ISSN 2522-9818 (print)
ISSN 2524-2296 (online) *Innovative technologies and scientific solutions for industries. 2023. No. 3 (25)*

УДК 681.326 DOI: https://doi.org/10.30837/ITSSI.2023.25.098

О. Шкіль, М. Мірошник, Д. Рахліс, О. Тріфанов

СТРУКТУРИ ДАНИХ ДЛЯ ДЕДУКТИВНОГО МОДЕЛЮВАННЯ УМОВНИХ ОПЕРАТОРІВ *HDL*

Предметом дослідження є кубітно-векторні моделі опису комбінаційних схем і процедури дедуктивного моделювання несправностей на основі цих моделей. **Об'єкт дослідження** – процеси побудови діагностичного забезпечення цифрових систем на основі використання векторних кубітних даних. **Мета роботи** – підвищення швидкості та якості створення діагностичного забезпечення цифрових пристроїв способом розроблення оптимальних структур даних і процедур дедуктивного моделювання несправностей на основі структурно-функціональних моделей комбінаційних схем. У статті вирішуються такі **завдання:** аналіз паралельних і послідовних умовних операторів мов опису апаратури та схемних структур, у які вони синтезуються; розроблення процедури формування таблиць істинності (*Q*-векторів) схемних структур, поданих мовами опису апаратури; створення універсальної структури даних для кубічного та аналітичного дедуктивного моделювання несправностей; удосконалення векторних моделей кубітного подання структур і компонентів цифрових систем на основі адресного кодування вхідних сигналів для підвищення технологічності та швидкодії моделювання несправностей; розроблення процедури отримання булевих похідних способом переставлень розрядів таблиць істинності (*Q*-векторів) та використання операції *XOR*; створення структури даних для дедуктивного моделювання несправностей на основі кубічного подання компонентів цифрових схем. Використовуються такі **методи**: дедуктивне, кубічне, дедуктивно-паралельне моделювання несправностей, моделювання несправностей за дедуктивними *Q*-векторами. Здобуто такі **результати**: показано еквівалентність паралельних і послідовних умовних операторів, а також їх схемна реалізація у вигляді мультиплексорів; запропоновано спосіб отримання таблиць істинності синтезованої схемної структури за допомогою *TestBench (Xilinx ISE)*; розглянуто різні технології та структури даних дедуктивного моделювання несправностей для табличного, аналітичного й кубітного способів опису цифрових схем; описано програмну реалізацію кубічного дедуктивного моделювання несправностей і показано еквівалентність отриманих результатів для схем мультиплексорів *MUX* 2-в-1 та *MUX* 4-в-1 з використанням програмного продукту *DCP*. **Висновки:** запропоновано новий *Q*-метод інтерпретативного моделювання несправностей цифрових схем, що визначається застосуванням компактних *Q*-векторів замість таблиць істинності; це дає змогу суттєво підвищити швидкодію аналізу завдяки адресному формуванню виходів функціональних примітивів і зменшити обсяги структур даних, що практично робить метод конкурентоспроможним із технологіями компілятивного моделювання.

Ключові слова: *HDL*-модель; структурно-функціональна модель; дедуктивне моделювання несправностей; кубічне покриття; таблиця істинності; *Q*-вектор.

Вступ

Процес проєктування високотехнологічних комп'ютерних систем стає дедалі більш складним. Це не може не призвести до зростання кількості відмов таких систем, причиною яких є відсутність верифікації та сумісності встановлених компонентів. Це пояснюється тенденцією обмеження інвестицій на сервісне діагностичне обслуговування комп'ютерних систем на стадіях проєктування та експлуатації. Отже, очевидною є необхідність розроблення та впровадження тестопридатного програмно-апаратного забезпечення цифрових систем на функціональному, алгоритмічному та системному рівнях. Для цього потрібно вирішувати завдання проєктування діагностичного забезпечення, а саме тестів, алгоритмів контролю та пошуку дефектів, системної верифікації сумісності компонентів. Виконання зазначених завдань потребує подальшого розвитку та вдосконалення методів і засобів моделювання справної поведінки та несправностей комп'ютерної системи на різних рівнях абстракції.

Мови опису апаратних засобів (*hardware description language, HDL*), зокрема *VHDL*, є потужним засобом для прискореного розроблення та верифікації складних цифрових систем. Мова *VHDL* може використовуватися на етапах проєктування, верифікації, синтезу й тестування апаратури так само, як і для передачі інформації про проєкт, модифікації та супроводу. Але сучасні промислові системи автоматизованого проєктування (САПР) цифрових схем на основі мов опису апаратури, як-от *Xilinx, Quartus, ModelSim* тощо, не містять інструментальних засобів розроблення діагностичного забезпечення

ISSN 2522-9818 (print)
Сучасний стан наукових досліджень та технологій в промисловості. 2023. № 3 (25) ISSN 2524-2296 (online)

систем, що проєктуються. Отже, розроблення методів побудови діагностичного забезпечення за умови автоматизованого проєктування цифрових систем, наданих мовами опису апаратури, є актуальним і технологічно значущим завданням.

HDL-модель – це опис алгоритмічною мовою з формалізацією мовних конструкцій. Цей опис за формальними правилами перетворюється (синтезується) на структурно-функціональні схемні моделі різного рівня ієрархії. Між кодом мовою опису апаратури та відповідної йому цифрової схеми в обраній технологічній платформі "стоїть" система синтезу, тобто комплекс програмних засобів, що перетворює *HDL*-модель на деяку схемну реалізацію за визначеними правилами. Система синтезу однозначно перетворює вихідний *VHDL*-опис у схемну реалізацію рівня регістрових передач незалежно від обраної технології. Правила перетворення операторних конструкцій стандартизовані (прикладом може бути стандарт *1076.6-1999 IEEE Standard for VHDL RTL Synthesis* та *1364.1-2002 IEEE Standard for Verilog RTL Synthesis*, а також їх пізніші версії). Можливі налаштування систем синтезу не впливають на реалізацію вихідного стандарту. Отже, припустимо, що в межах стандарту стандартизовані (шаблонні) фрагменти *HDL*-коду перетворюються на стандартні схемні реалізації.

Так, *HDL*-модель – це фактично схема, і до неї можна застосувати методи схемного аналізу, синтезу тестів, моделювання несправностей і побудови алгоритмів пошуку дефектів, що досить добре розроблені для цифрових схем.

Методи моделювання несправностей можна розрізняти таким чином: одиночне, паралельне, дедуктивне, кубічне та спільне моделювання [1]. Найбільш ефективним на практиці виявилося запропоноване ще Армстронгом [2] дедуктивне моделювання несправностей, яке полягає в одночасному обробленні всіх одиночних константних несправностей схеми на одному вхідному наборі та виділення в цьому разі підмножини дефектів, що перевіряються. Метод орієнтований на вентильний рівень опису моделі проєктованого об’єкта в базисі І-АБО-НІ. Необхідність отримання аналітичних формул для кожного типу примітивного елемента та великі витрати пам’яті з метою зберігання списків несправностей ускладнюють практичну реалізацію методу. Для усунення цієї проблеми широко використовуються модифікації дедуктивного методу, що ґрунтуються на кубічному поданні законів функціювання примітивних елементів структурно-функціональної моделі (СФМ) цифрової схеми [3, 4]. Усі наявні проблеми методів моделювання несправностей можна описати таким чином [5]: висока обчислювальна складність алгоритмів як на вентильному рівні, так і на рівні регістрових передач; значний обсяг структур даних для аналізу цифрових систем, що негативно впливає на продуктивність цих методів; складність аналізу та синтезу тестів для логіки великої розмірності. Отже, технології побудови тестів та визначення їх якості, що передбачають моделювання несправностей, мають відповідати вимогам швидкодії, уніфікації та простоти програмної реалізації для схем будь-якої розмірності та на будь-якому рівні подання.

Аналіз останніх досліджень і публікацій

Моделювання несправностей – добре вивчена тема в наукових колах і промисловості. Але, незважаючи на це, дослідження в цій сфері не припиняються. Так, у статті [6] запропоновано ефективний дедуктивний симулятор для малих дефектів затримки. Запропоновано окремі стратегії моделювання несправностей для різних дефектів. Експериментальні результати демонструють ефективність запропонованого симулятора, який може додатково прискорити моделювання несправності. Він досягає прискорення в середньому в 28,3 раза порівняно з методом послідовного моделювання та в середньому в 3,92 раза порівняно з методом трасування критичного шляху. Автори роботи [7] запропонували повністю автоматизовану систему моделювання несправностей *MetaFI*, яка підтримує різні моделі постійних і тимчасових несправностей. Цей фреймворк дає змогу додавати до системи, що тестується, константні одиночні несправності, перехідні збої, а також збої синхронізації. Моделювання несправності виконується на *RTL*-рівні. У статті [8] запропоновано підхід на основі *FPGA* для генерації тестових шаблонів і перевірки помилок комбінаційних схем. Замість моделювання автори використовують поняття емуляції для виявлення несправностей у схемі за допомогою апаратного забезпечення *FPGA*. Крім того, проблема незначної кількості вхідних і вихідних портів, доступних на платі *FPGA*, була подолана завдяки підходу мультиплексування перемикачів і кнопок.

