму, споживає на три порядки менше енергії при кількості транзисторів, що перевищує в п'ять разів існуючі апаратні рішення компанії Intel. Для програмування апаратно-орієнтованих алгоритмів використовуються компілятори: Nengo, Python. Шляхом використання компілятора Nengo сьогодні вже реалізовані цифрові системи на кристалах: vision systems, speech systems, motion control, adaptive robotic controllers, а також Spaun-chip для автономного інтерактивного спілкування комп'ютера з навколишнім середовищем. Software-Defined Security (SDS) або Catbird призначена для захисту системних об'єктів або логічних структур у віртуальному просторі. Це пов'язано з тим, що мережева безпека вже не має фізичних кордонів в рамках існування логічної архітектури хмарних сервісів. Тому створюється точна і гнучка SDS у вигляді доповнення до інфраструктур і центрів обробки даних без наявності спеціалізованих апаратних пристроїв захисту. Масштабування SDS дає можливість створювати або купувати мінімально необхідні умови безпеки в певному місці і часі, що істотно зменшує матеріальні витрати на формування якісного SDS сервісу.

1.3 Практика використання топ-технологій

Високі витрати на дослідження і розробки від Amazon, Apple, Baidu, Google, IBM, Microsoft і Facebook стимулюють створення оригінальних патентованих рішень в області Deep Learning і Machine Learning, серед яких слід відзначити: Amazon Alexa, Apple Siri, Google Now, Microsoft Cortana. Компанія Gartner Inc. впевнена, що інструменти для глибокого навчання становитимуть 80% стандартних засобів для вчених до 2018 року. Сьогодні вже на сайтах компаній стають доступними технології і дані про наукові дослідження: Amazon Machine Learning, Apple Machine Learning Journal, Baidu Research, Google Research, IBM AI, Cognitive Computing, Facebook Research.

Впровадження 5G-технології телекомунікацій (рис. 1.4) в найближче десятиліття надасть ринку очікувані інноваційні рішення з безпеки, масштабованості і продуктивності глобальних мереж і з'єднань в транспорті, IoT, індустрії, охороні здоров'я.

Gartner Inc. прогнозує, що до 2020 року 3% мережевих провайдерів послуг мобільного зв'язку запустять комерційні мережі в 5G-форматі, що забезпечить якісно нові умови повсюдного впровадження телекомунікацій для масштабованої глобалізації сервісів: IoT, cloud-transport control, UHD-телебачення. Лідерами 5G-впровадження в 2017-2018 році виступають: AT&T, NTT Docomo, Sprint USA, Telstra, T-Mobile і Verizon. Технологія 5G являє собою ультраширокополосний мобільний зв'язок в міліметровому діапазоні для Massive M2M транзакцій в реальному часі з допустимими для управління затримками (1мс), при одночасному підключенні близько 10 млн пристроїв на 1 км кв. 5G використовує технологію множинного доступу з поділом променя (Beam Division Multiple Access – BDMA) для взаємодії базової станції з мобільними пристроями. Бездротова стільникова архітектура 5G забезпечує пропускну здатність 10-50 Гбіт/с в міліметровому діапазоні частот 30-300 ГГц для додатків UHD відео і створення віртуальної реальності [5]. Інноваційна технологія 5G характеризується використанням: масиву приймально-

передавальних антен Massive MIMO, мережі Cognitive Radio, організацією безпосереднього зв'язку D2D для IoT, створенням мережі радіодоступу, як хмарної послуги (radio access network as a service) і хмари віртуальних мережевих функцій (network function virtualization cloud – NFV).

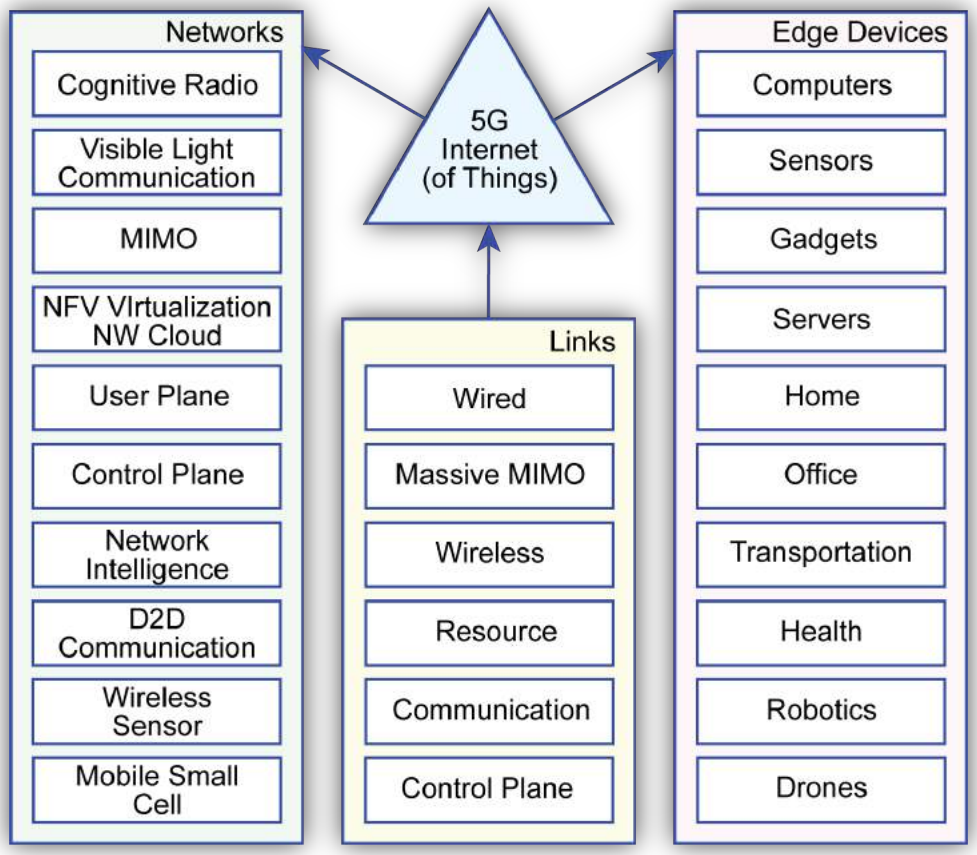


Рис. 1.4 – Три складових технології 5G

Green IoT – кіберфізична культура людської діяльності, спрямована на забезпечення якості життя людей і збереження екології планети, енергії, ресурсів і часу. Компонентами IoT є: Identification, Sensing, Controlling, Communication, Cumputation, Sevices Intelligent, Digital Infrastructure. Розумний світ (smart world) надає кожній людині сервіси від [6, 10-12]: розумних пристроїв (watches, mobile phones, computers), розумного транспорту (aircrafts, cars, buses, trains), розумної інфраструктури (homes, hospitals, offices , factories, cities, states), розумної освіти (school, university).

Компанія Gartner Inc. вважає за необхідне вивести з під парасольки актуальної ринкової моди такі технології, як не виправдали очікування IT-бізнесу: Affective Computing, Micro Data Centers, Natural-Language Question Answering, Personal Analytics, Smart Data Discovery and Virtual Personal Assistants.

Для створення успішних бізнесів і нових освітніх курсів компанія Gartner Inc. рекомендує враховувати свої припущення про стратегічне планування, які містять 10 пунктів (рис. 1.5) [2,10-12]: 1) До 2020 року 100 мільйонів споживачів будуть робити покупки в розширеній реальності, в тому числі з використанням Head-Mounted Displays (HMDs). 2) До 2020 року 30% сеансів перегляду веб-сторінок будуть виконуватися без використання екрану. Більше 5 з 550 мільйонів власників Apple iPhone використовуватимуть AirPods для обміну голосовими повідомленнями. П'ять відсотків веб-сайтів, орієнтованих на споживача, будуть оснащені аудіо-інтерфейсами (у тому числі голосові чати з підтримкою голосу).

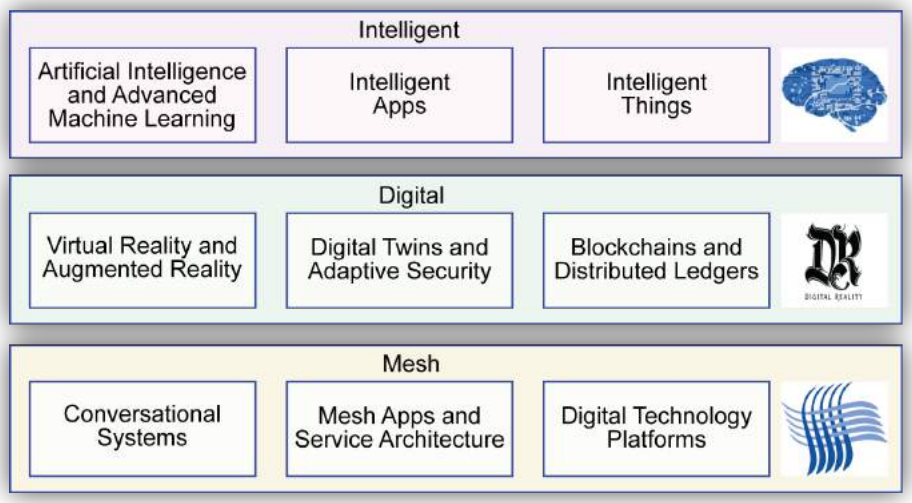


Рис. 1.5 – Топ-дизрапторні напрямки в IT-індустрії

3) До 2019 року 20% брендів відмовляться від своїх мобільних додатків (на користь MASA – Mesh App and Service Architecture). 4) До 2020 року розумні алгоритми позитивно вплинуть на поведінку понад 1 мільярду глобальних працівників. 5) До 2022 року бізнес на основі використання блокчейнів

буде коштувати 10 мільярдів доларів. 6) До 2021 року 20% усіх видів діяльності людини будуть додані, принаймні, в сервіси однієї з семи провідних глобальних компаній (Google, Apple, Facebook, Amazon, Baidu, Alibaba і Tencent). 7) До 2019 року кожний долар, інвестований в інновації, буде потребувати додаткових 7 доларів для основного виконання проекту. 8) Протягом 2020 року Internet of Things (IoT) на 3% збільшить попит, пов'язаний з data centers. Кімнатні екранні пристрої, такі як Amazon Echo і Google Home, будуть перебувати в більш, ніж 10 мільйонах будинків. 9) До 2022 року IoT і хмари (Google, Amazon, Microsoft) будуть економити споживачам і підприємствам 1 трильйон доларів на рік, орієнтованих на послуги і витратні матеріали. До 2020 року близько 40 мільйонів автомобілів будуть використовувати Android Auto, а 37 мільйонів транспортних засобів використовуватимуть CarPlay. 10) До 2020 року 40% співробітників зможуть скоротити свої витрати на охорону здоров'я, використовуючи фітнес-трекер.

1.4 Хмарні технології віртуального комп'ютингу

Кількість публікацій з тематики Cloud Computing в першоджерелах, які формуються бібліотекою IEEE Xplore дорівнює 44747. Така увага до віртуального комп'ютингу з боку вчених продиктована тотальною привабливістю даного сегмента ринку для промисловості, транспорту, охорони здоров'я, побуту, освіти і науки. Основна парадигма – завжди користуватися необмеженими масштабованими обчислювальними потужностями сучасного комп'ютингу всієї планети за помірну орендну плату. При цьому не потрібно піклуватися про захист сервісів і даних, оновлення швидко застарілого обладнання, на яке витрачається основна маса грошей для підтримки кіберінфраструктури компанії або університету. Крім того, компанії Microsoft, Amazon, Google, що надають хмарні платформи для бізнесу, створюють ідеальні умови і сервіси для перенесення всіх фізичних процесів і явищ у віртуальний кіберпростір, рис. 1.6. Підтвердженням сказаного нижче

наводяться приклади публікацій, де висвітлено цікаві хмарні рішення, що стосуються бізнесу, транспорту і освіти.

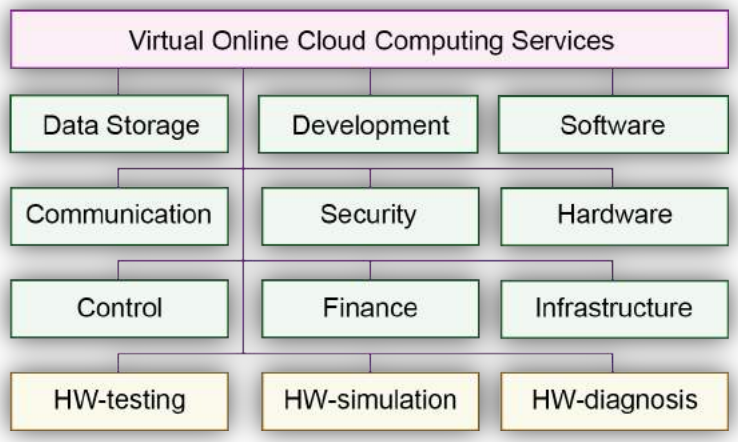


Рис. 1.6 – Cloud Computing Services

Практична реалізація виконуваних досліджень орієнтована на створення хмарних сервісів, пов'язаних з цифровим пристроєм, над яким виконується діагностичний експеримент в режимі online, що ілюструється структурою рис. 1.7.

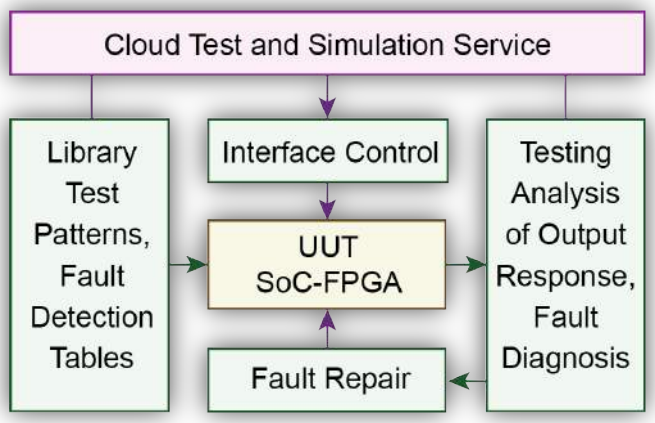


Рис. 1.7 – Практична реалізація результатів

Удосконалення технологічної культури, пов'язаної з Cloud Computing, створює потребу в наданні не тільки програмного, а й апаратного забезпечення «як послуги». Реконфігуровані апаратно-орієнтовані FPGA (Field Programmable Gate Arrays) є ідеальним рішенням для імплементації апарату-

ри в хмарні структури завдяки своїй високій гнучкості і масштабованості. Тут розглядаються [13-29] проблеми створення середовища проектування та верифікації на основі використання віддаленої реконфігурованої SoC (System on Chip) на базі FPGA з сервіс-орієнтованим інтерфейсом зв'язку через Інтернет, що ринково актуально для налагодження і тестування програмно-апаратних комплексів. Плата FPGA, яка може перебувати у віддаленому місці, налаштовується веб-службою, пов'язаною з веб-сторінкою JSP (Java Server Pages), де користувач може створювати файл конфігурації бітового потоку для запису в програмовну логіку (PL). На платі SoC/FPGA використовується спеціалізоване програмне забезпечення для керування завантаженням бітових потоків при конфігуруванні PL-блоків.

В даний час Cloud Computing (CC) затребуваний більше, ніж у 70 відсотках компаній планети і далі триває впевнене завоювання ринку привабливістю його сервісів [30]. Для задоволення потреб ринку у фахівцях з технологічної кіберкультури Cloud Computing починає впроваджуватися в освітні процеси університетів і середніх шкіл. Мається на увазі, що слухачі повинні бути ознайомлені з фізичною та логічною топологією і інфраструктурою, використовуваною під час кіберфізичних експериментів. Інтерес представляють описи експериментів, а також мінімальні фінансові та апаратні ресурси, що витрачаються на них, пов'язані з побудовою та використанням розумних будинків, керуванням соціальними і фізичними об'єктами в рамках технологічної культури IoT.

У міру розширення використання Інтернету, комп'ютерів і мобільних пристроїв виникає природна необхідність в розробці системи керування освітою студентів на основі хмарних обчислень для навчання програмуванню, проектуванню, тестуванню в рамках існуючих кіберфізичних технологій IoT [31]. Система керування навчанням має три платформи: SaaS, PaaS і IaaS. Навчальна платформа SaaS пропонує системи навчання, іспитів, документів, спілкування, програмування. Платформа навчання PaaS надає комунікації

для спілкування VS 2015, C-free, Eclipse, SQL Server, Oracle, IIS, Apache для навчання програмуванню. Навчальна платформа IaaS забезпечує інфраструктури: мережеву, обчислювальну, зберігання даних для навчання програмуванню. Інтеграція трьох навчальних платформ дає можливість реалізовувати будь-які проекти в системі хмарних обчислень OpenStack, використовуючи Microsoft ASP. Net (версія 4.0) і SQL Server (версія 2008). Експериментальні результати показують, що система керування освітою на основі хмарних обчислень для навчання кіберкультурі вельми ефективна і надає вичерпні навчальні послуги на платформах SaaS, PaaS, IaaS для студентів комп'ютерного напрямку.

В останні кілька років спостерігається експоненціальне збільшення кількості термінальних пристроїв з підтримкою Інтернету, що призводить до популярності туманних і хмарних обчислень серед кінцевих користувачів [32-47]. Останні чекають високих швидкостей передачі даних у поєднанні з безпечним доступом до даних для різних додатків, що виконуються на основі туманних обчислювачів або в базовій мережі хмарних обчислень. Однак двонаправлений потік даних між кінцевими користувачами і пристроями, розташованими у fog networks, може викликати перевантаження в хмарних центрах даних, які використовуються в основному для зберігання і аналізу даних. Висока мобільність транспортних засобів також може створювати додаткові проблеми щодо доступності і обробки даних в основних центрах обробки даних. Отже, необхідно більшість ресурсів перемістити в туманні мережі, щоб забезпечити online виконання додатків кінцевого користувача. Для вирішення проблем майбутніх вимог користувачів створюється архітектура, яка об'єднує хмарні і туманні обчислення в середовищі 5G спільно з передовими технологіями SDN, NFV і NSC. Модель обслуговування NSC допомагає автоматизувати віртуальні ресурси шляхом вибудовування послідовностей даних для швидкого обчислення в обох мережах. Архітектура також підтримує аналітику даних і керування ними для мобільних пристроїв.

Обслуговування завдань здійснюється шляхом використання гіпервізорів, віртуалізації, неоднорідності пристроїв і даних, безпеки простору при атаках на дані в середовищі 5G.

В даний час SaaS (програмне забезпечення як послуга) є новою парадигмою для галузей програмної індустрії і фахівців [48]. Як програмне забезпечення може пропонуватися як послуга? Які вимоги і проблеми SaaS? Що таке архітектурна різниця між SaaS та іншими хмарними обчисленнями? Які проблеми впровадження SaaS? Це кілька питань, на які потрібно відповісти при реалізації SaaS. Надання програмного забезпечення в якості послуги фактично зводить до мінімуму піратство в індустрії програмних продуктів. Це також спрощує процес оновлення та розповсюдження програмного забезпечення. Централізований супровід програмного забезпечення в безпечному середовищі CSP (Cloud Service Provider) звільняє користувача від забезпечення безпеки. SaaS – це верхній рівень хмарних обчислень, тому користувачам хмари не потрібна висока термінальна обчислювальна потужність на стороні користувача. Це допомагає істотно скоротити витрати користувача, зменшуючи покупки додаткового устаткування для інсталяції програмних продуктів. SaaS є сьогодні популярною і новою структурою хмарних обчислень, яку використовують багато розробників програмного забезпечення і галузі для розгортання свого програмного забезпечення. Розглядається інфраструктура для імплементації приватних SaaS з використанням мінімальних ресурсів, корисних для невеликих компаній і університетів, що бажають підвищити ефективність своєї роботи і зменшити витрати на обладнання.

Комп'ютерний світ став дуже великим і складним. Cloud – це нова прогресивна парадигма в світі інформаційних технологій. Хмарні обчислення пропонують IT-можливості як послуги [49]. Хмарні сервіси надаються на вимогу, легко масштабуються, не залежать від кінцевих пристроїв і є надійними. Хмарні обчислення базуються на концепції віртуалізації. Вона відокремлює апаратне забезпечення від програмного і має переваги консолідації сер-

верів і мобільності користувачів. Наведено огляд хмарних обчислень, а також описано інфраструктуру технології «віртуалізації».

Завдяки наявності конкуренції, що дозволяє максимізувати прибутковість своєї інфраструктури, сучасні хмарні платформи швидко розвиваються і наступають на серверні ринки, що пропонують, крім простих серверів на вимогу, все більш складні контракти, такі як спотові, превентивні, переривчасті і зарезервовані сервери [50]. Паралельно даному сегменту виникають успіхи в системній і мережевий віртуалізації, де сервер-час стає все більш затребуваним товаром. Ці тенденції мотивують появу відкритих хмарних ринків, аналогічних ринкам нафти, золота, напоїв, машин. Однак відкриті хмарні ринки ще не матеріалізувалися через ключові відмінності між хмарними ресурсами та іншими товарами. Зокрема, взаємозв'язок між додатками та їх базовими серверними ресурсами принципово відрізняється і є більш складним, ніж для інших товарів. На жаль, розробники програмного забезпечення поки не мають відповідних засобів ефективного керування ринком своїх додатків. Фінансові хмарні обчислення – це нова галузь, яка фокусується на адаптації та розширенні кіберкультури економіки і фінансів для online компромісного керування в цілях оптимізації додатків, вартості, ризиків, доступності та продуктивності на хмарних ринках. Основна задача фінансових хмарних обчислень – розробка моделей системного рівня і механізмів, які керують складністю і різноманіттям ринку. Нова область досліджень визначає потенційні вигоди становлення і використання хмарного товарного ринку.

Сучасний Cloud Computing в основному базується на запатентованих центрах обробки даних, де сотні тисяч виділених серверів налаштовані на розміщення хмарних сервісів [51]. На додаток до величезної кількості виділених серверів, розгорнутих в центрах обробки даних, існують мільйони недостатньо використовуваних персональних комп'ютерів (ПК), які зазвичай використовуються лише кілька годин в день, які належать приватним особам

і організаціям по всьому світу. Величезні невикористані можливості спільного використання та зберігання недовикористовуваних ПК можуть бути об'єднані в якості альтернативних хмарних мереж для надання інфраструктурних сервісів. Цей підхід «без центру обробки даних» доповнює модель надання хмарних даних на базі найбільших дата центрів. Таким чином, рішення «без центру обробки даних» дійсно працює і може мати на ринку послуг свій перспективний сегмент для капіталізації.

1.5 Тестування та моделювання несправностей цифрових структур

Запит за поєднанням слів "Fault Simulation" дає 32108 посилань на наукові публікації в IEEE Xplore Digital Library. Такий інтерес ринку визначається перманентними проблемами, пов'язаними з дефектами в технічних виробах різного призначення: комп'ютинг, транспорт, медицина, будівництво, космонавтика, озброєння, енергетика. Що стосується моделювання несправностей, тут загальна картина даної галузі знань може бути представлена взаємодією компонентів: моделі, методи і дефекти, представлені на рис. 1.8. Представлене різноманіття досить глибоко досліджено вченими протягом останніх 60 років, що відображено в десятках тисяч публікацій світового рівня, наприклад [52-61]. Окремі публікації останніх років орієнтовані на створення нових моделей і методів моделювання, які є наслідком появи нових компьютингових апаратних рішень і технологій, забезпечених штучним інтелектом, хмарними сервісами і квантовими структурами даних [62-79].

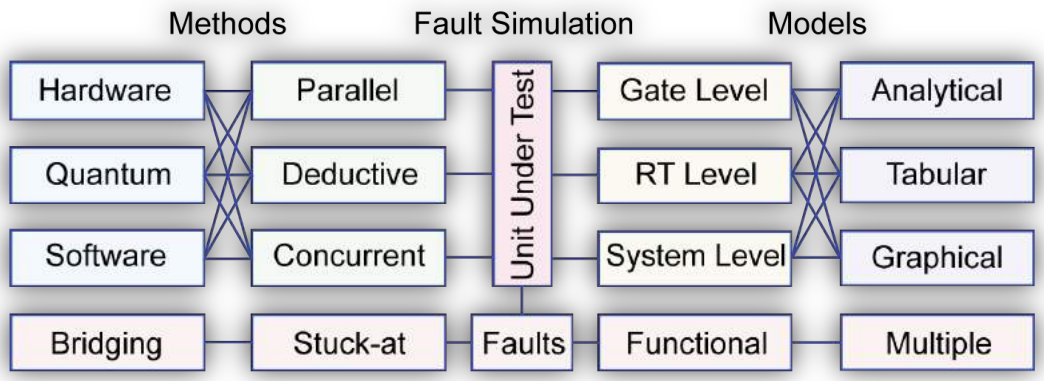


Рис. 1.8 – Моделі та методи моделювання несправностей

Уразливість хмарних сервісів є серйозною проблемою для державних структур і компаній. В роботі [66] представлена модель хмарної архітектури, яка містить широкий спектр засобів контролю захисту, а також модель оцінки хмарної безпеки Cloud-Trust, конфіденційності та цілісності. Імовірність проникнення істотно падає, якщо застосовується архітектура глибокого хмарного захисту віртуальної машини (VM) на основі постійного спостереження і керування всіма процесами, що відбуваються в хмарі.

Поліпшення покриття несправностей наборами тестів для реальних дефектів є важливою проблемою в області HBIC [69]. Якість моделі несправностей фактично впливає на валідність тестових послідовностей. Модель SAF (константні несправності) відіграє провідну роль в цифрових схемах протягом тривалого часу, але дослідження показують, що існує багато інших дефектів, які не можуть бути покриті підмножиною SAF. В результаті в останні роки постійно пропонуються нові моделі несправностей (затримки, замикання, розриви, функціональні несправності) і методи їх моделювання. Запропоновано алгоритм моделювання зазначених видів дефектів (у тому числі дефектів затримки), заснований на тестуванні IDDQ, а також метод визначення покриття несправностей. Робиться висновок, що з точки зору існуючих моделей важко описати всі види нових дефектів, викликані недосконалістю технологій при виготовленні нанометрових цифрових логічних схем.

Запропоновано швидкодіючий метод моделювання дефектів, який базується на точному трасуванні критичного шляху для комбінаційних схем [73]. Щоб перетворити послідовностну структуру моделювання несправностей в комбінаційну, в схему вводиться багатовходовий регістр MISR для покращення спостережуваності схеми. Роль таких регістрів полягає в моніторингу сигналів на лініях глобальних зворотних зв'язків і на окремих внутрішніх змінних. Послідовностна схема трансформується в ряд послідовностних або комбінаційних підсхем з додатковими точками спостереження на лініях глобальних зворотних зв'язків. Модельована тестова

послідовність організується в локальні набори вхідних послідовностей, що застосовуються до підсхем. Для локальних тестових наборів кожна підсхема є несправною, яка моделюється шляхом використання точного паралельного критичного трасування шляху, аналогічного комбінаційній еквівалентній схемі. Показано здійсненність і коректність методу, а також наведено експериментальні результати, які демонструють прискорення, досягнуте за допомогою методу.

Технології та засоби самодіагностування несправностей широко використовуються в різних бортових комп'ютерах транспортних засобів [76]. Щоб реалізувати тестування програмного забезпечення апаратури для авіоніки, в статті пропонується відмовитися від моделювання дефектів. Для цього був спроектований і впроваджений інжектор несправностей структур пам'яті. Після аналізу типового режиму відмови пам'яті і способу ініціювання дефектів була побудована модель несправностей пам'яті. На основі симулятора "open source QEMU" побудований модуль додавання помилок для підтримки моделювання дефектів модуля пам'яті і помилок декодера. Проведено численні експерименти для перевірки автентичності і достовірності інструменту додавання дефектів.

Надійність є ключовим фактором при прийнятті рішень в сучасному глобальному бізнес-середовищі [84]. Потужним методом, що дозволяє оцінювати надійність системи, є моделювання відмов у роботі. Ідея цього підходу полягає в тому, щоб генерувати відмови системи і відслідковувати її реакції, щоб спостерігати за її поведінкою при наявності несправностей. Були розроблені і експериментально перевірені кілька методів та інструментів для ін'єкцій дефектів. Вони можуть бути згруповані за трьома категоріями: апаратна ін'єкція дефектів, моделювання та емуляція відмов на основі використання мікропроцесорів.

Моделювання несправностей. Сутність моделювання несправностей полягає у визначенні впливу одного або декількох дефектів на стани ліній

об'єкта при подачі тестових послідовностей [90,91,94-97,99-101,103,104]. Методи моделювання несправностей можна класифікувати наступним чином: одиночне, паралельне, дедуктивне, кубічне і спільне моделювання.

Одиночне моделювання несправностей базується на внесенні однієї поодинокої константної несправності еквіпотенційної лінії в схему. При подачі тестових послідовностей виконується аналіз прояву несправності на зовнішніх виходах об'єкта діагностування. Метод орієнтований на обробку схем нерегістрового рівня і не вимагає значних витрат часу.

Паралельне моделювання несправностей грунтується на використанні машинних команд паралельної обробки слів (регістрів): логічне додавання, множення, інверсія, виключне АБО. Метод відноситься до компілятивного моделювання, оскільки поведінка примітивів схеми описується за допомогою алгоритмічних мов або асемблерів. В процесі моделювання одночасно виконується аналіз P несправностей на вхідному наборі, де P – розрядність машинного слова, доступного для паралельної обробки. До недоліків методу відносять складність проектування моделей та їх орієнтацію на конкретну обчислювальну платфрму. Швидкодія методу в P разів вище одиночного моделювання несправностей. Ідея паралельної обробки бінарного вектора за допомогою тільки логічних операцій може бути використана для істотного збільшення швидкості моделювання.

Дедуктивне моделювання несправностей полягає в одночасній обробці всіх поодиноких константних несправностей схеми на одному вхідному наборі і виділення при цьому підмножини перевірюваних дефектів. Метод орієнтований на вентильний рівень опису моделі проектованого об'єкта в базисі І-АБО-НЕ. Необхідність отримання аналітичних формул для кожного типу примітивного елемента і великі витрати пам'яті для зберігання списків несправностей ускладнюють практичну реалізацію методу.

У спільному (конкурентному) моделюванні, як і в дедуктивному, виявляються відразу всі перевірювані несправності для даного вхідного набору.

Метод орієнтований на обробку різних типів моделей схем, несправностей, затримок і сигналів. На відміну від дедуктивного методу, де дефекти моделюються неявно, конкурентний алгоритм аналізує явно справну роботу і ті несправності, які модифікують стани входів або виходів схеми, щоб використовувати ефективні моделі елементів, такі як табличні і функціональні.

Дедуктивно-паралельне моделювання несправностей цифрових систем ґрунтується на використанні переваг дедуктивного і паралельного алгоритмів [90] і дозволяє за одну ітерацію обробки модельованої схеми виявити всі несправності, що перевіряються на тест-векторі. Метод дозволяє істотно підвищити швидкодію моделювання поодиноких константних несправностей для оцінки якості синтезованих тестів цифрових систем, імплементованих в ПЛІС, що містять мільйони вентилів.

1.6 Кубітна схемотехніка квантового комп'ютингу

Представлено базові компоненти квантових обчислень для паралельного вирішення специфічних NP-повних задач комбінаторики, оптимізації, кібербезпеки і аналізу великих даних. Найактивнішими у створенні квантових комп'ютерів є компанії: Google, Microsoft, Intel, IBM, D-Vawe, Los-Alamos Lab, US Energy Department, HRL Laboratories, Lockheed Martin, Berkeley Lab, Univ of Sussex.

Останні дослідження в області квантових обчислень [1-33] свідчать про те, що вже в найближчі 10 років квантовий комп'ютер дозволить ефективно вирішувати такі завдання: моделювання соціальних відносин, логістична оптимізація, machine learning, error corrected computation, криптографія, паралельна обробка великих даних, паралельний аналіз множини питань з одночасним формуванням відповідей, моделювання фізичних і хімічних процесів на мікрорівні, створення штучного інтелекту і квантового криптографічного зв'язку. Всі завдання експоненціальної складності в класичному компьютингу квантовий комп'ютер переводить у площину поліноміальної

складності, що робить його унікально перспективним для всіх існуючих NP-повних (складних) рішень. Квантовий комп'ютинг містить 4 фази: Preparation Data, Evolution, Measurement, Extraction of Output Information. Але тільки в спільному проектуванні і використанні квантового та класичного комьпютингу можна отримати рішення глобальних задач, пов'язаних, наприклад, зі створенням штучного інтелекту людства.

Мета – аналітичний огляд квантових технологій тестування, моделювання та діагностування цифрових систем на кристалах, орієнтований на істотне підвищення швидкодії комбінаторного комп'ютингу. Джерела дослідження: квантові структури даних і обчислення [105-117]; застосування квантових методів до вирішення задач синтезу логічних схем, тестування і діагностування [118-137]. Квантові обчислення (quantum computing) є областю досліджень, яка швидко розвивається в останнє десятиліття [105-117], через 80 років після створення Максом Планком у 1900 році квантової механіки.

Квантовий комп'ютер – це обчислювальний пристрій, що використовує при роботі квантово-механічні явища: суперпозицію станів (superposition), квантове переплутування (entanglement), інтерференцію (interference), паралелізм (parallelism) і оборотність обчислень (reversible computation) [108], що дозволяє подолати обмеження швидкодії класичних комп'ютерів в області комбінаторики.

В класичних електронних цифрових обчислювальних машинах одиницею інформації є біт, який може приймати одне з двох можливих значень 0 або 1 і визначається як помітну відмінність (a difference that makes a difference) [116]: $\text{one bit} \in \{0,1\}$. Одиницею інформації квантової системи є квантовий біт, або кубіт [108], який відповідає станам $|\psi>$ квантової сутності, простір станів якої є комплексним двовимірним векторним простором $|\psi> \in C^2$. Кубіт визначається такою нескінченною кількістю кла-

сичної інформації: one qubit $=|\psi>=\alpha|0>+\beta|1>\in C^2$, де $|0>$, $|1>\in C^2$ означає ортонормальний базис; комплексні числа $\alpha,\beta\in C$ задовольняють виразу $|\alpha|^2+|\beta|^2=1$. Однак оскільки стани $|\psi>$ і $e^{i\eta}|\psi>$ для всіх $\eta\in[0,2\pi]$ еквівалентні з квантово-механічної точки зору, то α,β мають разом лише два ступеня свободи. Так, наприклад, можна прийняти: $\alpha=\cos\theta;\ \beta=e^{i\eta}\sin\theta;\ \eta,\theta\in[0,2\pi]$.

Таким чином, у порівнянні з класичним бітом кубіт має ширші можливості і дозволяє зберігати в одному кубіті фактично нескінченний в обидві сторони класичний інформаційний контент, в той час як один біт може містити мінімальний скінченний інформаційний контент. Тут проявляється одне з дивних квантових явищ, яке є предметом знаменитої загадки «кішки Шредінгера». З одного боку, квантовий комп'ютер може ефективно обробляти нескінченну в обидві сторони інформацію, яка знаходиться в кубіті, завдяки таким квантовим явищам, як суперпозиція, паралелізм, інтерференція, переплутування. Але при цьому обробка одного кубіта здійснюється за допомогою простої базової операції, як і обробка класичного біта в класичному компьютингу.

Для формування інформаційного змісту кубіта необхідно здійснити вимірювання у відповідній квантової системі. У загальному випадку це призводить до руйнування (згортання) відповідної хвильової функції, яка визначає стан $|\psi>$ кубіта. Класична інформація, яку можна отримати, знаходиться тільки в одному звичайному біті. Квантова система знаходиться в стані $|0>$ або $|1>$ з певною ймовірністю $|\alpha|^2$, $|\beta|^2$. У загальному випадку квантова система не знаходиться ні в одному з можливих станів $|0>;|1>$, а одночасно перебуває в обох станах $|0>,|1>$ (суперпозиція двох станів). Відмінності між квантовими і класичними комп'ютерами стають ще більш явними, коли необхідно обробляти будь-яку скінченну кількість $n\geq 1$ кубітів. Число n

можна визначити також як кількість класичних бітів, необхідних для отримання одного цілого $1 \le m \le 2^n$. Завдяки квантової суперпозиції, визначення n кубітів дає можливість отримати одночасно не менше, ніж 2^n цілих чисел). За аналогією зі звичайним електронним цифровим комп'ютером, вентиль квантової обчислювальної системи, може обробляти один або кілька кубітів [108,117]. Система перетворює їх відповідно до аксіом квантової механіки. Стани квантової системи перетворюються за допомогою унітарних, оборотних (інвертовних) операторів. При цьому квантові вентилі повинні мати однакову кількість вхідних і вихідних кубітів.

Простий квантовий вентиль A, що перетворює один вхідний кубіт в один вихідний кубіт, наведено на рис. 1.9.

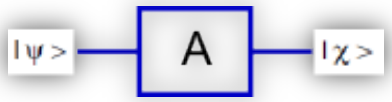


Рис. 1.9 – Квантовий вентиль

Квантові вентилі перетворюють кубіти за допомогою унітарних оборотних операторів. Тут мається на увазі, що $|\psi>, |\chi> \in C^2$ – кубіти; A: $C^2 \to C^2$ – унітарний лінійний оборотний оператор. Зручно використовувати матричне подання квантових вентилів: $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, $|\psi> = \alpha|0> + \beta|1>, |\chi> = \gamma|0> + \delta|1>$, для яких $A|\psi> = |\chi>$, справедливий вираз: $\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}$. Приклад квантового вентиля NOT, що задається унітарною матрицею: $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. Неважко помітити, що в цьому випадку останній вираз перетворюється до виду: $X(\alpha|0> + \beta|1>) = \beta|0> + \alpha|1>$, де вентиль NOT є перемикачем між станами

$|0>$ та $|1>$. Інший приклад квантових вентилів, описуваних унітарними матрицями:

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix},\ Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

які діють на даний кубіт у відповідності з виразами:

$$Y(\alpha|0> + \beta|1>) = -i(-\beta|0> + \alpha|1>),$$

$$Z(\alpha|0> + \beta|1>) = \alpha|0> - \beta|1>,$$

де X, Y, Z називають матрицями Паулі. Вентиль Адамара здійснює самооборотну операцію формування суперпозиції станів і визначається унітарною матрицею [108,117]: $H = (1/\sqrt{2})\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. Справедливим є вираз $X^2 = Y^2 = Z^2 = H^2 = I$, який означає, що кожен вентиль X, Y, Z, H визначається за допомогою кореня квадратного одиничної матриці і відповідного квантового вентиля I. У багатокубітному вентилі має бути однакова кількість кубітів на вході і виході [108]. Двокубітні вентилі відповідають операціям повороту в гільбертовому просторі двох взаємодіючих кубітів, які не можуть бути представлені у вигляді прямого добутку незалежних однокубітових операцій [117]. Основним двокубітовим вентилем є оборотний контрольований інвертор або оператор «контрольоване НЕ» (CNOT) з двома вхідними і двома вихідними кубитами (рис. 1.10), який функціонує відповідно до виразів:

$$|00> \mapsto |00>,\quad |01> \mapsto |01>,$$
$$|10> \mapsto |11>,\quad |11> \mapsto |10>.$$

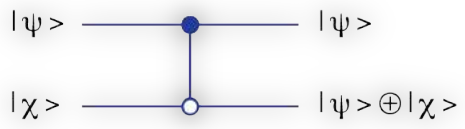


Рис. 1.10 – Контрольований інвертор

Коли $|\psi>=|0>$, то $|\psi>\oplus|\chi>=|\chi>$; для $|\psi>=|1>$ справедливо $|\psi>\oplus|\chi>=X|\chi>$. Вентиль CNOT описується матрицею:

$$\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}\alpha\\\beta\\\gamma\\\delta\end{pmatrix}=\begin{pmatrix}\alpha\\\beta\\\delta\\\gamma\end{pmatrix},$$

де $|\psi>=\alpha|0>+\beta|1>, |\chi>=\gamma|0>+\delta|1>$. Кубіт $|\chi>$ є контролюючим, кубіт $|1>$ – контрольованим, над яким здійснюється операція NOT за умови, що перший кубіт знаходиться в стані $|1>$. Двокубітовий оператор обміну станами кубітів SWAP може бути реалізований шляхом послідовного виконання трьох операцій CNOT (рис. 1.11) [117] і описується матрицею:

$$SWAP=\begin{pmatrix}1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\end{pmatrix}$$

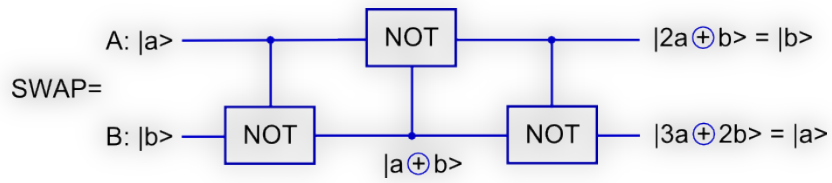


Рис. 1.11 – Оператор SWAP

Трикубітовий вентиль Тоффолі (CCNOT) представлений на рис. 1.12 і містить два керуючих кубіти A і B, а також один керований C.

