КОМПЬЮТЕРНАЯ ИНЖЕНЕРИЯ

УДК 658:512.011:681.326:519.713

СИНТЕЗ И АНАЛИЗ ЛОГИЧЕСКИХ Х-ФУНКЦИЙ

ЛЮБАРСКИЙ М.М., АБДУЛЛАЕВ В.Г., ХАХАНОВ В.И., ЧУМАЧЕНКО С.В., ЛИТВИНОВА Е.И., ХАХАНОВ И.В.

Предлагаются модели и методы кубитного синтеза и анализа логических Х-функций (xor, not-xor) от n переменных [1-8], которые являются мощным математическим средством для решения задач генерации тестов, моделирования неисправностей, создания тестопригодных схем. Их основное преимущество заключается в проверяемости неисправностей, инверсных по отношению к исправному поведению логической схемы.

Введение

Компьютинг – вычислительный процесс, представленный транзакциями записи-считывания данных на адресуемой памяти [1]. Такое определение на самом низком уровне убирает теоретические барьеры, создаваемые теоремой Поста и полнотой функционального базиса, устраняет логику (ALU) и шины передачи данных между процессором и памятью в классическом компьютинге. Более того, транзакционный (MAT – Memory-Address-Transaction) компьютинг настоятельно предлагает отказаться от квантовой логики, реализация которой создает технологические проблемы при создании квантового компьютера. Вполне достаточно реализовать фотонные транзакции записи-считывания данных на устойчивой структуре атомарных электронов, выполняемые со скоростью света.

Memory-driven компьютинг – вычислительный процесс, реализуемый в памяти, содержащей три компонента архитектуры: control-, computational- and conventional memory. Memory-driven классический компьютинг имеет существенный недостаток, который заключается в длительном цикле (read-write) обращения к памяти, в десятки раз большем, чем время выполнения логической операции. Однако реализация memory-driven компьютинга на квантовых (фотонных) транзакциях в атомарной структуре электронов устраняет низкое быстродействие обращения к памяти, благодаря световой скорости фотонного обмена данными, что дает возможность убрать из компьютинга технологически сложно реализуемую квантовую логику.

Квантовый компьютинг – отрасль знаний, занимающаяся теорией и практикой параллельного решения комбинаторных задач на вычислителях, использующих субатомные частицы при создании структур данных и их физического взаимодействия для реализации логических операций.

Квантовый эмулирующий компьютинг – структуры данных, модели, методы и алгоритмы для создания software приложений в целях параллельного решения комбинаторных задач на классических компьютерах путем использования дополнительной памяти.

Квантовый компьютинг – вычислительный процесс, использующий фотонные транзакции в атомарной структуре электронов для реализации параллельных алгоритмов и программных приложений. Квантовый компьютер без квантовых параллельных алгоритмов – всего лишь дорогая игрушка. Поэтому стратегия создания квантового компьютинга заключается в одновременной разработке квантовой аппаратуры и квантовых программных приложений на основе параллельных алгоритмов. Параллельное и раздельное развитие двух ветвей квантового компьютинга предоставляет в недалеком будущем возможность ученым из слаборазвитых стран активно участвовать в проектировании, моделировании и верификации квантовых алгоритмов и программных приложений на классических компьютерах в целях их последующей имплементации в рыночно доступные квантовые компьютеры.

Метрика квантового и классического компьютинга не имеет существенных структурных различий. Странно, но в научных публикациях отсутствует метрическое сравнение квантовой и классической логики. Естественно, существует аналогия между квантовыми операциями на кубитных структурах данных и теоретико-множественными операциями на символах алгебры Кантора. Изоморфизм двух структур: квантовой логики и алгебры множеств заключается в подобии носителей и сигнатур. Взаимно-однозначное соответствие носителей определяется отношениями между булеаном и кубитными структурами данных [1-3]:

$\text{Boolean } A^k =$	0	1	$X = 0 \cup 1$	$\varnothing = 0 \cap 1$
$\text{Qubit } \lvert\psi\rangle =$	$\lvert 0\rangle$	$\lvert 1\rangle$	$\alpha \lvert 0\rangle + \beta \lvert 1\rangle$	$\alpha \lvert 0\rangle \lvert \beta \lvert 1\rangle$

Изоморфизм сигнатур определяется взаимно-однозначным соответствием между двумя операциями: объединение – суперпозиция, дополнение – перепутывание. Для операции пересечения соответствующий аналог в квантовой логике не определен. При этом унитарное кодирование символов алфавита Кантора дает возможность параллельно выполнять любые логические операции даже на классическом компьютере. Но дизрапторное инновационное решение заключается в отказе от операций суперпозиции и пере-

путывания в сторону создания memory-driven квантового компьютинга, использующего только операции записи-считывания на памяти, где двоичные состояния реализованы, например, в спиновых моментах электронов.