Одним із популярних напрямів досліджень є різні методи діагностики несправностей

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2023. No. 3 (25)

за допомогою машинного навчання. Так, автори праці [9] пропонують метод, що не потребує ані моделювання несправностей, ані зберігання словників несправностей. Вихідні реакції схеми, що діагностується, застосовуються до навченої нейронної мережі, і зрештою визначаються потенційні помилки. У статті також досліджується генерація даних, які використовуються для навчання нейронної мережі. Ефективність запропонованого методу підтверджується результатами експериментів для еталонних схем. А в роботі [10] авторами запропоновано методичне навчання штучної нейронної мережі для керування автоматичним генератором тестових шаблонів. У статті [11] подано повністю автоматизований метод тестування з'єднань *FPGA*, що фокусується на виявленні та діагностиці окремої несправності за допомогою коду Уолша. Він дає змогу однозначно ідентифікувати й локалізувати будь-яку окрему несправність. Крім того, цей метод не потребує будь-яких зовнішніх конфігурацій апаратного забезпечення для визначення фактичного місця несправності. Автори роботи [12] розробили метод дедуктивного моделювання несправностей способом побудови дедуктивних формул на основі аналітичного отримання булевих похідних і реалізації дедуктивних формул у відповідні схемні структури для апаратного моделювання несправностей, що в сотні разів прискорює процес моделювання. У праці [13] запропонований метод дедуктивного моделювання несправностей на основі структурно-функціональної моделі, де примітивні елементи подані таблицями істинності (*Q*-векторами), але не наведений спосіб отримання *Q*-векторів для складних функціональних елементів цифрових систем.

Огляд публікацій показує актуальність створення нових технологій моделювання несправностей у межах загального циклу автоматизованого проєктування цифрових систем. **Метою дослідження** є підвищення швидкості та якості створення діагностичного забезпечення цифрових пристроїв способом розроблення оптимальних структур даних і процедур дедуктивного моделювання несправностей на основі структурно-функціональних моделей комбінаційних схем, наданих мовами опису апаратури.

Для досягнення поставленої мети необхідно виконати такі завдання:

1) провести аналіз паралельних і послідовних умовних операторів мов опису апаратури та схемних структур, у які вони синтезуються;

2) розробити процедури формування таблиць істинності (*Q*-векторів) схемних структур, поданих умовними операторами мов опису апаратури;

3) створити універсальні структури даних для кубічного, аналітичного та кубітного дедуктивного моделювання несправностей;

4) удосконалити векторні моделі кубітного подання структур і компонентів цифрових систем на основі адресного кодування вхідних сигналів для підвищення технологічності та швидкодії моделювання несправностей;

5) розробити процедури отримання булевих похідних способом переставлення розрядів таблиць істинності (*Q*-векторів) та використання операції *XOR*;

6) створити структури даних для дедуктивного моделювання несправностей на основі кубітного подання компонентів цифрових схем.

Побудова структурно-функціональних *VHDL*-моделей

У мові опису апаратури *VHDL* використовуються три стилі опису проєктів: потоковий (*data flow*), поведінковий (*behavioral*) і структурний (*structural*). Останній стиль є комбінацією компонентів, описаних одним із двох інших стилів. Потоковий стиль опису передбачає використання тільки паралельних операторів, а поведінковий – послідовних у середині процесу. Усі паралельні оператори виконуються одночасно під час одного й того самого циклу моделювання. Результати паралельних операторів доступні для інших операторів після завершення поточного циклу моделювання. До паралельних *VHDL*-операторів належать: оператор призначення сигналів (<=), умовний оператор призначення сигналів (*<=/when/else*), оператор призначення сигналів на вибір (*with/select/*<=). До послідовних *VHDL*-операторів належать: оператори призначення сигналів і змінних (<= / :=), умовний оператор призначення сигналів (*if/then/else*), оператор призначення сигналів на вибір (*case*), цикли (*while/for*). Паралельні та послідовні умовні оператори еквівалентні й застосовуються залежно від уподобань проєктувальника [14] як для реалізації комбінаційних схем, так і послідовнісних. На рис. 1 відтворено еквівалентність паралельного (*when/else*) та послідовного умовних операторів (*if*).

Аналогічно можна перейти від паралельного оператора вибору (*with/select*) до послідовного (*case*) (рис. 2).

Сучасний стан наукових досліджень та технологій в промисловості. 2023. № 3 (25)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

```
entity example_condition is
port ( x1, x2, x3, x4 : in bit; F : out bit;
       condition : in bit_vector (1 downto 0) );
end example_condition;
```

```
architecture first of example_condition is
begin
   F <= xl when condition = "00" else
        x2 when condition = "01" else
        x3 when condition = "10" else
        x4;
end first;
```

≡

```
architecture second of example_condition is
begin
 process (x1, x2, x3, x4, condition )
 begin
    if (condition = "00")      then F <= x1;
    elsif (condition = "01") then F <= x2;
    elsif (condition = "10") then F <= x3;
    else
       F <= x4;
    end if;
 end process;
end second;
```

Рис. 1. Еквівалентність умовних *VHDL*-операторів

```
architecture first1 of example_condition is
begin
  with condition select
    F <= xl when "00",
         x2 when "01",
         x3 when "10",
         x4 when others;
end first1;
```

≡

```
architecture second1 of example_condition is
begin
 process (x1, x2, x3, x4, condition )
 begin
  case condition is
     when "00" => F <= x1;
     when "01" => F <= x2;
     when "10" => F <= x3;
     when others => F <= x4;
  end case;
 end process;
end second1;
```

Рис. 2. Еквівалентність *VHDL*-операторів вибору

Важко уявити *VHDL*-модель без використання умовних операторів чи операторів вибору. Саме їх апаратна реалізація викликає найбільший інтерес. Розглянемо синтез умовного послідовного оператора *IF* зі складним умовним виразом:

```
architecture Behavioral of test1 is begin
process (a0,a1,a2,a3,A,B) begin
    if ((a3 and a2 and a1 and a0)='1') then y<=A;
   else                                  y<=B;
   end if;
end process;
end Behavioral;
```

На рис. 3 наведений результат синтезу вказаної мовної конструкції.

На цьому рисунку видно, що умовний послідовний оператор *IF* зі складним умовним виразом синтезований у комбінаційну схему, що містить мультиплексор *MUX* 2-в-1.

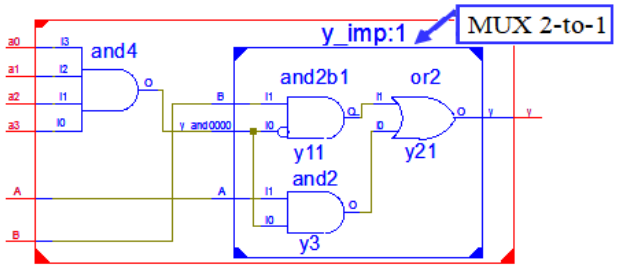


Рис. 3. *RTL*-схема для оператора *IF* зі складним умовним виразом

Розглянемо синтез послідовного оператора багатопозиційного вибору *CASE* зі складним умовним виразом:

```
architecture Beh of MUX is
begin
process(A,S) begin
      case S is
            when "00" => y<= A(0);
            when "01" => y<= A(1);
```

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2023. No. 3 (25)

```
            when "10" => y<= A(2);
            when "11" => y<= A(3);
            when others => y<='X';
        end case;
    end process;
    end Beh;
```

На рис. 4 видно, що послідовний оператор *CASE* зі складним умовним виразом синтезований у комбінаційну схему, що містить мультиплексор *MUX* 4-в-1.

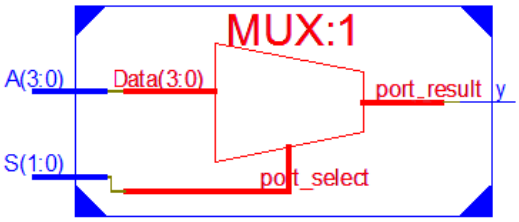


Рис. 4. *RTL*-схема для оператора *CASE* зі складним умовним виразом

З огляду на здобуті результати можна зробити висновки, що умовні оператори мови опису апаратури перетворюються в схемні конструкції мультиплексорів *MUX* 2-в-1 й *MUX* 4-в-1. Отже, необхідно розглянути різні методи моделювання несправностей саме для таких схем.

Декілька слів треба сказати про моделі несправностей за умови дедуктивного моделювання. Можливі помилки проєктування в *HDL*-моделях визначаються стилем опису *HDL*-коду. Під помилкою проєктування розуміється помилка в *HDL*-операторі, яка не належить до класу синтаксичних і порушує алгоритм функціювання моделі пристрою, що проєктується. Типи помилок проєктування: "заміна оператора" (логічного чи арифметичного) та "заміна операнда" (в операторі призначення або умовному) [15]. Але якщо переходимо до структурно-функціональної моделі пристрою, яка складається з ПЕ та ліній схеми, то основними типами несправностей, які історично розглядаються в діагностиці, є одиночні константні несправності на лініях структурно-функціональної моделі. Тому надалі будемо розглядати одиночні константні несправності ліній схеми, якщо не обумовлено інше.