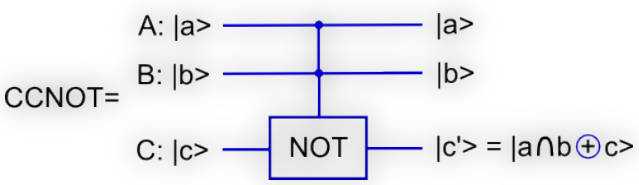


Рис. 1.12 – Вентиль Тоффолі

Вентиль Тоффолі описується матрицею 8x8 в базисних станах | 0,0,0>, | 0,0,1>, | 0,1,0>, | 1,0,0>, | 0,1,1>, | 1 , 0,1>, | 1,1,0>, | 1,1,1>:

$$\text{CCNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Ця операція може бути також реалізована у вигляді п'яти двокубітових операцій. N-бітний узагальнений вентиль Тоффолі описується як $(k_1,k_2,...,k_n) \rightarrow (k_1,k_2,...,(k_1,k_2,...,k_{n-1}) \oplus k_n)$[122]. Вентиль NOT є окремим випадком вентиля Тоффолі, для якого n=1, вентиль CNOT - окремим випадком, коли n=2.

Розширений n+1-бітовий вентиль Тоффолі (extended Toffoli gate, ETG) з двома керованими лініями (o_n,o_{n+1}) показаний на рис. 1.13. ETG має вхідний вектор $(k_1,k_2,...,k_n,k_{n+1})$ і вихідний вектор $(o_1,o_2,...,o_n,o_{n+1})$, де $o_j = k_j$ для $j = \overline{1,(n-1)}$, $o_n = k_1,k_2,...,k_{n-1} \oplus k_n$, $o_{n+1} = k_1,k_2,...,k_{n-1} \oplus k_{n+1}$. Перші n-1 бітів є керуючими, останні два – керованими.

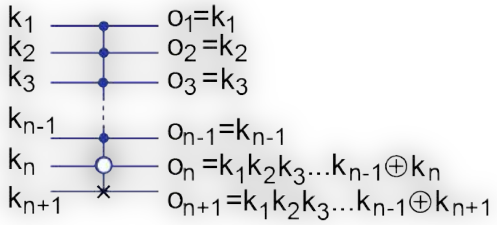


Рис. 1.13 – $n+1$-бітовий вентиль Тоффолі

Унікальною властивістю процесу тестування оборотної логіки є те, що будь-який тестовий набір, який може виявляти всі поодинокі несправності в оборотній логічній схемі, буде також виявляти всі кратні константні несправ-

ності. Ця властивість не може бути досягнута в необоротній логіці, де виявлення кратних константних несправностей є значно складнішим, ніж виявлення поодиноких константних несправностей.

Лише окремі існуючі моделі несправностей традиційної логіки можуть бути поширені на квантову логіку. Наприклад, при синтезі квантової логіки не використовується поведінковий рівень [118]. Несправності, пов'язані із затримками і дефектами, не мають аналогів в квантовій логіці. Моделі несправностей логічного рівня, орієнтовані на опис з'єднань квантових вентилів, повинні бути адаптовані до квантових схем. Під несправностями логічного рівня в квантовій логіці розуміють несправності рівня RTL (register transfer level).

Найбільш широко використовуваними моделями несправностей рівня RTL є константні несправності. У звичайній логіці вони моделюються шляхом присвоєння фіксованого логічного значення (0 або 1) певній точці тестованої схеми. Для квантових схем запропоновано дві моделі, перша – для двійкових переставних схем, друга – для квантових вентилів. У першій моделі для виявлення замикання провідника визначається багатозначний стан ланцюга $\{|0>, |1>, |V_0>|$ і $|V_1>\}$. Основна відмінність між двома моделями полягає в необхідності використання імовірнісних тестів при появі на виходах схеми комплексних значень.

У традиційній логіці часто використовуються спеціальні моделі несправностей, які можуть застосовуватися і для квантових схем. Запропоновано сімейство моделей несправностей логіки для оборотних схем, що базуються на k-входовому елементі «контрольоване НЕ» (k-CNOT) і відносяться до моделі несправностей, яка визначається як повне видалення вентиля – single missing-gate fault model (MGF) [119]. Несправності stuck-open і stuck-short використовуються для моделювання специфічних проблем транзистора в структурі КМОП вентиля.

Несправності витоку дуже важливі в традиційній цифровій логіці. Передбачається, що кубіт функціонує в повній ізоляції у двовимірному гільбертовому просторі H_2. При появі помилки кубіт може або взаємодіяти з навколишнім середовищем або непередбачувано повертатися в двовимірному просторі. Витік у квантових схемах відбувається, коли кубіт просочується з двовимірного гильбертова в більший простір. Тестування несправностей витоку квантових схем – це складне завдання. Несправності, які проявляються в поворотах кубіта, можуть бути змодельовані як ненавмисне додавання одного кубіта до вентиля Паулі, що викликає лише зміну фази кубіта, яким можна знехтувати, оскільки напрям, який він вказує в гільбертовому просторі, є тільки частиною інформації кубіта. Використання квантових властивостей, таких як перетворення Уолша-Адамара, для досягнення квантового паралелізму

$$H^{\otimes m} \mid 0>^{\otimes m} = \frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} |x> \text{ і } H^{\otimes m} \left(\frac{1}{\sqrt{2^m}} \sum_{x=0}^{2^m-1} \mid x>\right) = \mid 0>^{\otimes m}$$

дозволяє тестувати весь масив схем за допомогою одного тестового набору |000 ... 0>. В цьому випадку масив називається «1-тестопридатним» («1-testable») не тому, що для нього досить тільки одного тестового набору. Кількість тестових наборів не залежить від кількості входів і розміру масиву.

В роботі [132] наведено приклад побудови квантової системи вбудованого самотестування (Quantum Built-In Self-Test, QBIST) на основі використання властивості «1-тестопридатності». При вмиканні в тестовому режимі, первинні виходи встановлюються в нульовий стан. Ця операція виконується шляхом ініціалізації кубітів або створення резервної копії попереднього стану первинних входів в нормальному режимі. Контрольований нулем вентиль NOT може бути використаний для перевірки необхідних входів (чи встановлені вони всі в нуль). Контрольований вентиль Адамара використовується тут як генератор квантових тестових наборів (QTPG). Усі 2^m комбінацій те-

стових наборів піддаються суперпозиції і можуть бути одночасно використані для тестування. Тестована схема (CUT) перевіряється з використанням тільки одного тестового набору. Завдяки властивості «1-тестопридатності» для виконання вичерпного тестування не потрібні двійковий лічильник і лінійний зсувний регістр зі зворотним зв'язком, які використовуються в класичних BIST. Після перевірки схеми інформація передається на квантовий аналізатор відгуків (quantum output response analyzer, QORA), як це робиться в класичній BIST. Перевагою є те, що в даному випадку немає необхідності виконувати стиснення даних і складний сигнатурний аналіз. Спрощена структура CLB Xilinx FPGA показана на рис. 1.14.

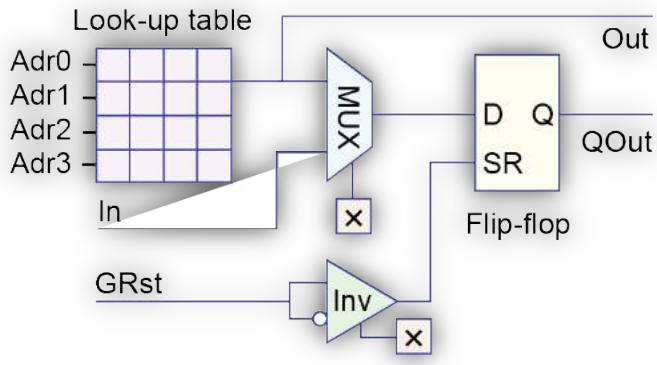


Рис. 1.14 – Джерела несправностей в структурі CLB

Конфігурований блок складається з набору таблиць перетворення (Look-up Tables – LUTs), тригерів і міжз'єднань. Джерелами несправностей SEU в CLB є: 1) біти таблиці перетворень, коли SEU змінює логічну функцію, імплементовану в look-up table; 2) конфігураційні біти таблиці перетворень; наприклад, біти, що встановлюються, якщо ресурс look-up table конфигурований як таблиця перетворень, двійковий порт RAM або регістр зсуву; SEU змінює функціональність таблиці перетворень; 3) мультиплексори та інвертори, де SEU змінює внутрішні зв'язки CLB.

Метод корекції помилок мультипроцесорних систем на базі FPGA описаний нижче. Одне з процесорних ядер сканує конфігураційні фрейми і

здійснює реконфігурацію в разі виявлення несправностей (перший прохід). Якщо несправність з'явилася безпосередньо в процесорному ядрі, інше ядро бере на себе функцію відновлення і виконує реконфігурацію (розширене відновлення). Під час нормальної роботи системи обрані процесорні ядра виконують перевірку конфігурації паралельно з виконанням цільового системного додатку. Схеми FPGA є уразливими для SEU і несправність може виникнути в будь-якій комірці конфігураційної пам'яті в будь-який час.

1.7 Постановка мети і задач дослідження

Виходячи з аналізу публікацій в області квантового, хмарного комп'ютингу, а також робіт, пов'язаних з проектуванням і тестуванням цифрових пристроїв, робиться висновок про актуальність хмарного рішення науково-практичної задачі квантового проектування, моделювання і тестування цифрових пристроїв і компонентів на основі використання memory-driven кубітних структур даних, вільних від застосування логіки.

Сутність дослідження структурно представлена на рис. 1.14 і полягає в розробці хмарних сервісів, моделей і методів квантового синтезу та аналізу memory-driven кубітних моделей цифрових пристроїв і компонентів для істотного підвищення швидкодії засобів проектування, тестування, моделювання та діагностування несправностей.

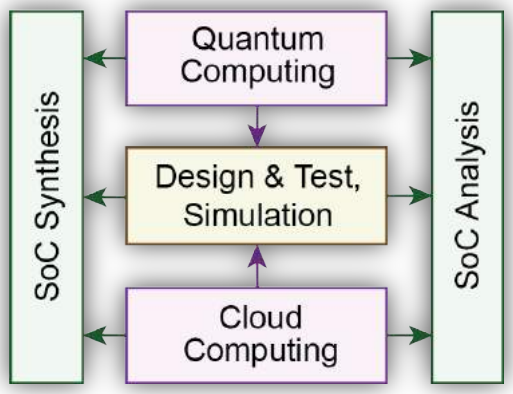


Рис. 1.14 – Структурна сутність дослідження

Об'єкт дослідження – технології квантового паралельного комп'ютингу на класичних обчислювальних пристроях, що використовують memory-driven кубітні структури даних без застосування логічних елементів.

Предмет дослідження – моделі, методи, алгоритми та засоби квантового синтезу та аналізу цифрових пристроїв і компонентів на основі використання кубітних memory-driven структур даних без застосування логічних елементів.

Мета дослідження – розробка квантових методів паралельного синтезу та аналізу цифрових пристроїв і компонентів для істотного підвищення швидкодії програмних хмарних сервісів і зменшення часу проектування програмно-апаратних комп’ютингових систем за рахунок збільшення пам'яті для зберігання кубітних структур даних.

Задачі дослідження:

1) Визначити модель метричної взаємодії класичного та квантового комп'ютингу за параметрами паралелізму, суперпозиційності та переплутування для реалізації квантового комп'ютингу в класичному виконанні за рахунок збільшення пам'яті.

2) Удосконалити метод невизначених коефіцієнтів для мінімізації булевих функцій шляхом паралельного виконання логічних операцій в цілях отримання двох векторів, відповідних мінімальній диз'юнктивній і кон'юнктивній нормальним формам.

3) Удосконалити кубітний метод пошуку дефектів для паралельного виконання операцій над двома групами векторів таблиці несправностей цифрового пристрою, отриманих після виконання діагностичного експерименту.

4) Реалізувати подальший розвиток квантового методу синтезу тестів для логічних функціональностей на основі використання булевих похідних за кубітними структурами даних для підвищення швидкодії за рахунок паралельного виконання логічних операцій.

5) Реалізувати подальший розвиток квантового методу моделювання справної поведінки на основі memory-driven кубітних структур даних шляхом використання транзакційних адресно-орієнтованих процедур аналізу цифрових пристроїв, що виключають логічні операції.

6) Розробити хмарні сервіси синтезу та аналізу кубітних моделей цифрових пристроїв і компонентів, верифікувати їх на різних прикладах цифрових схем і впровадити їх в навчальний процес. Середовище проектування: SWIFT, C++, Verilog, Jawa Script, Google Cloud Platform.

Методи дослідження – квантовий комп'ютинг, прикладна теорія цифрових автоматів, архітектури комп'ютерів, булева алгебра, теорія множин, теорія графів, квантово-кубітні методи обчислень і структури даних – для побудови моделей цифрових пристроїв; векторно-логічний аналіз, теорія алгоритмів, методи, засоби, мови проектування і моделювання цифрових систем – для синтезу та аналізу; методи і критерії якості створення обчислювальних проектів – для оцінювання тестопридатності цифрових виробів; засоби синтезу схем і аналізу кубітних покриттів – для тестування програмно-апаратних компонентів інфраструктури хмарних сервісів.

Функція мети визначається мінімізацією виразу, пов'язаного з проектними варіантами (i=1,n), що містять: зменшення часу проектування T цифрових систем на кристалах (time-to-market) за рахунок розробки D зручного інтерфейсу введення логічної структури і квантового паралельного моделювання схеми за рахунок збільшення пам'яті M для зберігання кубітних даних, а також істотним зменшенням фінансових витрат F завдяки хмарній реалізації засобів проектування, тестування і моделювання, яка виключає дорогу апаратну інфраструктуру A:

$$Q = \min_i \frac{T_i + F_i}{D_i + M_i + A_i}.$$

При цьому мова йде про проектування, моделювання, тестування і верифікацію цифрових схем і компонентів невеликої розмірності, що має на меті доведення переваг квантових технологій синтезу і аналізу шляхом надання відповідних хмарних сервісів для ергономічного навчання студентів і фахівців.

РОЗДІЛ 2

КВАНТОВІ МОДЕЛІ І МЕТОДИ СИНТЕЗУ ТА АНАЛІЗУ

Пропонуються логічні структури кібер-фізичного комп'ютингу, які розглядаються як компоненти хмарних технологій для точного моніторингу та метричного керування об'єктами. Дається аналітичний огляд кіберфізичних технологій, задекларованих у Gartner's Hype Cycle 2017, а також деякі роз'яснення, пов'язані з їх застосуванням в науці, освіті, транспорті, промисловості та державних структурах. Показано окремі напрямки, які не ввійшли в цикл ринково привабливих технологій і стосуються кібер-соціального моніторингу та керування суспільством. Пропонується розширений опис технологій, пов'язаних з розумним цифровим світом, зеленими містами і 5G-телекомунікаціями. Запропоновано рекомендації до використання топ 10 компонентів супер-циклу 2017 у бізнесі і науково-освітньому процесі університетів. Представлено memory-driven інноваційну архітектуру квантового комп'ютингу, яка характеризується використанням фотонних транзакцій запису-зчитування на структурі електронів при відсутності логіки, пов'язаної з суперпозицією і переплутуванням станів.

2.1 Вступ

Метою роботи є обґрунтування і розробка архітектури квантового комп'ютингу на основі використання кубітних структур даних для паралельного вирішення задач синтезу та аналізу цифрових пристроїв.

Задачі: 1) Аналіз сучасних кіберфізичних трендів розвитку цифрових технологій керування фізичними та соціальними процесами, представлених компанією Gartner. 2) Модернізація архітектури квантового memory-driven комп'ютингу без операцій суперпозиції і переплутування. 3) Метод трансфор-

мування табличних моделей цифрових компонентів в кубітні структури даних. 4) Метод квантової мінімізації булевих функцій шляхом унітарного кодування вхідних станів. 5) Метод квантового діагностування несправностей цифрових пристроїв.

2.2 Інновації для архітектури квантового комп'ютингу

Фізична основа класичного квантового комп'ютингу (рис. 2.1) полягає у використанні операцій суперпозиції і переплутування над станами електронів (p), яких цілком достатньо для організації обчислювального процесу [139-141]. Електрон виконує функцію пам'яті для зберігання біта інформації. Транзакції між електронами здійснюються за допомогою квантів або фотонів (q). Низька і висока орбіти електрона відповідають значенням нуля і одиниці.

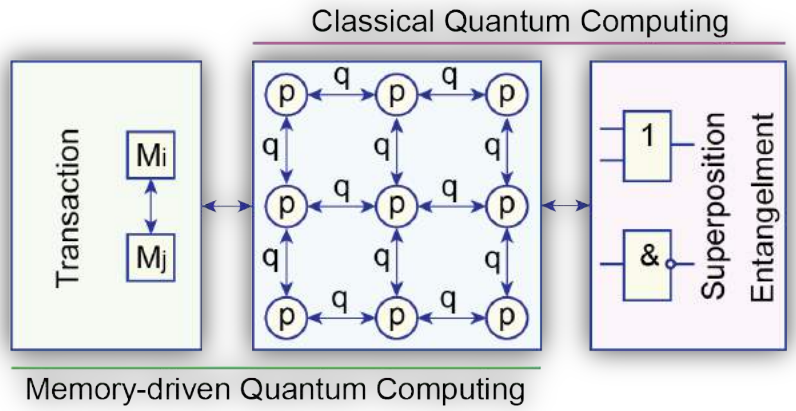


Рис. 2.1 – Два види квантового комп'ютингу

Функціонально повний базис для створення архітектури квантового комп'ютингу представлений операціями суперпозиції та переплутування, яким можна поставити у відповідність традиційний логічний базис або-ні. В теорії множин даному базису ставиться у відповідність ізоморфізм у формі пари «об'єднання-додаток». Однозначно, операція суперпозиції в квантовій фізиці ізоморфна логічній інверсії або теоретико-множинному доповненню в алгебрі логіки. Тому природно, що кожний стан двійкового розряду (електрона) після застосування даної операції до біту інформації знає один про одного все, де б

вони не знаходилися, з точністю до інверсії $a_i / \overline{a}_i$. Далі на основі згаданої пари операцій-примітивів (або, не) будується складніша система логічних компонентів і пристроїв для організації та оптимізації обчислювальних процесів. Недоліки квантового класичного комп'ютингу: 1) Висока вартість підтримки температурних умов для функціонування квантових атомарних структур на рівні -270 градусів Цельсія; 2) спостережуваність результатів обчислювальних процесів, що призводять до руйнування даних після їх читання.

Інновація в архітектурі квантового комп'ютингу визначається усуненням логіки, пов'язаної з суперпозицією і переплутуванням. Аналогом можуть виступати memory-driven архітектури класичного комп'ютера, вільні від reusable logic. В такому комп'ютері немає нічого, крім пам'яті, де реалізується транзакція (операції запису-зчитування даних) на адресовній пам'яті. Транзакції досить для організації будь-якого обчислювального процесу шляхом використання єдиного характеристичного рівняння [139,141]:

$$M_i = Q_i[M(X_i)].$$

Тут M_i, Q_i, X_i являють собою компоненти пам'яті: для вектора-стану обчислювального процесу, вектора-кубіта логічного примітиву, вектора-адреси комірки логічного примітиву. Q-логіка реалізується на адресній пам'яті, де також здійснюється асемблювання всіх примітивів за допомогою інтегруючого M-вектора стану обчислювального пристрою, який формує двійкові адреси M(X) на основі використання масиву номерів вхідних змінних X.

Інноваційна пропозиція полягає у створенні квантового memory-driven комп'ютингу без квантових операцій суперпозиції і переплутування (або, не) на основі використання наведеного вище характеристичного рівняння, що задає дві транзакції запису-зчитування на структурі електронів (див. рис. 2.1). Виключити дві складні операції з квантового комп'ютингу – означає істотно

спростити архітектуру і привести її до структури пам'яті на електронах для виконання транзакцій між ними за допомогою квантів або фотонів.

Підтвердженням спроможності запропонованої інноваційної квантової архітектури може служити кілька свіжих публікацій, які фіксують стійку тенденцію до створення квантового комп'ютингу на атомарній структурі пам'яті з передачею інформації за допомогою фотонів або квантів.

Вчені з Каліфорнійського технологічного інституту створили оптичну квантову пам'ять [142], в якій інформація передається шляхом кодування даних з використанням квантового стану фотонів. Пам'ять реалізована на рідкоземельних елементах і здатна зберігати стани фотонів за допомогою резонаторів-посередників між атомом і світлом. Розмірність квантової пам'яті в 1000 разів менше, ніж традиційні класичні рішення. Вона реалізована в нано порожнині, яка дозволяє зберігати інформацію в дуже невеликому обсязі.

Практична реалізація ідеї заміни електронів фотонами призводить до створення комп'ютингу з швидкодією, близькою до швидкості світла [143]. Корейські дослідники зробили ще один крок до оптичних обчислень. Вони створили photon-triggered нано-дротяний транзистор на основі кристалічного і пористого кремнію, де перемикання і посилення величини токового сигналу здійснюється під впливом фотона. Використання фотонів в логічних вентилях AND, OR і NAND приведе до появи ультракомпактних нанопроцесорів і нанорозмірних фотоприймачів для отримання зображень з високою роздільною здатністю.

Вчені з Колумбійського університету провели успішні досліди зі створення транзистора з одного атома в молекулярній електроніці [144]. Вони реалізували геометрично упорядкований кластер неорганічних атомів з центральним ядром, що складається з 14 атомів, яке пов'язали з золотими електродами, що дозволило керувати транзистором під впливом одного електрона при кімнатній температурі.

Вперше здійснено передачу цифрових сигналів між молекулами, що є істотним досягненням на шляху розвитку молекулярного комп'ютингу [145]. Створення електронних компонентів з окремих молекул є багатообіцяючою стратегією для мініатюризації та інтеграції електронних пристроїв. Однак практична реалізація молекулярних пристроїв і схем для передачі і обробки сигналів при кімнатній температурі виявилася складним завданням, яке вирішено шляхом розміщення молекул $SnCl_2Pc$ на поверхні міді (Cu). Площинна орієнтація молекул у міжмолекулярній взаємодії служить носієм інформації. В пов'язаних молекулярних масивах сигнал передається від однієї молекули до іншої за наперед заданими маршрутами, які реалізують логічні операції. Явища площинної орієнтації дозволяють використовувати молекули, які мають внутрішні бістабільні стани, для створення складних молекулярних пристроїв і схем.

Теоретична схожість класичного комп'ютингу з квантовим полягає в загальній моделі обчислювальної архітектури, яка використовує: пам'ять для зберігання даних і функціонально повний базис примітивних елементів (або, не) = (суперпозиція, переплутування) для реалізації арифметико-логічних операцій над даними.

Які ж формальні відмінності між класичним і квантовим комп'ютингом? Перший з них послідовно обробляє адресовні або впорядковані гетерогенні дані, витрачаючи на процедуру Q=n тактів. Він також здатний обробити гомогенні дані паралельно і за один автоматний такт. Якщо дані не впорядковані і являють собою множини, то гранична обчислювальна складність їх обробки на класичному комп'ютері залежить від потужності двох множин і визначається як $Q = n \times m.$ Наприклад, для перетину двох множин: $M_1 \cap M_2 = \{Q,E,H\} \cap \{E,H,J\} = \{E,H\}$ необхідно затратити 6 автоматних тактів. Квантовий комп'ютинг усуває даний недолік, пов'язаний з квадратичною або мультиплікативною обчислювальною складністю процедури перетину на класичному комп'ютері. Він вирішує задачу одночасної і паралельної обробки

теоретико-множинних даних. Прикладом тому може служити паралельне виконання наведеної вище операції перетину над множинами за один автоматний такт. Для цього попередньо виконується операція суперпозиції або об'єднання примітивних символів, що входять в множини $M_1 \cap M_2 = \{V\} \cap \{C\} = \{P\} = \{E,H\}$, але при цьому використовується замкнутий теоретико-множинний алфавіт [139]: B * (Y) = {Q, E, H, J, O = {Q, H}, I = {E, J}, A = {Q, E }, B = {H, J}, S = {Q, J}, P = {E, H}, C = {E, H, J}, F = {Q, H, J}, L = {Q , E, J}, V = {Q, E, H}, Y = {Q, E, H, J}, $U = \varnothing$ }. Символи алфавіту є множиною всіх підмножин на универсуме Y, які складені шляхом суперпозиції примітивів. Квантова суперпозиція дає можливість зосередити в одній точці гильбертового простору кілька дискретних станів. Аналогічно операція об'єднання також створює в одній точці дискретного простору символьний образ, що містить кілька станів. Виходячи зі сказаного, досить просто використовувати багатозначний замкнутий алфавіт для моделювання квантових обчислень на класичному комп'ютері. Але для цього необхідно попередньо створити символічну систему або алгебру множин для кодування станів. Найпростішою є алгебра Кантора, яка оперує двома дискретними станами і створює 4 символи: $A^k = \{0,1,X = \{0,1\},\varnothing\}$. Символіка цього алфавіту з точністю до ізоморфізму є теоретико-множинною інтерпретацію кубіта. Інакше, суперпозиція двох станів одного кубіта утворює 4 символи. Природно, що два кубіти здатні згенерувати 16 станів, три кубіти – 64 стани. У загальному випадку кількість станів Q має залежність від кількості кубітів n, представлену формулою: $Q = 2^{2^n}$.

Для паралельного виконання, але вже логічних операцій над кубитами інтерес представляє кодування примітивних символів алфавіту унарним двійковим кодом. Решта символів отримуються за допомогою суперпозиції кодів примітивів. Виняток становить код символу порожньої множини, який отримується шляхом застосування операції логічного множення (перетину).

Для алгебри Кантора таблиця відповідності «Symbol – Code» має такий вигляд:

$a_i \in A^k$	0	1	X	∅
$C(a_i)$	10	01	11	00

.

Платою за паралелізм виконання логічних операцій над множинами в класичному комп'ютері є істотне збільшення розрядності (регістра, пам'яті) для унарного кодування примітивних символів алфавіту. Аналогічна таблиця відповідності для кодування 16-кового алфавіту B*(Y) на чотирьох унарних кодах примітивів має вигляд:

$a_i \in B^*$	Q	E	H	J	O	I	A	B	S	P	C	F	L	V	Y	∅
$C(a_i)$	1	0	0	0	1	0	1	0	1	0	0	1	1	1	1	0
	0	1	0	0	0	1	1	0	0	1	1	0	1	1	1	0
	0	0	1	0	1	0	0	1	0	1	1	1	0	1	1	0
	0	0	0	1	0	1	0	1	1	0	1	1	1	0	1	0

.

Що стосується табличної моделі логічного елемента, він спочатку представлений квантоподібною сукупністю рівнозначних рядків або множиною дискретних відносин між вхідними та вихідними змінними. Натомість такої множини пропонується кубітний вектор вихідних станів, орієнтований на адресне паралельне моделювання цифрових логічних схем. Заміна невпорядкованої множини рівнозначних рядків таблиці істинності на упорядкований вектор адресовних станів дає можливість створювати паралельний комп'ютинг на класичних обчислювачах за рахунок збільшення обсягу пам'яті для унарного кодування кожного стану. Інакше, суперпозиція n рівнозначних елементів скінченної множини в квантовому (Q) компьютингу має взаємно-однозначну відповідність n-вимірному вектору в класичному (C) адресовному компьютингу, рис. 2.2. Даний вектор отриманий шляхом виконання or-операції над унарними кодами примітивних елементів вихідної множини. Природно, що будь-який перетин (and), об'єднання (or) або доповнення (not) в C-комп'ютері унарних кодів даних виконується паралельно, за один автоматний такт, як і в Q-комп'ютері. Платою за отриману швидкодію є збільшення

обсягу пам'яті (кількості бітів) для унарного кодування символів відносно позиційного кодування, яке визначається наступним виразом:

$$Q = n^2 / n \times \log n = n / \log n.$$

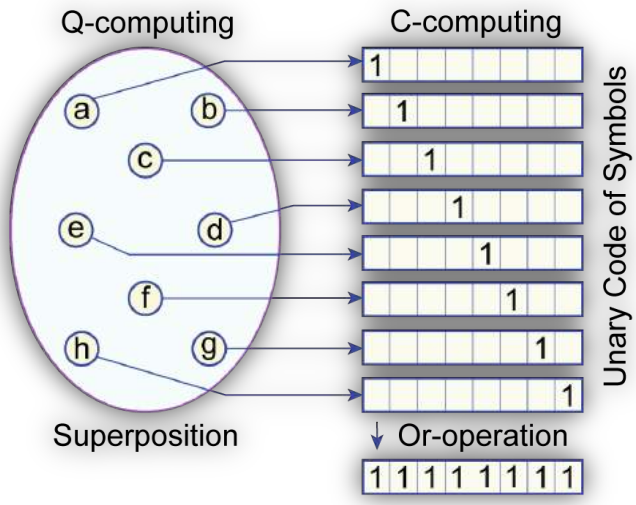


Рис. 2.2 – Суперпозиція елементів і логічне об'єднання векторів

Таким чином, розширенню потужності теоретико-множинного алфавіту можна поставити у відповідність нарощування кубітів в квантовому комп'ютері. Це дає можливість паралельно і в одному автоматному такті здійснювати обчислювальні процеси на основі використання логічних (теоретико-множинних) операцій. Суперпозиції n елементів скінченної множини в квантовому (Q) компьютингу взаємно-однозначно відповідає n-вимірний вектор в класичному (C) адресовному компьютингу, який виходить на основі використання or-операції над унарними кодами символів вихідної множини, виконуваної паралельно за один автоматний такт.

Інтерес представляє оптимальне рішення проблеми покриття шляхом використання цифрової регістрової структури (Quantum Coverage Processor – QCP), що створює всі можливі поєднання вхідних векторів у формі кубітних структур даних, представлених булеаном або множиною всіх підмножин, рис. 2.3.

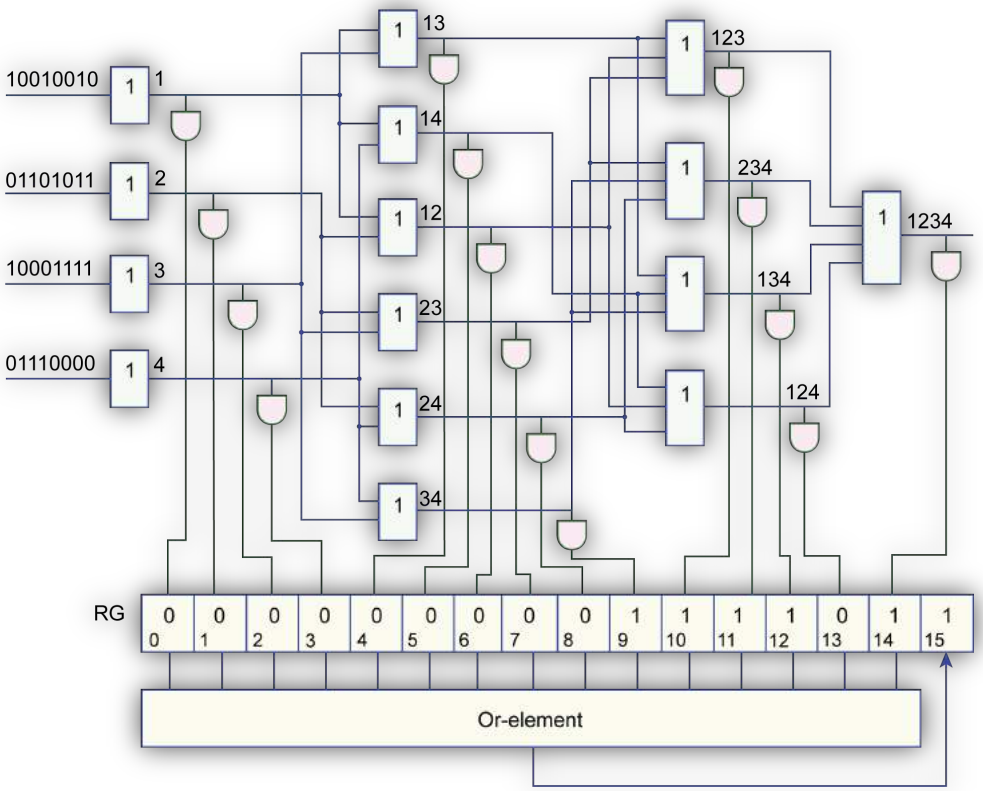


Рис. 2.3 – QC-Processor для визначення оптимального покриття

Схема має логічний аналізатор при кожному регістровому виході, який визначає повноту покриття шляхом виконання and-функції в усіх розрядах регістрової змінної. Кількість таких функцій-перетворювачів «вектор-біт» відповідає кількості елементів і дорівнює $Q = 2^n - 1$. Бітові результати обчислення and-функцій інтегруються в регістр аналізу RG, який своїми одиничними розрядами ідентифікує отримання покриття. Останній розряд регістра, що дорівнює 1, свідчить про існування позитивного результату, отриманого в процесі пошуку покриття. При цьому мінімальне покриття визначається крайньою лівою одиницею в регістрі-аналізаторі RG. Схема також призначена для визначення примітивізму або унікальності вхідних векторів, що ідентифікується нульовими значеннями в усіх розрядах регістру-аналізатора, крім останнього, рівного в цьому випадку одиниці. Апаратна складність регістрової цифрової схеми для пошуку оптимального покриття, де n – кількість рядків, m – довжина регістра, визначається аналітичним виразом: $Q = [2^{n+1} + 2^n] \times m + 2^n$.

Обчислювальна складність сукупних процедур при пошуку оптимального покриття дорівнює, в гіршому випадку, n автоматним тактам. Замість or-операцій в логічних елементах можна використовувати xor-функції, що дозволить вирішувати завдання з ідентифікації і розпізнавання кібер-об'єктів, представлених в векторній формі. Тут використовується аксіома: xor-операція двох векторів-примітивів створює їх логічне об'єднання або суперпозицію. Таким чином, n об'єктів в дискретно векторному просторі розпізнаються, якщо все xor-поєднання, крім останнього, формують нульові значення на виході and-аналізаторів.

Приклад 1. Визначити мінімальне покриття одиничними значеннями восьми розрядів наступної таблиці:

X – Inputs	0	1	2	3	4	5	6	7
1	1	0	0	1	0	0	1	0
2	0	1	1	0	1	0	1	1
3	1	0	0	0	1	1	1	1
4	0	1	1	1	0	0	0	0

Вплив на схему чотирма регістровими змінними: X = (1,2,3,4) формує наступний стан регістра аналізатора:

RG = 0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1

Одиничні значення регістра показують існування чотирьох можливих варіантів вирішення задачі: C = {3,4}, {1,2,3}, {2,3,4}, {1,3,4}, {1,2,3,4}. Мінімальне покриття забезпечується двома вхідними векторами: C = {3,4}, яке ідентифікується лівою крайньою одиницею в регістрі-аналізаторі RG.

QCP-структура може бути спрощена шляхом усунення регістра, який виконує функцію зберігання результатів пошуку оптимального покриття, рис. 2.4. В цьому випадку QCP-схема стає строго логічною, де позитивний результат пошуку визначається вже за один автоматний такт одиничним значенням стану виходу інтегрального or-елемента. Оптимальне покриття буде ідентифікуватися 1-значенням виходу and-елемента, який топологічно ближче знаходиться до зовнішніх входів схеми.

Приклад 2. Представлена структура з xor-елементів, яка вирішує завдання визначення множини векторів-примітивів, що подаються на входи цифрової схеми, рис. 2.5. Позитивний результат рішення визначається наявністю двох крайніх правих одиничних координат аналізатора RG.

Пояснення. Xor-операція для n примітивів (векторів) створює їх об'єднання. Тут мова йде про таке кодування n об'єктів в дискретному кіберпросторі, коли стає можливим *одночасне існування* або *суперпозиція всіх об'єктів в рамках форми, представленої одним вектором*. Це означає, що при правильному унарному кодуванні n об'єктів будь-яка їх суперпозиція, крім останньої (коли беруть участь всі n векторів), буде неповною, що призведе до появи нульових координат, які сформують 0-значення на виходах аналізаторів.

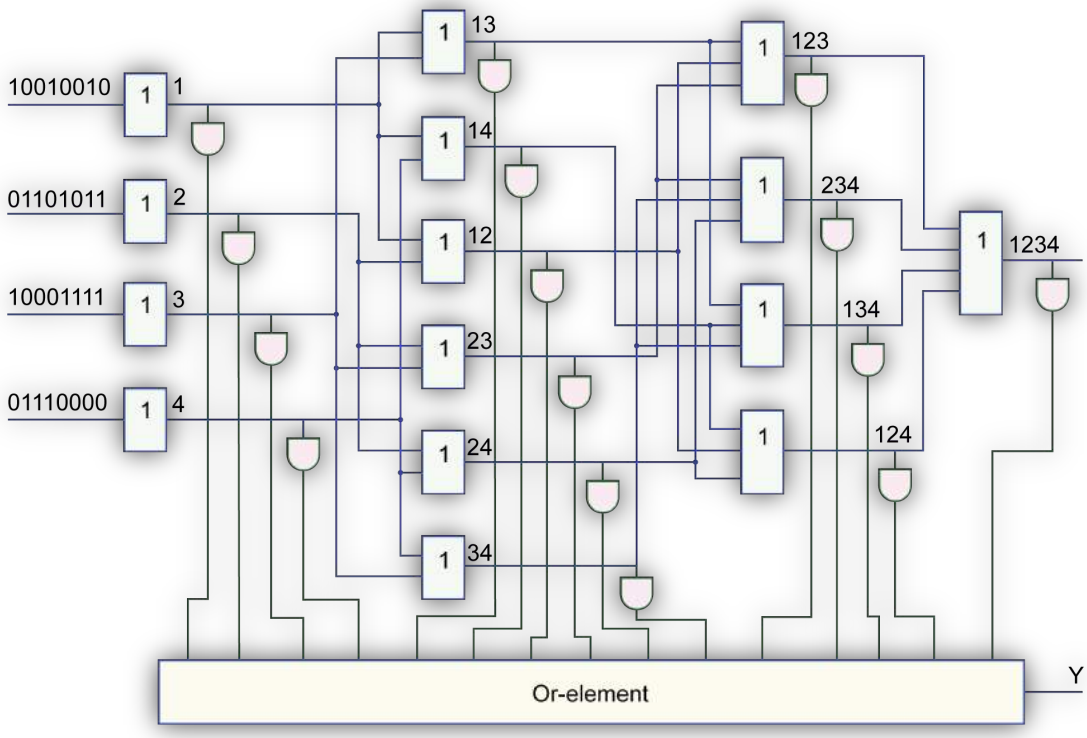


Рис. 2.4 – Комбінаційний QC-Processor для визначення покриття

Запропоновано комбінаційний QC-процесор для паралельного розв'язання задачі покриття, який характеризується одночасним обчисленням всіх можливих комбінаторних варіантів покриття за рахунок апаратної реалізації

операцій суперпозиції, що дає можливість в кілька разів підвищити швидкодію процедур при пошуку оптимального рішення.