Моделирование на классических компьютерах параллельных квантовых алгоритмов дает существенный прирост производительности за счет использования избыточной памяти для унитарного кодирования кубитных структур данных. Более того, моделирование квантовых алгоритмов стратегически призвано решать проблему создания квантового интеллекта планеты путем разработки банка параллельных программных приложений для будущего парка рыночно доступных квантовых компьютеров.

Design and Test является самой передовой областью деятельности ученых и компаний, направленной на создание новых технологий компьютинга по метрике: быстродействие, энергосбережение, компактность и надежность. Параметр time-to-market при создании компьютинговой продукции, наряду с качеством, является доминирующим. Поэтому квантовые параллельные методы и алгоритмы проектирования, тестирования, верификации, моделирования и диагностирования цифровых изделий на кубитных структурах данных являются актуальными при их имплементации в современные классические и в будущие квантовые компьютеры, в целях существенного уменьшения time-to-market. Особый интерес представляет использование кубитных покрытий примитивных функциональностей для минимизации, синтеза, анализа и диагностирования цифровых систем на кристаллах. Рассматриваются логические (xor, notxor) Х-функции, задаваемые симметричными кубитными покрытиями, для технологичного параллельного решения задач синтеза тестов и дедуктивного моделирования неисправностей. Синтез дедуктивных формул для Х-функций формирует компактную структуру для транспортирования входных списков неисправностей, которая инвариантна ко входным тестовым наборам. Это делает Х-функцию привлекательной для ее использования при проектировании тестопригодных цифровых систем. Уникальность Х-функций проявляется также в технологической простоте процедуры взятия булевых производных на основе встречного сдвига двоичных разрядов, что является основой для генератора тестов, использующего кубитные покрытия логических схем.

Цель исследования – анализ тестопригодных свойств логических Х-функций от n переменных, а также их синтез путем разработки рекурсивных моделей и формул, использующих симметрию свойств их кубитных покрытий.

Задачи: 1) Формирование модели замечательных свойств Х-функций, используемых для их синтеза и анализа. 2) Создание архитектуры секвенсора для синтеза Q-покрытий Х-функций от конечного числа переменных. 3) Разработка метода синтеза дедуктивных ДНФ для моделирования неисправностей на основе анализа таблиц истинности. 4) Разработка метода параллельного синтеза дедуктивных кубитных покрытий и дедуктивных ДНФ для моделирования неисправностей на основе анализа кубитных векторов описания функциональностей. 5) Разработка метода параллельного синтеза матриц кубитных покрытий дедуктивного анализа на основе свойств симметрии эвристически разработанных матриц перестановки битов. 6) Разработка метода синтеза тестов для логических Х-функций путем перестановки битов в кубитных покрытиях.

1. Свойства логических Х-функций

Практически полезными для синтеза и анализа цифровых схем могут быть следующие 12 свойств Х-функций, интегрированные в модель отношений, представленную на рис. 1.

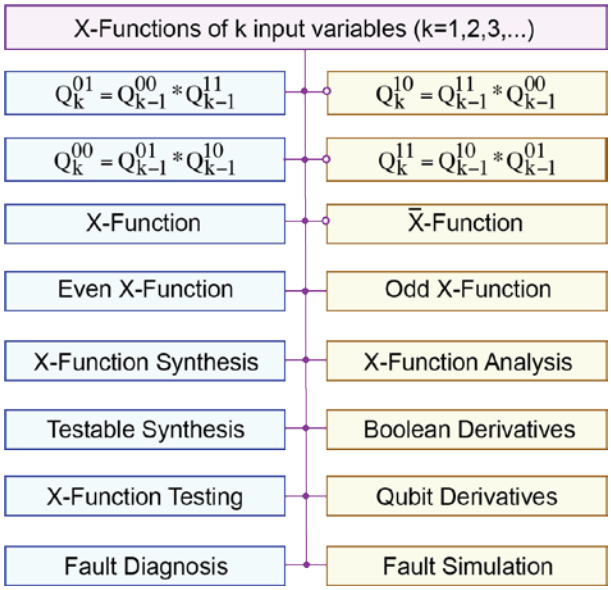

Рис. 1. Структурная модель взаимодействия Х-функций

1) Кубитное покрытие Х-функции имеет равное количество нулевых и единичных координат.

2) Количество Х-функций от n булевых переменных всегда равно двум:

$$Q^{2x}(n) = Q^{x}(n) \vee \bar{Q}^{x}(n).$$

Состояния координат кубитных покрытий обеих Х-функций от n переменных являются взаимноинверсными.

3) Х-функции от одной логической булевой переменной представлены повторителем и инвертором:

$$Y = X;\ Y = \bar{X}.$$

4) X-функции от двух булевых переменных представлены известными логическими примитивами xor, not-xor:

$$Y = X_1\bar{X}_2 \vee \bar{X}_1X_2;\ Y = X_1X_2 \vee \bar{X}_1\bar{X}_2.$$

5) Кубитная производная по любой переменной X-функции равна единичному вектору. Булева производная по любой переменной X-функции равна единице.