Після синтезу *VHDL*-моделі у *FPGA*-схему інструментальними засобами САПР (наприклад, *Xilinx*) можна отримати її схемну реалізацію на *LUT (Look-up table* – таблиця відповідності, кожна комірка якої здатна зберігати значення вихідного рядка таблиці істинності, тобто *Q*-вектора). Це дає змогу отримати повну таблицю істинності всієї схеми, що допомагає ефективно використовувати кубітні методи аналізу цифрових схем.

Кубічне дедуктивне моделювання несправностей схем

Найпростішим неявним методом моделювання несправностей є дедуктивний метод (*deductive method*), або метод Армстронга. Кубічне моделювання несправностей є машинно-орієнтованим алгоритмом в інтерпретативних системах моделювання. Воно ґрунтується на поданні опису примітивного едементу (ПЕ) структурно-функціональної моделі кубічними покриттями (КП) для комбінаційних схем в алфавіті $A^1 = \{0,\ 1,\ U,\ X\}$, а для послідовнісних схем в алфавіті A^2 [4].

Розглянемо приклад дедуктивного моделювання несправностей для мультиплексора *MUX* 2-в-1, функція якого має вигляд $y = x_1 x_3 \vee x_2 \overline{x_3}$, тобто $y = \sum m(2,\ 5,\ 6,\ 7)$. На рис. 5 наведено схемну познаку, вентильну схему, таблицю істинності (ТІ) та кубічне покриття зазначеної схеми.

Для отриманої схеми виконаємо ранжування способом нумерації її ліній, а потім – дедуктивне моделювання несправностей для вичерпного тесту – $2^n = 8$ наборів, де $n = 3$ – це кількість входів.

Рис. 5. Структурно-функціональна модель мультиплексора *MUX* 2-в-1

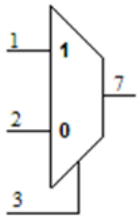
Сучасний стан наукових досліджень та технологій в промисловості. 2023. № 3 (25)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

Результати дедуктивного моделювання несправностей можуть подаватися як у вигляді списків несправностей, так і у вигляді таблиць несправностей (ТН), рядки яких відповідають вхідним наборам, а стовпці – входам та виходам схеми. Якщо на розглянутому наборі (рядок ТН) перевіряється несправність типу "0" на лінії, то у відповідному стовпці ставиться 0, а якщо перевіряється несправність типу "1" на лінії, то у відповідному стовпці ставиться 1.

Подамо різну послідовність наборів у ТІ за умови, який розряд вважається молодшим у двійковому наборі. Якщо *logic_vector* у *HDL*-моделі описується *downto*, то в молодшому розряді стоїть 3-я лінія, а якщо *to*, то відповідно 1-а лінія (рис. 6).

Розглянемо процес побудови структурно-функціональної моделі та кубічного моделювання комбінаційної схеми *MUX* 4-в-1, що складається з функціональних елементів *MUX* 2-в-1 (рис. 7). Алгоритм кубічного моделювання несправностей реалізований у комп'ютерній програмі *DCP (Deductive Circuit Processor)*, багаторічне використання якої в навчальному процесі Харківського національного університету радіоелектроніки підтверджує працездатність зазначеного методу.

На рис. 8 подані результати моделювання фрагмента вичерпного тесту схеми *MUX* 4-в-1 в програмі кубічного моделювання *DCP*. Важливо зазначити, що повнота наведеного фрагменту тесту дорівнює 100 %. Для поданого на рис. 7 набору 111101 результати ручного й комп'ютерного моделювання збігаються.



ТН для downto

	123 7	1	2	3	7
0	000 0	-	1	-	1
1	001 0	1	-	-	1
2	010 1	-	0	1	0
3	011 0	1	-	0	1
4	100 0	-	1	1	1
5	101 1	0	-	0	0
6	110 1	-	0	-	0
7	111 1	0	-	-	0

ТН для to

	321 7	1	2	3	7
0	000 0	-	1	-	1
4	001 0	-	1	1	1
2	010 1	-	0	1	0
6	011 1	-	0	-	0
1	100 0	1	-	-	1
5	101 1	0	-	0	0
3	110 0	1	-	0	1
7	111 1	0	-	-	0

ТН для КП

	123 7	1	2	3	7
0	000 0	-	1	-	1
1	001 0	1	-	-	1
3	011 0	1	-	0	1
4	100 0	-	1	1	1
2	010 1	-	0	1	0
5	101 1	0	-	0	0
6	110 1	-	0	-	0
7	111 1	0	-	-	0

Рис. 6. Варіанти ТН для різних варіантів СФМ мультиплексора *MUX* 2-в-1

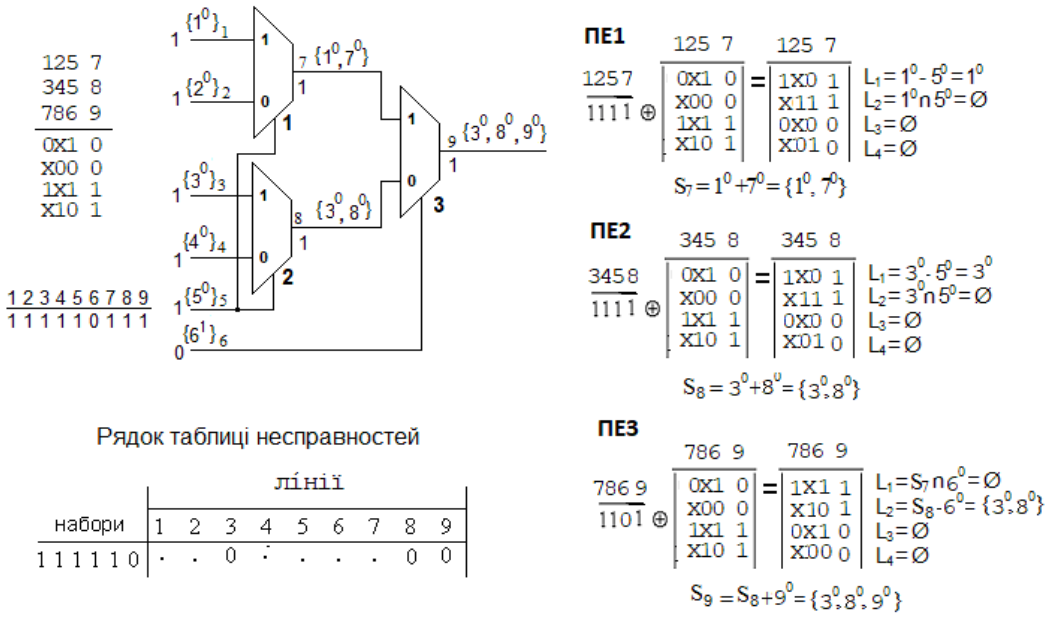


Рис. 7. Справне та несправне кубічне моделювання

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)
Innovative technologies and scientific solutions for industries. 2023. No. 3 (25)

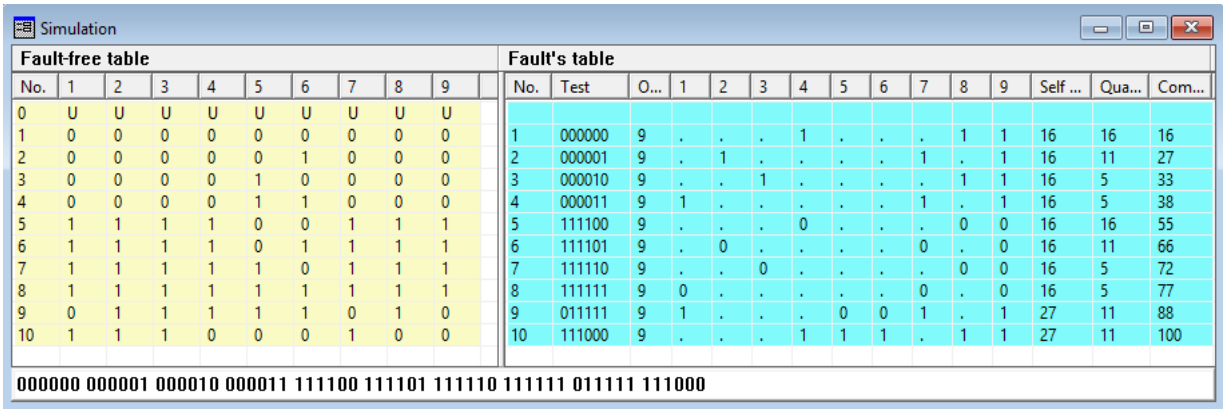


Рис. 8. Результати моделювання фрагмента тесту схеми *MUX* 4-в-1 в програмі кубічного моделювання *DCP*

Булеві похідні для графічного подання логічних функцій

Одним із наочних способів взяття булевих похідних є подання багаторівневих логічних функцій графічним способом з подальшим отриманням дужкових форм (ДФ) [16].

Розглянемо приклад отримання дужкових форм булевих функцій (БФ) на основі ДНФ. Наприклад, для функції трьох змінних $f(x_1,x_2,x_3)=\sum m(2,5,6,7)$ ДНФ має вигляд $f=x_1\times x_3\vee x_2\times\overline{x_3}$. Абсолютно аналогічні викладки можна застосувати й до КНФ. На рис. 9 показана графічна інтерпретація отримання дужкової форми булевої функції.