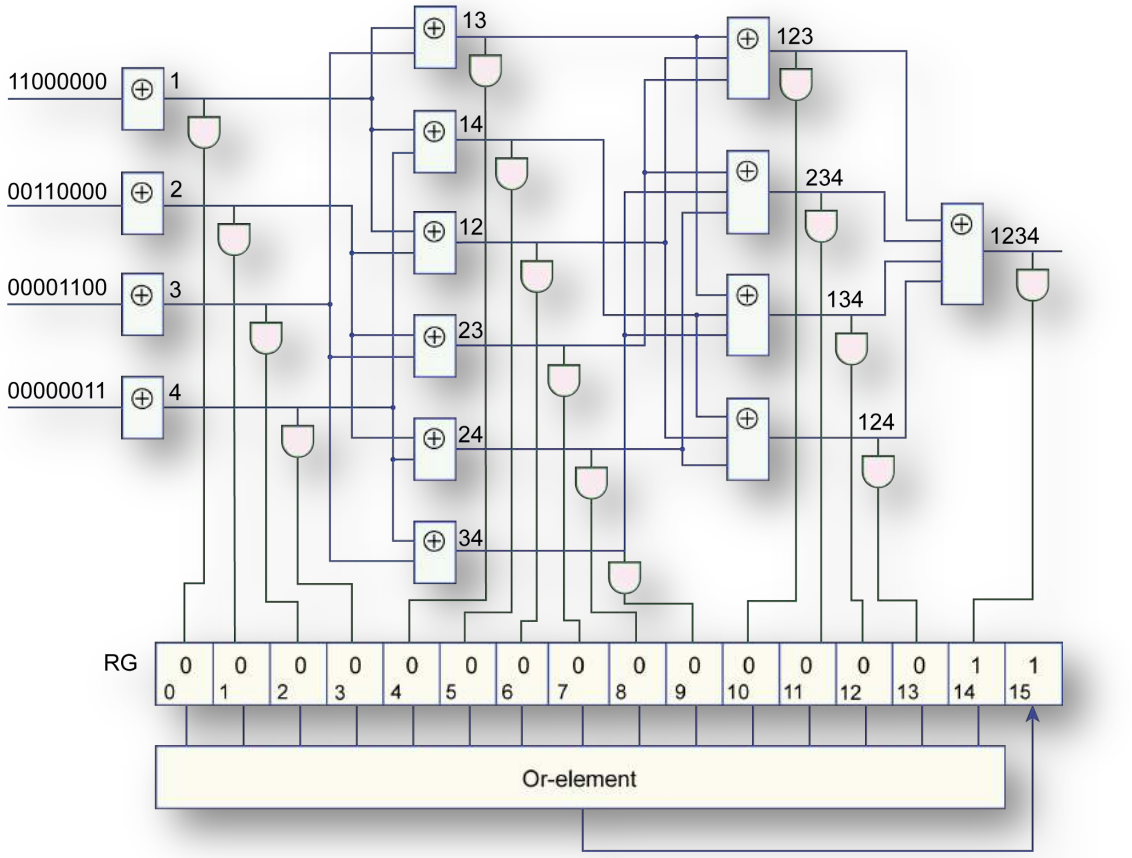


Рис. 2.5 – Схема визначення множини векторів-примітивів

2.3 Метод квантової мінімізації булевих функцій

Квантове представлення даних у вигляді суперпозиції унарних кодів можна використовувати для істотного спрощення методу невизначених коефіцієнтів при мінімізації булевих функцій.

Твердження. Будь-яка як завгодно складна таблиця істинності дискретного об'єкта може бути представлена не більше, ніж двома векторами квантового покриття при унітарному кодуванні (UC) вхідних станів. Процедура, що ілюструє дане твердження, наведена на рис. 2.6. Тут представлені нульові і одиничні куби таблиці істинності – вхідні стани, які унарно кодуються і логічно об'єднуються (VUC). В результаті виходять два вектори квантового по-

криття, кожен з яких може представляти логічну функцію в формі кубітного покриття.

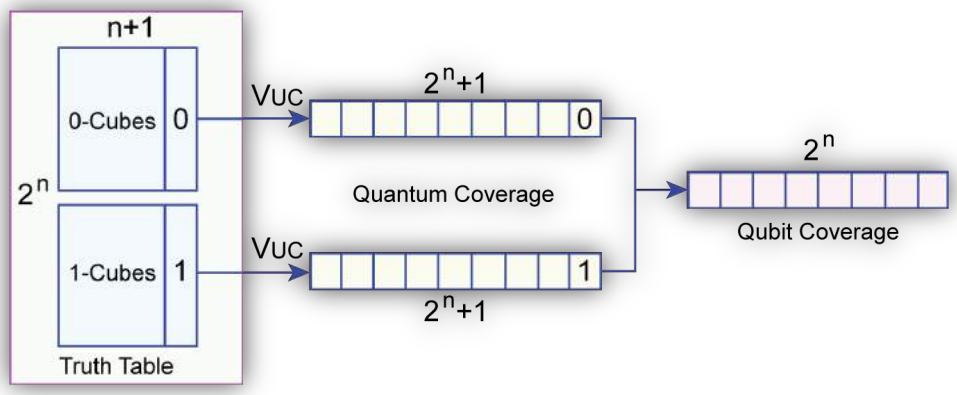


Рис. 2.6 – Модель отримання кубітного покриття

Нехай є таблиця $T = T_{ij},\ i = \overline{1,m};\ j = \overline{1,n}.$ У гіршому випадку кількість станів у кожному стовпчику таблиці (матриці) дорівнює m. Ця кількість станів може бути представлена m розрядами унітарного коду. (Наприклад, два символу алфавіту Кантора можна представити одним рядком (11), що складається з двох двійкових розрядів:

$$A^k = \{0 \to 10;\ 1 \to 01;\ X \to 11;\ \varnothing \to 00\}.)$$

В результаті виходить матриця розмірністю m на n, де кожна комірка містить m-розрядний вектор. Застосування операції суперпозиції або логічного об'єднання до всіх рядків таблиці, завдяки унітарному кодуванню, створює один рядок, який формує в компактному вигляді відносини, раніше представлені початковою таблицею. Наприклад, всі двійково-десяткові коди вхідних станів логічного елемента від трьох змінних представляються вектором (11111111), якщо враховувати наступне кодування: 000 – 10000000, 001 – 01000000, 010 – 00100000, 011 – 00010000, 100 – 00001000, 101 – 00000100, 110 – 00000010, 111 – 00000001. Однак, таблиця істинності являє собою функціональну відповідність

$$Y = f(X), X \to Y, X = \{x_1, x_2, ..., x_i, ..., x_n\}, Y = \{0,1\}.$$

З огляду на, що функція визначена на двох дискретних значеннях {0,1}, то двійкова по виходах таблиця істинності може бути завжди представлена двома рядками, кожний з яких суперпозиціонує або збирає нульові або одиничні унітарні коди вхідних впливів. Наприклад, таблиця істинності xor-елемента на три входи після логічного об'єднання унарних кодів вхідних станів за одиничним і нульовим значенням виходу буде мати такий вигляд:

a	b	c	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

→

a	b	c	U – code	Y
0	0	0	10000000	0
0	0	1	01000000	1
0	1	0	00100000	1
0	1	1	00010000	0
1	0	0	00001000	1
1	0	1	00000100	0
1	1	0	00000010	0
1	1	1	00000001	1

→

$\vee_{\forall Y=1}$
$\vee_{\forall Y=0}$

→

0	1	1	0	1	0	0	1	1
1	0	0	1	0	1	1	0	0

Таким чином, будь-яка таблиця істинності цифрового пристрою може бути представлена в явному вигляді двома рядками-кубами (назвемо його) квантового покриття, які суперпозиційно об'єднують одиничні і нульові (за станом виходу) вхідні впливи первинної таблиці. Враховуючи те, що куби квантового покриття завжди взаємно інверсні, то для завдання функціональності досить залишити один з них, вважаючи, що другий можна швидко і просто добудувати за допомогою операції інверсії, в разі необхідності.

З урахуванням отриманої інформації про можливості подання таблиці істинності двома кубами квантового покриття, далі пропонується модернізація відомого методу невизначених коефіцієнтів для мінімізації логічних функцій. Нехай є первинна таблиця невизначених коефіцієнтів для мінімізації булевої функції від трьох змінних [146]:

T_{ij}	$x_1x_2x_3$	k_1	k_2	k_3	k_{12}	k_{13}	k_{23}	k_{123}	f_i
0	000	k_1^0	k_2^0	k_3^0	k_{12}^{00}	k_{13}^{00}	k_{23}^{00}	k_{123}^{000}	1
1	001	k_1^0	k_2^0	k_3^1	k_{12}^{00}	k_{13}^{01}	k_{23}^{01}	k_{123}^{001}	0
2	010	k_1^0	k_2^1	k_3^0	k_{12}^{01}	k_{13}^{00}	k_{23}^{10}	k_{123}^{010}	1
3	011	k_1^0	k_2^1	k_3^1	k_{12}^{01}	k_{13}^{01}	k_{23}^{11}	k_{123}^{011}	0
4	100	k_1^0	k_2^0	k_3^0	k_{12}^{10}	k_{13}^{10}	k_{23}^{00}	k_{123}^{100}	1
5	101	k_1^1	k_2^0	k_3^1	k_{12}^{10}	k_{13}^{11}	k_{23}^{01}	k_{123}^{101}	0
6	110	k_1^1	k_2^1	k_3^0	k_{12}^{11}	k_{13}^{10}	k_{23}^{10}	k_{123}^{110}	0
7	111	k_1^1	k_2^1	k_3^1	k_{12}^{11}	k_{13}^{11}	k_{23}^{11}	k_{123}^{111}	1

Дана таблиця перетворюється у двійковий вид всіх можливих комбінаторних поєднань станів вхідних змінних, які здатні сформувати значення виходів функції, представленої в останньому стовпці:

T_{ij}	$x_1x_2x_3$	1	2	3	12	13	23	123	f
0	000	0	0	0	00	00	00	000	1
1	001	0	0	1	00	01	01	001	0
2	010	0	1	0	01	00	10	010	1
3	011	0	1	1	01	01	11	011	0
4	100	1	0	0	10	10	00	100	1
5	101	1	0	1	10	11	01	101	0
6	110	1	1	0	11	10	10	110	0
7	111	1	1	1	11	11	11	111	1

Природно, що отримані в комірках таблиці комбінації вхідних впливів: 0,1; 00, 01, 10, 11; 000, 001, 010, 011, 100, 101, 110, 111 тривіально трансформуються в унітарні коди двійкових станів 10, 01; 1000, 0100, 0010, 0001; 10000000, 01000000, 00100000, 00010000, 00001000, 00000100, 00000010, 00000001 відповідно:

T_{ij}	$x_1x_2x_3$	1	2	3	12	13	23	123	f
0	000	10	10	10	1000	1000	1000	10000000	1
1	001	10	10	01	1000	0100	0100	01000000	0
2	010	10	01	10	0100	1000	0010	00100000	1
3	011	10	01	01	0100	0100	0001	00010000	0
4	100	01	10	10	0010	0010	1000	00001000	1
5	101	01	10	01	0010	0001	0100	00000100	0
6	110	01	01	10	0001	0010	0010	00000010	0
7	111	01	01	01	0001	0001	0001	00000001	1

Далі виконується роздільне логічне об'єднання всіх одиничних і нульових рядків таблиці в два інтегруючих вектора. В результаті виходять всі можливі поєднання змінних, які формують одиничні і нульові значення функції.

Q	Operations	1	2	3	12	13	23	123	f
1	$Q^1 = \bigvee_{f_i=1} T_{ij}$	11	11	11	1111	1011	1011	10101001	1
2	$Q^0 = \bigvee_{f_i=0} T_{ij}$	11	11	11	1111	0111	0111	01010110	0
3	$Q = (\bigvee_{f_i=1} T_{ij}) \wedge \overline{(\bigvee_{f_i=0} T_{ij})}$	00	00	00	0000	1000	1000	10101001	Y

Щоб отримати диз'юнктивну форму мінімізованої розмірності (рядок 3 в наведеній Q-таблиці), необхідно з одиничного куба квантового покриття логічно відняти нульовий куб за правилом, представленим такою формулою:

$$Q = (\bigcup_{f_i=1} T_{ij}) \setminus (\bigcup_{f_i=0} T_{ij}) = (\bigvee_{f_i=1} T_{ij}) \wedge \overline{(\bigvee_{f_i=0} T_{ij})}.$$

Дешифрування отриманого квантового куба Q в диз'юнктивну нормальну форму дає наступний результат:

$$Y = \bar{x}_1\bar{x}_3 \vee \bar{x}_2\bar{x}_3 \vee \bar{x}_1\bar{x}_2\bar{x}_3 \vee \bar{x}_1x_2\bar{x}_3 \vee x_1\bar{x}_2\bar{x}_3 \vee x_1x_2x_3.$$

Ця форма не є мінімальною, тому потребує застосування засобів, що дозволяють вирішити задачу покриття найпростіших вихідних одиничних термів (000, 010, 100, 111) отриманими рішеннями. Для функції Y очевидно, що перші два терми (0x0, x00) покривають логічні складові 3,4,5 або (000, 010, 100), які, в даному випадку, є надлишковими відповідно до правила поглинання $a \vee ab = a$, що дає можливість записати мінімальну диз'юнктивну нормальну форму в наступному вигляді:

$$Y = \bar{x}_1\bar{x}_3 \vee \bar{x}_2\bar{x}_3 \vee x_1x_2x_3.$$

Інше рішення задачі покриття пов'язане із застосуванням QC-процесора, який має 6 регістрових входів, що дозволить визначити мінімальну ДНФ шляхом моделювання двійкових кодів-рядків $x_i \in X$ таблиці покриття:

T	000	010	100	111	$X_i \in X$
0x0	1	1	.	.	1100
x00	1	.	1	.	1010
000	1	.	.	.	1000
010	.	1	.	.	0100
100	.	.	1	.	0010
111	.	.	.	1	0001

Результат моделювання кодів визначає в якості мінімального покриття три коди-рядки, які створюють мінімальну функцію:

$$Y = 1100 \vee 1010 \vee 0001 \rightarrow 0x0 \vee x00 \vee 111 \rightarrow \bar{x}_1\bar{x}_3 \vee \bar{x}_2\bar{x}_3 \vee x_1x_2x_3.$$

Даний метод доцільно використовувати для отримання мінімальної ДНФ або КНФ за таблицями істинності, де кількість нульових і одиничних кубів-рядків не сильно відрізняється один від одного. Інше застосування методу пов'язано з істотною мінімізацією дефектної області при пошуку несправностей в цифрових системах.

Обчислювальна складність Q квантового методу невизначених коефіцієнтів знаходиться за допомогою виразу, який визначає час для унарного кодування станів таблиці істинності (для порівняння Q^b – складність базового методу мінімізації):

$$Q = 2^n \times 3^n;$$
$$Q^b = 2^n \times 2^n \times 2^n = 2^n \times 2^{n+n};$$
$$R = \frac{Q}{Q^b} = \frac{2^n \times 3^n}{2^n \times 2^{n+n}} = \frac{3^n}{2^{2n}}.$$

Таким чином, обчислювальна складність Q отримання компактного квантового покриття для мінімізації булевих функцій істотно менша в порівнянні з базовим методом Q^b невизначених коефіцієнтів, який використовує таблицю істинності спеціальної форми. Виключивши попередню

обробку таблиці істинності, яка полягає в унарному кодуванні станів, обчислювальна складність власне методу мінімізації булевих функцій визначаєься всього трьома векторними паралельними операціями.

Витрати пам'яті H для зберігання структур даних формуються розмірністю таблиці, необхідної для суперпозиційного отримання двох векторів квантового покриття, де елементи таблиці представлені унарними кодами станів:

$$H = 2^n \times 3^n;$$
$$H^b = 2^n \times 2^n = 2^{2n};$$
$$S = \frac{H}{H^b} = \frac{2^n \times 3^n}{2^n \times 2^n} = \frac{3^n}{2^n}.$$

Таким чином, щоб отримати компактне квантове покриття необхідно використовувати таблицю H істотно більшої розмірності в порівнянні з вихідною таблицею істинності H^b.

Запропоновано квантовий спосіб мінімізації булевих функцій, який відрізняється від методу невизначених коефіцієнтів паралельним виконанням операції суперпозиції над нульовими і одиничними станами вхідних змінних, представленими унітарними кодами, що дає можливість істотно підвищити швидкодію за рахунок надлишкової пам'яті.

2.4 Кубітний метод діагностування дефектів

Розглядається кубітний метод пошуку дефектів шляхом теоретико-множинної різниці двох векторів, відповідних одиничному і нульовому значенням станів виходів, як реакцій спостережуваних виходів на вхідний тест перевірки несправностей:

$$F = \left(\bigcup_{\forall R_i=1} Q_{ij}\right) \setminus \left(\bigcup_{\forall R_i=0} Q_{ij}\right) = \left(\bigvee_{\exists R_i=1} Q_{ij}\right) \wedge \overline{\left(\bigvee_{\exists R_i=0} Q_{ij}\right)}.$$

Структури даних представлені таблицею несправностей на декартовому добутку тестових наборів і множини ліній об'єкта діагностування, де кожна

комірка являє собою два біти: перший з них ідентифікує перевірювану константну несправність нуля (10), а другий – константну несправність одиниці (01):

$$
\begin{aligned}
&Q = \{F, T, L\},\\
&Q = Q_{ij}, i = \overline{1,m}; j = \overline{1,n};\\
&F = (F_1, F_2, ..., F_j, ..., F_n),\\
&F_j = \{10 \equiv 0; 01 \equiv 1; 11 = \{\equiv 0, \equiv 1\}; 00 = \varnothing\};\\
&T = (T_1, T_2, ..., T_i, ..., T_m);\\
&L = (L_1, L_2, ..., L_i, ..., L_n).
\end{aligned}
$$

Суперпозиція несправностей (дві одиниці на одній лінії-комірці) дає можливість істотно мінімізувати структури даних для зберігання інформації в цілях подальшого пошуку дефектів при виконанні діагностичного експерименту в режимі online.

Для перевірки методу пошуку дефектів далі пропонується логічна схема, представлена на рис. 2.7, яка має 6 елементів and-not, 11 ліній, 5 входів і два виходи.

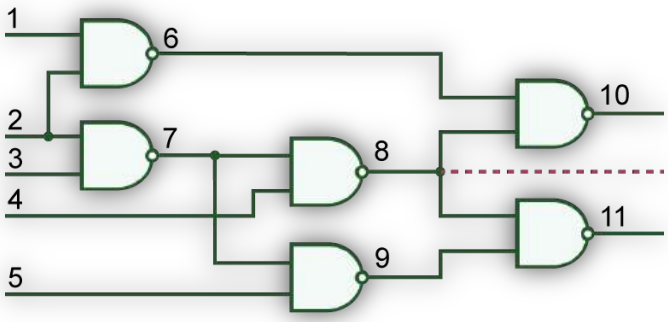


Рис. 2.7 – ISCAS-схема для верифікації

Наступна таблиця ілюструє виконання діагностичного експерименту з метою об'єднання множини дефектів, які формують некоректні стани виходів на тестових наборах {T1-R10; T5-R11; T6-R10, R11; T8-R11}:

$Q = Q_{ij}$	1	2	3	4	5	6	7	8	9	10	11	R_{10}	R_{11}
T_1	01	10	01	00	10	00	10	10	00	10	01	1	0
T_2	10	00	10	00	01	10	00	00	10	01	10	0	0
T_3	00	01	01	00	00	01	10	01	01	10	10	0	0
T_4	10	00	01	00	10	00	01	00	10	01	01	0	0
T_5	00	10	00	01	00	01	00	10	00	10	10	0	1
T_6	01	10	00	00	10	00	00	01	10	01	10	1	1
T_7	01	00	00	10	00	00	01	00	10	01	01	0	0
T_8	00	10	10	01	01	10	00	00	00	01	10	0	1
Q_1	01	11	11	01	11	11	10	11	10	11	11	1	1
Q_0	11	01	11	10	11	11	11	01	11	11	11	0	0
F	00	10	00	01	00	00	00	10	00	00	00	1/0	1/0

Тут диз'юнкція рядків T1, T5, T6, T8 формує вектор Q1, який збирає всі можливі дефекти, що перевіряються на тестових наборах. Вектор Q0 за допомогою рядків T2, T3, T4, T7 об'єднує всі неможливі, неперевірювані на тестових наборах дефекти. Віднімання всіх неможливих з усіх можливих дефектів дає шуканий результат у вигляді трьох несправностей, закодованих як F2 =10; F4=01; F8=10. Таким чином, паралельне виконання двох регістрових or-операцій на основі результатів проведеного діагностичного експерименту дозволило визначити три можливих несправності, кожна з яких може мати місце в логічній схемі:

$$F = \{2^0, 4^1, 8^0\}.$$

Більш жорсткою є обмежувальна умова існування в логічній схемі поодинокого константного дефекту, використання якого призводить до обчислення дефектів на основі такого виразу:

$$F = \left(\bigcap_{\forall R_i=1} Q_{ij}\right) \setminus \left(\bigcup_{\forall R_i=0} Q_{ij}\right) = \left(\bigwedge_{\exists R_i=1} Q_{ij}\right) \wedge \overline{\left(\bigvee_{\exists R_i=0} Q_{ij}\right)}.$$

Застосування цієї формули істотно уточнює результат діагностування і приводить його до виду: $F = \{2^0\}$ за рахунок суперечливості кодів дефектів по and-операції в стовпчиках 4 і 8. Умова наявності в логічній схемі поодинокої константної несправності ставить на чільне місце наступне твердження.

Твердження: Якщо в стовпці таблиці несправностей існує координата 00 або 01, яка створює на спостережуваних виходах некоректність R=1, пов'язану з несправністю 10 на інших координатах стовпчика, то такий поодинокий дефект (10) в логічній схемі неможливий.

Доведення. Нехай на n тестових наборах зафіксовано розбіжність на зовнішніх виходах еталонних і реальних значень сигналів. При цьому n-1 координата в розглянутому стовпці має значення 10 (01) і лише одна n-координата має значення 01 (10). Якщо припустити, що в логічній схемі є дефект 10, то на n-координаті також повинен бути присутнім дефект 10, який створює некоректний стан виходів. Але за умовами моделювання такий дефект там відсутній. Отже, неможливо вважати, що в схемі присутній дефект 10. Це підтверджується також формальним результатом – порожнім перетином всіх координат стовпчика, пов'язаних з некоректними станами виходів схеми:

$$F = (\bigwedge_{\exists R_i=1} Q_{ij}) = \begin{cases} 10 \wedge 10 \wedge \ldots \wedge 10 \wedge 01 = 00; \\ 10 \wedge 10 \wedge \ldots \wedge 10 \wedge 00 = 00. \end{cases}$$

Все сказане відноситься і до стану n-координати, який ідентифікується сигналом порожньої множини 00, взаємодія з яким також унеможливлює присутність в логічній схемі поодинокої константної несправності 10.

2.5 Висновки

1. Кібер-тенденції від Gartner Inc. надають можливість лідерам корпоративної архітектури та керівникам університетів не відставати безнадійно від цифрового бізнесу в науці, освіті та індустрії, своєчасно реагувати на кіберфізичні загрози, очолювати бізнес-інновації та визначати ефективну цифрову бізнес-стратегію сталого розвитку держав.

2. За фактом Hype-cycle є глибокою 4D-аналітикою в часі і просторі стану сучасного ринку стійкого кіберфізичного розвитку розумних хмарних технологій на найближчі 10-15 років.

3. Для університетів Hype-цикл визначає життєву необхідність інвестувати в знання студентів інноваційні технології, показані в фазах циклу, в цілях отримання через 5-10 років армії креативних фахівців, здатних підняти державу з руїн сучасного кібер-невігластва. Інакше, Gartner цикл для університету є стратегією його кіберфізичного сталого розвитку в часі і просторі. Будь-яка стратегія, розроблена без знання темпів і напрямів технологічних змін, буде страждати неправильним плануванням дій, руйнуванням бізнесу, науки та освіти. Наприклад, слід враховувати, що у 2018 році robobossи будуть точно моніторити і дистанційно online керувати 3 мільйонами працівників в світі в цілях: метрично оцінювати потенціал виконавців, роздавати завдання, логістично вірно маршрутизувати їх успішне виконання, інваріантне до позиціонування працівника в фізичному просторі, оцінювати якість і продуктивність праці, нараховувати заробітну плату за метричними результатами.

4. Hype-цикл неявно диференціює всі топ-технології на провідні та ведені (master-slave), які за фактом означають, що розвитку HardWare (Physical Space) платформ в сторону компактності завжди віддається пріоритет, оскільки решта віртуального світу (Cyber Space) прагне до безмежного розширення SoftWare додатків і завжди буде веденою [147,148]. Взаємодія двох світів, пов'язаних зі сталим розвитком обсягів апаратного і програмного забезпечення, що формує кіберфізичний простір, представлено на рис. 2.8.

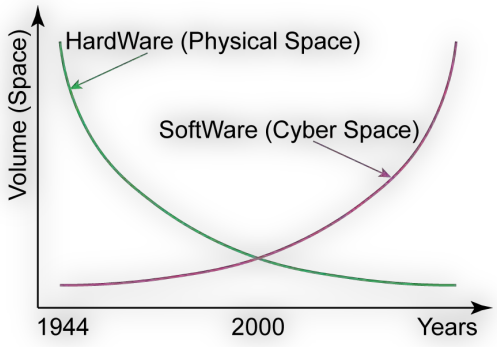


Рис. 2.8 – Взаємодія обсягів кіберфізичних компонентів

5. Проте, апаратні і програмні технології представлені в Hype-циклі (на ринку) практично в однакових пропорціях (50:50):

Hardware-driven technologies: 4D Printing, Volumetric Displays, Nanotube Electronics, Brain-Computer Interface, Human Augmentation, Autonomous Vehicles, Cognitive Computing, Commercial UAVs (Drones), Smart Dust, Smart Robots, Smart Workspace, Connected Home, 5G, IoT Platform, Edge Computing, Neuromorphic Hardware, Quantum Computing;

Software-driven technologies: Deep Learning, Deep Reinforcement Learning, Artificial General Intelligence, Enterprise Taxonomy, Ontology Management, Machine Learning, Virtual Assistants, Cognitive Expert Advisors, Digital Twin, Blockchain, Serverless PaaS, Software-Defined Security, Virtual Reality, Augmented Reality, Augmented Data Discovery, Conversational User Interfaces, Digital Humanity, Smart Cyber Digital State.

6. Однакове співвідношення апаратних і програмних технологій в Gartner-прогнозі означає, що рівні їх капіталізації на NASDAQ-ринку прагнуть до паритету, яскравим прикладом якого є компанії Apple (800 млрд доларів – індекс NASDAQ 2017) і Google (570 млрд). Ці виробники істотно відрізняються тим, що вони покладаються на мудрість своїх команд (експертів), озброєних доктриною: «споживачі не можуть передбачати свої власні потреби» (consumers could not predict their own needs) [11]. Альтернативою є політика компанії Microsoft (503 млрд), яка проводить великі дослідження перед запуском продукту, наприклад, такого як Windows Phone. За оцінками Gartner частка Apple на світовому ринку мобільних телефонів становить 14,2% проти 3,3% для Microsoft. Кому довіряти, експертам або споживачам? Відповідь однозначна – експертам, в форматі 4D (завжди, скрізь і з усіх питань).

7. Представлена memory-driven інноваційна архітектура квантового комп'ютингу, яка визначається можливістю усунення логіки, пов'язаної з суперпозицією і змішуванням станів на основі використання характеристичного рівняння, що реалізує транзакції запису-зчитування на структурі електронів.

Виключення логічних операцій з квантового комп'ютингу дозволяє істотно спростити архітектуру до рівня структури пам'яті на електронах для виконання транзакцій між ними за допомогою квантів або фотонів. Показано формальну відмінність квантового комп'ютингу від класичного, яка полягає в можливості паралельного і одночасного виконання логічних операцій над множинами.

8. Запропоновано квантовий метод мінімізіції булевих функцій, який відрізняється від методу невизначених коефіцієнтів паралельним виконанням операції суперпозиції над нульовими і одиничними станами вхідних змінних, представленими унітарними кодами, що дає можливість істотно підвищити швидкодію за рахунок надлишкової пам'яті.

9. Запропоновано квантовий метод діагностування несправностей цифрових пристроїв, який відрізняється від аналога паралельним виконанням логічних операцій над нульовими і одиничними станами вхідних змінних, представленими унітарними кодами, що дає можливість істотно підвищити швидкодію за рахунок надлишкової пам'яті.

РОЗДІЛ 3

QUANTUM MEMORY-DRIVEN COMPUTING ДЛЯ СИНТЕЗУ І ВЕРИФІКАЦІЇ ТЕСТІВ

Розглядається memory-driven комп'ютинг для проектування і тестування цифрових пристроїв. Пропонується одне з можливих рішень проблеми створення та апробування на класичних комп'ютерах теорії і методів квантового memory-driven комп'ютингу в цілях їх подальшого застосування в усіх сферах людської діяльності. Формулюються інженерно-орієнтовані визначення видів комп'ютингу, у тому числі квантового, який використовує поняття суперпозиції і переплутування, а також memory-driven комп'ютингу. Показується необхідність спільного і паралельного вирішення проблеми створення ринково доступного квантового комп'ютера і розробки квантоорієнтованих додатків і хмарних сервісів. Розглядаються приклади квантового memory-driven проектування і тестування фрагментів цифрових схем. Пропонується метод синтезу і мінімізації тестів для black-box функціональностей, що використовує матрицю кубітних похідних і секвенсор для знаходження квазіоптимального покриття.

3.1 Вступ

Існує поширене визначення квантового комп'ютера, як обчислювального пристрою, що використовує явища квантової суперпозиції і квантового переплутування для передачі та обробки даних [149-150]. В цілому воно правильне, але передачі даних у визначенні не повинно бути, оскільки вона не є функцією двох зазначених властивостей. У публікаціях відсутні чіткі пояснення для словосполучень, які зустрічаються у визначенні. Що стосується наступності квантового та класичного комп'ютингу, то тут ринкова ситуація схожа до революційної. Перший ще не може зайняти гідне місце в ніші ади-

тивних технологій, що розвиваються, а другий скоро вже не зможе підтримувати закон Мура, який має технологічну межу: роздільну здатність ліній на силіконовому кристалі – 2,5 нм [151]. Далі пропонується інженерно-орієнтована система доступних для розуміння визначень, яка пов'язує квантовий і класичний комп'ютинг, формуючи подібності та відмінності між ними шляхом розгляду структур, обчислювальних процесів і примітивних функцій.

Квантовий комп'ютер – обчислювальний пристрій, що використовує структуру елементарних частинок (електронів, атомів) для виконання квантових транзакцій.

Квантовий комп'ютинг – обчислювальний процес виконання квантових транзакцій на структурах елементарних частинок (електронів), що реалізують програмні інструкції.

Квантова транзакція – процес зчитування-запису даних на структурі елементарних частинок (електронів) шляхом випускання і прийому кванта електромагнітного випромінювання.

Квантова суперпозиція – об'єднання в просторі більше одного дискретного стану. Одне з основних правил дискретної математики свідчить, що будь-який об'єкт, компонент або система може перебувати одночасно в декількох дискретних станах. Це може бути описано алгеброю (алфавітом) Кантора, що використовує в границі n примітивних символів для одночасного подання n станів об'єкта. Більш того, діаграма Хассе, побудована на будь-якому алфавіті Кантора є не що інше, як паралельний процесор або секвенсор для одночасного одержання множини рішень, наприклад, при розгляді задачі покриття.

Квантова заплутаність – доповнення до стану елементарної частинки, що входить в універсум. Природно, що доповнення до будь-якої подмножини примітивів універсуму створює взаємозалежність двох його частин, кожна з яких має повну інформацію про своє доповнення.

Квантовий комп'ютинг (коректніше) – обчислювальний процес, який використовує квантову суперпозицію (superposition) і квантове переплутування (entangelment), які формують квантовий логічний базис sup-ent (or-not) для обробки даних.

Висновок 1. На ринку з'явилися кілька квантових комп'ютерів (IBM, d-wave, Google, NASA) на основі суперпозиції-заплутаності (or-not) елементарних частинок, що відповідає класичній теорії проектування комп'ютерів (див. рис. 2.1).

Висновок 2. Можливо, не слід створювати або шукати в природі структури логічного базису. Вони стають надлишковими з позиції memory-driven quantum computing (див. рис. 2.1). Простіше знайти структури пам'яті в будь-якій формі існування матерії для реалізації квантових транзакцій зчитування-запису. Але це вже руйнування класичної теорії проектування комп'ютерів.

Висновок 3. Memory-driven комп'ютинг стає домінуючим на ринку обробки великих даних. Архітектура FPGA Family Xilinx® UltraScaleTM Architecture Configurable Logic Block (архітектура класу повністю програмовних спеціалізованих систем на кристалі – the first ASIC-class All Programmable architecture – об'єднують програмовність процесора і конфігурованість апаратної частини) належить уже до класу ASIC і забезпечує продуктивність реалізованих смарт-функціональностей на рівні сотень гігабіт за секунду. Компанія Hewlett Packard Enterprise (HPE) створила прототип суперкомп'ютера The Machine – memory-driven computing architecture на основі фотонно-оптичних з'єднань, який на 4 порядки (в 8000 разів) перевершує по продуктивності існуючі комп'ютери [152]. Для обчислень використовується 8 терабайт пам'яті, створеної на мемристорах, що в десятки разів перевищує потужності сучасних серверів. Компанія планує за допомогою The Machine вирішити big data-driven проблеми: Transportation Systems, Smart Cities, Speech and Video Recognition, Security, Healthcare, and IoT-computing. Ідея проста і технологічна – перенести всі обчислення в практично безрозмірну пам'ять,

вилучити з архітектури комп'ютера логічний процесор, вирішивши при цьому проблему пляшкового горлечка – шинної взаємодії memory and logic processor. (Yervant Zorian, Yerevan, 2007 IEEE EWDTS). При цьому продуктивність обробки великих даних при вирішенні зазначених ринкових проблем в memory-driven computing зростає на 1-2 порядки.

Відповідно до принципу Гейзенберга [150], частинка (електрон) може знаходитися в двох точках простору одночасно, так само як і два електрони можуть одночасно перебувати в одній точці простору. Це означає, що два дискретних стани (електрона) можуть накладатися один на одний в одній точці простору, утворюючи їх суперпозицію. Математично природно і тривіально, що поділ частинки, що становить універсум станів, на дві взаємодоповнюючих субчастинки в просторі забезпечує знання стану кожної з них, утворюючи їх переплутування.

Квантовий перетин – визначення в просторі загальної частини для більш одного дискретного стану. Даній операції відповідає скалярний добуток кет-векторів в гільбертовому просторі, який дорівнює нулю, якщо вони ортогональні. Дана операція не фігурує в визначеннях квантового комп'ютингу, можливо, з причин її надлишковості. Проте, квантові операції (суперпозиції, переплутування і перетину), будучи надлишковими з позиції memory-driven computing, підвищують ефективність квантових обчислень [149, 150, 153-155]. Таким чином, для створення класичного квантового комп'ютера необхідно побудувати стійку в часі структуру (носій) елементарних частинок, на якій будуть визначені операції сигнатури: суперпозиція (or), переплутування (not), перетин (and). Транзакції на пам'яті, що реалізують всі логічні функції, руйнують один з основних принципів теорії обчислювальних процесів: не можна створити універсальний обчислювач без елемента інверсії. Юрій Петрович Шабанов-Кушнаренко (Харків) недаремно вважав інакше. Студент С. Римар, років двадцять тому, запропонував лазерний обчислювач, в якому функції і/або були реалізовані за допомогою поляризації когерентних світлових хвиль.

Метою роботи є поєднання трьох модних тенденцій в області сучасного комп'ютингу: quantum computing, memory-driven computing and cloud design computing для підвищення продуктивності вирішення завдань, пов'язаних з проектуванням, тестуванням і верифікацією цифрових систем на кристалах. Глобальна мета – створення нового квантового комп'ютера, що використовує парадигму <Memory – Address – Transaction> без властивостей суперпозиції і переплутування [156-158], що дає можливість істотного розширення умов для більш технологічної і простої реалізації обчислювача в практично будь-якій формі існування матерії (твердій, рідкій, газоподібній, плазмовій).

Мотивувувальним аргументом для публікації є наявність на ринку електроніки наступних очевидних проблем: 1) Вже існує квантовий комп'ютер, але немає комп'ютингу, який передбачає створення нових квантових технологій програмування моделей, методів і алгоритмів, орієнтованих на паралельне розв’язання традиційних і нових задач. 2) Наприклад, існує ефективне залізо від компанії Apple, для якого поки що відсутні такої ж якості програмні додатки і хмарні сервіси. 3) Існують на ринку мікроелектроніки тенденції відходу від структури <ALU – шини даних – пам'ять> в сторону реалізації регулярних обчислювальних архітектур типу memory-driven computing. Є бажання передових технологічних компаній піти від арифметики і логіки процесорів. Але є сильне лобі в середовищі провідних електронних фірм планети, яке не зацікавлене в ринковому закритті даного напрямку. 4) Існують також фахівці та вчені, які не зовсім розуміють, як можна синтезувати арифметичні і логічні пристрої тільки на елементах пам'яті. 5) Доставляти інструкції обробки великих даних в пам'ять набагато ефективніше, ніж перекачувати інформацію з пам'яті в ALU і назад по пляшковому горлечку шини даних, що вбиває продуктивність обчислювального пристрою.

Тому сьогодні буде актуальним вирішення проблеми створення та апробування на класичних комп'ютерах високопродуктивної теорії і методів квантового комп'ютингу в цілях їх подальшого застосування в усіх сферах людсь-

кої діяльності. Інакше, в даний час вченим необхідно спільно і паралельно вирішувати проблеми створення ринково доступного квантового комп'ютера і розробки кванто-орієнтованих теорій і додатків (хмарних сервісів).

Далі наведено визначення, що стосуються типів комп'ютерів і комп'ютингу, в залежності від області застосування і базису виконання архітектури:

1) Комп'ютер (classic) – радіоелектронна цифрова система, що об'єднує процесор, пямять і периферію для програмного керування виконанням арифметико-логічних інструкцій.

2) Комп'ютер (new) – кіберфізична система інтелектуального керування архітектурою і високопродуктивного виконання програмних інструкцій.

3) Комп'ютинг (системний) – інтелектуальне актюаторне керування кіберфізичною архітектурою на основі сенсорного моніторингу даних для високопродуктивного виконання програмних інструкцій.

4) Комп'ютинг (фізичний) – процес виконання адресних транзакцій в структурі пам'яті.

5) Комп'ютинг (кібер) – процес виконання адресних транзакцій в структурі даних.

6) Комп'ютинг (космологічний) – процес виконання квантових транзакцій в просторово-часовій структурі матеріально-енергетичної сутності в цілях виконання програми гармонійного розвитку Всесвіту.

3.2 Проектування суматора і синтез тесту без логіки

Мета – показати технологічність і ефективність використання квантових кубітних структур даних для синтезу і мінімізації тестів на основі кубітних булевих похідних. Задачі: 1) Синтез кубітних структур даних для memory-driven опису логічних схем на прикладі однорозрядного суматора. 2) Синтез тестів на основі використання кубітних похідних вхідних змінних логічної

схеми. 3) Мінімізація отриманих тестів шляхом вирішення задачі покриття при використанні отриманої матриці кубітних похідних.

Одноразрядний суматор являє собою примітивний логічний блок для отримання суми і перенесення при додаванні двійкових чисел. Таблиці істинності, кубітні покриття і логічні функції двох окремих модулів суматора, що формують суму й перенесення, мають такий вигляд:

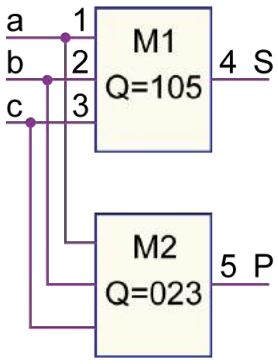
a	b	c	S	P
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

→

S	P
0	0
1	0
1	0
0	1
1	0
0	1
0	1
1	1

→ $\begin{cases} S = \bar{a}\bar{b}c \vee \bar{a}b\bar{c} \vee a\bar{b}\bar{c} \vee abc; \\ P = \bar{a}bc \vee a\bar{b}c \vee ab\bar{c} \vee abc. \end{cases}$

Структури даних (вектор моделювання M, списки вхідних змінних X і кубітні вектори-покриття Q) для аналізу сумматора мають вигляд, наведений на рис. 3.1.



N	1	2	3	4	5
M	1	0	1	0	1
X	.	.	.	123	123
Q	.	.	.	0	0
	.	.	.	1	0
	.	.	.	1	0
	.	.	.	0	1
	.	.	.	1	0
	.	.	.	0	1
	.	.	.	0	1
	.	.	.	1	1

Рис. 3.1 – Структура і дані для logic-free реалізації суматора

Всі дані розміщуються в структурах пам'яті, які за рахунок адресовних компонентів і комірок, можуть бути модифіковані в режимі online, а також відремонтовані, у разі виникнення відмов. Таким чином, використання адресовних елементів пам'яті при реалізації цифрових схем дає можливість позбутися логічних елементів, а також складних проблем тестування, верифікації та ремонту цифрових систем.

В основі синтезу та аналізу тестів для цифрових функціональностей лежить взяття булевої похідної за кубітним вектором, яке зводиться до одночасного виконання всього лише двох регістрових операцій (xor – write), показаних на рис. 3.2.