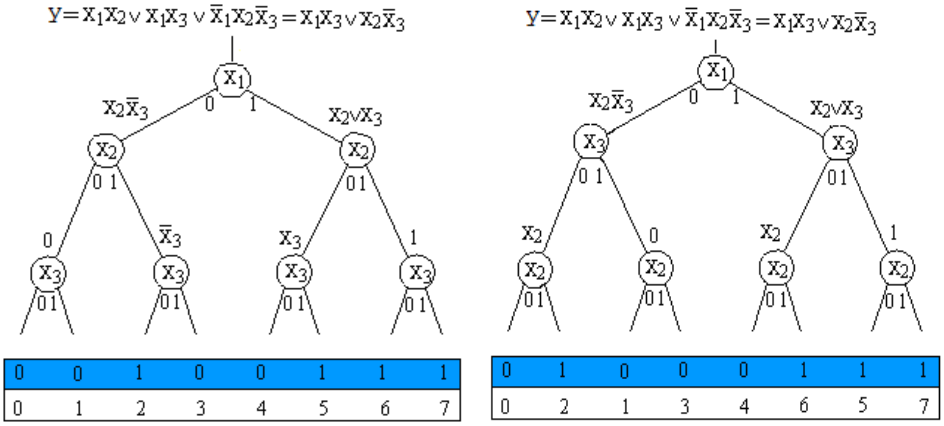


Рис. 9. Графова форма булевої функції мультиплексора *MUX* 2-в-1

Змінні в ярусах графа розташовуються в природному порядку (x_1,x_2,x_3) – від старших розрядів до молодших. У нижній частині рисунка в рамці показано таблицю істинності цієї функції (блакитне поле), а нижче зображено відповідні номери наборів у десятковому еквіваленті. Двійковий еквівалент шляху (складений з 0 і 1, що помічають вхідні ребра вузлів графа) від вершини верхнього ярусу графа до відповідного значення функції (блакитне поле) вказує на номер відповідного двійкового набору (порядок розрядів – x_1,x_2,x_3). Наприклад, шлях 0 1 1 (крізь вершини x_1,x_2,x_3) відповідає номеру набору 3, що підтверджується його десятковим еквівалентом. Залежно від порядку розміщення змінних у ярусах графа можна отримати різні варіанти скобкових форм для однієї і тієї самої булевої функції. Правило перестановки значення функції в таблиці істинності: для кожної вершини верхнього з ярусів, що переставляються, змінюються місцями між собою значення функцій у підграфах на "внутрішніх" ребрах вершин нижнього з ярусів, що переставляються. У прикладі під час переставлення ярусів 2 і 3 змінюються місцями набори (1, 2) і (5, 6).

Сучасний стан наукових досліджень та технологій в промисловості. 2023. № 3 (25)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

Графічне взяття булевої похідної для змінної нижнього ярусу графа визначається порівнянням парних і непарних позицій у поданій таблиці істинності, якщо відповідна змінна стоїть у нижньому ярусі графа. Якщо парні та непарні позиції в таблиці істинності розрізняються (0 1 або 1 0), то відповідна гілка дерева впливає на булеву змінну (на рисунку позначається "+"). Далі на другому ярусі відповідна змінна записується з інверсією, якщо "+" іде з лівого ребра, та без інверсії, якщо з правого.

Розглянемо аналітичне та графічне взяття булевих змінних для функції $f(x_1,x_2,x_3)=\left(x_1x_3 \vee x_2\overline{x_3}\right)$.

Графічне взяття булевих похідних наведено на рис. 10.

$$f_1 = df/dx_1 = \left(0\times x_3 \vee x_2\overline{x_3}\right)\oplus\left(1\times x_3 \vee x_2\overline{x_3}\right) = x_2\overline{x_3}\oplus\left(x_2 \vee x_3\right) =$$

$$= x_2\overline{x_3}\times\overline{\left(x_2 \vee x_3\right)} \vee \overline{x_2\overline{x_3}}\times\left(x_2 \vee x_3\right) = x_2\overline{x_3}\times\overline{x_2}\,\overline{x_3} \vee \left(\overline{x_2} \vee x_3\right)\times\left(x_2 \vee x_3\right) = 0 \vee x_3 = x_3\ ;$$

$$f_2 = df/dx_2 = \left(x_1x_3 \vee 0\times\overline{x_3}\right)\oplus\left(x_1x_3 \vee 1\times\overline{x_3}\right) = x_1x_3\oplus\left(x_1x_3 \vee \overline{x_3}\right) = x_1x_3\oplus\left(x_1 \vee \overline{x_3}\right) =$$

$$= x_1x_3\times\overline{\left(x_1 \vee \overline{x_3}\right)} \vee \overline{x_1x_3}\times\left(x_1 \vee \overline{x_3}\right) = x_1x_3\times\overline{x_1}x_3 \vee \left(\overline{x_1} \vee \overline{x_3}\right)\times\left(x_1 \vee \overline{x_3}\right) = 0 \vee \overline{x_3} = \overline{x_3}\ ;$$

$$f_3 = df/dx_3 = \left(x_1\times 0 \vee x_2\times 1\right)\oplus\left(x_1\times 1 \vee x_2\times 0\right) = x_2\oplus x_1 = x_1\overline{x_2} \vee \overline{x_1}x_2\ .$$

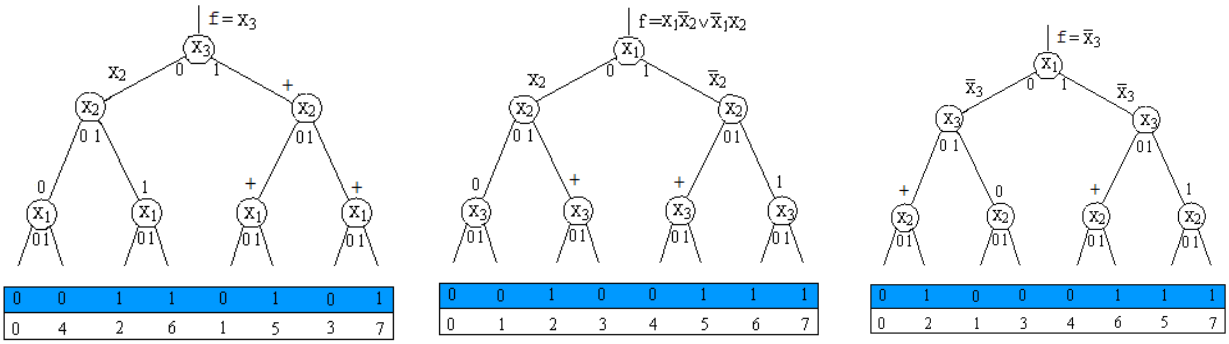


Рис. 10. Графічне взяття булевих похідних

Для трьох змінних маємо чотири умови активізації $\left(x_1\overline{x_2}\right.$, $\left.\overline{x_1}x_2\right.$, $\left.x_3\right.$, $\left.\overline{x_3}\right)$, що відповідають чотирьом логічним шляхам у схемній структурі диз'юнктивної форми цієї функції. Результати отримання булевих похідних для графічного та аналітичного способу подання булевих функцій повністю збігаються, що підтверджує ефективність застосування запропонованих методів.

Отже, взяття булевої похідної фактично полягає в переставленні розрядів таблиці істинності (Q-вектора) відповідно до обраного суттєвого входу, а застосування формули

$$L_{ti} = T_t \oplus F_i = F_{ti}\left\lfloor\left(x_{i1}\oplus T_{t1}\right),\left(x_{i2}\oplus T_{t2}\right),...,\left(x_{ij}\oplus T_{tj}\right),...,\left(x_{ini}\oplus T_{tni}\right)\right\rfloor\oplus T_{ti}\ , \qquad (1)$$

де F_i – компонент (примітив) функціонального опису цифрового пристрою;

T_t – тест-вектор;

L_{ti} – дедуктивна функція паралельного моделювання несправностей, дає змогу використовувати поданий Q-вектор для дедуктивного моделювання несправностей.

Булеві похідні для кубітного подання логічних функцій

Коректність використання прикметника "кубітна" для моделей цифрових пристроїв основана на порівнянні лінійної та булевої алгебри Кантора, що базується на алфавіті $A^1=\{0,\ 1,\ X,\ U\}$. Тут перші два символи – примітиви. Третій визначається суперпозицією: $X = 0\cup 1$. Символ U

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)
Innovative technologies and scientific solutions for industries. 2023. No. 3 (25)

є доповненням до універсуму. Загалом формулу обчислення булевих похідних для формування відповідної кубітної матриці функціональності можна подати як *XOR*-операцію над розрядами сусідніх груп бітів:

$$D_{(ij,ij+1)}(Q,x_i)=Q_j(x_i)\overset{2^{n-i}}{\underset{j=1}{\oplus}}Q_{j+1}(x_i),\quad i=\overline{1,n},$$

де n – кількість змінних у функціональності;

i – номер змінної, за якою береться похідна;

$(j,j+1)$ – номери сусідніх груп у кубітному векторі, що підлягають *XOR*-порівнянню, де число та потужність таких груп функціонально залежить від номера змінної;

$D_{(ij,ij+1)}(Q,x_i)$ – сусідні групи похідної по i-й змінній, що формуються *XOR*-операцією. Групою вважається сукупність бітів, кратних ступеню двійки, що підлягає порозрядному *XOR*-складанню з відповідними бітами сусідньої групи кубітного вектора [17].

Приклад взяття булевих похідних по кубітному покриттю для визначення умов суттєвості вхідних змінних у процесі моделювання несправностей мультиплексора *MUX* 2-в-1 $f(x_1,x_2,x_3)=\left(x_1x_3\vee x_2\overline{x_3}\right)$. Обчислення трьох похідних першого порядку за таблицею істинності дає такий результат: $f_1^{'}(01010101)=x_3$, $f_2^{'}(10101010)=\overline{x_3}$, $f_3^{'}(00111100)=x_1\oplus x_2$.

Результати отримання булевих похідних для кубічного, графічного та аналітичного способу подання булевих функцій на прикладі *MUX* 2-в-1 повністю збігаються, що підтверджує працездатність та ефективність застосування запропонованих методів.

Метод моделювання несправностей за дедуктивними *Q*-векторами

Математична основа дедуктивного моделювання несправностей полягає в транспортуванні бінарних комбінацій вхідних несправностей на вихід на заданому вхідному набору за формулою $L=T\oplus F$. Дедуктивне моделювання полягає в зміні логіки елемента F залежно від вхідних умов T. Дедуктивне моделювання є найефективнішим засобом аналізу якості тестів і синтезу таблиць несправностей для пошуку дефектів, простежування шляху поширення несправності. Пропонується його реалізація на основі векторної форми опису логіки, що передбачає логічні аналітичні форми й дає змогу істотно спростити алгоритми синтезу дедуктивних моделей та їх застосування для інтерпретативного моделювання цифрових елементів і схем великої розмірності [13]. Сутність методу моделювання несправностей за дедуктивними *Q*-векторами полягає в побудові дедуктивних векторів, що є впорядкованою сукупністю векторних булевих похідних за вхідними змінними, що визначаються стовпцями матриці. Сукупність похідних за вхідними змінними в матриці є тестом для перевірки одиночних константних несправностей вхідних та вихідних ліній. Одиничні координати дедуктивних векторів є умовами активізації вектора вхідної інформації як адреси для транспортування на вихід схеми. Будується дедуктивний вектор способом переставлення бітів *Q*-вектора та операції *XOR* між вхідним набором для функціонального елемента та його таблицею істинності (*Q*-вектором).

Розглянемо метод синтезу дедуктивних векторів за *Q*-вектором на прикладі мультиплексора *MUX* 2-в-1 (рис. 11).

L-матриця

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

а)

H-матриця

0	1	2	3	4	5	6	7
1	0	3	2	5	4	7	6
2	3	0	1	6	7	4	5
3	2	1	0	7	6	5	4
4	5	6	7	0	1	2	3
5	4	7	6	1	0	3	2
6	7	4	5	2	3	0	1
7	6	5	4	3	2	1	0

б)

D-матриця

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

в)

Рис. 11. Формування дедуктивних векторів для *MUX* 2-в-1

Сучасний стан наукових досліджень та технологій в промисловості. 2023. № 3 (25)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

1. Дано таблицю істинності n змінних у порядку вагомозначності позиційного коду вхідних наборів (від 000...0 до 2^{n-1}). Подамо вихідний стовпець таблиці як Q-вектор завдовжки 2^n. Для *MUX* 2-в-1 $Q=(00100111)$.

2. Отримання матриці L-векторів (взяття булевих похідних за i-ю змінною). Для кожного розряду Q_i обчислюється $L_i = Q \oplus Q_i$, де Q_i – стан i-біта Q-вектора. Для цього складається матриця $\left(2^{n+1} \times 2^{n+1}\right)$, у перший рядок і перший стовпець якої записується Q-вектор за умови, що нульова комірка порожня (жовті рядок і стовпець на рис. 11, *а*).

3. У кожен i-й рядок матриці L записується вектор Q без інверсії, якщо в стовпці $Q_i = 0$, та з інверсією, якщо $Q_i = 1$ (рис. 11, *а*).

4. Формується дедуктивна матриця D ($2^n \times 2^n$, рис. 11, *в*) способом переставлення елементів рядків матриці L за правилами H-матриці ($2^n \times 2^n$, рис. 11, *б*). H-матриця фактично відповідає встановленню по черзі відповідних змінних у молодший розряд і переставленню відповідно до цього елементів таблиці істинності (Q-вектора).

Розглянемо більш детально спосіб переставлення елементів Q-вектора залежно від вхідного набору та значення функції ПЕ на цьому наборі. Принцип переставлення такий: аналізуються розряди вхідного набору починаючи зі старшого і для тих розрядів вхідного набору, які дорівнюють 1, виконується переставлення елементів Q-вектора групами по 2^{n-1} розрядів, де n – номер розряду за умови, що молодший розряд (правий) вважається першим. Після всіх переставлень елементи отриманого дедуктивного вектора інвертуються, якщо значення Q-вектора на цьому наборі дорівнює 1 згідно з п. 3. Зауважимо, що запропонований принцип переставлень Q-вектора дає змогу уникнути використання H-матриці, яка, по-перше, займає багато пам'яті, а по-друге, її побудова достатньо складна. У табл. 1 наведено приклад переставлень Q-вектора 5 змінних $(n=5)$ для вхідного набору (11010). Початковий Q-вектор поданий у першому рядку за природним порядком змінних *downto*. Починаємо аналіз зі старших розрядів вхідного набору й переставляємо групи елементів у Q-векторі для тих розрядів, які визначаються одиницями у вхідному наборі. Для першого розряду **1**1010 переставляємо групи по 2^{n-1}, тобто по 16 елементів (перше переставлення). Для другого розряду 1**1**010 переставляємо групи по 2^{n-2}, тобто по 8 елементів (друге переставлення). Для четвертого розряду 110**1**0 переставляємо групи по 2^{n-4}, тобто по 2 елементи (третє переставлення).

Ця процедура не залежить від кількості змінних, а залежить тільки від числа 1 у вхідному наборі й повністю збігається з переставленнями в графовому поданні БФ [16] та з ітеративною процедурою у [13].

Викликає зацікавлення результат переставлень для вхідного набору 11111 (табл. 2).

Таблиця 1. *Приклад переставлень для набору 11010*

Початковий Q-вектор	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Перше переставлення	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Друге переставлення	24	25	26	27	28	29	30	31	16	17	18	19	20	21	22	23	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7
Третє переставлення	26	27	24	25	30	31	28	29	18	19	16	17	22	23	20	21	10	11	8	9	14	15	12	13	2	3	0	1	6	7	4	5

Таблиця 2. *Переставлення для набору 11111*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Процедура моделювання несправностей за дедуктивними векторами передбачає такі кроки.

1. Існує шаблон обчислення адрес у дедуктивному векторі для моделювання вхідних несправностей. Кількість стовпців дорівнює подвоєному числу несправностей, спочатку розташовуються нульові несправності, а потім одиночні. Кожен рядок має вагу, кратну ступеню двійки, до того ж *X1* (перший рядок) відповідає молодшому розряду.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) *Innovative technologies and scientific solutions for industries. 2023. No. 3 (25)*

Спочатку шаблон і вектор моделювання (*L*-вектор) заповнюються нулями (або мають порожні комірки).

2. Вхідні несправності мають інверсні значення від справних значень сигналів на входах ПЕ (вхідному набору).

3. У комірках, що відповідають вхідним несправностям, у шаблоні ставиться 1.

4. За стовпцями, де стоять 1, обчислюється адреса (індекс) координати дедуктивного вектора з огляду на ваги рядків (десятковий еквівалент двійкового числа стовпця шаблону 1, 2, 4, 8, …).

5. У вектор моделювання (L) у комірки, що відповідають стовпцям шаблону з 1, записуються значення розрядів відповідного дедуктивного вектора за обчисленими адресами (індексами). Зазначимо, що вихідні несправності ПЕ не моделюються й відповідні (вихідні) позиції *L*-вектора не обчислюються. Необхідно зважати, що дедуктивний вектор для одиничних значень вихідного сигналу на цьому наборі інвертується.

6. Якщо за одиничними значеннями вектора моделювання в шаблоні стоїть 1, то несправність, що визначає стовпець шаблону, перевіряється.

Розглянемо приклади моделювання несправностей за дедуктивними векторами для ПЕ *MUX* 2-в-1 для декількох вхідних наборів. Набір (0 0 0). Початковий список несправностей $\{1^1, 2^1, 3^1\}$. Дедуктивний вектор (0 0 1 0 0 1 1 1). Результат моделювання несправностей зображений на рис. 12.

Набір $(1\,1\,1)$. Початковий список несправностей $\{1^0, 2^0, 3^0\}$. Дедуктивний вектор (0 0 0 1 1 0 1 1). Результат подано на рис. 13.