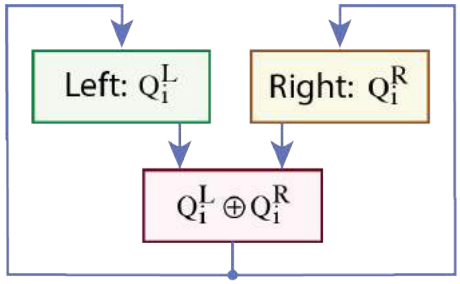


Рис. 3.2 – Отримання кубітної булевої похідної

По суті, похідна є умовою для активізації заданої вхідної змінної. Вона виражається функцією, де фігурують всі змінні за винятком заданої, за якою береться похідна.

$$\frac{df(x_1,x_2,...,x_i,...,x_n)}{dx_i} = f(x_1,x_2,...,x_i=0,...,x_n) \oplus f(x_1,x_2,...,x_i=1,...,x_n).$$

Що стосується кубітної похідної, то її сутність полягає у приведенні структури взаємодії вхідних змінних до формату вектора кубітного покриття функціональності. Більш того, кубітна похідна створює певну симетрію прихованої або неявної xor-взаємодії вхідних змінних шляхом виконання операції взяття похідної на двох частинах Q-покриття:

$$Q'(X) = \{Q_i^L, Q_i^R\} = Q_i^L \oplus Q_i^R,\ i = \overline{1,2^{k-1}},\ k = \overline{1,n}.$$

Таким чином, n кубітних похідних створюють умови активізації всіх n змінних, які записуються вже в формі матриці або сукупності відповідних векторів. Інакше, кубітна похідна одиничними значеннями своїх координат являє собою приховану форму тесту для перевірки всіх логічних шляхів від розглянутої змінної до виходу функціональності. З урахуванням викладеного, синтез

тесту, в даному випадку для суматора, можна виконати тільки на основі використання кубітного вектора. Наведена вище формула взяття булевих похідних за всіма змінними функціональності [149] дає можливість отримати тест шляхом регістрового об'єднання отриманих кубітних похідних:

$$T(Q') = \bigvee_{i=1}^{n} Q'(X_i)$$

Використовуючи дану формулу, досить просто синтезувати тест в форматі кубітного вектора, який в даному випадку дорівнює восьми двійковим наборам:

$Q'(X) \setminus Q$	0	1	1	0	1	0	0	1
$Q'(X_1)$	1	1	1	1	1	1	1	1
$Q'(X_2)$	1	1	1	1	1	1	1	1
$Q'(X_3)$	1	1	1	1	1	1	1	1
$T(Q') = \bigvee_{1=1}^{n} Q'(X_i)$	1	1	1	1	1	1	1	1

Одиничні координати отриманої матриці кубітних похідних означають, що будь-який вхідний набір за всіма координатами активізує (перевіряє) поодинокі константні несправності, інверсні відносно справного стану змінних. Дана властивість суматора є позитивною для діагностування, але її можна розглядати як негативну за метрикою синтезу мінімального тесту. Мінімізація тесту може бути реалізована на основі моделювання несправностей, що дає можливість побудувати таблицю покриття несправностей і вибрати мінімальну кількість тест-векторів, що покривають все поодинокі дефекти. Для цього необхідно побудувати дедуктивні формули аналізу поодиноких константних несправностей [157]. Інший підхід базується на аналізі матриці векторів булевих кубітних похідних, для якої необхідно знайти мінімальну кількість одиниць (стовпців), що покривають одиничними значеннями всі рядки або похідні вхідних змінних. Для останньої матриці такі 1-стовпці, які

складають мінімальний тест і активізують всі поодинокі константні дефекти на вхідних і вихідних змінних, представлені двома рішеннями:

Q'(X)\Q	0 1 1 0 1 0 0 1
$Q'(X_1)$	1 1
$Q'(X_2)$	1 1
$Q'(X_3)$	1 1
$T(Q') \vee_{i=1}^{n} Q'(X_i)$	1 0 0 0 0 0 0 1

Q'(X)\Q	0 1 1 0 1 0 0 1
$Q'(X_1)$	. . . 1 1 . . .
$Q'(X_2)$	. . . 1 1 . . .
$Q'(X_3)$	. . . 1 1 . . .
$T(Q') \vee_{i=1}^{n} Q'(X_i)$	0 0 0 1 1 0 0 0

Формально, вхідні набори {000-00, 111-11} або {011-01, 100-10} перевіряють всі поодинокі константні несправності для функціональності суми S. Якщо аналогічним чином побудувати тест для кубітного покриття функції перенесення P, то виявиться, що пара вхідних наборів {000-00, 111-11} не є перевіряльним тестом для другої функціональності S однорозрядного сумато-ра:

Q'(X)\Q	0 0 0 1 0 1 1 1
$Q'(X_1)$	0 0 1 1 1 1 0 0
$Q'(X_2)$	0 1 0 1 1 0 1 0
$Q'(X_3)$	0 1 1 0 0 1 1 0
$T(Q') = \vee_{1=1}^{n} Q'(X_i)$	0 1 1 1 1 1 1 0

Це пов'язано з тим, що не кожне змінення однієї вхідної координати (0-1, 1-0) на тестових наборах 000-00 і 111-11 приведе до зміни стану виходу P.

У загальному випадку метод синтезу мінімального тесту за матрицею кубітних похідних для будь-якої black-box функціональності пов'язаний з вирішенням задачі покриття:

1) Поділ множини стовпців Q-матриці похідних на нульові і одиничні відповідно до значень координат вихідного Q-покриття функціональності.

2) Визначення мінімальної кількості 0-стовпців, що покривають всі рядки одиничними значеннями своїх координат.

3) Визначення мінімальної кількості 1-стовпців, що покривають всі рядки одиничними значеннями своїх координат.

4) Об'єднання отриманих мінімальних покриттів для підмножини 0- і 1-стовпців дозволяє отримати мінімальний тест перевірки поодиноких констрантних несправностей для будь-якої black-box функціональності.

Виконання пунктів описаного вище методу синтезу мінімального тесту для black-box функціональностей зводиться до виконання процедур, представлених у формулі:

$$T = \min[T(Q^0) \vee T(Q^1)] = \min[(\bigvee_{\forall i(Q_i=0)}^{i=\overline{1,n}} Q_{ij}) \vee (\bigvee_{\forall i(Q_i=1)}^{i=\overline{1,n}} Q_{ij})].$$

Використання даної формули при синтезі тестів для функціональності перенесення P однорозрядного суматора на основі отриманої матриці кубітних похідних дає такий результат:

Q'(X) \ Q	0	0	0	1	0	1	1	1
$Q'(X_1)$	.	.	1	1	1	1	.	.
$Q'(X_2)$	.	.	0	1	1	0	.	.
$Q'(X_3)$	.	.	1	0	0	1	.	.
$\min[T(Q^0) \vee T(Q^1)]$	0	0	1	1	1	1	0	0

Спільна мінімізація тесту для двох функціональних елементів однорозрядного суматора формує оптимальний результат, який налічує 4 тестових послідовності:

Q'(S) \ Q(S)	0	1	1	0	1	0	0	1
$Q'(X_1)$	.	.	.	1	1	.	.	.
$Q'(X_2)$	.	.	.	1	1	.	.	.
$Q'(X_3)$	.	.	.	1	1	.	.	.
T(S)	0	0	0	1	1	0	0	0

$\vee$

Q'(P) \ Q(P)	0	0	0	1	0	1	1	1
$Q'(X_1)$	.	.	1	1	1	1	.	.
$Q'(X_2)$	.	.	0	1	1	0	.	.
$Q'(X_3)$	.	.	1	0	0	1	.	.
T(P)	0	0	1	1	1	1	0	0

$=$

$= T(S) \vee T(P) = (00111100).$

Таким чином, метод синтезу мінімального тесту на основі взяття кубітних похідних відрізняється від відомих тим, що тут не розглядаються вхідні та

вихідні змінні функціональності, а всі операції використовують тільки кубітні вектори та їх похідні. Обчислювальна складність взяття кубітних похідних має лінійну залежність від кількості змінних. Однак процедура мінімізації тесту, зведена до задачі покриття, має ступеневу залежність від кількості змінних, яка може бути істотно зменшена за рахунок розбиття матриці кубітних похідних і паралельного знаходження квазіоптимальних покриттів для фрагментів матриці [151].

3.3 Метод отримання квазіоптимального тесту або покриття

Метод є складовою частиною для синтезу мінімального тесту цифрових схем. Він базується на використанні регістрових паралельних операцій над апаратно-орієнтованими структурами даних, які представляють собою матрицю кубітних похідних для black-box функціональності. Архітектура секвенсора для реалізації методу представлена компонентами, зображеними на рис. 3.3.

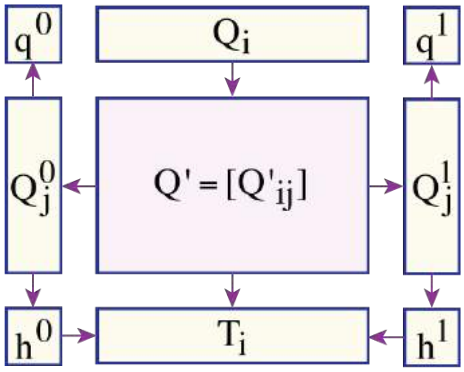


Рис. 3.3 – Архітектура секвенсора для визначення квазіоптимального тесту

Аналітична модель і обчислювальні процедури для мінімізації перевіряльного тесту на основі використання матриці кубітних покриттів мають такий вигляд:

$$A = < Q, Q', Q^0, Q^1, T, q^0, q^1, h^0, h^1, p >,$$

1) $Q = (Q_1, Q_2, ..., Q_i, ..., Q_{2^n})$;

2) $Q' = [Q'_{ij}], i = \overline{1, 2^n}, j = \overline{1, n}$;

3) $Q^0, = (Q_1^0, Q_2^0, ..., Q_j^0, ..., Q_n^0)$;

4) $Q^1, = (Q_1^1, Q_2^1, ..., Q_j^1, ..., Q_n^1)$;

5) $T = (T_1, T_2, ..., T_i, ..., T_{2^n})$;

6) $q^0 = (Q_1^0 \wedge Q_2^0 \wedge ... \wedge Q_j^0 \wedge ... \wedge Q_n^0)$;

7) $q^1 = (Q_1^1 \wedge Q_2^1 \wedge ... \wedge Q_j^1 \wedge ... \wedge Q_n^1)$;

8) $h^0 = 1 \leftrightarrow q^0 = 1$;

9) $h^1 = 1 \leftrightarrow q^1 = 1$;

10) $p = \bigvee_{j=1}^{n} [(Q_j^{0(1)} \wedge Q'_{ji}) \oplus Q'_{ji}]$.

Тут представлені: 1) кубітне покриття black box функціональності; 2) матриця кубітних похідних за всіма змінними; 3) буферний накопичувальний регістр для індикації процесу отримання квазіоптимального покриття за нульовими координатами Q-вектора; 4) буферний накопичувальний регістр для індикації процесу отримання квазіоптимального покриття за одиничними координатами Q-вектора; 5) кубітний тест-вектор, де одиничними координатами відзначені тестові набори, які необхідно подавати на Unit Under Test; 6) індикатор досягнення квазіоптимального покриття за нульовими координатами Q-вектора; 7) індикатор досягнення квазіоптимального покриття за одиничними координатами Q-вектора; 8) перемикач закінчення аналізу стовпців матриці похідних за нульовими координатами Q-вектора; 9) перемикач закінчення аналізу стовпців матриці похідних за одиничними координатами Q-вектора; 10) покажчик збільшення ступеня покривальності рядків матриці кубітних похідних для розглянутого стовпчика. Якщо покажчик дорівнює нулю (збільшення немає), то стовпець з нульової або одиничної підмножини матриці не додається до тесту:

$$p = \bigvee_{i=1}^{n} [(Q^{0(1)} \wedge Q_i^{'}) \oplus Q_i^{'}].$$

Структурна схема алгоритму синтезу квазіоптимального тесту на основі розбиття матриці похідних має дві симетричні гілки, кожна з яких орієнтована на аналіз нульової та одиничної підмножини стовпців відповідно (рис. 3.4).

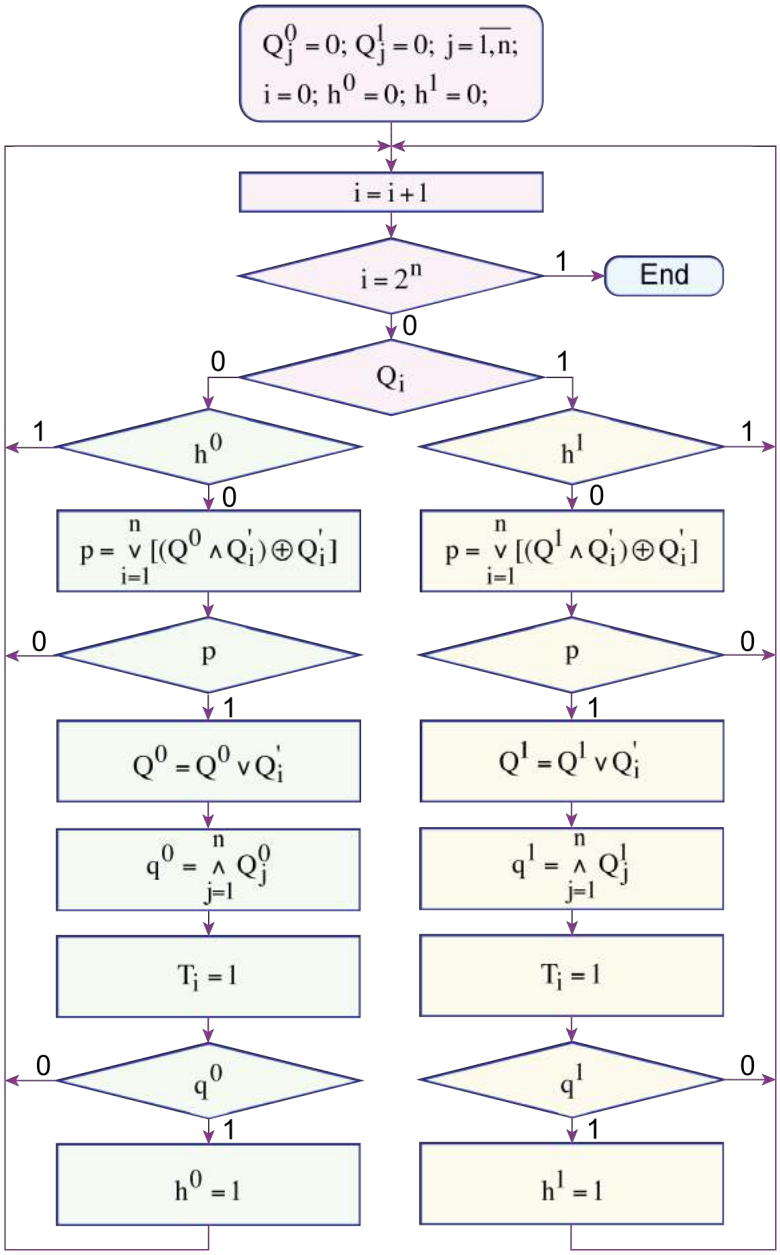


Рис. 3.4 – Алгоритм пошуку квазіоптимального тесту

Основна ідея отримання квазіоптимального тесту полягає у визначенні мінімальної кількості стовпців в нульовій та одиничній підмножині матриці

кубітних похідних, які своїми одиничними координатами покриють всі рядки або змінні розглянутої функціональності. При цьому, якщо черговий стовпець не додає перевіряльних властивостей до раніше доданих до тесту векторів, то він вилучається зі списку.

Далі пропонується використання описаних процедур методу синтезу квазімінімального тесту для логічного прикладу Шнейдера, наведеного на рис. 3.5. Кубітне покриття цифрової схеми представлено вектором (1000000000000001), за яким були отримані чотири кубітних похідних. Істотно, що для кожної змінної кубітна похідна складена парами одиничних координат у векторі. В сукупності, кожна пара при виконанні or-операції дає функціональний терм з трьох вхідних змінних, де відсутня четверта, за якою береться похідна.

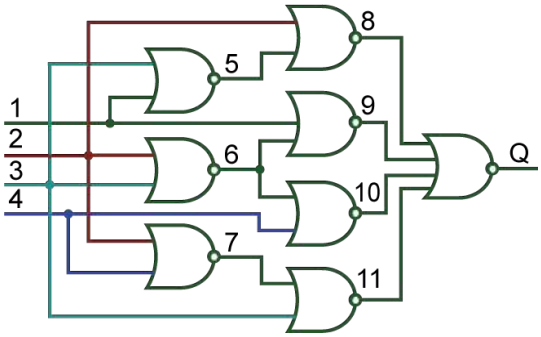


Рис. 3.5 – Логічна схема Шнейдера для синтезу тесту

Наприклад, похідна $Q'(X_1)=11$ на адресних координатах вхідних змінних 0000 і 0001 означає умови активізації 000 для першої змінної. Таким чином, кубітная похідна, як пара вхідних наборів, являє собою тест для перевірки поодиноких константних несправностей розглянутої вхідної змінної. Природно, що таких пар для кожної вхідної лінії може бути кілька. В цьому випадку їх використання пов'язане з перевіркою внутрішніх змінних функціональності. У наступній таблиці наведено кубітне покриття, булеві похідні і стани вхідних змінних, які відповідають значенням координат кубітного покриття (для візуального сприйняття загальної картини даних точками відзначено нульові стани координат таблиці істинності та матриці похідних):

X_1	. . . . . 1 1 1 1 1 1 1 1
X_2	. . . . 1 1 1 1 . . . . 1 1 1 1
X_3	. . 1 1 . . 1 1 . . 1 1 . . 1 1
X_4	. 1 . 1 . 1 . 1 . 1 . 1 . 1 . 1
Q	1 1
$Q'(X_1)$	1 . . . . . 1 1 1
$Q'(X_2)$	1 . . . 1 . . . . . 1 . . . 1
$Q'(X_3)$	1 . 1 . . . . . 1 . 1
$Q'(X_4)$	1 1 . . . . . 1 1
Test =	1 1 1 . 1 . . 1 1 . . 1 . 1 1 1
T_{min} =	1 1 . . . . . 1 1 . . . . . 1 1

В результаті виконання or-операції над векторами кубітних похідних було отримано тест, який містить 10 вхідних наборів T = (1110100110010111). Він є тестом діагностування (надлишковим перевіряльним) для поодиноких константних несправностей цифрової логічної схеми. Процедура мінімізації тесту з урахуванням структури схеми шляхом знаходження квазімінімального покриття всіх вхідних змінних нульовими та одиничними підмножинами кубітних похідних видає результат з шести тестових наборів, наведений в наступній таблиці (точками відзначено координати з неперевірюваними несправностями):

Test	1	2	3	4	5	6	7	8	9	10	11	12	FD	FC
000011100001	1	1	1	1	0	0	0	1	1	1	1	0	50	50
000111000010	.	.	.	0	.	.	1	.	.	.	0	1	16	66
011100001000	.	.	.	.	.	1	.	.	0	.	.	1	12	75
100001110000	0	.	.	.	1	.	.	0	.	.	.	1	16	87
111000000100	.	.	.	.	.	1	.	.	.	0	.	1	12	91
111100000001	0	0	0	0	.	.	.	1	1	1	1	0	37	100
∪ =	X	X	X	X	X	X	X	X	X	X	X	X		

Таким чином, мінімальний тест для логічної схеми Шнейдера містить всього шість вхідних послідовностей, які перевіряють 100 відсотків поодиноких константних несправностей для вхідних, внутрішніх і вихідних ліній пристрою.

Проведені експерименти мінімізації тестів над 16 фрагментами цифрових пристроїв (4-10 вхідних змінних) свідчать про таке: 1) у 25 відсотках випадків були отримані оптимальні тести; 2) у 70 відсотках випадків тести

відрізнялися від оптимальних не більше, ніж на 25 відсотків; 3) у 5 відсотках випадків тести були близькі до вичерпної кількості вхідних послідовностей; 4) обчислювальна складність запропонованого кубітного методу синтезу мінімального тесту для логічної функціональності від n змінних визначається оцінкою, яка формує часові витрати на взяття кубітних похідних і мінімізацію тесту:

$$E = 2n + 2n^2 = 2n(n+1).$$

Таким чином, метод синтезу тестів на основі використання кубітних похідних дозволяє згенерувати вхідні набори, які перевіряють всі поодинокі константні несправності вхідних і внутрішніх ліній. Однак для синтезу мінімального тесту необхідно використовувати структуру цифрового пристрою.

3.4 Дедуктивний аналіз схеми Шнейдера

Далі описано синтез тестів і дедуктивних формул моделювання несправностей на основі булевих похідних за вхідними змінними для схеми Шнейдера (див. рис. 3.5), спочатку описаної таблицею істинності і Q-покриттям. Мета – показати технологічність і продуктивність методів і алгоритмів, що використовують кубітні покриття функціональних примітивів.

Q	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
X_1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
X_2	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
X_3	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
X_4	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Спрощена формула визначення похідних Q' за вхідними змінними оперує xor-взаємодією (рис. 3.2) пар сусідніх компонентів кубітного покриття, яке для даного прикладу має вигляд: (1000000000000001):

$$Q'(X) = \{Q_i^L, Q_i^R\} = Q_i^L \oplus Q_i^R,\ i = \overline{1, 2^{k-1}},\ k = \overline{1, n}.$$

Потужність кожного компонента, як кількість бітів і, залежить від номера даної змінної k. Іншими словами, сусідніми парами можуть бути: біти, двійки, четвірки, вісімки бітів. Загальна кількість змінних задається номером n. Збільшення номера змінної призводить до зростання розрядності сусідніх компонентів і до відповідного зменшення їх загальної кількості, необхідної для покриття кубітного вектора.

Для отримання похідної за першою змінною схеми Шнейдера слід послідовно xor-скласти сусідні біти в кожній парі кубіт-вектора Q, а результат записати в кожен біт пари за правилом: (a,b)=a⊕b: {1,1}=1⊕0, (0,0)=0⊕0, (0,0)= 0⊕0, (0,0)=0⊕0, (0,0)=0⊕0, (0,0)= 0⊕0, (0,0)=0⊕0, (1,1)=0⊕1.

Для отримання похідної за другою змінною слід послідовно xor-скласти сусідні пари кубіт-вектора Q а результат записати в кожну пару: {a,b}=a⊕b: {10,10}=10⊕00, (00,00)=00⊕00, (00,00)=00⊕00, (01,01)=00⊕01.

Для отримання похідної за третьою змінною необхідно послідовно xor-скласти пари сусідніх тетрад кубіт-вектора Q, а результат записати в обидві тетради: {a,b}=a⊕b: {1000,1000}=1000⊕0000, (0001,0001)=0000⊕0001.

Для отримання похідної за четвертою змінною слід xor-скласти дві сусідні вісімки бітів кубіт-вектора Q, а результат записати в обидві вісімки бітів: {a,b}=a⊕b: {10000001,10000001} = 10000000⊕00000001. Наступна таблиця містить Q-покриття і 4 вектори, що являють собою похідні за чотирма змінними:

Q'(X)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
X_1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1
X_2	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1
X_3	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1
X_4	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1

Аналітична форма похідних, записана за одиничними значеннями кубітного покриття в форматі вхідних змінних має такий вигляд:

$$Q'(X_1) = \overline{X}_1\overline{X}_2\overline{X}_3\overline{X}_4 \vee X_1\overline{X}_2\overline{X}_3\overline{X}_4 \vee \overline{X}_1X_2X_3X_4 \vee X_1X_2X_3X_4 =$$
$$= \overline{X}_2\overline{X}_3\overline{X}_4 \vee X_2X_3X_4.$$

$$Q'(X_2) = \overline{X}_1\overline{X}_2\overline{X}_3\overline{X}_4 \vee \overline{X}_1X_2\overline{X}_3\overline{X}_4 \vee X_1\overline{X}_2X_3X_4 \vee X_1X_2X_3X_4 =$$
$$= \overline{X}_1\overline{X}_3\overline{X}_4 \vee X_1X_3X_4.$$

$$Q'(X_3) = \overline{X}_1\overline{X}_2\overline{X}_3\overline{X}_4 \vee \overline{X}_1\overline{X}_2X_3\overline{X}_4 \vee X_1X_2\overline{X}_3X_4 \vee X_1X_2X_3X_4 =$$
$$= \overline{X}_1\overline{X}_2\overline{X}_4 \vee X_1X_2X_4.$$

$$Q'(X_4) = \overline{X}_1\overline{X}_2\overline{X}_3\overline{X}_4 \vee \overline{X}_1\overline{X}_2\overline{X}_3X_4 \vee X_1X_2X_3\overline{X}_4 \vee X_1X_2X_3X_4 =$$
$$= \overline{X}_1\overline{X}_2\overline{X}_3 \vee X_1X_2X_3.$$

Процедура синтезу тесту для схеми Шнейдера на основі похідних описана нижче. Кожна отримана функція являє собою умови активізації змінної, за якою береться похідна. Це означає, якщо на кожний вхід подати в двох часових тактах зміни сигналів 01 або 10, то в сукупності виходить тест, який перевіряє константні несправності на всіх логічних шляхах, які задаються умовами активізації. Стосовно прикладу Шнейдера, такий тест буде складатися з 16 вхідних наборів. Мінімальний тест в форматі змінних $(X_1X_2X_3X_4)$, який не містить повторюваних вхідних наборів, має всього 10 векторів, записаних в правому стовпці:

Activation	Test	Min
$\overline{X}_2\overline{X}_3\overline{X}_4 \vee X_2X_3X_4$	0000 1000 0111 1111	0000 1000 0111 1111
$\overline{X}_1\overline{X}_3\overline{X}_4 \vee X_1X_3X_4$	0000 0100 1011 1111	0100 1011
$\overline{X}_1\overline{X}_2\overline{X}_4 \vee X_1X_2X_4$	0000 0010 1101 1111	0010 1101
$\overline{X}_1\overline{X}_2\overline{X}_3 \vee X_1X_2X_3$	0000 0001 1110 1111	0001 1110

Результат синтезу тесту, отриманого за допомогою взяття похідних в форматі Q-вектора, має вигляд:

$$T(A) = \boxed{1\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 1}.$$

Синтез Q-тестів на основі зустрічного S-зсуву компонентів кубітного покриття використовує наступну формулу:

$$T(S) = \bigvee_{j=1}^{n} [Q \overline{\oplus}_{j=1}^{n} S_j(\overline{Q})].$$

Результат використання формули, яка містить чотири логічних регістрових операції, представлений таблицею:

Q – Test Synthesis	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
$\overline{Q}$	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
$S_1(\overline{Q})$	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1
$S_2(\overline{Q})$	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1
$S_3(\overline{Q})$	1	1	1	1	0	1	1	1	1	1	1	0	1	1	1	1
$S_4(\overline{Q})$	1	1	1	1	1	1	1	0	0	1	1	1	1	1	1	1
$T_1 = Q\overline{\oplus}S_1(\overline{Q})$	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1
$T_2 = Q\overline{\oplus}S_2(\overline{Q})$	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1
$T_3 = Q\overline{\oplus}S_3(\overline{Q})$	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1
$T_4 = Q\overline{\oplus}S_4(\overline{Q})$	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1
$T(S) = T_1 \vee T_2 \vee T_3 \vee T_4$	1	1	1	0	1	0	0	1	1	0	0	1	0	1	1	1

Таким чином, обидва методи синтезу тестів (на основі зустрічного зсуву і активізації) дали однаковий результат, що містить по 10 вхідних векторів, які перевіряють всі логічні шляхи в схемі та всі поодинокі константні несправності ліній:

$$T(S) = T(A) = (1110100110010111).$$

Однак тест можна отримати більш просто, шляхом використання паралельної логічної векторної операції диз'юнкції над координатами Q-векторів похідних, за правилом:

$$T(Q') = \bigvee_{i=1}^{n} Q'(X_i).$$

Результат виконання регістрової операції диз'юнкції представлений в нижньому рядку такої таблиці похідних:

$Q'(X)$	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
$Q'(X_1)$	1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1
$Q'(X_2)$	1 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1
$Q'(X_3)$	1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1
$Q'(X_4)$	1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1
$T(Q') = \bigvee_{i=1}^{n} Q'(X_i)$	1 1 1 0 1 0 0 1 1 0 0 1 0 1 1 1

Таким чином, представлені три методи синтезу тестів на основі аналізу кубітного покриття функціональності, що мають різні обчислювальні складності, дають однаковий результат:

$$T(Q') = T(S) = T(A) = (1110100110010111).$$

Подальша мінімізація тесту на основі моделювання дає всього 6 наборів, представлених наступними двома таблицями справного моделювання та аналізу несправностей:

1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	0	1	0
0	0	1	0	1	1	1	1	1	1	0	0
0	1	0	0	1	1	1	0	1	1	1	0
1	0	0	0	1	1	1	1	0	1	1	0
1	1	1	1	0	0	0	1	1	1	1	1

1	2	3	4	5	6	7	8	9	10	11	12	FD	FC
1	1	1	1	.	.	.	0	0	0	0	0	37	37
.	.	.	0	.	0	.	.	.	1	.	1	16	54
.	.	0	.	.	.	0	.	.	.	1	1	16	66
.	0	.	.	0	.	.	1	.	.	.	1	16	79
0	.	.	.	.	0	.	.	1	.	.	1	16	87
0	0	0	0	1	1	1	0	0	0	0	0	50	100

Кубітно-дедуктивний метод аналізу несправностей описано нижче. Аналітична інтерпретація таблиці кубітних похідних Q'(X), повернена на 90 градусів для написання дедуктивних формул і подальшого моделювання несправностей, також проста для розуміння. Практично процедура формування вихідного списку (вектора) транспортовуваних поодиноких константних несправностей L(X) полягає в диз'юнкції списків транспортовуваних несправностей, записаних за одиничними значеннями координат рядка таблиці похідних,

помноженої на заперечення кон'юнкцій, записаних за нульовими координатами рядка таблиці похідних:

$$L^S(X) = \bigvee_{i=1}^{2^n} \{[\bigvee_{X_{ij}=1} L(X_{ij})] \& [\overline{\bigwedge_{X_{ij}=0} L(X_{ij})}]\} = \bigvee_{i=1}^{2^n} \{[\bigvee_{X_{ij}=1} L(X_{ij})] \& [\bigvee_{X_{ij}=0} \overline{L}(X_{ij})]\}.$$

Використовуючи цю формулу, далі синтезуються таблиці виразів для дедуктивного моделювання поодиноких несправностей з кількістю термів, що дорівнює розмірності або довжині кубітного покриття:

x_1	x_2	x_3	x_4	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	0	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

X_1	X_2	X_3	X_4	$L^S(X)$
1	1	1	1	$X_1 \vee X_2 \vee X_3 \vee X_4$
0	0	0	1	$X_4(\overline{X_1X_2X_3})$
0	0	1	0	$X_3(\overline{X_1X_2X_4})$
0	0	0	0	$X_1X_2X_3X_4$
0	1	0	0	$X_2(\overline{X_1X_3X_4})$
0	0	0	0	$X_1X_2X_3X_4$
0	0	0	0	$X_1X_2X_3X_4$
1	0	0	0	$X_1(\overline{X_2X_3X_4})$
1	0	0	0	$X_1(\overline{X_2X_3X_4})$
0	0	0	0	$X_1X_2X_3X_4$
0	0	0	0	$X_1X_2X_3X_4$
0	1	0	0	$X_2(\overline{X_1X_3X_4})$
0	0	0	0	$X_1X_2X_3X_4$
0	0	1	0	$X_3(\overline{X_1X_2X_4})$
0	0	0	1	$X_4(\overline{X_1X_2X_3})$
1	1	1	1	$X_1 \vee X_2 \vee X_3 \vee X_4$

Тут запис термів ДНФ за одиничними значеннями стовпців-похідних (права таблиця) являє собою наближені умови транспортування списків вхідних поодиноких несправностей на вхідних наборах, що задаються відповідними адресами рядків (0000, 0001 ...). Наприклад, перший і останній рядки об'єднують всі списки вхідних несправностей. Другий рядок таблиці свідчить про транспортування на вихід функціональності всіх дефектів від першого входу за винятком всіх тих, які будуть одночасно присутніми на вхо-

дах 2,3,4. При наявності в рядку таблиці всіх нульових координат виконується логічний перетин списків поодиноких несправностей для всіх входів, які будуть транспортуватися на вихід функціональності. Точні формули дедуктивного моделювання мають недолік великої розмірності, тому, що для кожного вхідного вектора необхідно будувати не один терм, а ДНФ, яка за розміром має верхньою межею таблицю істинності.

На рис. 3.6 представлені списки-вектори вхідних дефектів, які необхідно транспортувати через примітив, схему Шнейдера, на шести тестових наборах 0000, 0001 0010, 0100, 1000, 1111. Один результат, як вектор вихідних несправностей, що перевіряються на вхідному наборі 0000 отримано завдяки використанню першого стовпчика матриці векторних похідних Q'(X) для моделювання поодиноких константних дефектів. Природно, що кожна несправність з вхідного списку вектора є інверсною по відношенню до тестового станум даної вхідної змінної.

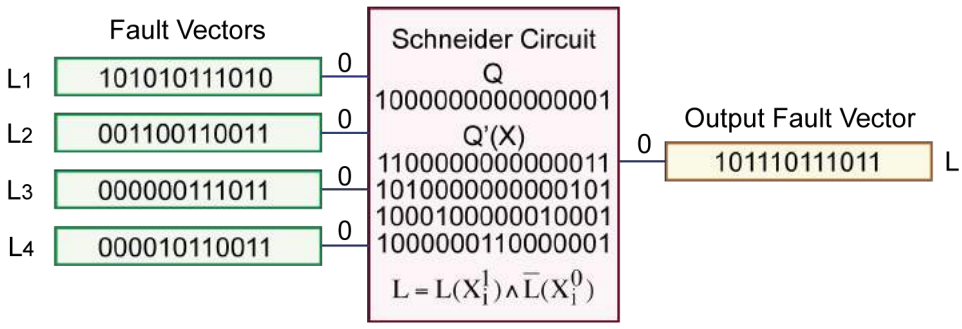


Рис. 3.6 – Транспортування векторів несправностей

Решта результатів моделювання вхідних несправностей на шести вхідних наборах (Test Column), шляхом використання стовпців матриці похідних (Derivatives) для прикладу Шнейдера, представлені нижче:

Input Faults	Test Column Derivatives							Output Faults
101010111010	0	1	0	0	0	1	1 1 0 0 0 1	111111111111
001100110011	0	0	1	0	0	1	1 0 1 0 0 1	000010000000
000000111011	0	0	0	1	0	1	1 0 0 1 0 1	000000000000
111101001100	0	0	0	0	1	1	1 0 0 0 1 1	000000000000
								010001000100
								111111111111

Зворотний бік технологічності кубітно-дедуктивного моделювання за допомогою кубітних похідних – незначне зменшення точності. Це теоретично впливає на результати моделювання. Для порівняння далі пропонуються таблиці побудови точних формул дедуктивного аналізу для семи тестових вхідних послідовностей схеми Шнейдера. Лівий стовпець являє собою таблицю істинності, внизу представлені: вхідний тестовий вектор, а також дві рівнозначні формули дедуктивного моделювання поодиноких константних несправностей. У формуванні термів дедуктивного аналізу беруть участь тільки ті рядки таблиці істинності, які задають одиничні значення функції Y. Якщо вхідний тестовий набір перетворює функціональність в 1, то всі дедуктивні терми підлягають інверсії.

X_1	X_2	X_3	X_4	Y	L_i	$L_i \oplus Y(x)$	DNF	$L^s(x,X)$
0	0	0	0	1	1111	$\overline{1111}$	$(\overline{X_1X_2X_3X_4})$	$\bar{X}_1 \vee \bar{X}_2 \vee \bar{X}_3 \vee \bar{X}_4$
0	0	0	1	0	.	.	.	.
0	0	1	0	0	.	.	.	.
0	0	1	1	0	.	.	.	.
0	1	0	0	0	.	.	.	.
0	1	0	1	0	.	.	.	.
0	1	1	0	0	.	.	.	.
0	1	1	1	0	.	.	.	.
1	0	0	0	0	.	.	.	.
1	0	0	1	0	.	.	.	.
1	0	1	0	0	.	.	.	.
1	0	1	1	0	.	.	.	.
1	1	0	0	0	.	.	.	.
1	1	0	1	0	.	.	.	.
1	1	1	0	0	.	.	.	.
1	1	1	1	1	0000	$\overline{0000}$	$\overline{\bar{X}_1\bar{X}_2\bar{X}_3\bar{X}_4}$	$X_1 \vee X_2 \vee X_3 \vee X_4$

$x = 1111;\ L_i = x(1111) \oplus X_i^1;\ Y(x) = 1;\ L_i = L_i \oplus Y(x) = \overline{L_i};$

$L = (X_1 \vee X_2 \vee X_3 \vee X_4)(\bar{X}_1 \vee \bar{X}_2 \vee \bar{X}_3 \vee \bar{X}_4);$

$L = (X_1 \vee X_2 \vee X_3 \vee X_4)(\overline{X_1X_2X_3X_4}).$

Нижче наведено скорочені таблиці, з яких вилучено рядки, які не є суттєвими для отримання результатів у вигляді аналітичних виразів дедуктивного моделювання поодиноких несправностей функціональних компонентів.

X_1	X_2	X_3	X_4	Y	L_i	$L_i \oplus Y(x)$	DNF	$L^S(x,X)$
0	0	0	0	1	0000	$\overline{0000}$	$\overline{\bar{X}_1\bar{X}_2\bar{X}_3\bar{X}_4}$	$X_1 \vee X_2 \vee X_3 \vee X_4$
0	0	0	1	0	.	.	.	.
.	.	.	.	.	.	.	.	.
1	1	1	1	1	1111	$\overline{1111}$	$\overline{X_1X_2X_3X_4}$	$\bar{X}_1 \vee \bar{X}_2 \vee \bar{X}_3 \vee \bar{X}_4$

$x = 0000$; $L_i = x(0000) \oplus X_i^1$; $Y(x) = 1$; $L_i = L_i \oplus Y(x) = \bar{L}_i$;
$L = (X_1 \vee X_2 \vee X_3 \vee X_4)(\bar{X}_1 \vee \bar{X}_2 \vee \bar{X}_3 \vee \bar{X}_4)$;
$L = (X_1 \vee X_2 \vee X_3 \vee X_4)(\overline{X_1X_2X_3X_4})$.

X_1	X_2	X_3	X_4	Y	L_i	$L_i \oplus Y(x)$	DNF	$L^S(x,X)$
0	0	0	0	1	0001	0001	$\bar{X}_1\bar{X}_2\bar{X}_3X_4$	$(\overline{X_1 \vee X_2 \vee X_3})X_4$
0	0	0	1	0	.	.	.	.
.	.	.	.	.	.	.	.	.
1	1	1	1	1	1110	1110	$X_1X_2X_3\bar{X}_4$	$(X_1X_2X_3)\bar{X}_4$

$x = 0001$; $L_i = x(0001) \oplus X_i^1$; $Y(x) = 0$; $L_i = L_i \oplus Y(x) = L_i$;
$L = \bar{X}_1\bar{X}_2\bar{X}_3X_4 \vee X_1X_2X_3\bar{X}_4$;
$L = (\overline{X_1 \vee X_2 \vee X_3})X_4 \vee (X_1X_2X_3)\bar{X}_4$.

X_1	X_2	X_3	X_4	Y	L_i	$Y(x) = 0 \to L_i$	DNF	$L^S(x,X)$
0	0	0	0	1	1100	1100	$X_1X_2\bar{X}_3\bar{X}_4$	$X_1X_2(\overline{X_3 \vee X_4})$
0	0	0	1	0	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	1	1	.	0011	0011	$\bar{X}_1\bar{X}_2X_3X_4$	$(\overline{X_1 \vee X_2})X_3X_4$

$x = 1100$; $L_i = x(1100) \oplus X_i^1$; $L_i = L_i \oplus Y(x) = L_i$;
$L = X_1X_2\bar{X}_3\bar{X}_4 \vee \bar{X}_1\bar{X}_2X_3X_4$;
$L = X_1X_2(\overline{X_3 \vee X_4}) \vee (\overline{X_1 \vee X_2})X_3X_4$.

X_1	X_2	X_3	X_4	Y	L_i	$L_i \oplus Y(x)$	DNF	$L^S(x,X)$
0	0	0	0	1	0010	0010	$\bar{X}_1\bar{X}_2X_3\bar{X}_4$	$(\overline{X_1 \vee X_2 \vee X_4})X_3$
0	0	0	1	0	.	.	.	.
.	.	.	.	.	.	.	.	.
1	1	1	1	1	1101	1101	$X_1X_2\bar{X}_3X_4$	$(X_1X_2X_4)\bar{X}_3$

$x = 0010$; $L_i = x(0010) \oplus X_i^1$; $Y(x) = 0$; $L_i = L_i \oplus Y(x) = L_i$;
$L = \bar{X}_1\bar{X}_2X_3\bar{X}_4 \vee X_1X_2\bar{X}_3X_4$;
$L = (\overline{X_1 \vee X_2 \vee X_4})X_3 \vee (X_1X_2X_4)\bar{X}_3$.

X_1	X_2	X_3	X_4	Y	L_i	$L_i \oplus Y(x)$	DNF	$L^s(x,X)$
0	0	0	0	1	0100	0100	$\bar{X}_1X_2\bar{X}_3\bar{X}_4$	$(\overline{X_1 \vee X_3 \vee X_4})X_2$
0	0	0	1	0	.	.	.	.
.	.	.	.	.	.	.	.	.
1	1	1	1	1	1011	1011	$X_1\bar{X}_2X_3X_4$	$(X_1X_3X_4)\bar{X}_2$

$x = 0100;\ L_i = x(0100) \oplus X_i^1;\ Y(x) = 0;\ L_i = L_i \oplus Y(x) = L_i;$
$L = \bar{X}_1X_2\bar{X}_3\bar{X}_4 \vee X_1\bar{X}_2X_3X_4;$
$L = (\overline{X_1 \vee X_3 \vee X_4})X_2 \vee (X_1X_3X_4)\bar{X}_2.$

X_1	X_2	X_3	X_4	Y	L_i	$L_i \oplus Y(x)$	DNF	$L^s(x,X)$
0	0	0	0	1	1000	1000	$X_1\bar{X}_2\bar{X}_3\bar{X}_4$	$(\overline{X_2 \vee X_3 \vee X_4})X_1$
0	0	0	1	0	.	.	.	.
.	.	.	.	.	.	.	.	.
1	1	1	1	1	0111	0111	$\bar{X}_1X_2X_3X_4$	$(X_2X_3X_4)\bar{X}_1$

$x = 1000;\ L_i = x(1000) \oplus X_i^1;\ Y(x) = 0;\ L_i = L_i \oplus Y(x) = L_i;$
$L = X_1\bar{X}_2\bar{X}_3\bar{X}_4 \vee \bar{X}_1X_2X_3X_4;$
$L = (\overline{X_2 \vee X_3 \vee X_4})X_1 \vee (X_2X_3X_4)\bar{X}_1.$

Використання вже точних формул дедуктивного моделювання чотирьох списків несправностей на шести вхідних наборах для схеми Шнейдера дає такий же результат, який був отриманий за допомогою аналізу дефектів на основі Q-векторів булевих похідних:

Input Faults	Deductive formulas	Output Faults
101010111010	$L = 0000 \wedge (X_1 \vee X_2 \vee X_3 \vee X_4)(\overline{X_1X_2X_3X_4})$	111111111111
001100110011	$L = 0001 \wedge (\overline{X_1 \vee X_2 \vee X_3})X_4 \vee (X_1X_2X_3)\bar{X}_4$	000010000000
000000111011	$L = 0010 \wedge (\overline{X_1 \vee X_2 \vee X_4})X_3 \vee (X_1X_2X_4)\bar{X}_3$	000000000000
111101001100	$L = 0100 \wedge (\overline{X_1 \vee X_3 \vee X_4})X_2 \vee (X_1X_3X_4)\bar{X}_2$	000000000000
	$L = 1000 \wedge (\overline{X_2 \vee X_3 \vee X_4})X_1 \vee (X_2X_3X_4)\bar{X}_1$	010001000100
	$L = 1111 \wedge (X_1 \vee X_2 \vee X_3 \vee X_4)(\overline{X_1X_2X_3X_4})$	111111111111

Таким чином, для підвищення продуктивності дедуктивного аналізу несправностей можна пожертвувати кількома відсотками точності, яка істотно не впливає на проектування цифрових систем великої розмірності. Неточність кубітно-дедуктивного методу пов'язана з виявленням несправностей, які є наслідком багатовимірної активізації дефектів на лініях східних розгалужень. Виграшем від використання практично орієнтованого методу є істотне змен-

шення витрат пам'яті для зберігання структур даних, а також технологічність та висока швидкодія реалізації алгоритму дедуктивного аналізу, заснованого тільки на використанні векторів кубітних похідних.

Кубітна форма опису цифрових систем за метрикою (компактність, швидкодія і якість), перевершує всі існуючі способи завдання обчислювальних пристроїв. Кубітне покриття функціональності є найбільш технологічним засобом для вирішення задач аналізу, синтезу, тестування і моделювання цифрових компонентів.

З огляду на те, що кубітні покриття великої розмірності, як правило, визначені не за всіма координатами, іноді має сенс створювати компактний кубіт вектор за допомогою введення надлишковості у вигляді вектора-дешифратора фактичних і модельних адрес. Наприклад, кубітне покриття з невизначеними значеннями (10xxxx0xxxxxxxx1) можна записати компактно за істотними станами розрядів, як (1010), зберігаючи первинні адреси бітів в додатковому векторі (0,1,6,15). Крім того, грунтуючись на кубітній теорії проектування і тестування, необхідно відходити від моделей несправностей для ліній reusable logic за рахунок створення нових методів моделювання вже кубітних дефектів.

3.5 Висновки

Інноваційність отриманих результатів полягає в наступному:

1) Сформульовано інженерно-орієнтовані визначення видів комп'ютингу, у тому числі квантового, на основі використання суперпозиції і переплутування, а також memory-driven комп'ютингу. 2) Запропоновано інноваційне рішення проблеми створення та апробування на класичних комп'ютерах теорії і методів квантового memory-driven комп'ютингу в цілях її подальшого застосування в усіх сферах людської діяльності. 3) Обумовлено необхідність спільного і паралельного вирішення проблеми створення ринково доступного квантового комп'ютера і розробки кванто-орієнтованих додатків і

хмарних сервісів. 4) Розглянуто приклади квантового memory-driven проектування і тестування фрагментів цифрових схем. 5) Запропоновано метод, алгоритм і структуру секвенсора для синтезу і мінімізації тестів black-box функціональностей, що використовують матрицю кубітних похідних для отримання квазіоптимального покриття. 6) Проведено експерименти на фрагментах цифрових схем, які показали практичну значущість і технологічність запропонованої архітектури і методу синтезу квазімінімального тесту для black-box логічних функціональностей. 7) Подальші дослідження будуть спрямовані на створення сімейства інтелектуальних алгоритмів синтезу тестів, моделювання і діагностування несправностей за допомогою апарату кубітних похідних.

РОЗДІЛ 4

ХМАРНИЙ СЕРВІС ТЕСТУВАННЯ І МОДЕЛЮВАННЯ ЛОГІЧНИХ СХЕМ

Запропоновано інноваційні методи взяття булевих похідних, синтезу тестів на їх основі, а також дедуктивного моделювання несправностей для функціональних елементів, заданих кубітними покриттями. Методи аналізу використовують векторні логічні операції: and, or, not, xor, а також зустрічного зсуву частин кубітної форми функціональності. Розглянуто приклади комбінаційних схем для верифікації та порівняльного аналізу продуктивності базових і запропонованих методів. Описано структуру вбудованого процесора, який виконує операції взяття похідних, синтезу тестів, дедуктивного моделювання несправностей для оцінки якості перевіряльних вхідних наборів і діагностування. Запропоновані технології орієнтовані на їх імплементацію у вигляді хмарного сервісу або IP-Infrastructure в архітектурі SoC.

4.1 Технології хмарного комп'ютингу

На ринку електроніки спостерігається стійке домінування інноваційної кіберкультури, яка базується на Нано-Адитивних, Біо-Інформаційних, Кібер-Фізичних, Соціально-Когнітивних компьютингових технологіях, що створюють умови для кіберфізичного моніторингу та керування в значущих для людини галузях, таких як: освіта, промисловість, транспорт, зелена екологія і висока якість життя. В рамках сталого розвитку кіберфізичного комп'ютингу цікавими для ринку є: Quantum Computing, Memory-Driven Computing, Cloud Computing, IoT Computing, Big Data Computing, які створюють основу для другої фази комп'ютингу, яка характеризується активним керуванням (рис. 4.1) всіма фізичними процесами і явищами.

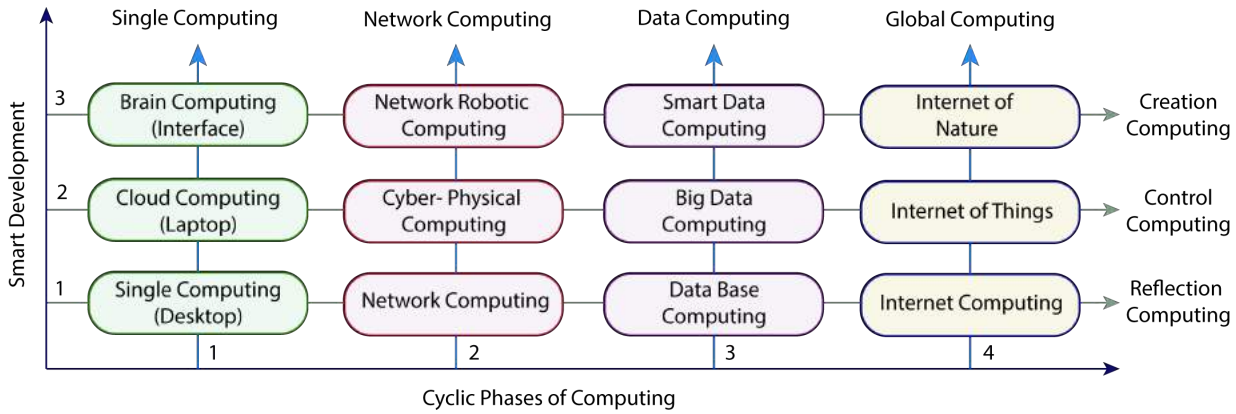


Рис. 4.1 – Комп'ютинг в часі і просторі

Таким чином, можна визначити дві парадигми комп'ютингу: віртуальний та фізичний комп’ютинг, які базуються на memory-driven технологіях і вписуються в структуру, представлену на рис. 4.2.

Далі слід також інтегрувати чотири найістотніших компонента, які отримали сталий розвиток на ринку електронних технологій і пояснюють появу практично орієнтованих трикутних взаємодій (IoT, Big Data Analytics, Cyber Physical Systems), представлених на рис. 4.3.

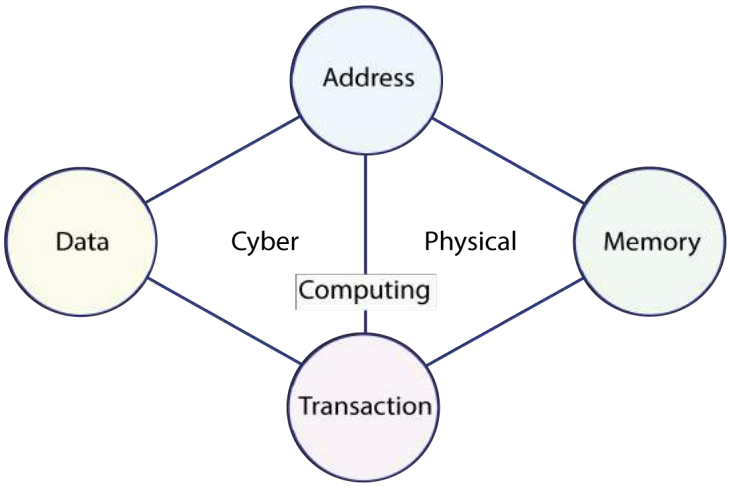


Рис. 4.2 – Дві технології комп'ютингу

З огляду на, що в найближчі 5 років з'являться квантові обчислювачі, які вимагатимуть переділу послідовної парадигми алгоритмів на паралельні архітектури, то сьогодні актуальною видається емуляція технологій паралельного синтезу та аналізу на сучасних напівпровідникових комп'ютерах [159-160].

В рамках такої мотивації далі пропонуються кубітні структури даних і методи їх синтезу-аналізу для memory-driven комп'ютингу, орієнтовані на паралельне виконання всіх операцій (рис. 4.4).

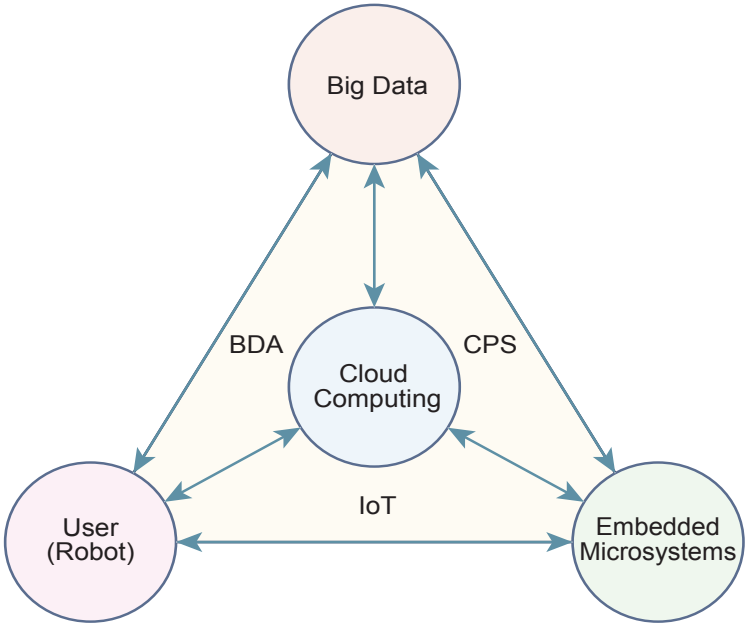


Рис. 4.3 – Стійкі комп’ютингові тріади ринку

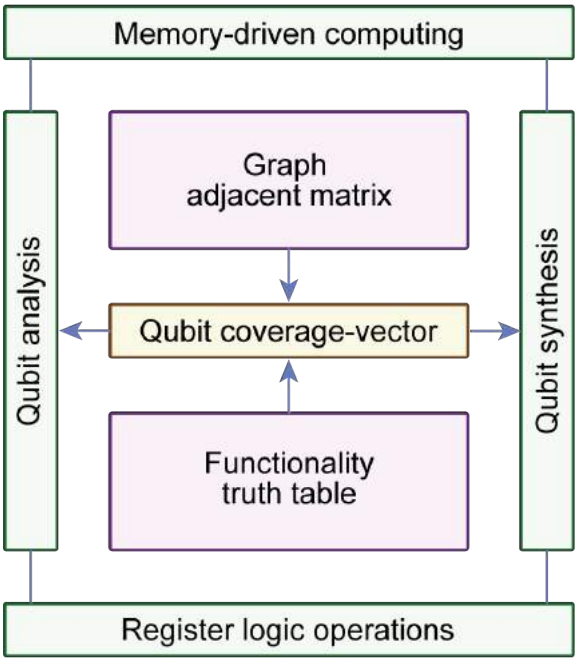


Рис. 4.4 – Структура кубітного (квантового) комп'ютингу

Представлені далі моделі і методи є логічним продовженням публікації [159], де викладено визначення і структури даних кубітного (квантового) комп'ютингу, а також метод синтезу тестів на основі аналізу кубітного покриття.

Мета роботи – підвищення продуктивності методів тестування цифрових пристроїв, пов'язаних з синтезом тестів і дедуктивним моделюванням несправностей на основі використання кубітної форми булевої похідної.

Задачі: 1) Метод взяття булевої похідної на основі кубітного або векторного представлення логічної функції. 2) Метод синтезу тестів на основі використання кубітної форми булевої похідної. 3) Метод дедуктивного моделювання несправностей на основі використання кубітної форми булевої похідної. 4) Архітектура спеціалізованого процесора вбудованого тестування на основі використання кубітної форми завдання функціональності.

4.2 Імплементація квантового методу синтезу тестів для прикладів

Пропонується модифікація методу синтезу тестів на основі взяття похідних за кубітним покриттям цифрової логічної схеми. Для верифікації результатів використовується квантовий симулятор, який дає можливість моделювати справну поведінку і несправності в цілях отримання тестів перевірки дефектів при читанні курсів з проектування і тестування цифрових систем і компонентів. Симулятор має простий і ефективний інтерфейс графічного зображення логічних елементів, портів введення тестових наборів і виведення результатів моделювання, а також сервіси для виправлення помилок і зберігання структур даних. Структура раніше розглянутої логічної схеми представлена на рис. 4.5.

Усередині елемента є інформація про порядковий номер примітиву і тип функціональності, представлена десятковим числом, яке є згорткою кубітного вектор-покриття логічного елемента. Симулятор працює в покроковому ре-

жимі, коли моделюється один вектор, який вручну подається на зовнішні входи схеми. Є також автоматичний режим подання вичерпного тесту для отримання таблиці істинності всього цифрового пристрою зі значеннями на всіх (вхідних, внутрішніх і вихідних) лініях схеми.

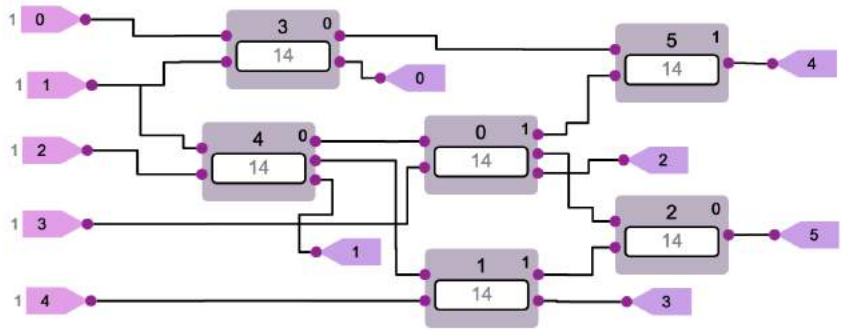


Рис. 4.5 – ISCAS-схема в квантовому симуляторі

За допомогою квантового симулятора шляхом застосування характеристичного рівняння Mi = Qi[M(Xi)] отримано таблицю істинності, представлену нижче, де інтерес представляють два останніх стовпчика (10,11) – кубітні вектор-покриття, які далі використовуються для синтезу тестів, що перевіряють поодинокі константні несправності:

$T = T_{ij}$	1	2	3	4	5	6	7	8	9	10	11
0	0	0	0	0	0	1	1	1	1	0	0
1	0	0	0	0	1	1	1	1	0	0	1
2	0	0	0	1	0	1	1	0	1	1	1
3	0	0	0	1	1	1	1	0	0	1	1
4	0	0	1	0	0	1	1	1	1	0	0
5	0	0	1	0	1	1	1	1	0	0	1
6	0	0	1	1	0	1	1	0	1	1	1
7	0	0	1	1	1	1	1	0	0	1	1
8	0	1	0	0	0	1	1	1	1	0	0
9	0	1	0	0	1	1	1	1	0	0	1
10	0	1	0	1	0	1	1	0	1	1	1
11	0	1	0	1	1	1	1	0	0	1	1
12	0	1	1	0	0	1	0	1	1	0	0
13	0	1	1	0	1	1	0	1	1	0	0
14	0	1	1	1	0	1	0	1	1	0	0
15	0	1	1	1	1	1	0	1	1	0	0

$T = T_{ij}$	1	2	3	4	5	6	7	8	9	10	11
16	1	0	0	0	0	1	1	1	1	0	0
17	1	0	0	0	1	1	1	1	0	0	1
18	1	0	0	1	0	1	1	0	1	1	1
19	1	0	0	1	1	1	1	0	0	1	1
20	1	0	1	0	0	1	1	1	1	0	0
21	1	0	1	0	1	1	1	1	0	0	1
22	1	0	1	1	0	1	1	0	1	1	1
23	1	0	1	1	1	1	1	0	0	1	1
24	1	1	0	0	0	0	1	1	1	1	0
25	1	1	0	0	1	0	1	1	0	1	1
26	1	1	0	1	0	0	1	0	1	1	1
27	1	1	0	1	1	0	1	0	0	1	1
28	1	1	1	0	0	0	0	1	1	1	0
29	1	1	1	0	1	0	0	1	1	1	0
30	1	1	1	1	0	0	0	1	1	1	0
31	1	1	1	1	1	0	0	1	1	1	0

Структурна організація кубітних даних для моделювання справної поведінки цифрової логічної схеми на тестовому наборі 26 наведена на рис. 4.6.

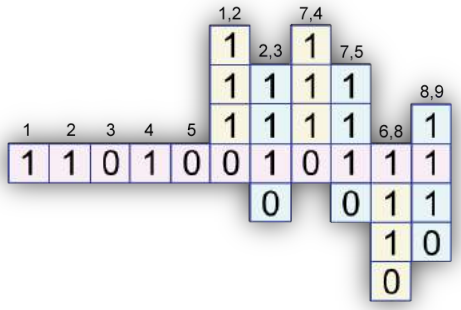


Рис. 4.6 – Структури кубітних даних для логічної схеми

Тут горизонтальний вектор M формує структуру зв'язків кубітних покриттів, а вертикальні стовпчики являють собою кубітні Q-вектори, які рухаються вгору-вниз для формування станів комірок вектора моделювання. Кубітні покриття цифрової логічної схеми, що має п'ять вхідних змінних, чотири внутрішніх і дві вихідних, представлені такими двома векторами:

Q_{10}	0011001100110000 0011001111111111
Q_{11}	0111011101110000 0111011101110000

Вони призначені для синтезу тестів на основі отримання булевих похідних шляхом зустрічного зсуву частин кубіта з подальшим обчисленням паралельної операції xor і занесенням результатів за згаданими частинами:

$$Q'(X) = \{Q_i^L, Q_i^R\} = Q_i^L \oplus Q_i^R,\ i = \overline{1,2^{k-1}},\ k = \overline{1,n}.$$

Оскільки схема має п'ять вхідних змінних, то п'ять булевих похідних формують відповідну матрицю, яка одиничними значеннями координат вказує на зміну стану виходу на парі вхідних тестових наборів:

Q_{10}	0011001100110000 0011001111111111
Q_1	0000000011001111 0000000011001111
Q_2	0000001100000011 1100110011001100
Q_3	0000000000110011 0000000000000000
Q_4	1111111111110000 1111111100000000
Q_5	0000000000000000 0000000000000000
T =	1111111111111111 1111111111001111

Q_{11}	0111011101110000 0111011101110000
Q_1	0000000000000000 0000000000000000
Q_2	0000011100000111 0000011100000111
Q_3	0000000001110111 0000000001110111
Q_4	1010101010100000 1010101010100000
Q_5	1100110011000000 1100110011000000
T =	1110111111110111 1110111111110111

При розгляді матриці кубітних похідних нескладно помітити, що пари, четвірки, вісімки сусідніх одиниць в кожному векторі мінімізуються в умови активізації вхідних змінних, де деякі змінні виявляються несуттєвими. Наприклад, кубітная матриця Q10 в рядку Q1 має 1111 на позиціях 12-15 і 28-31. Це означає, що для активізації першої змінної необхідно виконати умови:

$$\frac{df}{dx_1} = \begin{matrix} 011000 \\ 011010 \\ 011100 \\ 011110 \end{matrix} \oplus \begin{matrix} 111001 \\ 111011 \\ 111101 \\ 111111 \end{matrix} = (011xx0) \oplus (111xx1).$$

Інакше, для зміни стану виходу шляхом активізації першого входу необхідно і достатньо використовувати будь-яку з чотирьох наведених вище умов. Така закономірність дає підстави для істотної мінімізації тестових наборів шляхом структурно-логічної мінімізації матриці кубітних похідних. Наступні таблиці ілюструють процес і результат мінімізації тесту для перевірки несправностей всіх ліній логічної схеми:

Q_{10}	0011001100110000 0011001111111111
Q_1	00000000x100xxx1 00000000x100xxx1
Q_2	000000x1000000x1 x100x100x100x100
Q_3	0000000000x100x1 0000000000000000
Q_4	xxxxxxxxxxx10000 xxxxxxx100000000
Q_5	0000000000000000 0000000000000000
$T(Q_{10})$ =	0000000101010001 0100010101000101

Q_{11}	0111011101110000 0111011101110000
Q_1	0000000000000000 0000000000000000
Q_2	00000xx100000xx1 00000xx100000xx1
Q_3	000000000xx10xx1 000000000xx10xx1
Q_4	x0x0x0x010100000 x0x0x0x010100000
Q_5	xx00xx0011000000 xx00xx0011000000
$T(Q_{11})$ =	0000000111110001 0000000111110001

Об'єднання двох тестів для кожного з виходів дає наступний результат, який містить дванадцять тестових наборів і перевіряє поодинокі константні несправності вхідних, внутрішніх і вихідних ліній логічної схеми:

T_{10} = 0000000101010001 0000000101000001
T_{11} = 0000000111110001 0000000111110001
T = 0000000111110001 0000000111110001

Таким чином, метод синтезу тестів за кубітними покриттями логічної схеми має такі пункти:

1) Обчислення кубітного покриття логічної схеми за допомогою структурно-функціональної моделі.

2) Визначення кубітних похідних на основі використання кубітного покриття за формулою зустрічного зсуву його частин і виконання xor-операцій над ними:

$$Q'(X) = \{Q_i^L, Q_i^R\} = Q_i^L \oplus Q_i^R,\ i = \overline{1, 2^{k-1}},\ k = \overline{1, n}.$$

3) Мінімізація кількості одиничних координат в матриці кубітних покриттів на основі об'єднання умов для активізації вхідних логічних змінних.

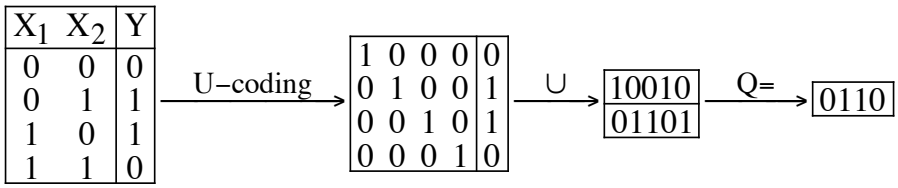
4) Верифікація отриманого тесту методом моделювання дефектів для отримання таблиці несправностей в цілях проведення діагностичного експерименту і пошуку дефектів заданого класу.

4.3 Паралельне моделювання дефектів по кубітним покриттям

Квантовий комп'ютинг орієнтований на вирішення завдань паралельної комбінаторики, де підмножини даних обробляються за один часовий такт. Класичний комп'ютинг служить для паралельної обробки регулярних адресовних даних в одному часовому такті. Поєднання двох видів комп'ютингу створює обчислювальні потужності для глобального хмарного інтелекту. Обмеження за паралелізмом квантового комп'ютингу пов'язані з кількістю комбінацій на множині з n-примітивів, що дорівнює 2**n. Практичні обмеження за паралелізмом класичного комп'ютингу пов'язані з кількістю n роз-

рядів регістрової пам'яті, на якій виконуються паралельні логічні операції. Основа квантового комп'ютингу – множини, а класичного – байти – впорядковані структури адресовних даних пам'яті.

Розглядається паралельний метод кубітного моделювання несправностей, який характеризується одночасною обробкою векторів поодиноких константних дефектів в цілях побудови таблиці несправностей. Алгоритм отримання кубітного покриття функціональності з таблиці істинності пов'язаний з процедурами унітарного кодування вхідних наборів, об'єднанням рядків з однаковими станами на виході примітиву і залишенням 1-кубіта, що має одиничне значення на виході логічного елемента:



Природно, що 0-кубіт може бути отриманий за принципом доповнення, шляхом інверсії всіх розрядів 1-вектора. Процес моделювання зводиться до визначення стану комірки 1-кубіта за її адресою.

Отже, логічний елемент, представлений кубітним покриттям, служить для визначення його реакції на вхідне слово. Нескладно підвищити швидкодію процесу моделювання шляхом одночасного і паралельного впливу на примітив кінцевою множиною вхідних наборів. Логічний елемент, який має n входів і один вихід трансформується в кубітний примітив, який має 2**n входів, станів і виходів. Це дає можливість паралельно обробляти повну множину вхідних станів логічного елемента шляхом суперпозиційного виконання регістрової операції над вхідними станами і кубітом примітиву, з подальшим записом результату в регістр виходів. Суперпозиції вхідних кубітів, функціональних і вихідних дають можливість паралельно обробляти великі обсяги структур даних. Наприклад, суперпозиція кубітів однакової розмірності X = 0111 і Q = 0110 дає результат:

$$Y = 0111 \wedge 0110 = 0110.$$

Фактично, вхідний вектор виконує роль маски для кубітного покриття, що дає можливість за один автоматний такт визначити стани виходів логічної схеми при впливі на неї не більше, ніж 2**n вхідними наборами.

Класична реалізація паралельного методу моделювання несправностей пов'язана з одночасною обробкою 2n+1 векторів, де один вектор – справне моделювання, а решта – дефекти, що підлягають перевірці на тестовому вхідному наборі. Сказане демонструється на прикладі схеми, представленої на рис. 4.7, наступною таблицею, яка формує вектор перевірюваних дефектів, позначених символами «+» на фоні вектора моделювання справної поведінки G:

L	G	1^0	2^0	3^0	4^0	5^0	1^1	2^1	3^1	4^1	5^1
1	1	0	1	1	1	1	1	1	1	1	1
2	1	1	0	1	1	1	1	1	1	1	1
3	0	0	0	0	0	0	0	0	1	0	0
4	0	1	1	0	0	0	0	0	0	1	0
5	0	1	1	0	0	0	0	0	1	1	1
D	=	+	+	−	−	−	−	−	+	+	+

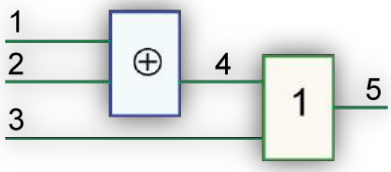


Рис. 4.7 – Структура для паралельного моделювання

На рис. 4.8 представлено структури даних на основі використання кубітних покриттів, які також орієнтовані на паралельне моделювання несправностей.

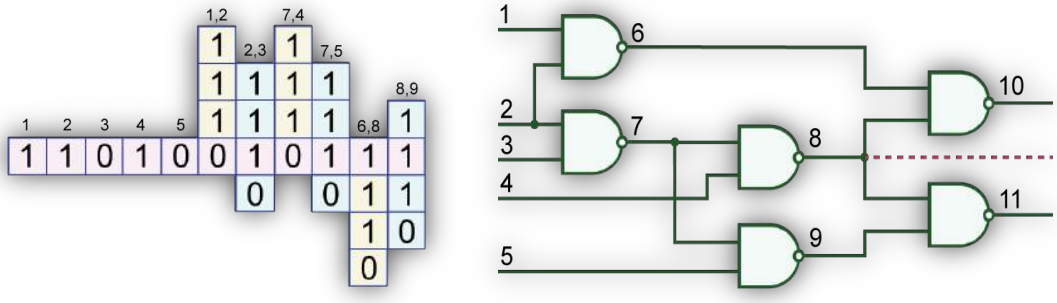


Рис. 4.8 – Структури даних для паралельного моделювання

Результат паралельного моделювання поодиноких константних несправностей на вхідному тестовому наборі (00111110011) представлений в

наступній таблиці, яка нижнім рядком ідентифікує перевірювані дефекти, відмічені символами «+»:

L	G	1^1	2^1	3^0	4^0	5^0	6^0	7^0	8^1	9^1	10^0	11^0
1	0	1	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0
3	1	1	1	0	1	1	1	1	1	1	1	1
4	1	1	1	1	0	1	1	1	1	1	1	1
5	1	1	1	1	1	0	1	1	1	1	1	1
6	1	1	1	1	1	1	0	1	1	1	1	1
7	1	1	1	1	1	1	1	0	1	1	1	1
8	0	0	0	0	1	0	0	1	1	0	0	0
9	0	0	0	0	0	1	0	1	0	1	0	0
10	1	1	1	1	0	1	1	0	0	1	0	1
11	1	1	1	1	1	1	1	0	1	1	1	0
D	=	.	.	.	+	.	.	+	+	.	+	+

Символи «+» ставляться в координатах, відповідних стовпцям дефектів, якщо стани виходів (10 і 11) змінюються при внесенні поодинокої несправності даної лінії.

Таким чином, зменшити обчислювальну і структурну складність алгоритму паралельного моделювання можна шляхом аналізу тільки тих несправностей, які є інверсними по відношенню до станів ліній справної поведінки цифрової схеми. В результаті виходить таблиця, зменшена в два рази, яка містить наступні вектори моделювання і трансформуються в вектор перевірюваних на вхідному тестовому наборі дефектів:

L	G	1^0	2^0	3^1	4^1	5^1
1	1	0	1	1	1	1
2	1	1	0	1	1	1
3	0	0	0	1	0	0
4	0	1	1	0	1	0
5	0	1	1	1	1	1
D	=	+	+	+	+	+

→

L	G	F	D
1	1	0	+
2	1	0	+
3	0	1	+
4	0	1	+
5	0	1	+

→

L	1	2	3	4	5
G	1	1	0	0	0
F	0	0	1	1	1
D	+	+	+	+	+

Отже, вхідний тестовий набір 11000 перевіряє поодинокі константні несправності, представлені вектором: 00111.

Обчислювальна складність такого алгоритму визначається формулою:

$$Q = \frac{k}{w} \times (n+1),$$

де k – кількість тестових вхідних наборів, w – довжина регістра для паралельного запису і аналізу фрагмента вектора моделювання, n –- кількість ліній цифрової схеми.

Перевага паралельного методу моделювання несправностей полягає в можливості використання історії обробки дефектів на кубітних покриттях логічних елементів для прискорення процесу аналізу дефектів на тестових наборах. При цьому швидкодія моделювання істотно підвищується в порівнянні з базовим методом паралельного аналізу дефектів.

4.4 Хмарний сервіс «Quantum Modeling»

Пропонується хмарний сервіс проектування і тестування цифрових пристроїв, який відрізняється від існуючих аналогів ергономічним інтерфейсом швидкого введення структурно-функціонального графічного опису у вигляді рисунка, а також адресно-орієнтованим кубітно-векторним моделюванням справної поведінки і несправностей в цілях істотного підвищення швидкодії інтерпретативного аналізу, синтезу тестів і діагностування.

Мета – надання хмарних сервісів проектування і тестування цифрових пристроїв широкому колу фахівців і студентів, пов'язаному з впровадженням і відпрацюванням нової технології квантового моделювання на основі використання кубітних структур даних.

Задачі: 1) Структури кубітних даних для проектування моделі цифрового пристрою. 2) Структура функціональних компонентів хмарного сервісу «Quantum Modeling». 3) Тестування сервісу моделювання на реальних цифрових логічних схемах. 4) Хмарна імплементація сервісів моделювання і тестування логічних схем. 5) Практичні результати і висновки.

Структури кубітних даних цифрового пристрою орієнтовані на паралельний аналіз цифрових систем. Чи можна відмовитися від структурно-логічної схеми опису моделі при синтезі і аналізі цифрового виробу? Відповідь пов'язана з memory-driven реалізацією логічних або комбінаційних

схем. Для цього необхідно створювати нові оригінальні формати візуалізації, нетрадиційні по відношенню до історично сформованих шаблонів проектування комп'ютингових компонентів. Далі розглядається приклад логічної схеми (рис. 4.9) з бібліотеки міжнародного симопзіума ISCAS, для якої здійснюється побудова структур даних.

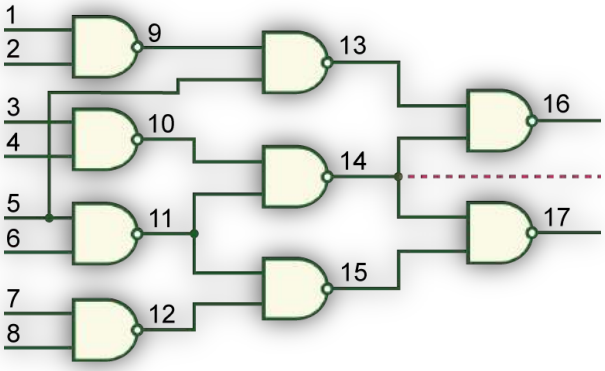


Рис. 4.9 – Структура логічних елементів з бібліотеки ISCAS

Одним з можливих варіантів інноваційної форми опису цифрових логічних схем є суперпозиційна структура кубітних векторів, представлена на рис. 4.10.

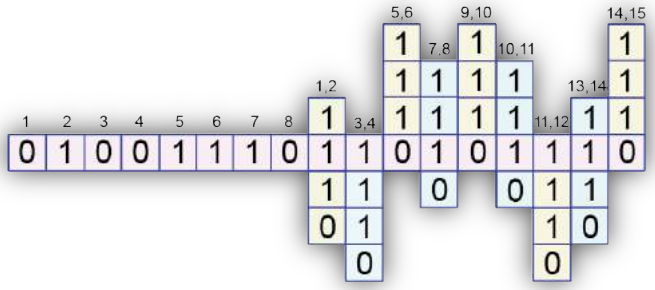


Рис. 4.10 – Інноваційні кубітні структури даних логічної схеми

Тут результат аналізу визначається суперпозицією кубітних векторів, що створюють вектор моделювання M, де алгоритм аналізу містить одну процедуру – зсув векторів вгору-вниз щодо вектора моделювання. Віртуальний зсув технологічно просто реалізується шляхом обчислення адрес комірок кубітних векторів Mi=Qi[M(Xi)], з яких зчитується інформація і заноситься в вектор

моделювання справної поведінки. Структурний взаємозв'язок кубітних векторів-примітивів здійснюється за допомогою нумерації логічних змінних: вхідних, внутрішніх і вихідних. Змінні формують своїми двійковими станами адреси комірок кубітних векторів для обчислення реакції цифрового пристрою на вхідний вплив (01001110 110101110).

Структури кубітних даних для синтезу та аналізу цифрових систем і компонентів оперують наступною моделлю опису логічних схем:

$$S = \{M, X, Y, Q\},$$
$$M = (M_1, M_2, ..., M_i, ..., M_m),$$
$$X = (X_1, X_2, ..., X_i, ..., X_n),$$
$$Y = (Y_1, Y_2, ..., Y_i, ..., Y_k),$$
$$Q = (Q_1, Q_2, ..., Q_i, ..., Q_q),$$
$$M_i = Q_i[M(X_i)].$$

Тут представлені такі системні компоненти: M – вектор моделювання цифрового пристрою, який пов'язує всі кубітні покриття примітивних елементів в структуру; X – вектор вхідних змінних, заданих двійковими значеннями; Y – вектор вихідних змінних, які формують реакцію цифрового пристрою; Q – кубітне покриття, представлене у вигляді вектора вихідних станів логічного елемента і призначене для формування його функції. Основне характеристичне рівняння для моделювання цифрового пристрою оперує обчисленням адрес для запису-зчитування даних, що створює прості і швидкодіючі транзакції між вектором моделювання і кубітними покриттями:

$$M_i = Q_i[M(X_i)].$$

Щоб визначити двійкове значення логічної змінної або лінії Mi, необхідно сформувати адресу комірки кубітного покриття, яка створюється конкатенацією двійкових станів вектора моделювання M, де адреси комірок вектора задаються номерами-ідентифікаторами вхідних змінних X. Характеристичне рівняння безпосередньо впливає на швидкодію квантового методу моделювання, яка залежить від операцій конкатенації k, зчитування r та запису w

бітів, кількості q кубітних покриттів в цифровій схемі або логічних примітивів, а також довжини тесту (вхідних наборів) t:

$$Q = (k + r + w) \times q \times t.$$

Таким чином, лише три транзакційних операції необхідні для обробки логічного елемента будь-якої функціональної складності!

4.5 Структура хмарного сервісу «Quantum Modeling»

Сучасна стратегія компьютингового бізнесу полягає в масовому переведенні всіх процесів і явищ з чисто фізичного в кібер-фізичний простір. Це робить сервіси менш уразливими для кібер злочинності, економічно вигідними і технологічно доступними в просторі і часі, без будь-яких обмежень. Тому високотехнологічний бізнес Design and Test, який визначається компаніями, що формують індекс капіталізації NASDAQ, посиленими темпами переходить в хмарний кіберфізичний комп'ютинг, який має приблизну структуру, наведену на рис. 4.11.