	3^0	2^0	1^0	3^1	2^1	1^1
X1				1		
X2					1	
X3						1
Адреси стовпців початкових несправностей				1	2	4

	0	1	2	3	4	5	6	7
					1	1	1	1
			1	1			1	1
		1		1		1		1
L	0	0	0	0	1	0	0	0

Це правильний результат

3^0	2^0	1^0	3^1	2^1	1^1
			0	1	0

Рис. 12. Моделювання несправностей для набору 0 0 0 ПЕ *MUX* 2-в-1

	3^0	2^0	1^0	3^1	2^1	1^1
X1	1					
X2		1				
X3			1			

	0	1	2	3	4	5	6	7
					1	1	1	1
			1	1			1	1
		1		1		1		1
L	0	0	1	0	0	0	0	0

Це правильний результат

3^0	2^0	1^0	3^1	2^1	1^1
0	0	1			

Рис. 13. Моделювання несправностей для набору (1 1 1) ПЕ *MUX* 2-в-1

Розглянемо, як впливає послідовність змінних у вхідному наборі на результат моделювання за дедуктивними *Q*-векторами. Зазначимо, що в мовах опису апаратури використовуються два способи подання векторних змінних:

– *std_logic_vector (3 downto 0)*; – молодший розряд правий;

– *std_logic_vector (3 to 0)*; – молодший розряд лівий.

Для прикладу розглянемо моделювання симетричного набору (0 1 0). Початковий список несправностей $\{1^1, 2^0, 3^1\}$. Послідовність змінних у вхідному наборі впливає тільки на формування шаблону обчислення адрес. Дедуктивний вектор (0 1 1 1 0 0 1 0).

Спочатку розглянемо шаблон для *downto*. Тут молодший розряд правий (третій за порядком), що відповідає загальноприйнятій системі позиційного коду.

Результати моделювання несправностей наведені на рис. 14.

	3^0	2^0	1^0	3^1	2^1	1^1
X1						1
X2		1				
X3				1		

	0	1	2	3	4	5	6	7
					1	1	1	1
			1	1			1	1
		1		1		1		1
L	0	1	0	1	0	0	0	0

Це правильний результат

3^0	2^0	1^0	3^1	2^1	1^1
	1		1		0

Рис. 14. Моделювання несправностей набору (0 1 0) для шаблону *downto*

Сучасний стан наукових досліджень та технологій в промисловості. 2023. № 3 (25)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

Далі розглянемо шаблон для *to*. Тут молодший розряд лівий (перший за порядком), що не відповідає загальноприйнятій системі позиційного коду.

Результати моделювання несправностей зображені на рис. 15.

	1^0	2^0	3^0	1^1	2^1	3^1
X1				1		
X2		1				
X3						1

	0	1	2	3	4	5	6	7
					1	1	1	1
			1	1			1	1
		1		1		1		1
L	0	1	0	1	0	0	1	0

Це результат з точністю до навпаки

1^0	2^0	3^0	1^1	2^1	3^1
	1		1		0

Рис. 15. Моделювання несправностей набору (0 1 0) для шаблону *to*

Так, результати моделювання для шаблону *downto* повністю збігаються з результатами простого дедуктивного моделювання (рис. 6) і не збігаються для шаблону *to*.

Отже, у використанні позиційного коду номерів розрядів ПЕ рекомендується застосовувати шаблон обчислення адрес *downto*.

Розглянемо приклади моделювання несправностей за дедуктивними векторами для більш складного ПЕ *MUX* 4-в-1. На рис. 16 подано його познаку (*а*), кубічне покриття (*б*), закон функціювання (*в*) і фрагмент таблиці істинності (*г*), отримані в *Xilinx ISE*. Останній стовпець на рис. 16, *г* – це фактично *Q*-вектор.

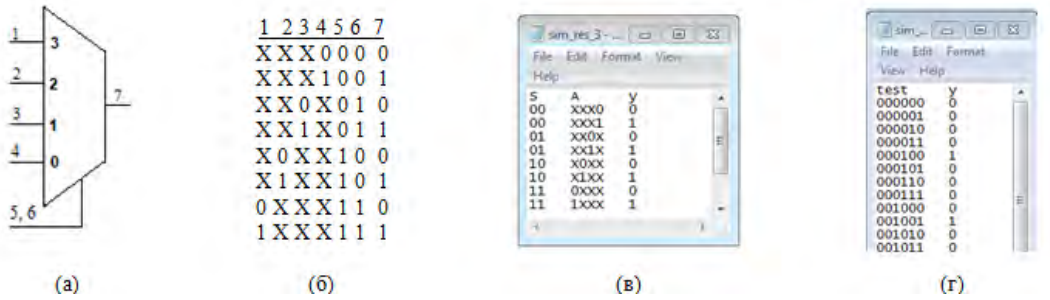


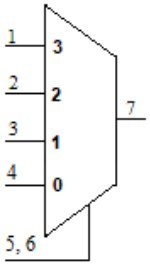
Рис. 16. Познака, кубічне покриття та фрагмент таблиці істинності ПЕ *MUX* 4-в-1

На рис. 17 поданий останній стовпець повної таблиці істинності за зростанням номерів наборів для *MUX* 4-в-1 (2^6 = 64), розбитий на дві частини по 32 набори. Фактично це *Q*-вектор, або переставлений *D*-вектор, позначений кольором. Зелений колір – для набору (0 0 0 0 0 0), а синій – для набору (1 1 1 1 1 1).

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

Рис. 17. Повна таблиця істинності (*Q*-вектор) для *MUX* 4-в-1

На рис. 18 наведені результати моделювання несправностей для ПЕ *MUX* 4-в-1 за дедуктивними *Q*-векторами на наборах (0 0 0 0 0 0) та (1 1 1 1 1 1).



Набір	0	0	0	0	0	0	
Несправності	6^1	5^1	4^1	3^1	2^1	1^1	4^1
Позиції в D-векторі	1	2	4	8	16	32	
Значення в D-векторі	0	0	1	0	0	0	
Набір	1	1	1	1	1	1	
Несправності	6^0	5^0	4^0	3^0	2^0	1^0	1^0
Позиції в D-векторі	62	61	59	55	47	31	
Значення в D-векторі. (інв)	0	0	0	0	0	1	

Рис. 18. Результати моделювання несправностей ПЕ *MUX* 4-в-1 за дедуктивними Q-векторами

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)
Innovative technologies and scientific solutions for industries. 2023. No. 3 (25)

Результати досліджень та їх обговорення

Перевіримо отримані результати способом моделювання несправностей на програмі *DCP* (кубічне моделювання) повного тесту 000000 000001 000010 000011 111100 111101 111110 111111 011111 111000, в якому присутні зазначені набори. Результат повністю збігається, що підтверджує працездатність і ефективність методу моделювання несправностей за дедуктивними *Q*-векторами (рис. 19).

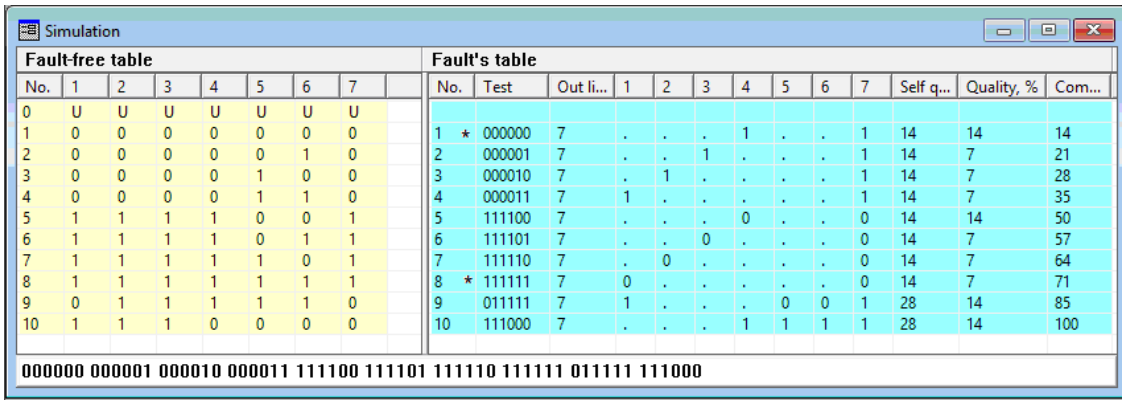


Рис. 19. Результати моделювання програмою *DCP* повного тесту для ПЕ *MUX* 4-в-1

Розглянемо співвідношення витрат пам'яті для кубічного моделювання несправностей і моделювання за дедуктивними Q-векторами.

Для кодування символів алфавіту A^2 (16 символів) використовується кубітне кодування по 4 біти (полубайт) на кожен символ. Так, для *MUX* 4-в-1 розмір КП становитиме 56 полубайтових комірок (56x4 = 224 біти), у цьому разі *Q*-вектор – 64 біти, тобто *Q*-вектор займає менше пам'яті. Якщо умовно взяти $n = 8$ (*MUX* 5-в-1), то КП становитиме 90 полубайтових комірок (360 бітів), а *Q*-вектор – 256 бітів, що близько за витратами пам'яті одне до одного.

Тобто якщо $n > 8$ (а в середньому таке співвідношення буде для всіх регулярних КП основних логічних елементів і мультиплексорів) пам'ять для зберігання *Q*-вектора перевищуватиме пам'ять для зберігання КП. І що більше n, то зростатиме різниця не на користь кубітних *Q*-векторів. Тобто для ПЕ з кількістю входів $n > 8$ застосовувати *Q*-вектор не доцільно з огляду на витрати пам'яті.