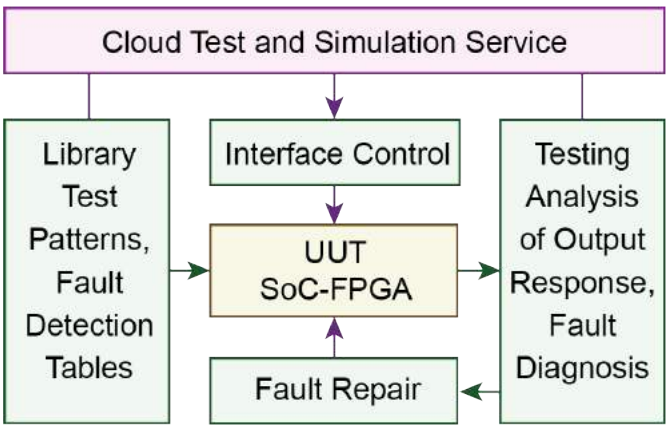


Рис. 4.11 – Desing and Test Cloud Online Service

Тут є тільки один фізичний модуль з IP-адресою, Unit Under Test (будь-які цифрові пристрої, сенсори, актюатори, мобільні пристрої, автомобільні комп'ютери, кінцеві гаджети), який знаходиться в фізичному просторі. Решта блоків, зазначених в структурі, створюють хмарні сервіси, доступні в режимі

онлайн 24/7. Таким чином, будь-який фізичний пристрій може бути легко продіагностований завдяки його під’єднанню до хмарних сервісів в режимі автономного тестування або навіть в режимі робочого функціонування. Все сказане відноситься і до освітнього процесу, коли студенти можуть використовувати як завгодно складну апаратуру, під’єднану через інтернет до власного гаджету для отримання кіберфізичних сервісів, пов'язаних з виконанням реальних технологічних експериментів на комп'ютингових пристроях.

Хмарний сервіс «Quantum Modeling» реалізований на мові SWIFT (2300 рядків коду) і містить такі структурні компоненти або модулі: 1) Q-function – кубітна модель функціонального примітиву, що входить до складу логічної схеми. 2) Data Structures – структура цифрового пристрою, заснована на використанні наскрізної нумерації вхідних внутрішніх і вихідних змінних, які фігурують в якості вхід-вихідних ідентифікаторів ліній логічних елементів. 3) Interface – інтерфейс зв'язку цифрового пристрою з зовнішнім оточенням створює контакти для впливу на виріб зовнішніми функціональними або тестовими наборами, а також для зняття двійкової інформації з зовнішніх спостережуваних виходів схеми. 4) Run-Step – функціональні сервіси моделювання вхідних впливів в режимах: покрокове моделювання тесту шляхом ручного завдання або введення двійкового вхідного набору; автоматичне моделювання вичерпного тесту на всіх 2**n вхідних послідовностях. 5) Visual – сервіс візуалізації схеми цифрового пристрою в складі: логічних елементів; з'єднувальних ліній між елементами, зовнішніми входами і виходами; візуалізація зовнішніх вхідних і вихідних контактів; візуалізація станів логічних елементів на одному вхідному наборі і на повному тесті; візуалізація вектора моделювання на одному тестовому наборі і на повному тесті; візуалізація нумерації логічних елементів, вхідних і вихідних портів, а також десяткових кодів, відповідних двійковим кубітним векторам логічних примітивів. 6) Infrastructure – інфраструктурні сервіси для обслуговування основних функцій, таких як збереження файлу зі схемним описом цифрового

пристрою; корекція схемного опису шляхом видалення-додавання структурних компонентів, кубітних покриттів і міжелементних зв'язків. 7) Optimizer – оптимізація візуального розміщення схемних компонентів і зв'язків шляхом застосування хвильового алгоритму і обмежень, пов'язаних з мінімальною відстанню між лініями і елементами. 8) Fault Simulation – моделювання несправностей цифрових пристроїв на основі використання кубітних покриттів логічних елементів і булевих похідних, що дають можливість одночасної обробки списків несправностей за допомогою дедуктивного алгоритму. 9) Test Synthesis – синтез тестових наборів для перевірки поодиноких константних несправностей шляхом взяття булевих похідних за кубітним покриттям для всіх вхідних змінних цифрового пристрою на основі зустрічного зсуву і логічного підсумовування симетричних сусідніх частин кубітного вектора. Screen-Shot екрану, який візуалізує інтерфейс сервісу «Quantum Modeling», представлений на рис. 4.12.

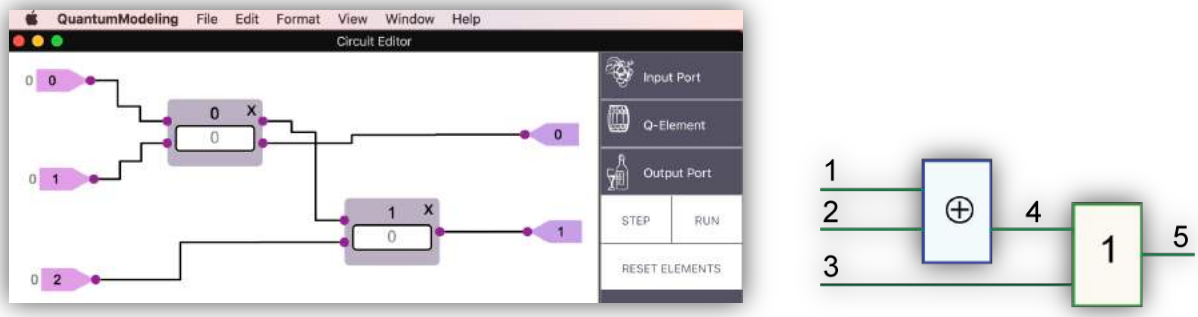


Рис. 4.12 – Візуалізація екрану з сервісом Quantum Modeling

З огляду на, що початковий стан логічного елемента до початку моделювання невідомий, то в правому верхньому куті демонструється стан «x», який довизначається двійковим значенням, що ілюструється на рис. 4.13, який відображає результат моделювання справної поведінки схеми з двох логічних елементів (Q0 (xor) = 0110, Q1(or) = 0111) на вхідному наборі 111.

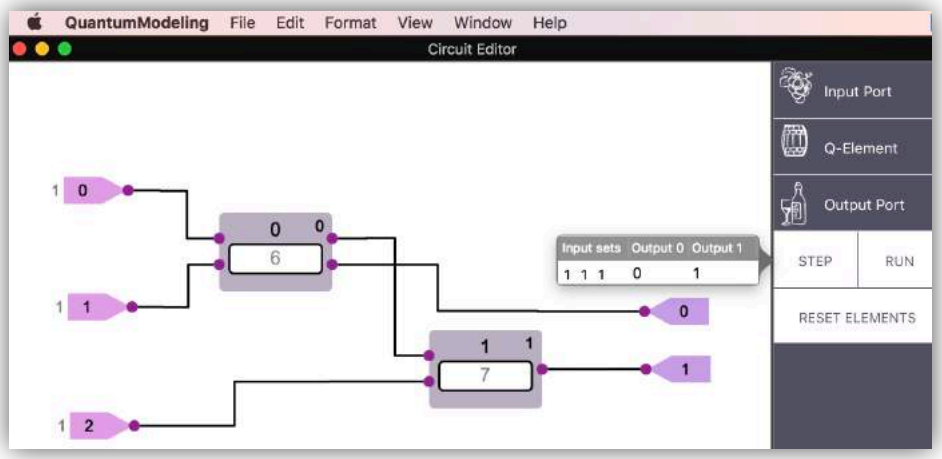


Рис. 4.13 – Візуалізація результату моделювання схеми

Наступна схема, представлена на рис. 4.14, містить дані про моделювання повного тесту, що містить 8 вхідних наборів, за допомогою сервісної процедури RUN.

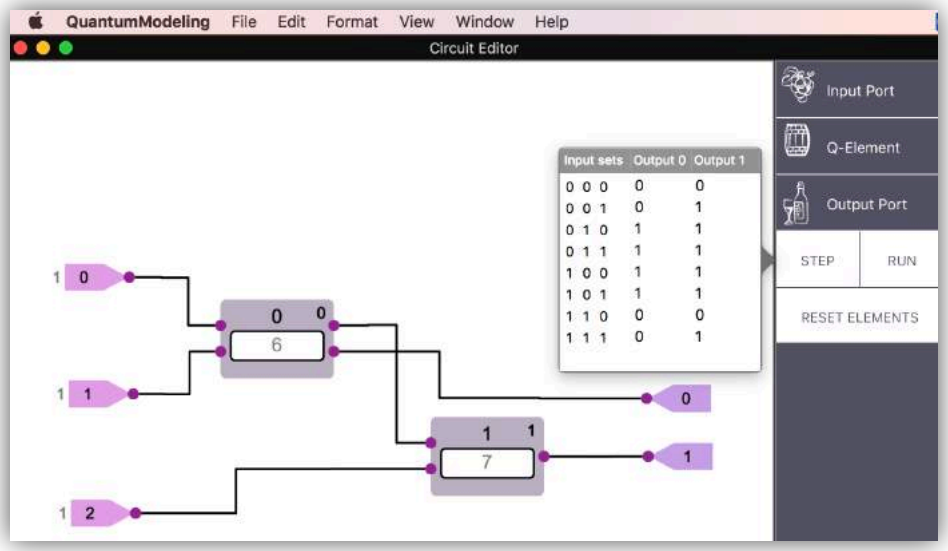


Рис. 4.14 – Моделювання схеми на повному тесті

Більш складна структура, раніше зображена на рис. 1, представлена візуальним інтерфейсом (рис. 4.15), де присутні обнулені функціональності з невизначеними станами логічних елементів.

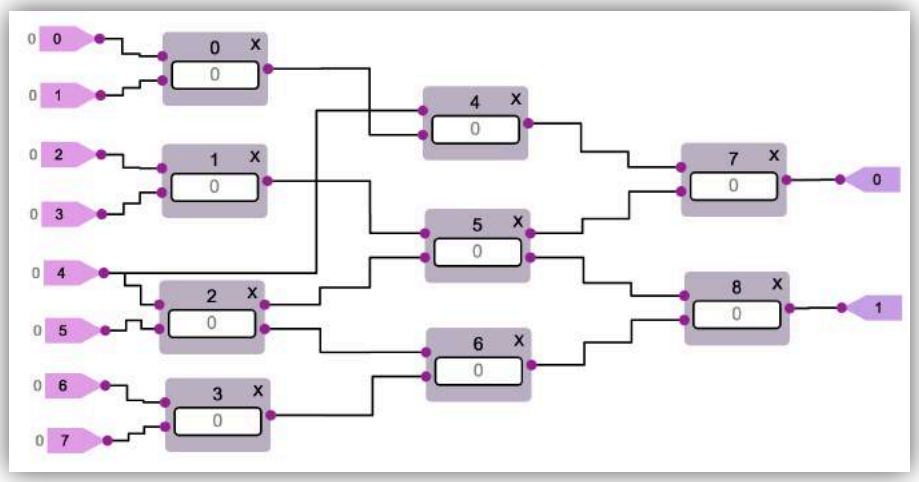


Рис. 4.15 – Вихідна схемна структура-шаблон

Для моделювання цифрової схеми вводяться кубітні покриття в десятковому вигляді, що досить зручно для простих логічних елементів, рис 4.16. Щоб спостерігати стани вихідних ліній можна скористатися з'єднанням внутрішніх ліній з вихідними значками інтерфейсу. Це також дає можливість автоматично формувати таблицю моделювання на повному перевіряльному тесті, що містить 2**n вхідних послідовностей.

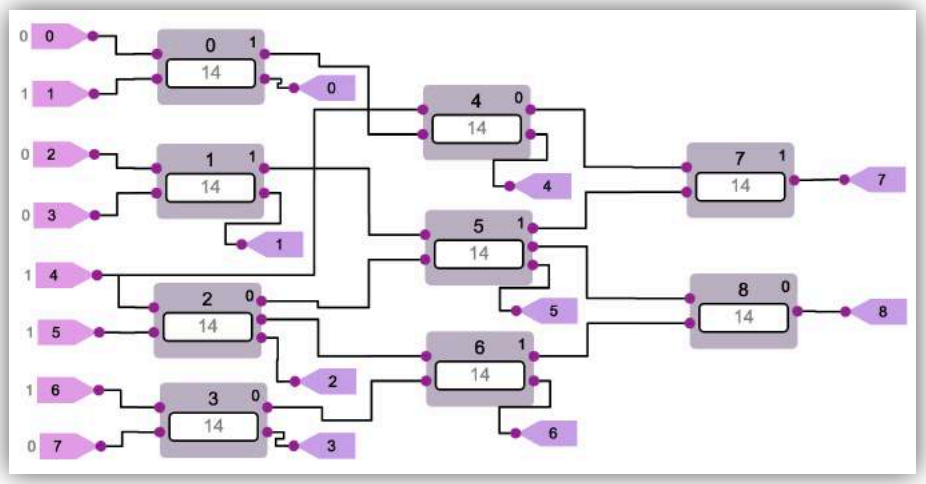


Рис. 4.16 – Моделювання схемної структури на тест-векторі

Реалізація Хассе-процесора (рис. 4.17) для вирішення задачі покриття складається з логічних or-елементів, які містять одно-входові примітиви, дво-

входових, три-входові і один чотири-входовий елемент. Практично за один автоматний такт здійснюється пошук оптимального покриття, якщо елементи реалізують регістрові змінні за входами і виходами. Стан найближчого до зовнішніх входів виходу логічного елемента, що дорівнює одиниці, свідчить про наявність оптимального рішення.

Таким чином, сервіс логічного моделювання здатний продемонструвати переваги гнучких кубітних структур даних і адресно-орієнтованого аналізу цифрових пристроїв, як альтернативної технології для memory-driven квантового паралельного комп'ютингу для вирішення задач синтезу та аналізу.

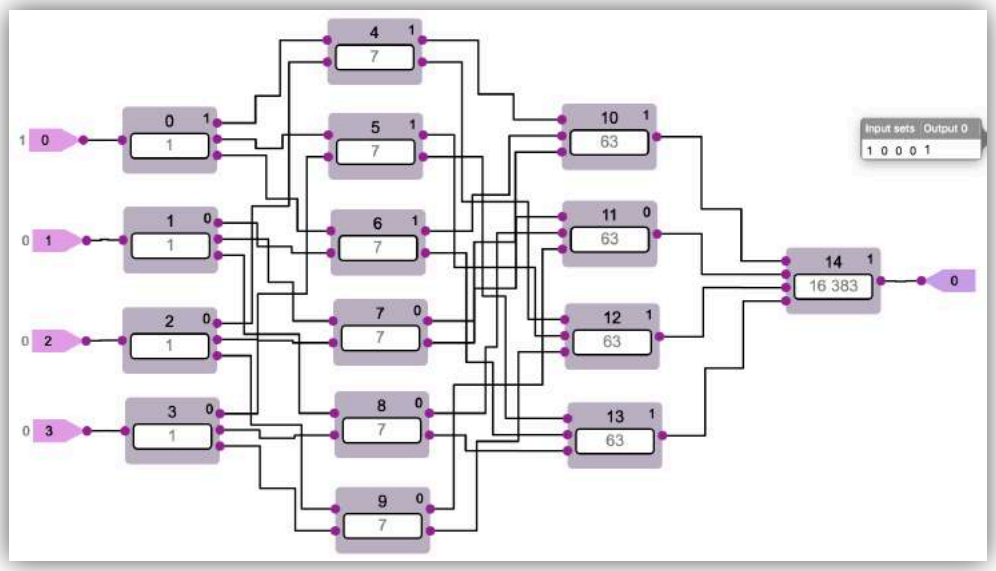


Рис. 4.17 – Моделювання Хассе-структури на тест-векторі

Тут слід також наголосити на необхідності введення в майбутньому: нової моделі кубітних несправностей, орієнтованої на memory-driven комп'ютинг; завдання кубітного формату вхідних впливів, аналогічного кубітним векторам логічних елементів; синтезу кубітних моделей для цифрових автоматів, де вже не потрібні тригери для синтезу функцій збудження; паралельного моделювання без наявності адрес пам'яті; моделювання несправностей, прив'язаних до кубітів.

4.6 Хмарна імплементація сервісу "Quantum modeling"

Реалізація квантового моделювання в якості хмарного сервісу має ряд переваг перед програмним додатком: 1) Інваріантність до апаратних обчислювальних засобів – основи функціонування сервісу, яка швидко застаріває. 2) Відсутність необхідності закупівлі апаратних засобів для реалізації програмного додатку як сервісу. 3) Низький рівень фінансових витрат для оренди хмарного простору, як платформи для реалізації комп'ютингового сервісу. 4) Інваріантність місця знаходження розробника (-ів) для швидкої імплементації хмарного сервісу. 5) Інваріантність хмарного сервісу до географічних координат всіх користувачів, що знаходяться на планеті. 6) Доступність хмарного сервісу з будь-якої точки земної кулі в часі і просторі. 7) Високий рівень надійності дата центрів, які формують хмарно-орієнтовані платформи від компаній: Google, Amazon, Microsoft. 8) Високий рівень software-driven кібербезпеки хмарних сервісів, які надаються згаданими платформами. 9) Швидка технологічна масштабованість хмарних сервісів, орієнтована на розширення функціональних можливостей і велику кількість реальних користувачів.

Конкретно, хмарний сервіс "Quantum modeling" реалізований на платформі Google, яка забезпечує засоби для реалізації інфраструктурних мікросервісів: зберігання даних і доступу до них, кібер захисту хмарної функціональності і автентифікації користувачів. Інфраструктурні мікросервіси від Google платформи істотно впливають на продуктивність і зменшення часу проектування хмарних сервісів, а також на технологічні зручності підтримки працюючого сервісу в актуальному режимі функціонування.

Хмарний сервіс "Quantum modeling" представлений доменно і IP-ідентифікованою структурою кіберфізичних компонентів, що реалізують комп'ютинг-сервіс моделювання цифрових пристроїв в контейнерно-орієнтованому середовищі Docker, що працює в Google Computing Engine.

Мета сервісу пов'язана з популяризацією нових технологій квантового синтезу та аналізу цифрових пристроїв в середовищі студентів, вчених і фахівців завдяки згаданим вище перевагам хмарного комп'ютингу.

Задачі: 1) Створення хмарних мікро-сервісів квантового проектування, моделювання, діагностування та верифікації широкого класу цифрових пристроїв. 2) Популяризація кубітних структур даних і квантових технологій проектування memory-driven обчислювальних пристроїв.

Сучасний ринок проектування хмарних систем показує, що в останні три роки набирають популярність гнучкі і масштабовані технології проектування сервісів, які використовують контейнеризацію, спрямовану на мінімізацію time-to-market. Основна ідея контейнера полягає в створенні захищеного і доступного віртуального середовища проектування і експлуатації в кіберпросторі на основі використання локальної або глобальної платформи, рис. 4.18.

На інфраструктуру платформи встановлюється практично будь-яка операційна система Host operating system, зручна для розробника, наприклад, Ubuntu, Debian. У локально виконаній hardware інфраструктурі контейнерів можна інсталювати також OSX та Windows.

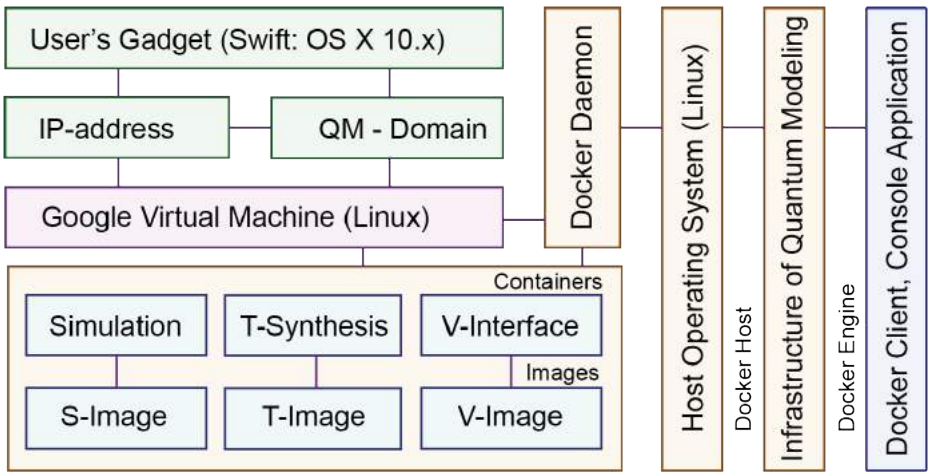


Рис. 4.18 – Інфраструктура сервісу моделювання

Розробник інсталює Docker Engine, якщо він не встановлений спочатку, який за допомогою Docker Daemon (реалізація на Golang, C), керує контейне-

рами, запущеними на основі попередньо побудованих образів (Images). Контейнери Containers є легкою віртуальною обчислювальною машиною, під якою працюють функціональні блоки: Modeling, Simulation, Test Synthesis, Fault Simulation, Fault Diagnosis, реалізовані мовою Swift (Jawa), і мають відповідні бібліотеки. Docker Engine являє собою комп'ютингову систему хмарної контейнеризації, яка складається з двох взаємодіючих між собою компонентів: Docker Host, Docker Client (Console Application), які можна інтерпретувати як виконавчий Host і керуючий (Client) механізми хмарної системи Quantum Modeling. Далі Docker Engine є компонентом, який входить до складу Google Virtual Machine, якій ставиться у відповідність IP-address і Quantum Modeling Domain, що робить розробку, а далі хмарний сервіс, доступним для широкого кола фахівців і студентів.

Останньою ланкою у вертикальній зв'язці (Back-End, Front-End) є User's Gadget (Swift OS X.10x). На кінцевому пристрої користувача установлюються Front-End додатки, здатні привести результати роботи хмарних сервісів моделювання до мульти-віконного формату візуалізації даних, зручного для сприйняття людиною. Слід зазначити, що набагато технологічніше створювати Front-End у вигляді додатку на JavaScript, HTML, CSS, який буде працювати через Internet Browser. Це гарантує відсутність блокування клієнта з боку постачальника конкретної ОС (vendor lock). Для мобільних пристроїв доцільніше використовувати додаток індивідуального клієнта під кожну ОС. Всі модулі хмарного сервісу запрограмовані мовою Swift, операційна система OSX 10.9, компілятор XCode 7. Кількість вихідних файлів 42, загальна кількість рядків коду – 2300.

Дослідна експлуатація сервісу Quantum Modeling показала: 1) Слухачі досить легко переходять на кубітний формат опису логічних елементів в цифровий структурі. 2) Істотно зменшується час проектування цифрових пристроїв при використанні зручного графічного інтерфейсу. 3) Значно простіше вирішуються всі задачі, пов'язані з тестуванням, моделюванням і діагносту-

ванням цифрових компонентів і схем при застосуванні кубітних структур даних і квантових методів синтезу та аналізу.

Для отримання статистичної інформації та верифікації Quantum Modeling було проведено 10 експериментів на логічних схемах з бібліотеки ISCAS, а також на інших структурах: 1) Adder SP. 2) Circuit Schneider. 3) Circuit C5. 4) Circuit C17. 5) RFO Circuit. 6) MUX16 Circuit. 7) DFA Circuit. 8) Hasse processor. 9) DC4-16 Circuit. 10) Circuit C432. Порівнянню підлягали час введення схемної структури (Modeling Time), а також час проектування діагностичної інформації (Designing Time), до якого входять: введення моделі пристрою, синтез і аналіз тестів для отримання таблиці несправностей. Базовим варіантом для порівняння послужив продукт Active HDL, Aldec Inc., де інформація про моделі схеми вводилася VHDL мовою опису апаратури. Статистика порівняння двох часів: Modeling Time і Designing Time представлена на рис. 4.19 і 4.20 відповідно.

Вона показує переваги візуального образного схемотехнічного проектування логічних схем невеликої розмірності в порівнянні з введенням схем на основі HDL-опису. Особливо це прийнятно і ергономічно в процесі навчання студентів технологіям проектування і верифікації цифрових систем і компонентів. Однак для великих промислових проектів поки що доцільно використовувати мови опису апаратури.

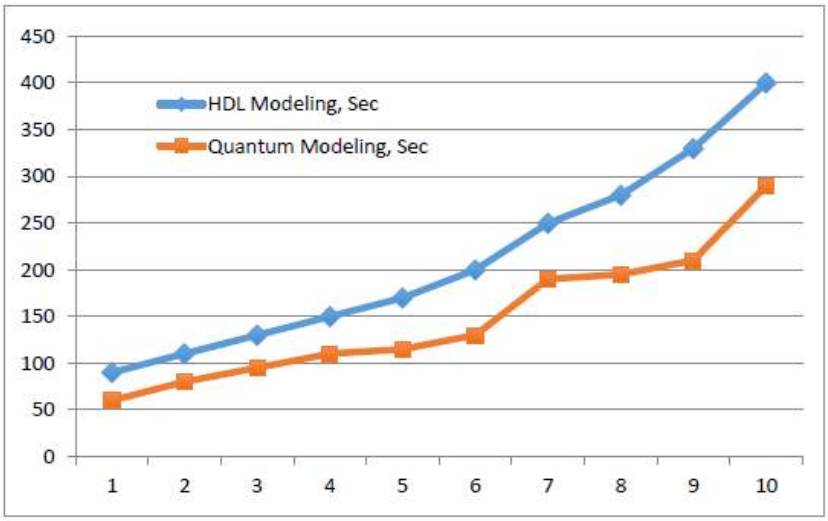


Рис. 4.19 – Аналіз часу введення логічних схем

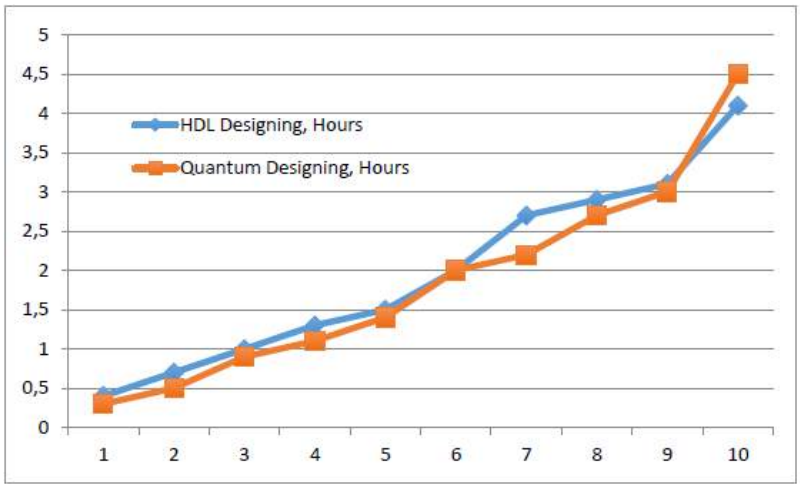


Рис. 4.20 – Аналіз часу проектування логічних схем

4.7 Висновки

Становлення нової теорії проходить через її первісне відторгнення традиційно орієнтованою науковою громадськістю. Проте, очевидні окремі переваги практично спрямованого квантового комп'ютингу знаходять відображення в публікаціях вчених, які працюють в областях, пов'язаних з паралельними memory-driven обчисленнями і комбінаторикою задач захисту інформації, верифікації та діагностування. Немає кращого способу просунути інноваційну теорію, ніж шлях використання відкритого кіберфізичного простору, де створюються хмарні сервіси, що покривають інтереси широкої наукової громадськості в часі і просторі. Тому з'єднання квантового комп'ютингу з його хмарним виконанням є вдалою технологічною зв'язкою для вирішення задач кіберфізичного формату, орієнтованого на проектування і тестування цифрових систем і програмно-апаратних компонентів. В рамках зазначеного науково-технологічного напрямку були створені такі засоби: 1) Графічний інтерфейс, зручний для ручного проектування кубітних моделей цифрових пристроїв і компонентів, який дає можливість в режимі online здійснювати корекцію помилок. При високій швидкодії сучасного комп'ютингу головним фактором, що впливає на time-to-market, стає час ручного введення моделей цифрових систем і компонентів. 2) Структури даних для кубітного опису циф-

рових пристроїв і компонентів, які відрізняються компактністю і високим паралелізмом їх обробки. 3) Інфраструктура для проектування і тестування цифрових пристроїв і компонентів з функціями зберігання, видалення та корекції даних, що дає можливість здійснювати одночасне створення цифрових схем декількома проектувальниками. 4) Програмні модулі для квантового моделювання цифрових пристроїв і компонентів в режимах ручного і автоматичного введення вхідних тестових послідовностей, що дає можливість наочного навчання студентів методам синтезу та аналізу. 5) Програмні засоби для синтезу таблиць несправностей на основі дедуктивного методу квантового моделювання дефектів шляхом використання кубітних структур даних, які забезпечують пошук поодиноких і кратних константних дефектів в режимі реального часу. 6) Напрями майбутніх досліджень пов'язані зі створенням memory-driven моделей несправностей, орієнтованих на кубітні форми опису функціональностей з подальшою розробкою сімейства memory-driven паралельних методів квантового тестування, моделювання та діагностування комп'ютингових систем і компонентів.

ВИСНОВОК

Існуючі обчислювальні потужності всієї планети не здатні задовільно вирішити згадану проблему через низьку продуктивність глобального комп'ютингу і високу енерговитратність існуючих класичних і квантових комп'ютерів. Це дає підстави для пошуку альтернативних технологій реалізації комп'ютингових інноваційних архітектур. Можливим вирішенням проблеми може бути квантовий memory-driven і logic-free комп'ютинг, який виключає дорогі, енерговитратні, логіко-подібні (or, not) операції суперпозиції і переплутування.

Проведені науково-технологічні дослідження в рамках дисертаційної роботи характеризуються успішним вирішенням актуальної науково-практичної задачі квантового проектування, моделювання і тестування цифрових пристроїв і компонентів на основі використання memory-driven кубітних структур даних, вільних від застосування логіки.

Автором одержано такі наукові та практичні результати:

1) Запропоновано нову модель метричної взаємодії класичного та квантового комп'ютингу, яка характеризується взаємно-однозначною відповідністю за параметрами паралелізму, суперпозиційності та переплутування в обох видах обчислень, що дає можливість реалізувати квантовий комп'ютинг в класичному виконанні за рахунок збільшення пам'яті. Модель охоплює всі метричні компоненти, необхідні для реалізації комп'ютингу.

2) Удосконалений метод невизначених коефіцієнтів для мінімізації булевих функцій, який відрізняється від класичного унітарним кодуванням даних для паралельного виконання логічних операцій в цілях отримання двох векторів, відповідних мінімальній диз'юнктивній і кон'юнктивній нормальним формам. Паралельне виконання операції суперпозиції над нульовими і одиничними станами вхідних змінних, представленими

унітарними кодами, дає можливість істотно підвищити швидкодію за рахунок надлишкової пам'яті.

3) Удосконалений кубітний метод пошуку дефектів, який відрізняється від існуючого унітарним кодуванням таблиці перевірюваних дефектів для паралельного виконання логічних операцій над нульовими і одиничними станами вхідних змінних, що дає можливість істотно підвищити швидкодію за рахунок надлишкової пам'яті.

4) Розвинений квантовий метод синтезу тестів для логічних функціональностей за рахунок використання булевих похідних за змінними на кубітних структурах даних, що дає можливість підвищити швидкодію методу шляхом паралельного виконання логічних операцій. Метод дозволяє згенерувати вхідні набори для перевірки всіх поодиноких константних несправностей вхідних і внутрішніх ліній схеми.

5) Розвинений квантовий метод моделювання справної поведінки за рахунок memory-driven реалізації кубітних структур даних, що дає можливість використовувати транзакційні адресно-орієнтовані процедури аналізу цифрових пристроїв, які виключають логічні операції. Структури даних для кубітного опису цифрових пристроїв і компонентів відрізняються компактністю і високим паралелізмом їх обробки.

6) Розроблені хмарні сервіси синтезу та аналізу кубітних моделей цифрових пристроїв і компонентів протестовані на різних прикладах комбінаційних схем. Отримані результати підтверджують висновок, що з'єднання квантового комп'ютингу з його хмарним виконанням є вдалою технологічною зв'язкою для вирішення задач кіберфізичного формату, орієнтованого на проектування і тестування цифрових систем і програмно-апаратних компонентів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Vladimir Hahanov. Cyber Physical Computing for IoT-driven Services. New York. Springer. 2018. 279p.

2. https://www.forbes.com/sites/louiscolumbus/2017/08/15/gartners-hype-cycle-for-emerging-technologies-2017-adds-5g-and-deep-learning-for-first- time / # 646a4cf34be2

3. http://www.gartner.com/newsroom/id/3784363

4. http://www.wired.co.uk/article/ai-neuromorphic-chips-brains

5. A. Gupta and RK Jha, "A Survey of 5G Network: Architecture and Emerging Technologies," in IEEE Access, vol. 3, pp. 1206-1232, 2015.

6. C. Zhu, VCM Leung, L. Shu and ECH Ngai, "Green Internet of Things for Smart World," in IEEE Access, vol. 3, pp. 2151-2162, 2015.

7. K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," in IEEE Access, vol. 4, pp. 2292-2303, 2016.

8. Blockchains: How They Work and Why They'll Change the World IEEE Spectrum. October 2017.

9. https://spectrum.ieee.org/computing/networks/blockchains-how-they-work-and-why-theyll-change-the-world

10. A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, "Internet of Things for Smart Cities," in IEEE IoT Journal, vol. 1, no. 1, pp. 22-32, Feb. 2014.

11. https://www.gartner.com/doc/3471559?srcId=1-7578984202&utm_campaign = RM_GB_2017_TRENDS_QC_E2_What & utm_medium = email & utm_source = Eloqua & cm_mmc = Eloqua -_- Email-LM_RM_GB_2017_TRENDS_QC_E2_ What -_- 0000

12. http://www.gartner.com/smarterwithgartner/three-digital-marketing-habits-to-break-2/

13. Machidon O. Remote SoC / FPGA platform configuration for cloud applications / O. Machidon, F. Sandu, C. Zaharia, P. Cotfas and D. Cotfas // 2014 Inter-

national Conference on Optimization of Electrical and Electronic Equipment (OPTIM). - Bran.- 2014.- P. 827-832.

14. Stanik A. Hardware as a Service (HaaS): Physical and virtual hardware on demand / A. Stanik, M. Hovestadt, O. Kao // 4th IEEE Int. Conference on Cloud Computing Technology and Science Proceedings.- Taipei.- -2012 P. 149-154.

15. Wang C. Regarding Processors and Reconfigurable IP Cores as Services / C. Wang, X. Li, P. Chen, J. Zhang, X. Feng, X. Zhou // 2012 IEEE Ninth Int. Conference on Services Computing.- Honolulu, HI.- -2012 P. 668-669.

16. Ogrutan P. Microcontroller Based System for Accelerated Reliability Tests for Electronic Equipment / P. Ogrutan, LE Aciu // International Conference AFASES.- Brasov.- 2013.- P. 387-392.

17. Expanding the All Programmable SoC Portfolio. Електронний ресурс: www.xilinx.com/zynq (accessed 28 Nov 2013).

18. ZedBoard Development System. Електронний ресурс: http://www.zedboard.org (accessed 10 Dec 2013).

19. LabVIEW Manual National Instruments. Електронний ресурс: http://www.ni.com (accessed 30 Dec 2018).

20. Sarangi A. MacMahon-Xilinx LightWeight IP-XAPP1026 Application Note. v3.2-October 28 2012. Електронний ресурс: http://www.xilinx.com/ (accessed 3 Jan 2014 року).

21. Kohn-Partial C. Reconfiguration of a Hardware Accelerator on Zynq-7000. All Programmable SoC Devices-XAPP1159. Application Note. V1.0-January 21 2013. Електронний ресурс: http://www.xilinx.com/ (accessed 20 Dec 2013).

22. Alonso G. Web services / G. Alonso, C. Fabio, K. Harumi, M. Vijay.- Springer, Berlin Heidelberg.- 2004.

23. Chappell D. Java Web Services / D. Chappell, T. Jewell.- OReilly.- 2002.

24. Madhavapeddy A. Reconfigurable Data Processing for Clouds / A. Madhavapeddy, S. Singh // Proceedings of the IEEE 19th Annual International

Symposium on Field-Programmable Custom Computing Machines (FCCM) .- 2011.- P.141-145.

25. Eguro K. FPGAs for Trusted Cloud Computing / K. Eguro, R. Venkatesan // Proceedings of 22nd International Conference on Field Programmable Logic and Applications. -2012 pp.63-70.

26. Moller L. Reconfigurable systems enabled by a Network-on-Chip / L. Moller, G. Ismael, N. Calazans, F. Moraes // Proceedings of FPL'06 IEEE International Conference on Field Programmable Logic and Applications-Madrid. - 28-30 Aug. 2006.- P. 857-860.

27. El-Medany WM FPGA remote laboratory for hardware e-learning courses / WM El-Medany // Proceedings of the 8th IEEE International Conference on Computational Technologies in Electrical and Electronics Engineering, SIBIRCON.- 2008.- P.106-109.

28. Morgan F. Enhancing Learning of Digital Systems using a Remote FPGA Lab / F. Morgan, S. Cawley // Proceedings of the 6th International Workshop on Reconfigurable Communication-centric Systems-on-Chip.- 2011.- P.1- 8.

29. Hardware at a Glance. National Instruments. Електронний ресурс: http://www.ni.com/white-paper/14604/en/-myRIO (accessed 10 Jan 2014 року).

30. Moravčík M. Teaching cloud computing in cloud computing / M. Moravčík, P. Segeč, J. Uramová, M. Kontšek // 2017 15th International Conference on Emerging eLearning Technologies and Applications (ICETA) .- Stary Smokovec.- 2017. - P. 1-6.

31. Yi H. Implementation of Learning Management System Based on Cloud Computing / H. Yi, Z. Nie, W. Li // 2017 4th International Conference on Information Science and Control Engineering (ICISCE) .- Changsha, China.- 2017. - P. 380-384.

32. Chaudhary R. Network Service Chaining in Fog and Cloud Computing for the 5G Environment: Data Management and Security Challenges / R. Chaudhary, N.

Kumar, S. Zeadally // IEEE Communications Magazine.- vol. 55, no. 11.- November, 2017.- P. 114-122.

33. Frahim J. Securing the Internet of Things: A Proposed Framework / J. Frahim // Cisco White Paper.- 2015.

34. Aujla GS Data Offloading in 5G-Enabled Software-Defined Vehicular Networks: A Stackelberg Game-Based Approach / GS Aujla // IEEE Commun. Mag.- vol. 55, no. 7.- July 2017.

35. Stallings W. Cryptography and Network Security: Principles and Practices / W. Stallings.- Pearson Education, India.- 2006.

36. Peng M. Energy-Efficient Resource Assignment and Power Allocation in Heterogeneous Cloud Radio Access Networks / M. Peng // IEEE Transactions on Vehicular Technology.- vol. 64, no. 11.- Nov. 2015.- P. 5275-5287.

37. Gonzales D. Cloud-trust - A Security Assessment Model for Infrastructure as a Service (Iaas) Clouds / D. Gonzales // IEEE Transactions on Cloud Computing.- vol. 5, no. 3.- July-Sept. 1, 2017.- P. 523-536.

38. John W. Research Directions in Network Service Chaining / W. John // 2013 IEEE SDN for Future Networks and Services.- Nov. 2013.- p. 1-7.

39. Varrette S. HPC Performance and Energy-Efficiency of Xen KVM and VMware Hypervisors / S. Varrette // 2013 25th Int'l. Symp. Computer Architecture and High Performance Computing.- Oct. 2013.- P. 89-96.

40. Kim C. Credit-Based Runtime Placement of Virtual Machines on a Single Numa System for QoS of Data Access Performance / C. Kim, KH Park // IEEE Trans. Computers.- vol. 64, no. 6.- June 2015.- P. 1633-46.

41. Peng M. Fog-Computing Based Radio Access Networks: Issues and Challenges / M. Peng // IEEE Network.- vol. 30, no. 4.- July 2016, P. 46-53.

42. Chen M. Privacy Protection and Intrusion Avoidance for Cloudletbased Medical Data Sharing / M. Chen // IEEE Transactions on Cloud Computing.- vol. PP, no. 99.- 2016.- P. 1-1.

43. Sarkar S. Assessment of the Suitability of Fog Computing in the Context of Internet of Things / S. Sarkar, S. Chatterjee, S. Misra // IEEE Trans. Cloud Computing.- vol. PP, no. 99.- 2015.- P. 1-15.

44. Modi C. A Survey on Security Issues and Solutions at Different Layers of Cloud Comp. / C. Modi // J. Supercomputing.- vol. 63, no. 2.- 2013.- P. 561-592.

45. Shi Y. Cloudlet Mesh for Securing Mobile Clouds from Intrusions and Network Attacks / Y. Shi, S. Abhilash, K. Hwang // 2015 3rd IEEE Int'l. Conf. Mobile Cloud Computing Services and Engineering.- Mar. 2015.- P. 109-18.

46. He D. Efficient and Anonymous Mobile User Authentication Protocol Using Self-Certified Public Key Cryptography for Multi-Server Architectures / D. He // IEEE Trans. Info. Forensics and Security.- vol. 11, no. 9.- Sept. 2016.- P. 2052-2064.

47. Verma OP Notice of Violation of IEEE Publication Principles Peformance Analysis of Data Encryption Algorithms / OP Verma // 2011 3rd Int'l. Conf. Electronics Computer Technolog.- vol. 5.- Apr. 2011.- P. 399-403.

48. Rumale AS Cloud computing: Software as a service / AS Rumale, DN Chaudhari // 2017 Second International Conference on Electrical, Computer and Communication Technologies.- Coimbatore, Tamil Nadu, India.- 2017.- P. 1-6.

49. Kapil D. Cloud Computing: Overview and Research Issues / D. Kapil, P. Tyagi, S. Kumar and VP Tamta // 2017 International Conference on Green Informatics (ICGI) .- Fuzhou, China.- 2017.- P. 71-76.

50. Irwin D. The Financialization of Cloud Computing: Opportunities and Challenges / D. Irwin, P. Sharma, S. Shastri, P. Shenoy // 2017 26th International Conference on Computer Communication and Networks (ICCCN) .- Vancouver, BC. - 2017.- P. 1-11.

51. Mengistu T. A "No Data Center" Solution to Cloud Computing / T. Mengistu, A. Alahmadi, A. Albuali, Y. Alsenani, D. Che // 2017 IEEE 10th International Conference on Cloud Computing (CLOUD) .- Honolulu, CA.- 2017.- P. 714-717.

52. Abramovici M. Digital System Testing and Testable Design / M. Abramovici, MA Breuer and AD Friedman.- Comp. Sc. Press.- 1998.- 652 p.

53. Fujiwara H. Fault Simulation. In Logic Testing and Design for Testability, 1.- MIT Press.- 1985.- P. 84-108.

54. Карибський В.В. Основи технічної діагностики. Кн. 1. / В.В. Карибський, П.П. Пархоменко, Е.С. Согомонян, В.Ф. Халчев.- М .: «Енергія» .- 1976.- 346 с.

55. Чжен Г. Діагностика відмов цифрових обчислювальних систем / Г. Чжен, Е. Меннінг, Г. Метц.- М .: світ.- 1972. - 230 с.

56. Пархоменко П.П. Основи технічної діагностики (Оптимізація алгоритмів діагностування, апаратурні засоби) / П.П. Пархоменко, Е.С. Согомонян. Під ред. П.П. Пархоменко.- М .: Енергія.- 1981.- 320 с.

57. Marinissen EJ Guest Editors 'Introduction: The Status of IEEE Std -1500 / EJ Marinissen, Y. Zorian // IEEE Design & Test of Computers.- 2009.- No26 (1) .- P. 6-7.

58. Малишенко Ю.В. Автоматизація діагностування електронних пристроїв / Ю.В.Малишенко і ін. / Под ред. В.П.Чіпуліса.- М .: Вища школа, 1986.- 216с.

59. Надійність технічних систем / Под ред. І.А.Ушакова.- М .: 1985.- 512 с.

60. Хаханов В.І. Проектування і тестування цифрових систем на кристалах / В.І. Хаханов, Є.І. Литвинова, І.В. Хаханова, О.А. Гузь. - Харків: ХНУРЕ.- 2009. - 484 с.

61. Pomeranz I. Aliasing Computation Using Fault Simulation with Fault Dropping / I. Pomeranz, M. Reddy Sudhakar // IEEE Transactions on Computers.- 1995.- P. 139-144.

62. Lee SM Static fault analysis for resilient System-on-Chip design / SM Lee and SE Lee // 2015 International SoC Design Conference (ISOCC) .- Gyungju.- 2015.- P. 5-6.

63. Admane NC Fault tolerant system for FPGA using simulation based fault injection technique / NC Admane, DR Rotake // 2015 International Conference on Communications and Signal Processing (ICCSP) .- Melmaruvathur.- 2015.- P. 0855-0859.

64. Yıldız C. Fault emulation on heterogeneous architectures / C. Yıldız, C. Gürsoy, S. Gören // 2017 International Conference on Computer Science and Engineering (UBMK) .- Antalya, Turkey.- 2017.- P. 905-910 .

65. Luo T. DaDianNao: A Neural Network Supercomputer / T. Luo // IEEE Transactions on Computers.- vol. 66, no. 1.- Jan. 1, 2017.- P. 73-88.

66. Gonzales D. Cloud-Trust-a Security Assessment Model for Infrastructure as a Service (IaaS) Clouds / D. Gonzales, JM Kaplan, E. Saltzman, Z. Winkelman, D. Woods // IEEE Transactions on Cloud Computing.- vol. 5, no. 3.- July-Sept. 1, 2017.- P. 523-536.

67. Hosokawa T. A Diagnostic Fault Simulation Method for a Single Universal Logical Fault Model / Hosokawa, H. Takano, H. Yamazaki, K. Yamazaki // 2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC) .- Christchurch, 2017.- P. 217-218.

68. Dhiliban C. Fault simulation and analysis of VLSI circuit using n-detect test sets / Dhiliban C and Govindaraju S // 2016 Online Int. Conference on Green Engineering and Technologies (IC-GET) .- Coimbatore, 2016.- P. 1-5.

69. Jinling D. A fault simulation method based on mutated truth table of logic gates / D. Jinling and X. Aiqiang // 2016 International Conference on Integrated Circuits and Microsystems (ICICM) .- Chengdu, 2016.- P. 28-32 .

70. Hadjitheophanous S. Scalable parallel fault simulation for shared-memory multiprocessor systems / S. Hadjitheophanous, SN Neophytou, MK Michael // 2016 IEEE 34th VLSI Test Symposium (VTS) .- Las Vegas, NV.- 2016.- P. 1 -6.

71. Chiang KY Fault Simulation and Test Pattern Generation for Cross-gate Defects in FinFET Circuits / KY Chiang, YH Ho, YW Chen, CS Pan, JCM Li // 2015 IEEE 24th Asian Test Symposium.- Mumbai, 2015.- P. 181-186.

72. Wu X. Substation Grounding Studies with More Accurate Fault Simulation Strategy and SI / MI Method / X. Wu, V. Simha, Y. Xue and R. Wellman // IEEE Transactions on Power Delivery.- vol. PP, no. 99.- P. 1-1.

73. Ubar R. Combinational fault simulation in sequential circuits / R. Ubar, J. Kõusaar, M. Gorev, S. Devadze // 2015 IEEE International Symposium on Circuits and Systems (ISCAS) .- Lisbon.- 2015.- P. 2876-2879.

74. Rivière L. A novel simulation approach for fault injection resistance evaluation on smart cards / L. Rivière, J. Bringer, TH Le, H. Chabanne // 2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW) .- Graz.- 2015.- P. 1-8.

75. Mirkhani S. EAGLE: A regression model for fault coverage estimation using a simulation based metric / S. Mirkhani, JA Abraham // 2014 International Test Conference.- Seattle, WA.- 2014.- P. 1-10.

76. Xu J. The Research of Memory Fault Simulation and Fault Injection Method for BIT Software Test / J. Xu, P. Xu // 2012 Second International Conference on Instrumentation, Measurement, Computer, Communication and Control.- Harbin.- 2012. - P. 718-722.

77. Kochte MA Efficient BDD-based Fault Simulation in Presence of Unknown Values / MA Kochte, S. Kundu, K. Miyase, X. Wen, HJ Wunderlich // 2011 Asian Test Symposium.- New Delhi.- 2011.- P. 383-388.

78. Reinsalu U. Fast RTL Fault Simulation Using Decision Diagrams and Bitwise Set Operations / U. Reinsalu, J. Raik, R. Ubar, P. Ellervee // 2011 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems.- Vancouver, BC.- 2011.- P. 164-170.

79. Janning A. A Cost-Effective FPGA-based Fault Simulation Environment / A. Janning, J. Heyszl, F. Stumpf, G. Sigl // 2011 Workshop on Fault Diagnosis and Tolerance in Cryptography.- Nara.- 2011.- P. 21-31.

80. Goudarzi H. Design of a universal logic block for fault-tolerant realization of any logic operation in trappedion quantum circuits / H. Goudarzi, MJ Dousti, A.

Shafaei et al. // Quantum Infirmation Processing Journal.- Springer US.- 2014.- vol. 13, iss. 5.- P. 1267-1299.

81. Daley AJ Quantum computing and quantum simulation with group-II atoms / AJ Daley // Quantum Information Processing Journal.- 2011.- no 10.- Springer US.- P. 865-884.

82. Kharchenko V. Green IT Engineering: Concepts, Models, Complex Systems Architectures / V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.) // In the book series "Studies in Systems, Decision and Control" (SSDC) .- vol. 1.- Berlin, Heidelberg: Springer International Publishing.- 2017.

83. Kharchenko V. Green IT Engineering: Components, Networks and Systems Implementation / V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.) // In the book series "Studies in Systems, Decision and Control" (SSDC). - vol. 2.- Berlin, Heidelberg: Springer International Publishing.- 2017.

84. Kooli M. A survey on simulation-based fault injection tools for complex systems / M. Kooli, G. Di Natale // 2014 9th IEEE Int. Conference on Design & Technology of Integrated Systems in Nanoscale Era.- Santorini.- 2014.- P. 1-6.

85. Rozier K. Dependability management-Part 1: Dependability management systems / K. Rozier.- International Electrotechnical Commission.- IEC Std.- 2003.

86. Avizienis A. Basic concepts and taxonomy of dependable and secure computing / A. Avizienis, J.-C. Laprie, B. Randell, C. Landwehr // IEEE Trans. Dependable Secur. Comput.- vol. 1, no. 1.- Jan. 2004. P. 11-33. [Online]. Available: http://dx.doi.org/10.1109/TDSC.2004.2

87. Raimund U. Design and Test Technology for Dependable Systems-on-Chip / U. Raimund, R. Jaan, TV Heinrich // United State of America by Information Science Reference.- 2011 року.

88. Di Natale G. Software-implemented system dependability for safety critical applications / G. Di Natale // Ph.D. diss.- Politecnico di Torino.- 2003.

89. Huang K.-H. Algorithm-based fault tolerance for matrix operations / K.-H. Huang, JA Abraham // IEEE Trans. Comput.- vol. 33, no. 6.- Jun. 1984.- P. 518-528. [Online]. Available: http://dx.doi.org/10.1109/TC.1984.1676475

90. Benso A. Validation of a software dependability tool via fault injection experiments / A. Benso, S. Di Carlo, G. Di Natale, L. Tagliaferri, P. Prinetto // Proceedings Seventh International On-Line Testing Workshop.- Taormina .- 2001.- P. 3-8 [Online]. Available: http://dl.acm.org/citation.cfm?id=876896.880975

91. Benso A. Software dependability techniques validated via fault injection experiments / A. Benso, S. Di Carlo, G. Di Natale, P. Prinetto, L. Tagliaferri // RADECS 2001. 2001 6th European Conference on Radiation and Its Effects on Components and Systems (Cat. No.01TH8605) .- 2001.- P. 269-274.

92. Benso A. A c / c ++ source-to-source compiler for dependable applications / A. Benso, S. Chiusano, P. Prinetto, L. Tagliaferri // Proceeding International Conference on Dependable Systems and Networks. DSN 2000.- New York, NY.- 2000.- P. 71-78. [Online]. Available: http://dl.acm.org/citation.cfm?id=647881.737934

93. Benso A. Control-flow checking via regular expressions / A. Benso, S. Di Carlo, G. Di Natale, P. Prinetto, L. Tagliaferri // Proceedings 10th Asian Test Symposium.- Kyoto.- 2001.- P . 299-303. [Online]. Available: http://dl.acm.org/citation.cfm?id= 872025.872649

94. De Oliveira Moraes R. Jaca-a software fault injection tool / R. de Oliveira Moraes, E. Martins // Proceedings of 2003 International Conference on Dependable Systems and Networks.- 2003.- p. 667.

95. Ziade H. A survey on fault injection techniques / H. Ziade, R. Ayoubi, R. Velazco // The International Arab Journal of Information Technology.- vol. 1, no. 2.- July 2004. P. 171-186.

96. Ningfang S. Fault injection methodology and tools / S. Ningfang et al. // 2011 International Conference on Electronics and Optoelectronics (ICEOE) .- July 2011.- P. V1-47-V1-50.

97. Kanawati GA Ferrari: A flexible software-based fault and error injection system / GA Kanawati, NA Kanawati, JA Abraham // IEEE Trans. Comput.- vol. 44, no. 2.- Feb. 1995.- P. 248-260. [Online]. Available: http://dx.doi.org/10.1109/12.364536

98. Potyra S. Evaluating fault-tolerant system designs using faumachine / S. Potyra, V. Sieh, MD Cin // Proceedings of the 2007 Workshop on Engineering Fault Tolerant Systems ser. EFTS 07.- New York, NY.- 2007. [Online]. Available: http://doi.acm.org/10.1145/1316550.1316559

99. GML Nelson. Jaca software fault injection tool. 2009 September. [Online]. Available: http://www.ic.unicamp.br/eliane/JACA.html

100. Marinescu PD Lfi: A practical and general librarylevel fault injector / PD Marinescu, G. Candea // Proceedings of the Intl. Conference on Dependable Systems and Networks (DSN) .- Portugal.- June 2009.

101. Lfi: Library-level fault injector. [Online]. Available: http://sourceforge.net/apps/trac/lfi/

102. Carreira J. Xception: A technique for the experimental evaluation of dependability in modern computers / J. Carreira, H. Madeira, JG Silva // IEEE Trans. Softw. Eng.- vol. 24, no. 2.- Feb. 1998.- P. 125-136. [Online]. Available: http://dx.doi.org/10.1109/32.666826

103. Madeira H. Rifle: A general purpose pin-level fault injector / H. Madeira, MZ Rela, F. Moreira, JG Silva // Proceedings of the First European Dependable Computing Conference on Dependable Computing ser. EDCC-1.- London UK.- UK: Springer-Verlag.- 1994.- P. 199-216. [Online]. Available: http://dl.acm.org/citation.cfm?id=645330.649779

104. Bosio A. Lifting: A flexible open-source fault simulator / A. Bosio, G. Di Natale // Proceedings of the 2008 17th Asian Test Symposium ser. ATS 08.- Washington DC USA: IEEE Computer Society.- 2008.- P. 35-40. [Online]. Available: http://dx.doi.org/10.1109/ATS.2008.17

105. Thibodeau P. IBM's new future: Quantum computing / Patrick Thibodeau // [Online]. Available:

http://www.computerworld.com/s/article/9217669/IBM_s_new_future_Quantum_computing?source=rss_keyword_patrick+thibodeau

106. Benenti G., Casati G., Strini G. Principles of Quantum Computation and Information. Volume 1: Basic Concepts.-World Scientific.- 2004.- 256 p.

107. Vedral V., Plenio MB Basics of Quantum Computation.- 1998.- 28 p. [Online]. Available:

http://www.tfp.uni-karlsruhe.de/~cuevas/Lehre/SS04/9802065.pdf

108. Rosinger EE Basics of Quantum Computation (Part 1) .- 2004.- 87 p. [Online]. Available: http://chaos.swarthmore.edu/courses/TSG/2004d.pdf

109. Stenholm S., Suominen K.-A. Quantum approach to informatics.- A John Wiley & Sons, Inc., Publication.- 2005.- 249 p.

110. Imai Hiroshi, Hayashi Masahito. Quantum Computation and Information. From Theory to Experiment.- Springer.-2006.- 234 p.

111. Nielsen MA, Chuang IL Quantum Computation and Quantum Information.- Cambridge University Press.- 2010.- 710 p.

112. Mikio Nakahara. Quantum computing: an overview // Mathematical Aspects of Quantum Computing.- 2007.- 53 p. [Online]. Available:

http://www.worldscibooks.com/physics/6851.html

113. Whitney MG Practical Fault Tolerance for Quantum Circuits.- A dissertation submitted in partial satisfaction of the requirements for the degree of Doctor of Philosophy in Computer Science in the Graduate Division of the University of California, Berkeley.- 2009. 206 p.

114. DiVincenzo DP The Physical Implementation of Quantum Computation // IBM TJ Watson Research Center, Yorktown Heights, NY 10598 USA.- 9 p. [Online]. Available:

http://www.unifiedfieldtheories.com/0002077_DiVincenzo_Phys_Imp.pdf

115. Svore KM, Terhal BM, DiVincenzo DP Local Fault-Tolerant Quantum Computation // [Online]. Available:

http://research.microsoft.com/pubs/143764/local2005.pdf

116. Бейтсон Г. Кроки в напрямку екології розуму / Г. Бейтсон.- М .: КомКніга.- 2005.- 248 с.

117. Валієв К.А. Квантові комп'ютери: надії та реальність / К.А.Валіев, А.А.Кокін.- Іжевськ: РХД.- 2001.- 352 с.

118. Feinstein DDY Computer-Aided-Design Methods for Emerging Quantum Computing Technologies / DDY Feinstein.- BiblioLabsII.- 2011.- 184 p.

119. Hayes JP Testing for Missing-Gate Faults in Reversible Circuits / John P. Hayes, Ilia Polian, Bernd Becker // Proc. Asian Test Symposium.- Taiwain.- November, 2004.

120. Матюшкін І.В. Квантові клітинні автомати / І.В. Матюшкін // Електронний науковий журнал «витрачаються на дослідження В РОСІЇ» .- 2011.- с. 367-392. [Online]. Available: http://zhurnal.ape.relarn.ru/articles/2011/029.pdf

121. Feinstein DY Partially Redundant Logic Detection Using Symbolic Equivalence Checking in Reversible and Irreversible Logic Circuits / DY Feinstein, MA Thornton, DM Miller // Design, Automation and Test in Europe, DATE '08 .- 2008. - P. 1 378 - 1381 .

122. Nayeem NM Online Fault Detection in Reversible Logic / NM Nayeem, JE Rice // Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT) .- 2011.- P.426-434.

123. Zamani M. Fault Masking and Diagnosis in Reversible Circuits / M. Zamani, N. Farazmand, MB Tahoori // 16th IEEE European Test Symposium (ETS) .- 2011.- P.69-74.

124. Hongyan Z. Improved Fault Diagnosis for Reversible Circuits / Zhang Hongyan, R. Wille, R.Drechsler // 20th Asian Test Symposium (ATS) .- 2011.- P. 207 - 212.

125. De Vos A. The Roots of the NOT Gate / A. De Vos, S. De Baerdemacker // 42nd IEEE Intern. Symposium on Multiple-Valued Logic (ISMVL) .- -2012 P. 167 - 172.

126. Kerntopf P. Synthesis of multipurpose reversible logic gates / P. Kerntopf // Proceedings of the Euromicro Symposium on Digital System Design (DSD'02) .- 2002. - P. 259 - 266.

127. Boykin PO Reversible Fault-Tolerant Logic / Boykin, PO; Roychowdhury, VP // Proceedings of the 2005 Intern. Conf. on Dependable Systems and Networks (DSN'05) .- 2005.- P. 444 - 453.

128. Wille R .; Equivalence Checking of Reversible Circuits / R. Wille, D. Grosse, DM Miller, R. Drechsler // 39th International Symposium on Multiple-Valued Logic.- 2009.- P. 324 - 330.

129. Patel KN Fault testing for reversible circuits / KN Patel, JP Hayes, IL Markov // Proceedings of the 21st IEEE VLSI Test Symposium (VTS 03) .- 2003.- Vol. 23, Iss. 8.- P. 1 220 - 1230.

130. Chakraborty A. Synthesis of reversible circuits for testing with universal test set and C-testability of reversible iterative logic arrays / A. Chakraborty // Proc. of the 18th Intern. Conf. on VLSI Design (VLSID'05) .- 2005.- P. 249 - 254.

131. Miller DM A synthesis method for MVL reversible logic [multiple value logic] / DM Miller, GW Dueck, D. Maslov // Proceedings of the 34th Int. Symposium on Multiple-Valued Logic (ISMVL'04) .- 2004.- P. 74 - 80.

132. Chou Yao-Hsin QBIST: Quantum Built-In Self-Test for any Boolean Circuit / Yao-Hsin Chou, Sy-Yen Kuo, I-Ming Tsai // 26th IEEE VLSI Test Symposium.- 2008.- P. 261 - 266.

133. Perkowski M. Test generation and fault localization for quantum circuits / M. Perkowski, J. Biamonte, M. Lukac // Proc. of the 35th Intern. Symposium on Multiple-Valued Logic (ISMVL'05) .- 2005.- P. 62 - 68.

134. Paler A. Tomographic Testing and Validation of Probabilistic Circuits / A. Paler, A. Alaghi, I. Polian, JP Hayes // Sixteenth IEEE European Test Symposium (ETS) .- 2011.- P. 63 - 68.

135. Golubitsky O. A Study of Optimal 4-bit Reversible Toffoli Circuits and Their Synthesis / O. Golubitsky, D. Maslov // IEEE Transactions on Computers.- 2011.- P. 1-14.

136. Feinstein DY Advances in Quantum Computing Fault Tolerance and Testing / DY Feinstein, VSS Nair, MA Thornton // 10th IEEE High Assurance Systems Engineering Symposium.- 2007.- P. 369 - 370.

137. Lukac, M. Quantum Finite State Machines as Sequential Quantum Circuits / M. Lukac, M. Perkowski // 39th Intern. Symposium on Multiple-Valued Logic.- 2009.- P. 92 - 97.

138. Dorta T. Overview of FPGA-Based Multiprocessor Systems / T. Dorta, J. Jimenez, JL Martin, U. Bidarte, A. Astarloa // 2009 Intern. Conf. on Reconfigurable Computing and FPGAs.- 2009.- P. 273 - 278.

139. Hahanov V. Cyber Physical Computing for IoT-driven Services / V. Hahanov. - New York.- Springer.- 2017.- 243p.

140. Hahanov VI Qubit technology for analysis and diagnosis of digital devices / VI Hahanov, T. Bani Amer, SV Chumachenko, EI Litvinova // Electronic modeling. - 2015.- № 37 (3) .- С. 17-40.

141. Hahanov V. Quantum memory-driven computing for test synthesis / V. Hahanov, W. Gharibi, E. Litvinova, M. Liubarskyi, A. Hahanova // IEEE East-West Design and Test Symposium. - 2017. - Novi Sad, Serbia.- P. 123-128.

142. Zhong T. Nanophotonic rare-earth quantum memory with optically controlled retrieval / Tian Zhong, Jonathan M. Kindem, John G. Bartholomew at all. // Science.- 29 Sep 2017.- Vol. 357, Issue 6358.- P. 1392-1395.

143. Kim J. Photon-triggered nanowire transistors / J. Kim, H.-Ch. Lee, K.-Ho Kim at all. // Nature Nanotechnology 12.- 2017.- P. 963-968.

144. Lovat G. Room-temperature current blockade in atomically defined single-cluster junctions / G. Lovat, B. Choi, DW Paley at all. // Nature nanotechnology 12 (11) .- 1050.

145. Li C. Conformation-based signal transfer and processing at the single-molecule level / C. Li, Z. Wang, Y. Lu, X. Liu, L. Wang // Nature Nanotechnology.- 2017.- Nature Nanotechnology, 12.- P. 1071-1076.

146. Савельєв А.Я. Прикладна теорія цифрових автоматів / А.Я. Савельєв - М .: Вища. шк.- 1987.- 272 с.

147. Kharchenko V. Green IT Engineering: Concepts, Models, Complex Systems Architectures / V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.) // In the book series "Studies in Systems, Decision and Control" (SSDC) .- vol. 1.- Berlin, Heidelberg: Springer International Publishing.- 2017.

148. Kharchenko V. Green IT Engineering: Components, Networks and Systems Implementation / V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.) // In the book series "Studies in Systems, Decision and Control" (SSDC). - vol. 2.- Berlin, Heidelberg: Springer International Publishing.- 2017.

149. Almudever CG The engineering challenges in quantum computing / CG Almudever et al. // Design, Automation & Test in Europe Conference & Exhibition (DATE) .- 2017.- Lausanne.- P. 836-845.

150. Nielsen MA Quantum Computation and Quantum Information / MA Nielsen, IL Chuang. - Cambridge University Press.- 2010 року.

151. Williams RS What's Next? [The end of Moore's law] / RS Williams // Computing in Science & Engineering.- vol. 19, no. 2.- Mar.-Apr. 2017.- P. 7-13.

152. Memory-Driven Computing. [Online]. Available: https://www.labs.hpe.com/next-next/mdc

153. Singh J. Evolution in Quantum Computing / J. Singh, M. Singh // Int. Conference on System Modeling & Advancement in Research Trends (SMART) .- Moradabad.- 2016.- P. 267-270.

154. Shaikh TA Quantum Computing in Big Data Analytics: A Survey / TA Shaikh, R. Ali // IEEE Int. Conference on Computer and Information Technology (CIT) .- Nadi.- 2016.- P. 112-115.

155. Vandersypen L. 1.4 Quantum computing - the next challenge in circuit and system design / L. Vandersypen, A. van Leeuwenhoek // IEEE International Solid-State Circuits Conference (ISSCC) .- San Francisco, CA.- 2017.- P . 24-29.

156. Hahanov V. Qubit test synthesis of the functionality / V. Hahanov et al. 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM) .- Lviv.- 2017.- P. 251-255.

157. Hahanov I. Deductive qubit fault simulation / I. Hahanov, S. Chumachenko, I. Iemelianov, V. Hahanov, L. Larchenko, T. Daniyil // 14th Int. Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM) .- Lviv.- 2017.- P. 256-259.

158. Hahanov VI Qubit technologies for analysis and diagnosis of digital devices / VI Hahanov, T. Bani Amer, SV Chumachenko, EI Litvinova // Electronic Modeling.- vol. 37, no. 3.- 2015.- P. 17-40.

159. Хаханов В.І. Кубітние структури даних обчислювальних пристроїв / В. І. Хаханов, В. Гарібі, Е. І. Литвинова, А. С. Шкіль // Електронне моделювання. - 2015. - Т. 37, № 1. - С. 76-99.

160. Хаханов В.І. Кубітние технології аналізу і діагностування цифрових пристроїв / В. І. Хаханов, Т. Лазні Амер, С. В. Чумаченко, Є. І. Литвинова // Електронне моделювання. - 2015. - Т. 37, № 3. - С. 17-40.

161. Hahanov V. Cyber Physical Computing for IoT-driven Services [4. Hahanov I., Iemelianov I., Liubarskyi M., Hahanov V. Qubit Description of the Functions and Structures for Service Computing Synthesis; 5. Hahanov V., Bani Amer T., Iemelianov I., Liubarskyi M. Quantum Computing for Test Synthesis; 6. Hahanov I., Bani Amer T., Iemelianov I., Liubarskyi M., Hahanov V. QuaSim – Cloud Service for Quantum Circuits Simulation.] – New York. – Springer. – 2018.– С. 73-143.

162. Hahanov V. Matrix-Model for Diagnosing SoC HDL-Code / V. Hahanov, E. Litvinova, V. Obrizan, I. Yemelyanov // Radioelektroniks and informatics.– 2013.– №1.– P. 12-19. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

163. Хаханов В.И. Процессорные структуры для анализа Big Data / В.И. Хаханов, Е.И. Литвинова, С.В. Чумаченко, И.В. Емельянов, Т. Bani Amer // Радіоелектронні і комп'ютерні системи. – 2016.– № 6 (80).– С. 163-175. (Входить до міжнародних бібліометричних і наукометричних баз даних: наукової електронної бібліотеки eLIBRARY.RU (Російська Федерація); Index Copernicus (Польща); INSPEC IDEAS (Institution of Engineering and Technology, Великобританія); CiteFactor; Academic Keys; Infobase Index; Google Scholar).

164. Bani Amer T. Кубитная форма описания вычислительных структур / T. Bani Amer, С.В. Чумаченко, И.В. Емельянов // Радиоэлектроника и информатика. – 2016. – № 1.– С.47-52. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

165. Хаханов В.И. Синтез Q-тестов по кубитному описанию функциональностей / В.И. Хаханов, T. Bani Amer, И.В. Емельянов, М.М. Любарский // Радиоэлектроника и информатика.– 2016.– № 2(72).– С. 38-48. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

166. Хаханов В.И. Кубитный метод дедуктивного анализа неисправностей для логических схем / В.И. Хаханов, И.В. Емельянов, М.М. Любарский, С.В. Чумаченко, Е.И. Литвинова, Т. Бани Амер // Электронное моделирование.– 2017.– Том 39, № 6.– С. 59-92 [Входить до міжнародних наукометричних баз Cambridge Scientific Abstracts, Computer and Information Systems Abstracts, INIS Collection, Inspec, ВИНИТИ РАН].

167. Хмарний сервіс для тестування і верифікації систем на кристалах / Є.І. Литвинова, І.В. Ємельянов, І.В. Хаханов // Радиоэлектроника и информатика.– 2017.– №3.– С. 60-69. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

168. Емельянов И.В. Квантовые модели и облачные сервисы для анализа и диагностирования логических схем / И.В. Емельянов, М.М. Любарский, В.И. Хаханов // Радиоэлектроника и информатика. – 2017. – № 4.– С.28-45. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR)

169. Хаханов В.И. Квантовый метод синтеза тестов на основе кубитных структур данных / В.И. Хаханов, И.В. Емельянов, М.М. Любарский, С.В. Чумаченко, Е.И. Литвинова // Электронное моделирование.– 2018.– Том 40, № 1.– С. 63-80 [Входить до міжнародних наукометричних баз Cambridge Scientific Abstracts, Computer and Information Systems Abstracts, INIS Collection, Inspec, ВИНИТИ РАН].

170. Hahanov V. Cloud-driven Cyber Managing Resources / V. Hahanov, S. Chumachenko, E. Litvinova, O. Mishchenko, I. Yemelyanov, Bani Amer Tamer // Australian Journal of Scientific Reseach.– № 1(5).– 2014.– С. 202-215.

171. Hahanov V. Quantum Data Structures for SoC Component Testing / V. Hahanov, W. Gharibi, S. Chumachenko, E. Litvinova, I. Iemelianov, M. Liubarskyi // International Journal of Design, Analysis & Tools for Integrated Circuits & Systems.– Oct. 2017.– vol. 6, iss. 1.– P. 23. (Входить до міжнародної наукометричної бази EBSCO Information Services).

172. Хаханов В.И. Gartner 2017 топ-технологии: их анализ и применение / В.И. Хаханов, А.С. Мищенко, И.В. Емельянов, М.М. Любарский, Т.И. Соклакова, В.Г. Абдулаев // Paradigmata poznání. Vědecko vydavatelské centrum «Sociosféra-CZ», s.r.o., Praha, Česká republika. – 2017. – №4. – P. 33-62.

(The journal is indexed by Electronic Research Library, Russia; Research Bible, China; Scientific Indexing Services, USA; Cite Factor, Canada; General Impact Factor, India; Scientific Journal Impact Factor, India; CrossRef, USA; ORCID, USA).

173. Bani Amer T. Компьютинговые модели облачных сервисов / T. Bani Amer, В.И. Хаханов, И.В. Емельянов, М. Любарский // АСУ и приборы автоматики.– 2015.– Вип. 173.– С.48-57. (Входить до міжнародних наукометричних баз Google Scholar, Cyberleninka).

174. Bani Amer T. Синтез и анализ кубитных моделей цифровых систем / T. Bani Amer, И.В. Хаханов, Е.И. Литвинова, И.В. Емельянов // АСУ и приборы автоматики.– 2016.– Вып. 174.– С. 24-41. (Входить до міжнародних наукометричних баз Google Scholar, Cyberleninka).

175. Emelyanov I. Qubit Modeling Digital Systems / I. Emelyanov, I. Hahanova, T. Bani Amer // Proc. of IEEE East-West Design and Test Symposium. – Kiev, 26-29 September.– 2014.– P. 246-248. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

176. Hahanova A. Metric for Analyzing Big Data / A. Hahanova, Yu. Hahanova, I. Yemelyanov, V. Obrizan, D. Krulevska, M. Skorobogatiy // Матеріали XIII Міжнародної науково-технічної конференції CADSM 2015 «Досвід розробки та застосування приладо-технологічних САПР в мікроелектроніці».– 24-27 лютого, 2015.– Львів – Поляна.– С.81-83. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

177. Hahanov V. «Quantum» diagnosis and simulation of SOC / V. Hahanov, I. Yemelyanov, V. Obrizan, I. Hahanov // Proc. of XIth International Conference “Perspective Technologies And Methods In Mems Design”.– Lviv-Polyana.– 2-6 September, 2015.– P.58-60. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

178. Hahanov V. «Quantum» Processor for Digital Systems Analysis / V. Hahanov, W. Gharibi, I. Iemelianov, D. Shcherbin // Proceedings of IEEE East-

West Design & Test Symposium (EWDTS-2015).– 2015.– Batumi, Georgia.– P. 104-110. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

179. Soklakova T. Technological culture of Big Data / T. Soklakova, I. Iemelianov, T. Bani Amer, I. Hahanov // Матеріали XIII Міжнародної конференції TCSET 2016.– 23-26 лютого, 2016.– Львів – Славське.– С.549-554. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

180. Hahanov I. QuaSim – Cloud Service for Digital Circuits Simulation / I. Hahanov, W. Gharibi, I. Iemelianov, T. Bani Amer // Proceedings of IEEE East-West Design & Test Symposium.– 2016.– Yerevan, Armenia.– P. 363-370. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

181. Hahanov I. Deductive qubit fault simulation / I. Hahanov, I. Iemelianov, T. Bani Amer, D. Timofieiev, S. Chumachenko, V. Hahanov, L. Larchenko // Матеріали XIV Міжнародної науково-технічної конференції CADSM 2017 «Досвід розробки та застосування приладо-технологічних САПР в мікроелектроніці».– 21-25 лютого, 2017.– Львів – Поляна.– С.256-259. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

182. Hahanov V. Qubit test synthesis for the black box functionalities / V. Hahanov, E. Litvinova, S. Chumachenko, I. Iemelianov, M. Liubarskyi // Proc. of 5th Prague Embedded Systems Workshop.– June 29-30, 2017.– Roztoky u Prahy, Czech Republic.– P.45-51.

183. Hahanov V. Quantum Sequencer for the Minimal Test Synthesis of Black-box Functionality / V. Hahanov, S. Chumachenko, I. Hahanova, I. Iemelianov, I. Hahanov // Proc. of IEEE East-West Design and Test Symposium.– Novi Sad.– October, 2017.– P.445-450. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

184. Hahanov V. Quantum Data Structures for SoC Component Testing / V. Hahanov, W. Gharibi, K. L. Man, S. Chumachenko, E. Litvinova, I. Iemelianov, M. Liubarskyi // The International Conference on Recent Advancements in Com-

puting, IoT and Computer Engineering Technology.– The Tamkang University.– Taipei, Taiwan.– 2017.– 7 p.

185. Емельянов И.В. Models of Quantum Sequential Primitives / И.В. Емельянов, Д.И. Тимофеев, Б.Д. Ларченко // Материалы XXI Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– 25-27 апреля, 2017.– Ч. 5.– С.70-71.

186. Емельянов И. Телеметрический модуль «SherLock» для управления мобильными объектами / И. Емельянов, А. Котляров // Материалы XVII Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– Ч. 5. – 22-24 апреля, 2013.– С. 64-65.

187. Bani Amer T. Кибер-компьютинг – новый бренд IoT-рынка / T. Bani Amer, И. Емельянов // Материалы XX Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– 2016.– Ч. 5.– С.36-37.

ДОДАТОК А

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

в яких опубліковані основні наукові результати дисертації:

1. Hahanov V. Cyber Physical Computing for IoT-driven Services [4. Hahanov I., Iemelianov I., Liubarskyi M., Hahanov V. Qubit Description of the Functions and Structures for Service Computing Synthesis; 5. Hahanov V., Bani Amer T., Iemelianov I., Liubarskyi M. Quantum Computing for Test Synthesis; 6. Hahanov I., Bani Amer T., Iemelianov I., Liubarskyi M., Hahanov V. QuaSim – Cloud Service for Quantum Circuits Simulation.] – New York. – Springer. – 2018.– С. 73-143.

2. Hahanov V. Matrix-Model for Diagnosing SoC HDL-Code / V. Hahanov, E. Litvinova, V. Obrizan, I. Yemelyanov // Radioelektroniks and informatics.– 2013.– №1.– P. 12-19. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

3. Хаханов В.И. Процессорные структуры для анализа Big Data / В.И. Хаханов, Е.И. Литвинова, С.В. Чумаченко, И.В. Емельянов, Т. Bani Amer // Радіоелектронні і комп'ютерні системи. – 2016.– № 6 (80).– С. 163-175. (Входить до міжнародних бібліометричних і наукометричних баз даних: наукової електронної бібліотеки eLIBRARY.RU (Російська Федерація); Index Copernicus (Польща); INSPEC IDEAS (Institution of Engineering and Technology, Великобританія); CiteFactor; Academic Keys; Infobase Index; Google Scholar).

4. Bani Amer T. Кубитная форма описания вычислительных структур / T. Bani Amer, С.В. Чумаченко, И.В. Емельянов // Радиоэлектроника и информатика. – 2016. – № 1.– С.47-52. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

5. Хаханов В.И. Синтез Q-тестов по кубитному описанию функциональностей / В.И. Хаханов, Т. Bani Amer, И.В. Емельянов, М.М. Любарский // Ра-

диоэлектроника и информатика.– 2016.– № 2(72).– С. 38-48. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

6. Хаханов В.И. Кубитный метод дедуктивного анализа неисправностей для логических схем / В.И. Хаханов, И.В. Емельянов, М.М. Любарский, С.В. Чумаченко, Е.И. Литвинова, Т. Бани Амер // Электронное моделирование.– 2017.– Том 39, № 6.– С. 59-92 [Входить до міжнародних наукометричних баз Cambridge Scientific Abstracts, Computer and Information Systems Abstracts, INIS Collection, Inspec, ВИНИТИ РАН].

7. Хмарний сервіс для тестування і верифікації систем на кристалах / Є.І. Литвинова, І.В. Ємельянов, І.В. Хаханов // Радиоэлектроника и информатика.– 2017.– №3.– С. 60-69. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR).

8. Емельянов И.В. Квантовые модели и облачные сервисы для анализа и диагностирования логических схем / И.В. Емельянов, М.М. Любарский, В.И. Хаханов // Радиоэлектроника и информатика. – 2017. – № 4.– С.28-45. (Входить до міжнародних наукометричних баз Index Copernicus, Google Scholar, OECSP, OAJI, Scholar Steer, SIS, Cyberleninka, CiteFactor, TIU Hannover, I2OR)

9. Хаханов В.И. Квантовый метод синтеза тестов на основе кубитных структур данных / В.И. Хаханов, И.В. Емельянов, М.М. Любарский, С.В. Чумаченко, Е.И. Литвинова // Электронное моделирование.– 2018.– Том 40, № 1.– С. 63-80 [Входить до міжнародних наукометричних баз Cambridge Scientific Abstracts, Computer and Information Systems Abstracts, INIS Collection, Inspec, ВИНИТИ РАН].

10. Hahanov V. Cloud-driven Cyber Managing Resources / V. Hahanov, S. Chumachenko, E. Litvinova, O. Mishchenko, I. Yemelyanov, Bani Amer Tamer // Australian Journal of Scientific Reseach.– № 1(5).– 2014.– С. 202-215.

11. Hahanov V. Quantum Data Structures for SoC Component Testing / V. Hahanov, W. Gharibi, S. Chumachenko, E. Litvinova, I. Iemelianov, M. Liubarskyi // International Journal of Design, Analysis & Tools for Integrated Circuits & Systems.– Oct. 2017.– vol. 6, iss. 1.– P. 23. (Входить до міжнародної наукометричної бази EBSCO Information Services).

12. Хаханов В.И. Gartner 2017 топ-технологии: их анализ и применение / В.И. Хаханов, А.С. Мищенко, И.В. Емельянов, М.М. Любарский, Т.И. Соклакова, В.Г. Абдулаев // Paradigmata poznání. Vědecko vydavatelské centrum «Sociosféra-CZ», s.r.o., Praha, Česká republika. – 2017. – №4. – P. 33-62. (The journal is indexed by Electronic Research Library, Russia; Research Bible, China; Scientific Indexing Services, USA; Cite Factor, Canada; General Impact Factor, India; Scientific Journal Impact Factor, India; CrossRef, USA; ORCID, USA).