І що далі? А далі лише декомпозиція. Наведемо приклад декомпозиції *MUX* 4-в-1 на ПЕ *MUX* 2-в-1 та моделювання несправностей декомпозованої структури. На рис. 20 зображена декомпозована структура *MUX* 4-в-1 через *MUX* 2-в-1 та результати моделювання несправностей на наборі (0 0 0 0 0 0) з огляду на результати, здобуті в процесі моделювання набору (0 0 0) за дедуктивними векторами для *MUX* 2-в-1.

Результати повністю збігаються з результатами кубічного моделювання зазначеного набору програмою *DCP* (рис. 8). Це підтверджує працездатність методу моделювання несправностей за дедуктивними *Q*-векторами внаслідок декомпозиції.

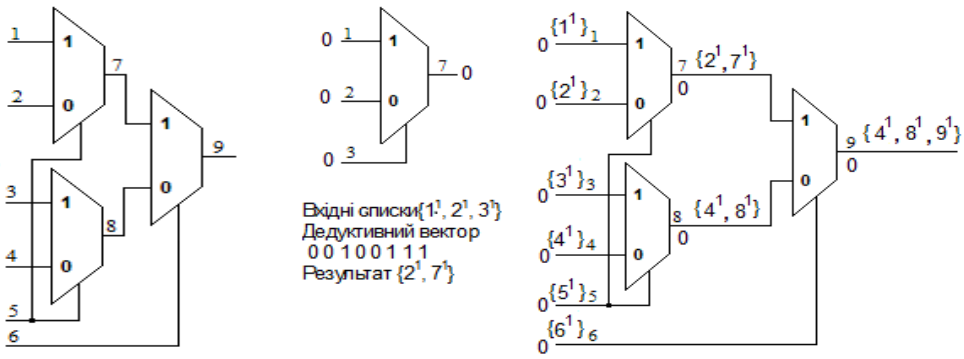


Рис. 20. Результати моделювання *MUX* 4-в-1 після декомпозиції

Сучасний стан наукових досліджень та технологій в промисловості. 2023. № 3 (25)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

Висновки й перспективи подальшого розвитку

У статті розглянуто різні стилі опису цифрових пристроїв мовами опису апаратури. Показано еквівалентність паралельних і послідовних умовних операторів, а також їх схемна реалізація у вигляді мультиплексорів. Запропоновано спосіб отримання таблиць істинності синтезованої схемної структури за допомогою *TestBench (Xilinx ISE)*. Необхідно зауважити, що запропонований підхід має бути корисним проєктувальникам цифрових систем, які документують та обмінюються налагодженими проєктами мовами опису апаратури.

Розглянуто різні технології та структури даних дедуктивного моделювання несправностей для табличного й аналітичного способу опису цифрових схем. Запропоновано програмну реалізацію кубічного дедуктивного моделювання несправностей і показно еквівалентність здобутих результатів для схем мультиплексорів *MUX* 2-в-1 та *MUX* 4-в-1 з використанням програмного продукту *DCP*.

Для моделювання застосовуються кубітні моделі опису структури цифрових систем і функцій компонентів, що визначаються компактністю опису таблиць істинності у формі *Q*-векторів. Це дало змогу підвищити швидкодію програмних та апаратних засобів для справного й несправного моделювання завдяки адресній реалізації аналізу кубітних векторів.

Запропоновано метод інтерпретативного моделювання несправностей цифрових схем, що базується на взятті булевих похідних способом переставлення розрядів *Q*-вектора без використання *H*-матриці. Це дало змогу скоротити витрати пам’яті в процесі програмної реалізації методу та підвищити його швидкодію.

Список літератури

1. Abramovici M. A., Breuer A. M., Friedman D. Digital system testing and testable design. Comp. Sc. Press., 1998. 652 p.

2. Armstrong D. B. A deductive method of simulating faults in logic circuits. *IEEE Trans. on Computers*. Vol. № 5. 1972. P. 464–471. DOI: 10.1109/T-C.1972.223542

3. Шкиль А. С., Хаханов В. И., Ханько В. В. Дедуктивный метод кубического моделирования неисправностей цифровых устройств. *Радиоэлектроника и информатика*. 1999. № 1(6). С. 77–84. URL: https://openarchive.nure.ua/bitstreams/1e30e364-445b-488f-97ea-9394cd98e7e2/download

4. Хаханов В. И., Ковалев Е. В., Джахирул Х. М., Мехеди Масуд М. Д. Кубическое моделирование неисправностей цифровых проектов на основе FPGA, CPLD. *Радиоэлектроника и информатика*. 1999. № 4. С. 64–71.

5. Хаханова А. В., Хаханов В. І., Чумаченко С. В., Литвинова Є. І., Рахліс Д. Ю. Векторні моделі логіки і структури для тестування та моделювання цифрових схем. *Радіоелектроніка. Інформатика. Управління*. Запоріжжя: ЗНТУ. 2021. №3. С. 69–85. DOI: 10.15588/1607-3274-2021-3-7

6. Liu T., Yu T., Wang S., Cai S. An efficient degraded deductive fault simulator for small-delay defects. *Institute of Electrical and Electronics Engineers (IEEE Access)*. 2020. Vol. 8. P. 855–862. DOI: 10.1109/ACCESS.2020.3037292

7. Kaja E., Gerlin N., Rivas L., Bora M.K., Devarajegowda K., Ecker W. MetaFS: model-driven fault simulation. *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT),* 19-21 Oct. 2022. P. 1–8. DOI:10.1109/DFT56152.2022.9962369

8. Gupta I. Stuck at fault testing in combinational circuits using FPGA. Proceedings of emerging trends and technologies on intelligent systems. *Advances in Intelligent Systems and Computing*. Springer, Singapore. Noida, India, Vol 1371. P. 275–284. DOI: 10.1007/978-981-16-3097-2_23

9. Higami Y., Yamauchi T., Inamoto T., Wang S., Takahashi H., Saluja K. K. Machine learning based fault diagnosis for stuck-at faults and bridging faults. *37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC’22)*, Phuket, Thailand, 2022. P. 477–480. DOI: 10.1109/ITC-CSCC55581.2022.9894966

10. Soham R., Millican S., Agrawal V. Training neural network for machine intelligence in automatic test pattern generator. *34th International Conference on VLSI Design and 20th International Conference on Embedded Systems (VLSID’21),* 2021, Guwahati, India. P. 16–32. DOI: 10.1109/VLSID51830.2021.00059

11. Nirmalraj T. Radhakrishnan S., Pandiyan S. K. Automatic diagnosis of single fault ininterconnecttesting of SRAM-based FPGA. *IET Computers & Digital Techniques*. John Wiley& Sons Ltd. 2021. №15 (5). P. 362–371. DOI: 10.1049/cdt2.12028

12. Хаханов В. И., Ктейман Хассан, Парфентий А. Н., Хаханова И. В. HFS-процессор аппаратного моделирования неисправностей цифровых проектов. *АСУ и приборы автоматики*. 2007. № 1 (134). С. 93–108.

13. Gharibi W., Hahanova A., Hahanov V., Chumachenko S., Litvinova E., Hahanov I. Vector-logic synthesis of deductive matrices for fault simulation. *Èlektronic modeling.* 2023. №45 (2). P. 16–33. DOI: 10.15407/emodel.45.02.016.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) *Innovative technologies and scientific solutions for industries. 2023. No. 3 (25)*

14. Pong P. C. RTL Hardware design using VHDL: coding for efficiency, portability, and scalability. *Wiley-IEEE Press*, 2006. 694 p. DOI: 10.1002/0471786411

15. Shkil A.S., Miroshnyk M., Kulak E., Filippenko I., Kucherenko D., Grebenyuk A. Synchronizing Sequences for Verification of Finite State Machines. *9th International IEEE Conference Dependable Systems, Services and Technologies, DESSERT'2018*, Ukraine, Kyiv, 2018. P. 226–230. DOI: 10.1109/UkrMiCo47782.2019.9165509

16. Шкиль А. С., Кривуля Г. Ф. Автоматизация получения булевых разностей. *АСУ и приборы автоматики*. 1981. Вып. 59. С. 73–78.

17. Хаханов В. И., Емельянов И. В., Любарский М. М., Чумаченко С. В., Литвинова Е. И., Бани А. Т. Кубитный метод дедуктивного анализа неисправностей для логических схем. *Электронное моделирование*. 2017. Т. 39(6). С. 59–91.

References

1. Abramovici, M. A., Breuer, A. M., Friedman, D. (1998), Digital system testing and testable design, Comp. Sc. Press, 652 p.

2. Armstrong, D. B. (1972), "A deductive method of simulating faults in logic circuits", *IEEE Trans. on Computers*, Vol. № 5, P. 464–471. DOI: 10.1109/T-C.1972.223542

3. Shkil, A. C., Hahanov, V. I., Han'ko, V. V. (1999), "Deductive method of cubic faults simulation of digital devices" ["Deduktivnyj metod kubicheskogo modelirovanija neispravnostej cifrovyh ustrojstv"], *Radioelectronics and informatics*, № 1(6), P. 77–84, available at: https://openarchive.nure.ua/bitstreams/1e30e364-445b-488f-97ea-9394cd98e7e2/download

4. Hahanov, V. I., Kovalev, E. V., Djahirul, Hak H. M., Mehedi, Masud M.D. (1999), "Cubic fault simulation of digital systems based on FPGA, CPLD" ["Kubicheskoe modelirovanie neispravnostej cifrovyh proektov na osnove FPGA, CPLD"], *Radioelectronics and informatics*, № 4, P. 64–71.