13. Bani Amer T. Компьютинговые модели облачных сервисов / T. Bani Amer, В.И. Хаханов, И.В. Емельянов, М. Любарский // АСУ и приборы автоматики.– 2015.– Вип. 173.– С.48-57. (Входить до міжнародних наукометричних баз Google Scholar, Cyberleninka).

14. Bani Amer T. Синтез и анализ кубитных моделей цифровых систем / T. Bani Amer, И.В. Хаханов, Е.И. Литвинова, И.В. Емельянов // АСУ и приборы автоматики.– 2016.– Вып. 174.– С. 24-41. (Входить до міжнародних наукометричних баз Google Scholar, Cyberleninka).

які засвідчують апробацію матеріалів дисертації:

15. Emelyanov I. Qubit Modeling Digital Systems / I. Emelyanov, I. Hahanova, T. Bani Amer // Proc. of IEEE East-West Design and Test Symposium. – Kiev, 26-29 September.– 2014.– P. 246-248. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

16. Hahanova A. Metric for Analyzing Big Data / A. Hahanova, Yu. Hahanova, I. Yemelyanov, V. Obrizan, D. Krulevska, M. Skorobogatiy // Матеріали XIII Міжнародної науково-технічної конференції CADSM 2015 «Досвід розробки

та застосування приладо-технологічних САПР в мікроелектроніці».– 24-27 лютого, 2015.– Львів – Поляна.– С.81-83. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

17. Hahanov V. «Quantum» diagnosis and simulation of SOC / V. Hahanov, I. Yemelyanov, V. Obrizan, I. Hahanov // Proc. of XIth International Conference "Perspective Technologies And Methods In Mems Design".– Lviv-Polyana.– 2-6 September, 2015.– P.58-60. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

18. Hahanov V. «Quantum» Processor for Digital Systems Analysis / V. Hahanov, W. Gharibi, I. Iemelianov, D. Shcherbin // Proceedings of IEEE East-West Design & Test Symposium (EWDTS-2015).– 2015.– Batumi, Georgia.– P. 104-110. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

19. Soklakova T. Technological culture of Big Data / T. Soklakova, I. Iemelianov, T. Bani Amer, I. Hahanov // Матеріали XIII Міжнародної конференції TCSET 2016.– 23-26 лютого, 2016.– Львів – Славське.– С.549-554. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

20. Hahanov I. QuaSim – Cloud Service for Digital Circuits Simulation / I. Hahanov, W. Gharibi, I. Iemelianov, T. Bani Amer // Proceedings of IEEE East-West Design & Test Symposium.– 2016.– Yerevan, Armenia.– P. 363-370. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

21. Hahanov I. Deductive qubit fault simulation / I. Hahanov, I. Iemelianov, T. Bani Amer, D. Timofieiev, S. Chumachenko, V. Hahanov, L. Larchenko // Матеріали XIV Міжнародної науково-технічної конференції CADSM 2017 «Досвід розробки та застосування приладо-технологічних САПР в мікроелектроніці».– 21-25 лютого, 2017.– Львів – Поляна.– С.256-259. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

22. Hahanov V. Qubit test synthesis for the black box functionalities / V. Hahanov, E. Litvinova, S. Chumachenko, I. Iemelianov, M. Liubarskyi // Proc. of 5th Prague

Embedded Systems Workshop.– June 29-30, 2017.– Roztoky u Prahy, Czech Republic.– P.45-51.

23. Hahanov V. Quantum Sequencer for the Minimal Test Synthesis of Black-box Functionality / V. Hahanov, S. Chumachenko, I. Hahanova, I. Iemelianov, I. Hahanov // Proc. of IEEE East-West Design and Test Symposium.– Novi Sad.– October, 2017.– P.445-450. (Входить до міжнародних наукометричних баз Scopus, IEEE Xplore).

24. Hahanov V. Quantum Data Structures for SoC Component Testing / V. Hahanov, W. Gharibi, K. L. Man, S. Chumachenko, E. Litvinova, I. Iemelianov, M. Liubarskyi // The International Conference on Recent Advancements in Computing, IoT and Computer Engineering Technology.– The Tamkang University.– Taipei, Taiwan.– 2017.– 7 p.

25. Емельянов И.В. Models of Quantum Sequential Primitives / И.В. Емельянов, Д.И. Тимофеев, Б.Д. Ларченко // Материалы XXI Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– 25-27 апреля, 2017.– Ч. 5.– С.70-71.

які додатково відображають наукові результати дисертації:

26. Емельянов И. Телеметрический модуль «SherLock» для управления мобильными объектами / И. Емельянов, А. Котляров // Материалы XVII Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– Ч. 5. – 22-24 апреля, 2013.– С. 64-65.

27. Bani Amer T. Кибер-компьютинг – новый бренд IoT-рынка / T. Bani Amer, И. Емельянов // Материалы XX Международного молодежного форума «Радиоэлектроника и молодежь в XXI веке».– 2016.– Ч. 5.– С.36-37.

ДОДАТОК Б

ВІДОМОСТІ ПРО АПРОБАЦІЮ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЇ

1. The 12-th IEEE East-West Design and Test Symposium (м. Київ, Україна, 2014 року) - очна участь.
2. The 13-th IEEE East-West Design & Test Symposium (м. Батумі, Грузія, 2015) - очна участь.
3. The 14-th IEEE East-West Design & Test Symposium (м. Єреван, Арменія, 2016) - очна участь.
4. The 13-th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics (м. Львів, Україна, 2015) - очна участь.
5. The XIth International Conference "Perspective Technologies and Methods in MEMS Design" (м. Львів, Україна, 2015) - очна участь.
6. The 13-th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (м. Львів-Славське, Україна, 2016) - очна участь.
7. The 14-th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics (м. Львів, Україна 2017) - очна участь.
8. The 5th Prague Embedded Systems Workshop (Roztoky u Prahy, Czech Republic 2017) - очна участь.
9. The 15-th IEEE East-West Design and Test Symposium (Novi Sad, Serbia 2017) - очна участь.
10. The International Conference on Recent Advancements in Computing, IoT and Computer Engineering Technology (The Tamkang University, Taipei, Taiwan 2017) - очна участь.
11. XXI Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI сторіччі» (м. Харків, Україна 2017) - очна участь.

ДОДАТОК В

ДОКУМЕНТИ, ЩО ПІДТВЕРДЖУЮТЬ ВПРОВАДЖЕННЯ

«ЗАТВЕРДЖУЮ»
Проректор з НМР ХНУРЕ
Рубан І.В.
"01" 01 2018 р.

АКТ

про впровадження у навчальний процес ХНУРЕ результатів дисертаційної роботи Ємельянова Ігора Валерійовича «Моделі і методи кубітного тестування цифрових пристроїв на основі memory-driven структур даних», поданої на здобуття наукового ступеня кандидата технічних наук за спеціальністю 05.13.05 – комп'ютерні системи та компоненти

Комісія у складі: зав. каф. АПОТ проф. Чумаченко С.В., доц. каф. АПОТ Шкиля О.С., проф. каф. АПОТ Литвинової Є.І. розглянула матеріали дисертаційної роботи пошукача Ємельянова І.В., які використовуються в навчальному процесі кафедри АПОТ ХНУРЕ у 2017/2018 навчальному році, і прийшла до наступного висновку.

Розроблені у дисертаційній роботі моделі і методи квантового тестування, моделювання і аналізу якості тестових наборів реалізовані у вигляді хмарних сервісів, що придатні до використання у навчальних цілях для курсів «Квантові обчислення», «Схемотехнічне проектування та моделювання НВІС», «Проектування та тестування цифрових пристроїв на ПЛІС», «Комп'ютерна логіка», а саме: 1) модель метричної взаємодії класичного та квантового комп'ютингу за параметрами паралелізму, суперпозиційності та змішування для реалізації квантового комп'ютингу в класичному виконанні за рахунок збільшення пам'яті; 2) метод невизначених коефіцієнтів для мінімізації булевих функцій для паралельного виконання логічних операцій з метою отримання двох векторів, відповідних мінімальній диз'юнктивній і кон'юнктивній нормальним формам; 3) кубітний метод пошуку дефектів для паралельного виконання операцій над двома групами векторів таблиці несправностей цифрового пристрою, отриманих після виконання діагностичного експерименту; 4) квантовий метод синтезу тестів для логічних функціональностей на основі використання булевих похідних за кубітними структурами даних для підвищення швидкодії за рахунок паралельного виконання логічних операцій; 5) квантовий метод моделювання справної поведінки на основі memory-driven кубітних структур даних, шляхом використання адресно-орієнтованих процедур аналізу цифрових пристроїв, що виключають логічні операції; 6) хмарні комп'ютингові сервіси квантового тестування, моделювання справної поведінки цифрових пристроїв є працездатними, перевіреними на представницької вибірці стандартних тестових схем з бібіліотеки конференції ISCAS. Середовище проектування: SWIFT, C ++, Verilog, Jawa Script, Google Cloud Platform.

Зав. каф. АПОТ проф. Чумаченко С.В.,
Доц. каф. АПОТ Шкиль О.С.,
Проф. каф. АПОТ Литвинова Є.І.

ALDEC
The Design Verification Company

ООО «Алдек-КТС»
ул. Космическая 23а
61145, г.Харьков, Украина

ЕГРПОУ 36374067
Р/с 26008154763 в ВАТ ХОД «Райффайзен Банк Аваль»,
МФО 380805
Тел.: (+38 057) 760-47-25

СПРАВКА

о внедрении облачных сервисов квантового тестирования и верификации SoC-компонентов в методологическое и технологическое обеспечение «ООО АЛДЕК-КТС» в целях повышения качества проектирования цифровых изделий

Внедрение в научно-практическую деятельность квантовых моделей и методов векторно-кубитного синтеза и анализа цифровых устройств, облачных компьютинговых сервисов тестирования и моделирования, разработанных на кафедре АПВТ, ХНУРЭ при 30 процентном участии соискателя Игоря Валерьевича Емельянова позволило на 7 процентов уменьшить время проектирования и выхода годной продукции на рынок.

Установлено, что разработанные квантовые модели и методы кубитно-векторного синтеза тестов, моделирования и оценки качества тестов, реализованные в виде облачных сервисов, потенциально позволяют на 5-10 процентов повысить выход годной продукции за счет online восстановления работоспособности memory-driven логических схем.

Основными результатами работы следует считать:

1) кубитный метод поиска дефектов для параллельного выполнения операций над двумя группами векторов таблицы неисправностей цифрового устройства, полученных после выполнения диагностического эксперимента.

2) квантовый метод синтеза тестов для логичеких функциональностей, на основе использования булевых производных по кубитным структурам данных для повышения быстродействия за счет параллельного выполнения логических операций.

3) квантовый метод моделирования исправного поведения на основе memory-driven кубитных структур данных, путем использования адресно-ориентированных процедур анализа цифровых устройств, исключающих логические операции.

Облачные компьютинговые сервисы квантового тестирования, моделирования исправного поведения цифровых устройств являются работоспособными, проверенными на представительной выборке стандартных тестовых схем из библиотек ISCAS. При реализации сервисов использовались технологии SWIFT, C++, Verilog, JavaScript, Google Cloud Platform. Моделирование и отладка тестов, синтезированных при помощи разработанных сервисов, проведены с применением среды верификации Aldec Riviera-PRO, а синтез и имплементация в FPGA произведены при помощи среды Xilinx Vivado.

Директор ООО «Алдек-КТС», к.т.н: _____ / Зайченко С.О. /

09.01.2018г.

ДОДАТОК Г

SOFTWARE OF CLOUD-DRIVEN QUANTUM MODELING SERVICES

```
//
// QuantumModeling
//

import Cocoa

class QVectorTableController: NSViewController {

    @IBOutlet weak var truthTableView: NSTableView!
    // MARK: Properties
    var truthTable = [([Bool], [Bool])] ()
    var elements = [Element] ()
    var circuit: Circuit?

    override func viewDidLoad () {
        super.viewDidLoad ()
        assembleTruthTable ()
      // let width = 42 + 3 + max (60, circuit! .inputPorts.count ∗
14)
      // let height = 20 ∗ Int (pow (2, Double (circuit!
.inputPorts.count)))
    // view.frame = CGRectMake (view.frame.minX, view.frame.minY,
CGFloat (width), CGFloat (height))

        truthTableView.tableColumns [0] .sizeToFit ()
        for outElem in elements {
            let tableColumn = NSTableColumn (identifier: "Output
\(outElem.index) ")// Додати індекс портів
            truthTableView.addTableColumn (tableColumn)
            tableColumn.headerCell = NSTableHeaderCell (textCell:
"Output \(outElem.index) ")
            tableColumn.sizeToFit ()
        }
        var width = CGFloat (0)
        for column in truthTableView.tableColumns {
            width + = column.width + 3
        }

        let rows = truthTable.count +1
        let height = CGFloat (min (rows ∗ 20, 500))
        view.setFrameSize (CGSize (width: width +10, Height: height))
    }

    fileprivate func assembleTruthTable () {
// guard circuit! = Nil else {
// print ( "circuit not found")
```

```
// return
//}
// let inputsNumber = circuit! .inElems.count
// let inputStatesNum = Int (pow (2, Double (inputsNumber)))
// for i in 0 .. <inputStatesNum {
// var inputValues = [Bool] (count: inputsNumber, repeatedValue:
false)
// for var j = inputsNumber-1, z = 0; j> = 0; j - = 1, z + = 1 {
// inputValues [z] = Bool ((i & (1 << j)) >> j)
//}
// let outputValues = circuit! .simulateForInput (inputValues)
// truthTable.append ((inputValues, outputValues))
//}
        let inputsNumber = circuit! .inPorts.count
        let inputStatesNum = Int (pow (2, Double (inputsNumber)))
        for i in 0.. <inputStatesNum {
            var inputValues = [Bool] (repeating: false, Count: in-
putsNumber)
            var j = inputsNumber - 1
            var z = 0
            while j> = 0 {
                inputValues [z] = Bool (((i & (1<< j)) >> j) as
NSNumber)
                z + = 1
                j - = 1
            }
            var outputValues = [Bool] ()
            for elem in elements {
                outputValues.append (circuit! .modelElement (elem,
forInputValues: inputValues))
            }
            truthTable.append ((inputValues, outputValues))
        }
    }

}

extension QVectorTableController: NSTableViewDataSource {
    func numberOfRows (in tableView: NSTableView) -> Int {
        return truthTable.count
    }
}

extension QVectorTableController: NSTableViewDelegate {

    func tableView (_ tableView: NSTableView, ViewFor tableColumn:
NSTableColumn?, Row: Int) -> NSView? {
        let cell = tableView.make (withIdentifier: "QTableCell", Own-
er: self) as! NSTableCellView
        if tableColumn! .identifier == "InputSetsColumn" {
            var str = ""
            for i in truthTable [row].0 {
                str + = String (Int (i as NSNumber))
                str + = ""
            }
```

```
            cell.textField! .stringValue = str
        } else {
            let index = tableView.column (withIdentifier: tableColumn!
.identifier) -1
            cell.textField! .stringValue = String (Int (truthTable
[row].1[Index] as NSNumber))
        }
        return cell
    }

}

//
// CircuitDelegate.swift
// QuantumModeling
//
//

import Foundation

protocol CircuitDelegate {
    // func modelingVectorChanged ()
    func onAddingElement (_ element: Element)
    func reloadData ()
}
//
// CircuitDesignerDataSource.swift
// QuantumModeling
//
//

import Cocoa

protocol CircuitDesignerDataSource {
    var circuit: CircuitView? {get set }
    var updatedFrame: NSRect { get }
    var circuitDesignerView: CircuitDesignerView! {get set }
    var tracker: NSBezierPath? {get set }
    var dragInterType: DragInteractionType { get set }
                              }
//
// CircuitDesignerView.swift
// QuantumModeling
//

import Cocoa

class CircuitDesignerView: NSView {

    var dataSource: CircuitDesignerDataSource?

    override func updateTrackingAreas () {
        super.updateTrackingAreas ()
        removeTrackingArea (trackingAreas [0])
        let trackingArea = NSTrackingArea (rect: bounds, options:
```

```
[.activeInKeyWindow, .mouseMoved], owner: self, UserInfo: nil)
        addTrackingArea (trackingArea)
    }

    override var isFlipped: Bool {
        return true
    }


    override func draw (_ dirtyRect: NSRect) {
        super.draw (dirtyRect)

        NSColor.black.setStroke ()

        for (_, Link) in dataSource! .circuit! .links {
            link.stroke ()
        }
        if dataSource? .dragInterType == .linking {
            NSColor.blue.setStroke ()
        } else if dataSource? .dragInterType == .selection {
            NSColor.black.setStroke ()
            NSColor.blue.withAlphaComponent (0.1) .setFill ()
            dataSource? .tracker? .fill ()
        }

        dataSource! .tracker? .stroke ()
    }}
//
// CircuitEditorWindowController.swift
// QuantumModeling
//
//

import Cocoa

class CircuitEditorWindowController: NSWindowController {

    // MARK: Properties
    @IBOutlet weak var toolbar: NSToolbar!
    @IBOutlet weak var simulateButton: NSToolbarItem!

    // MARK: Methods
    override func windowDidLoad () {
        super.windowDidLoad ()
        let splitViewController = contentViewController as! SplitView-
Controller
        let editorViewController = splitViewControl-
ler.childViewControllers [1] as! EditorViewController
   // NSNotificationCenter.defaultCenter (). AddObserver (editorView-
Controller, selector: Selector ( "resizeDesignerViewToFit"), name:
NSWindowDidResizeNotification, object: window)

    }

    @IBAction func collapsingSideMenu (_ sender: AnyObject) {
```

```
        let splitVC = contentViewController as! SplitViewController

        let segControl = sender as! NSSegmentedControl
        if segControl.selectedSegment == 0 {
            splitVC.splitViewItems [0] .isCollapsed =!
SplitVC.splitViewItems [0] .isCollapsed
        } else {
            splitVC.splitViewItems [2] .isCollapsed =!
SplitVC.splitViewItems [2] .isCollapsed
        }
    }

    override func prepare (for segue: NSStoryboardSegue, sender: Any?)
{
        if segue.identifier == "SimulationTableSegue" {
            let editorViewController = contentViewController as! Edi-
torViewController
            let tableViewController = segue.destinationController as!
ViewController
            tableViewController.circuit = editorViewController.circuit
        } else if segue.identifier == "QTableSegue" {
            let splitViewController = contentViewController as!
SplitViewController
            let editorViewController = splitViewControl-
ler.childViewControllers [1] as! EditorViewController
            let targerViewController = segue.destinationController as!
QVectorTableController
            targerViewController.circuit = editorViewControl-
ler.circuit
            for outPort in editorViewController.circuit.outPorts {
                if outPort.element! = nil {
                    targerViewController.elements.append (out-
Port.element!)
                }
            }
        }
    }


}

extension CircuitEditorWindowController: NSToolbarDelegate {

}

//
// CircuitTableDataSource.swift
// QuantumModeling
//
//

import Cocoa

protocol CircuitTableDataSource {
```

```
    var circuit: CircuitView? {get set }
   // var cellSize: CGSize {get}
 // var circuitTableView: CircuitTableView! {Get set}
                              }
////
//// CircuitTableView.swift
//// QuantumModeling
////
////
//
import Cocoa
//
class CircuitTableView: NSView {
//
// var dataSource: CircuitTableDataSource?
//
// override func drawRect (dirtyRect: NSRect) {
// super.drawRect (dirtyRect)
//
// let cellSize = dataSource! .cellSize
//
// // 1
// var startPoint = NSPoint (x: 0, y: bounds.height - cellSize.height-
3)
// var endPoint = NSPoint (x: bounds.width, y: startPoint.y)
// var rectangle = NSRect (x: 0, y: bounds.height - cellSize.height,
width: cellSize.width, height: cellSize.height)
// let paragraphStyle = NSMutableParagraphStyle ()
// paragraphStyle.alignment = .Center
// let attrs = [NSFontAttributeName: NSFont (name: "Helvetica", size:
cellSize.height * 0.75) !, NSParagraphStyleAttributeName: para-
graphStyle]
// NSColor.blackColor (). SetStroke ()
//
// let L: NSString = "L"
// L.drawInRect (rectangle, withAttributes: attrs)
// drawLine (startPoint, endPoint)
//
// var i = 1
// let sortedElements = dataSource! .circuit! .elements.sort ({
// if $ 0.1 is InputPortElementView && $ 1.1 is QElementView {
// return true
//} else if $ 1.1 is InputPortElementView && $ 0.1 is QElementView {
// return false
//} else {
// return $ 0.0 <$ 1.0
//}})
// for (index, _) in sortedElements {
// let shift = CGFloat (i) * cellSize.width
// let x = shift
// let y = bounds.height - cellSize.height
// let rect = NSRect (x: x, y: y, width: cellSize.width, height:
cellSize.height)
//
// let strValue: NSString = "\ (index)"
```

```
// strValue.drawInRect (rect, withAttributes: attrs)
// i + = 1
//}
//
// // 2
// let M: NSString = "M"
// startPoint.y - = cellSize.height
// endPoint.y - = cellSize.height
// rectangle.origin.y - = cellSize.height
// M.drawInRect (rectangle, withAttributes: attrs)
// drawLine (startPoint, endPoint)
//
// var vectorShift = dataSource! .circuit! .modelingVectorLength-1
// let modelingVector = dataSource! .circuit! .modelingVector
// for i in 1 .. <dataSource! .circuit! .modelingVectorLength + 1 {
// let shift = CGFloat (i) * cellSize.width
// let x = shift
// let y = bounds.height - 2 * cellSize.height
// let rect = NSRect (x: x, y: y, width: cellSize.width, height:
cellSize.height)
//
// let value = (modelingVector & (UInt64 (1) << UInt64 (vectorShift)))
>> UInt64 (vectorShift)
// vectorShift - = 1
//
// let strValue: NSString = "\ (value)"
// strValue.drawInRect (rect, withAttributes: attrs)
//
//}
//
//
// // 3
// var lowestValue = 0
// let XTableXShift = dataSource! .circuit! .inputPorts.count
// let XTableYShift = 2
//
// i = XTableXShift
// for (_, element) in sortedElements {
// if element is QElementView {
// let qElementView = element as! QElementView
// let height = qElementView.links.count
// if lowestValue <height {
// lowestValue = height
//}
// let horizontalShift = CGFloat (i + 1) * cellSize.width
// var j = XTableYShift
// let sorted = qElementView.links.sort ({$ 0.0 <$ 1.0})
// for (key, _) in sorted {
// let verticalShift = bounds.height - cellSize.height - CGFloat (j) *
cellSize.height
//
// let x = horizontalShift
// let y = verticalShift
//
// let rect = NSRect (x: x, y: y, width: cellSize.width, height:
```

```
cellSize.height)
// let value = key
// let strValue: NSString = "\ (value)"
// strValue.drawInRect (rect, withAttributes: attrs)
// j + = 1
//
//}
// i + = 1
//}
//}
//
// let X: NSString = "X"
// var vertShift = bounds.height - 2 * cellSize.height - (CGFloat
(lowestValue) * cellSize.height / 2) -cellSize.height / 2
// startPoint.y - = cellSize.height * CGFloat (lowestValue)
// endPoint.y - = cellSize.height * CGFloat (lowestValue)
// rectangle.origin.y = vertShift
// X.drawInRect (rectangle, withAttributes: attrs)
// drawLine (startPoint, endPoint)
//
// // 4
// let QTableXShift = XTableXShift
// let QTableYShift = lowestValue * Int (cellSize.height) + 2 * Int
(cellSize.height)
// lowestValue = 0
// i = QTableXShift
// for (_, element) in sortedElements {
// if element is QElementView {
// let qElementView = element as! QElementView
// let height = Int (pow (2.0, Double (qElementView.links.count)))
// if lowestValue <height {
// lowestValue = height
//}
// let x = CGFloat (i + 1) * cellSize.width
// var currentShift = height - 1
// for j in QTableYShift .. <height + QTableYShift {
//
// let y = bounds.height - CGFloat (QTableYShift) - cellSize.height -
cellSize.height * CGFloat (j-QTableYShift)
// let rect = NSRect (x: x, y: y, width: cellSize.width, height:
cellSize.height)
// let qVector = qElementView.qVector
// let value = ((qVector & (1 << UInt (currentShift))) >> UInt (cur-
rentShift))
// currentShift - = 1
// let strValue: NSString = "\ (value)"
// strValue.drawInRect (rect, withAttributes: attrs)
//
//}
// i + = 1
//}
//
//}
//
// let Q: NSString = "Q"
```

```
// vertShift = bounds.height - CGFloat (QTableYShift) - (CGFloat
(lowestValue) * cellSize.height / 2) -cellSize.height / 2
// rectangle.origin.y = vertShift
// Q.drawInRect (rectangle, withAttributes: attrs)
//
// startPoint = NSPoint (x: cellSize.width, y: bounds.height)
// endPoint = NSPoint (x: cellSize.width, y: 0)
// drawLine (startPoint, endPoint)
//
// startPoint.x + = CGFloat (XTableXShift) * cellSize.width
// endPoint.x + = CGFloat (XTableXShift) * cellSize.width
// drawLine (startPoint, endPoint)
//}
//
//
//}
//
// public func drawLine (startPoint: CGPoint, _ endPoint: CGPoint) {
// let path = NSBezierPath ()
// path.lineWidth = 2
// path.moveToPoint (startPoint)
// path.lineToPoint (endPoint)
// path.stroke ()
}
//
// CircuitView.swift
// QuantumModeling
//
//

import Cocoa

typealias HashCode = Double

protocol CircuitView: Circuit, NSCoding {

    var elemViews: [HashCode: ElementView] { get set }
    var links: [HashCode: LinkView] { get set }

    func linkElementAtHash (_ from: HashCode, ToElemAtHash to: Hash-
Code)
    func deleteLinkAtHash (_ hash: HashCode)
    func deleteQElemAtHash (_ hash: HashCode)
    func deleteInputPortAtHash (_ hash: HashCode)
    func deleteOutputPortAtHash (_ hash: HashCode)
    func addInPortElementView ()
    func addQElementView ()
    func addOutPortElementView ()
    func isReady () ->Bool
    //
// CircuitViewDocument.swift
// QuantumModeling
//
//
```

```
import Cocoa

class CircuitViewDocument: NSDocument {

    var circuit: CircuitView

    / *
    override var windowNibName: String? {
        // Override returning the nib file name of the document
        // If you need to use a subclass of NSWindowController or if
your document supports multiple NSWindowControllers, you should remove
this method and override -makeWindowControllers instead.
        return "CircuitDocument"
    }
    * /

    override func makeWindowControllers () {
        let editorWC = NSStoryboard (name: "Main", Bundle: Bun-
dle.main) .instantiateController (withIdentifier: "CircuitEditorWin-
dowController") as! CircuitEditorWindowController
        let editorVC = ((editorWC.contentViewController as! SplitView-
Controller) .childViewControllers [1] as! EditorViewController)
        editorVC.circuit = circuit
        addWindowController (editorWC)
    }

    override func windowControllerDidLoadNib (_ aController: NSWindow-
Controller) {
        super.windowControllerDidLoadNib (aController)
        // Add any code here that needs to be executed once the win-
dowController has loaded the document's window.
    }

    override func data (ofType typeName: String) Throws -> Data {
        var data = Data ()
        let circuitData = NSArchiver.archivedData (withRootObject:
circuit)
        let str = circuitData.base64EncodedString (options: NSDa-
ta.Base64EncodingOptions.lineLength64Characters)
        data = str.data (using: String.Encoding (rawValue: 0))!
        return data
        // Insert code here to write your document to data of the
specified type. If outError! = NULL, ensure that you create and set an
appropriate error when returning nil.
        // You can also choose to override -fileWrapperOfType: error
:, -writeToURL: ofType: error :, or -writeToURL: ofType: forSaveOpera-
tion: originalContentsURL: error: instead.
        // throw NSError (domain: NSOSStatusErrorDomain, code: unim-
pErr, userInfo: nil)
    }

    override func read (from data: Data, OfType typeName: String)
Throws {
        circuit = NSUnarchiver.unarchiveObject (with: data) as! Cir-
cuitView
```

```
        // Insert code here to read your document from the given data
of the specified type. If outError! = NULL, ensure that you create and
set an appropriate error when returning false.
        // You can also choose to override -readFromFileWrapper:
ofType: error: or -readFromURL: ofType: error: instead.
        // If you override either of these, you should also override -
isEntireFileLoaded to return false if the contents are lazily loaded.
    // throw NSError (domain: NSOSStatusErrorDomain, code: unimpErr,
userInfo: nil)
    }

    override class func autosavesInPlace () -> Bool {
        return true
    }

    init(CircuitView: CircuitView) {
        self.circuit = circuitView
    }

}
//
// CircuitViewDocument.swift
// QuantumModeling
//
//

import Cocoa

class CircuitViewDocument: NSDocument {

    var circuit: CircuitView

    / *
    override var windowNibName: String? {
    // Override returning the nib file name of the document
    // If you need to use a subclass of NSWindowController or if your
document supports multiple NSWindowControllers, you should remove this
method and override -makeWindowControllers instead.
    return "CircuitDocument"
    }
    * /

    override func makeWindowControllers () {
        let editorWC = NSStoryboard (name: "Main", Bundle: Bun-
dle.main) .instantiateController (withIdentifier: "CircuitEditorWin-
dowController") as! CircuitEditorWindowController
        let editorVC = ((editorWC.contentViewController as! SplitView-
Controller) .childViewControllers [1] as! EditorViewController)
        editorVC.circuit = circuit
        addWindowController (editorWC)
    }

    override func windowControllerDidLoadNib (_ aController: NSWindow-
Controller) {
        super.windowControllerDidLoadNib (aController)
```

```
        // Add any code here that needs to be executed once the win-
dowController has loaded the document's window.
    }

    override func data (ofType typeName: String) Throws -> Data {
    var data = Data ()
    let circuitData = NSArchiver.archivedData (withRootObject: cir-
cuit)
    let str = circuitData.base64EncodedString (options: NSDa-
ta.Base64EncodingOptions.lineLength64Characters)
    data = str.data (using: String.Encoding (rawValue: 0))!
    return data
    // Insert code here to write your document to data of the speci-
fied type. If outError! = NULL, ensure that you create and set an ap-
propriate error when returning nil.
    // You can also choose to override -fileWrapperOfType: error :, -
writeToURL: ofType: error :, or -writeToURL: ofType: forSaveOperation:
originalContentsURL: error: instead.
    // throw NSError (domain: NSOSStatusErrorDomain, code: unimpErr,
userInfo: nil)
    }

    override func read (from data: Data, OfType typeName: String)
Throws {
    circuit = NSUnarchiver.unarchiveObject (with: data) as! Cir-
cuitView
    // Insert code here to read your document from the given data of
the specified type. If outError! = NULL, ensure that you create and
set an appropriate error when returning false.
    // You can also choose to override -readFromFileWrapper: ofType:
error: or -readFromURL: ofType: error: instead.
    // If you override either of these, you should also override -
isEntireFileLoaded to return false if the contents are lazily loaded.
    // throw NSError (domain: NSOSStatusErrorDomain, code: unimpErr,
userInfo: nil)
    }

    override class func autosavesInPlace () -> Bool {
        return true
    }

    init(CircuitView: CircuitView) {
        self.circuit = circuitView
    }

}
//
// DictionaryExtensions.swift
// QuantumModeling
//
//

import Cocoa
```

```
enum Errors: Error {
    case unconvertable (String)
}
//
// EditorViewController.swift
// QuantumModeling
//
//

import Cocoa

enum DragInteractionType {
    case linking
    case selection
    case drag
}

class EditorViewController: NSViewController, NSTextFieldDelegate {

    // MARK: Outlets
    @IBOutlet weak var scrollView: NSScrollView!
    @IBOutlet weak var circuitDesignerView: CircuitDesignerView!
    @IBOutlet weak var heightConstraint: NSLayoutConstraint!
    @IBOutlet weak var widthConstraint: NSLayoutConstraint!

    // MARK: Properties
    var circuit: CircuitView = SimpleCircuit ()
    var circuitDesignerModel: CircuitDesignerDataSource = CircuitView-
Supply ()

    var selectedObjects = [HashCode: Selectable] ()
    var linkHash: HashCode?
    var hitHash: HashCode?
    var dragInterType = DragInteractionType.drag

    var prevDragLoc = CGPoint.zero

    // MARK: Methods
    override func viewDidLoad () {
        super.viewDidLoad ()
        view.wantsLayer = true
        view.acceptsTouchEvents = true
        circuitDesignerView.dataSource = circuitDesignerModel
        circuitDesignerModel.circuitDesignerView = circuitDesignerView
        circuit.delegate = self
        scrollView.hasVerticalScroller = true
        scrollView.hasHorizontalScroller = true
      // scrollView.borderType = .NoBorder
        circuitDesignerModel.circuit = circuit
        scrollView.translatesAutoresizingMaskIntoConstraints = false
        circuitDesignerView.translatesAutoresizingMaskIntoConstraints
= false
        let trackingArea = NSTrackingArea (rect: circuitDesign-
erView.bounds, options: [.mouseMoved, .activeInKeyWindow], owner:
```

```
self, UserInfo: nil)
        circuitDesignerView.addTrackingArea (trackingArea)
        scrollView.allowsMagnification = true
        scrollView.maxMagnification = 1
        scrollView.minMagnification = 0.5
    }

    override func viewDidLayout () {
        reloadData ()
    }

// private func resizeDesignerViewBounds () {
// guard! Circuit.elemViews.isEmpty else {return}
//// var occupiedArea = (circuit.elements.first! .1 as! ElementView)
.view.frame
//// for (_, element) in circuit.elements {
//// let elemView = element as! ElementView
//// occupiedArea = CGRectUnion (occupiedArea, elemView.view.frame)
////}
//// if occupiedArea.width> circuitDesignerView.bounds.width {
//// circuitDesignerView.bounds = CGRectMake (0, 0, occu-
piedArea.width, circuitDesignerView.bounds.height - (circuitDesign-
erView.bounds.height-occupiedArea.width))
////}
////
//// let frame = CGRectMake (circuitDesignerView.frame.minX, cir-
cuitDesignerView.frame.minY, max (circuitDesignerView.bounds.width,
occupiedArea.width + 20), max (circuitDesignerView.bounds.height, oc-
cupiedArea.height + 20) )
//// circuitDesignerView.frame = frame
// var occupiedArea = (circuit.elements.first! .1 as! ElementView)
.view.frame
// for (_, element) in circuit.elements {
// let elemView = element as! ElementView
// occupiedArea = CGRectUnion (occupiedArea, elemView.view.frame)
//}
// circuitDesignerView.frame = CGRectUnion (occupiedArea, cir-
cuitDesignerView.frame)
//}

    override func prepare (for segue: NSStoryboardSegue, sender: Any?)
{
        if segue.identifier == "QTableSegue" {
            let targerViewController = segue.destinationController as!
QVectorTableController
            targerViewController.circuit = circuit
        } else if segue.identifier == "PopoverSegue" {
            let targerViewController = segue.destinationController as!
QVectorTableController
            targerViewController.circuit = circuit
        }
    }

    // MARK: Help Functions
    fileprivate func positionElement (_ elemView: ElementView,
```

```
AtCoords coordinates: NSPoint) {
        let roundedCoords = CGPoint (x: coordinates.x- (coordi-
nates.x.truncatingRemainder (dividingBy: 10)), Y: coordinates.x- (co-
ordinates.x.truncatingRemainder (dividingBy: 10)))
        elemView.frame.origin = roundedCoords
    }

    fileprivate func moveObjectsToLoc (_ location: NSPoint) {
        let roundedLocation = NSPoint (x: location.x- (loca-
tion.x.truncatingRemainder (dividingBy: 10)), Y: location.y- (loca-
tion.y.truncatingRemainder (dividingBy: 10)))
        let dx = roundedLocation.x - prevDragLoc.x
        let dy = roundedLocation.y - prevDragLoc.y
        moveObjctsByDelta (dx: dx, dy: dy)
// for (_, obj) in selectedObjects {
// if let elementView = obj as? ElementView {
// elementView.frame.origin.x + = roundedLocation.x - prevDragLoc.x
// elementView.frame.origin.y + = roundedLocation.y - prevDragLoc.y
//
//} else {
//
//}
//}
        prevDragLoc = roundedLocation
    }
// circuitDesignerView.setNeedsDisplayInRect (circuitDesign-
erView.bounds)
    }

    override func mouseDragged (with theEvent: NSEvent) {
  // elementInputField? .removeFromSuperview ()
      // let newLocation = view.convertPoint (theEv-
ent.locationInWindow, toView: circuitDesignerView!)
        // let newLocation = NSApplication.sharedApplication (). Key-
Window! .contentView! .convertPoint (theEvent.locationInWindow,
toView: circuitDesignerView)
        let splitViewController = parent as! SplitViewController
        let newLocation = splitViewController.view.convert (theEv-
ent.locationInWindow, to: circuitDesignerView)
        if dragInterType == .drag {
            moveObjectsToLoc (newLocation)
        } else if dragInterType == .linking {
            circuitDesignerModel.dragInterType = .linking
            if ! (SelectedObjects.first ?.1 is OutputPortElementView)
{

                let mouseTracker = NSBezierPath ()
                mouseTracker.lineWidth = 2
                mouseTracker.move (to: prevDragLoc)
                mouseTracker.line (to: newLocation)
                circuitDesignerModel.tracker = mouseTracker

                let hit = hitObject (theEvent.locationInWindow)
                if hit! = nil &&! (Hit !.1 is InputPortElementView)
&&! (Hit !.1 is LinkView) {
```

```
                    hitHash = hit !.0
                    circuit.elemViews [hitHash!] !. highlighted = true
// circuit.linkElementAtHash (selectedObjects.first! .0, toElemAtHash:
hit! .0)
// linkHash = hit! .0 / selectedObjects.first! .0
// circuitDesignerModel.tracker = nil
                } else {
                    if hitHash! = nil {
                        circuit.elemViews [hitHash!] !. highlighted =
false
                        hitHash = nil
                    }
                 // circuitDesignerModel.tracker = Tracker (fromEle-
ment: selectedObjects.first! .1 as! ElementView, cursor: newLocation)
                }
            }
        } else {
            circuitDesignerModel.dragInterType = .selection
            let roundedPrev = CGPoint (x: Double (Int (prevDragLoc.x))
+0.5, Y: Double (Int (prevDragLoc.y)) +0.5)
            let roundedNew = CGPoint (x: Double (Int (newLocation.x))
+0.5, Y: Double (Int (newLocation.y)) +0.5)
            let selectionArea = CGRect (x: roundedPrev.x, y: round-
edPrev.y, width: roundedNew.x-roundedPrev.x, height: roundedNew.y-
roundedPrev.y)
            let selectionAreaPath = NSBezierPath (rect: selectionArea)
            selectionAreaPath.lineWidth = 1
            circuitDesignerModel.tracker = selectionAreaPath
            for (Hash, elemView) in circuit.elemViews {
                if elemView.frame.intersects (selectionArea) {
                    selectObject ((hash, elemView))
                } else {
                    deselectObjectAtHash (hash)
                }
            }

        }
        reloadData ()
    }

    override func mouseUp (with theEvent: NSEvent) {
        linkHash = nil
        circuitDesignerModel.tracker = nil
        resizeDesignerViewToFit ()
        if hitHash! = nil {
            circuit.elemViews [hitHash!] !. highlighted = false
            circuit.linkElementAtHash (selectedObjects.first !.0,
ToElemAtHash: hitHash!)
            hitHash = nil
        }
// let splitViewController = parentViewController as! SplitViewCon-
troller
// let newLocation = splitViewController.view.convertPoint (theEv-
ent.locationInWindow, toView: circuitDesignerView)
// for (_, obj) in selectedObjects {
```

```
// if let elemView = obj as? ElementView {
// if elemView.frame.minX <0 {
// moveObjectsToLoc (CGPointMake (100, newLocation.y))
//}
// if elemView.frame.minY <0 {
// moveObjectsToLoc (CGPointMake (newLocation.x, 100))
//}
//}
//}

        reloadData ()
    }

    // MARK: Handling Keyboard
    override func keyDown (with theEvent: NSEvent) {

        if Int (theEvent.characters! .unicodeScalars.first! .value) ==
NSDeleteCharacter {
            for (Hash, obj) in selectedObjects {
                if obj is QElementView {
                    circuit.deleteQElemAtHash (hash)
                } else if obj is InputPortElementView {
                    circuit.deleteInputPortAtHash (hash)
                } else if obj is OutputPortElementView {
                    circuit.deleteOutputPortAtHash (hash)
                } else if obj is LinkView {
                    circuit.deleteLinkAtHash (hash)
                }
            }
            deselectObjects ()
            reloadData ()

        }
    }
}
```