5. Hahanova, A. V., Hahanov, V. I., Chumachenko, S. V., Litvinova, E. I., Rakhlis, D. Y. (2021), "Vector-driven logic and structure for testing and deductive fault simulation" ["Vektorni modeli logiky` i struktury` dlya testuvannya ta modelyuvannya cy`frovy`x sxe"], *Radio Electronics, Computer Science, Control*, Zaporizhzhia: ZNTU, №3, P.69–85. DOI: 10.15588/1607-3274-2021-3-7

6. Liu, T., Yu, T., Wang, S., Cai, S. (2020), "An efficient degraded deductive fault simulator for small-delay defects", *Institute of Electrical and Electronics Engineers (IEEE Access),* Vol. 8, P. 855–862. DOI: 10.1109/ACCESS.2020.3037292

7. Kaja, E., Gerlin, N., Rivas, L., Bora, M. K., Devarajegowda, K., Ecker, W. (2022), "MetaFS: model-driven fault simulation", *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT),* P. 1–8. DOI:10.1109/DFT56152.2022.9962369

8. Gupta, I. (2021), "Stuck at fault testing in combinational circuits using FPGA", Proceedings of emerging trends and technologies on intelligent systems, *Advances in Intelligent Systems and Computing*, Springer, Singapore, Noida, India, Vol. 1371, P. 275–284. DOI: 10.1007/978-981-16-3097-2_23

9. Higami, Y., Yamauchi, T., Inamoto, T., Wang, S., Takahashi, H., Saluja, K. K. (2022), "Machine learning based fault diagnosis for stuck-at faults and bridging faults", *37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC'22),* Phuket, Thailand, P. 477–480. DOI: 10.1109/ITC-CSCC55581.2022.9894966

10. Soham, R., Millican, S., Agrawal, V. (2021), "Training neural network for machine intelligence in automatic test pattern generator", *34th International Conference on VLSI Design and 20th International Conference on Embedded Systems (VLSID'21),* Guwahati, India, P. 16–32. DOI: 10.1109/VLSID51830.2021.00059

11. Nirmalraj, T. Radhakrishnan, S., Pandiyan, S. K. (2021), "Automatic diagnosis of single fault ininterconnecttesting of SRAM-based FPGA", *IET Computers & Digital Techniques*, John Wiley&Sons, №15 (5), P. 362–371. DOI: 10.1049/cdt2.12028

12. Hahanov V. I., Kteyman H., Parfentiy A. N., Hahanova I. V. (2007), "HFS-processor for hardware fault simulation of digital projects" ["HFS-processor apparatnogo modelirovanija neispravnostej cifrovyh proektov"], *ACS and automation devices*, № 1 (134), P. 93–108.

13. Hahanov V., Gharibi W., Hahanova A., Chumachenko S., Litvinova E., Hahanov I. (2023), "Vector-logic synthesis of deductive matrices for fault simulation", *Electronic simulation*, Vol. 45, № 2, P. 16–33. DOI: 10.15407/emodel.45.02.016

14. Pong, P. C. (2006), RTL hardware design using VHDL: coding for efficiency, portability, and scalability, *Wiley-IEEE Press*, 694 p. DOI: 10.1002/0471786411

15. Shkil A.S., Miroshnyk M., Kulak E., Filippenko I., Kucherenko D., Grebenyuk A. (2018), "Synchronizing sequences for verification of finite state machines", *9th International IEEE Conference Dependable Systems, Services and Technologies, DESSERT'2018,* Ukraine, Kyiv May 24-27, P. 226–230. DOI: 10.1109/UkrMiCo47782.2019.9165509

16. Shkil A. S., Kruvulia G. F. (1981), "Automation of Boolean differences obtention" ["Avtomatizacija poluchenija bulevyh raznostej"], *ACS and automation devices*, Vol. 59, P. 73–78.

17. Hahanov V. I., Yemelianov I. V., Lubarskiy M. M., Chumachenko S. V., Litviniva E. I., Bani A. T. (2017), "A qubit method for deductive fault analysis for logic circuits" ["Kubitnyj metod deduktivnogo analiza neispravnostej dlja logicheskih shem"], *Electronic simulation*, Vol. 39, № 6, P. 59–91.

Надійшла 21.07.2023

Сучасний стан наукових досліджень та технологій в промисловості. 2023. № 3 (25)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

Відомості про авторів / About the Authors

Шкіль Олександр Сергійович – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри автоматизації проєктування обчислювальної техніки, Харків, Україна; e-mail: oleksandr.shkil@nure.ua; ORCID ID: https://orcid.org/0000-0003-1071-3445

Мірошник Марина Анатоліївна – доктор технічних наук, професор, Харківський національний університет імені В. Н. Каразіна, професор кафедри теоретичної та прикладної системотехніки, Харків, Україна; e-mail: m.miroshnyk@karazin.ua; ORCID ID: https://orcid.org/0000-0002-2231-2529

Рахліс Дарія Юхимівна – кандидат технічних наук, доцент, Харківський національний університет радіоелектроніки, доцент кафедри автоматизації проєктування обчислювальної техніки, Харків, Україна; e-mail: dariia.rakhlis@nure.ua; ORCID ID: https://orcid.org/0000-0002-6652-1840

Тріфанов Олег Віталійович – Харківський національний університет радіоелектроніки, магістрант кафедри автоматизації проєктування обчислювальної техніки, Харків, Україна; e-mail: oleh.trifanov@nure.ua; ORCID ID: https://orcid.org/0009-0003-9732-9522

Shkil Alexander – PhD (Technical Sciences), Associated Professor, Kharkiv National University of Radio Electronics, Associated Professor at the Department of design automation, Kharkiv, Ukraine.

Miroshnyk Maryna – Doctor of Sciences (Engineering), Professor, V. N. Karazin Kharkiv National University, Professor at the Department of theoretical and applied systems engineering, Kharkiv, Ukraine.

Rakhlis Dariia − PhD (Technical Sciences), Associated Professor, Kharkiv National University of Radio Electronics, Associated Professor at the Department of design automation, Kharkiv, Ukraine.

Trifanov Oleh – Kharkiv National University of Radio Electronics, master student at the Department of design automation, Kharkiv, Ukraine.

DATA STRUCTURES FOR DEDUCTIVE SIMULATION OF HDL CONDITIONAL OPERATORS

The subject of research in the article is qubit-vector models for combinational circuits' description and procedures for faults deductive simulation based on these models. **The object of research** is the processes of diagnostic support creation for digital systems based on the usage of vector qubit data. **The purpose of the work** is to increase the speed and quality of diagnostic support creation for digital devices by creating optimal data structures and deductive fault simulation procedures based on structural-functional models of combinational circuits. The following **tasks** are solved in the article: analysis of concurrent and sequential conditional operators of hardware description languages and schematic structures into which they are synthesized; development of the procedure for the truth tables (Q-vectors) formation for schematic structures presented by HDL; development of an universal data structure for cubic and analytical deductive faults simulation; vector models improvement of qubit representation of structures and components of digital systems based on address coding of input signals to increase the manufacturability and speed of faults simulation; development of the procedure for obtaining Boolean derivatives by permuting the lines of the truth tables (Q-vectors) and using the XOR operation; development of a data structure for deductive fault simulation based on the cubic representation of digital circuit components. The following **methods** are used: deductive, cubic, deductive-parallel simulation of faults, faults simulation by deductive Q-vectors. The following **results** were obtained: the equivalence of concurrent and sequential conditional operators, as well as their schematic implementation in the form of multiplexers, was shown; method of obtaining truth tables of the synthesized circuit structure using TestBench (Xilinx ISE) was proposed; different technologies and data structures of deductive fault simulation for tabular, analytical and qubit methods of digital circuits description were considered; the software implementation of faults cubic deductive simulation is considered and the equivalence of the obtained results for the multiplexers circuits (MUX 2-in-1 and MUX 4-in-1) using the DCP software product was demonstrated. **Conclusions**: a new Q-method of interpretative faults simulation of digital circuit is proposed, which is characterized by the usage of compact Q-vectors instead of truth tables, which makes it possible to significantly increase the analysis speed due to the addressable formation of the functional primitives outputs and reduce the volume of data structures, which practically makes the method competitive with compilative simulation technologies.

Keywords: HDL-model; structural-functional model; deductive fault simulation; cubic coverage; truth table; Q-vector.

Бібліографічні описи / Bibliographic descriptions

Шкіль О. С., Мірошник М. А., Рахліс Д. Ю., Тріфанов О. В. Структури даних для дедуктивного моделювання умовних операторів HDL. *Сучасний стан наукових досліджень та технологій в промисловості*. 2023. № 3 (25). С. 98–113. DOI: https://doi.org/10.30837/ITSSI.2023.25.098

Shkil, A., Miroshnyk, M., Rakhlis, D., Trifanov, O. (2023), "Data structures for deductive simulation of HDL conditional operators", *Innovative Technologies and Scientific Solutions for Industries*, No. 3 (25), P. 98–113. DOI: https://doi.org/10.30837/ITSSI.2023.25.098