6) Для активизации входной переменной X-функции в целях изменения выхода не требуется никаких условий по состоянию других переменных.

7) Пара входных наборов, имеющая инверсные сигналы по всем координатам, всегда изменяет состояние выхода X-функции от нечетного числа переменных. Изменение состояния входа X-функции всегда приводит к изменению состояния выхода.

8) Синтез двух X-функций от n переменных реализуется путем конкатенации (*) кубитных векторов X-функций от n-1 переменной:

$$Q^{2x}(n) = Q^{x}(n-1) * \bar{Q}^{x}(n-1) \vee \bar{Q}^{x}(n-1) * Q^{x}(n-1).$$

Структура секвенсора, предназначенного для синтеза кубитных покрытий X-функций от n=1,2,3,4 переменных посредством конкатенации и инверсии, обозначенной кружочком, представлена на рис. 2.

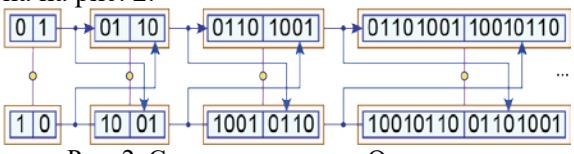

Рис. 2. Секвенсор синтеза Q-покрытия для X-функции

Другая интерпретация X-функций связана с их идентификацией по первому и последнему биту кубитного покрытия: 01–10, 00–11. Тогда синтез кубитных векторов X-функций от k переменных можно реализовать путем применения операции конкатенации к двум Q-покрытиям X-функций от k-1 переменных:

$$Q_k^{01} = Q_{k-1}^{00} * Q_{k-1}^{11};$$
$$Q_k^{10} = Q_{k-1}^{11} * Q_{k-1}^{00};$$
$$Q_k^{00} = Q_{k-1}^{01} * Q_{k-1}^{10};$$
$$Q_k^{11} = Q_{k-1}^{10} * Q_{k-1}^{01}.$$

Естественно, что между двумя кубитными покрытиями X-функций от k переменных существует взаимно-однозначное отношение инверсии:

$$Q_k^{01} = \bar{Q}_k^{10};$$
$$Q_k^{10} = \bar{Q}_k^{01};$$
$$Q_k^{00} = \bar{Q}_k^{11};$$
$$Q_k^{11} = \bar{Q}_k^{00}.$$

Это означает, что если известна одна X-функция от k переменных, то легко можно получить вторую функцию, как двоичное дополнение к первой.

9) Любой входной набор для X-функции проверяет 50% неисправностей по внешним входам, которые являются инверсными по отношению к состояниям исправного поведения упомянутых линий. Два взаимно-инверсных тестовых набора проверяют все одиночные константные неисправности входных переменных и выхода в X-функции от нечетного числа переменных.

10) Дедуктивная формула X-функции транспортирует на выход симметрическую разность входных списков неисправностей. Это означает объединение входных списков проверяемых дефектов, за исключением случая, когда списки неисправностей на всех входах идентичны. X-функция от n переменных, которая отождествляется с xor-примитивом, является единственной, где логическая и дедуктивная функции равны между собой на любом входном двоичном наборе.

11) Тестом для одиночных константных неисправностей всех линий логической X-функции от n переменных является ее СДНФ, дополненная любым термом обратной СДНФ данной функции:

$$T = T^1 \vee T_i^0.$$

Размерность полного проверяющего теста для X-функции всегда равна

$$Q = 1 + \frac{1}{2} \times 2^{2^n}.$$

12) Покоординатная xor-сумма всех кодов X-функции, соответствующих СДНФ (обратной СДНФ), равна нулевому (по всем координатам) вектору.

Таким образом, логические X-функции, обладающие уникальными свойствами тестирования, могут быть использованы для синтеза тестопригодных и самовосстанавливаемых логических цифровых устройств, а также для транспортирования дефектов от внешних входов до выходов булевой структуры.

2. Метод синтеза дедуктивного кубитного покрытия и ДНФ по таблице истинности

Метод синтеза дедуктивной формулы по аналитической форме функциональности [5-7], представленной в виде ДНФ, является достаточно сложной и нерегулярной вычислительной процедурой, которая нуждается в упрощении. Далее предлагается простая и тривиальная формула-процедура получения для входной последовательности (T) дедуктивной таблицы (L), а по ней и записи дедуктивной формулы для функциональности от n переменных, заданной таблицей

истинности (C) или кубическим покрытием функциональности (рис. 3):

$$L_i = T_i \oplus C_j,\ \{i, j\} = 1, 2^{2^n}.$$

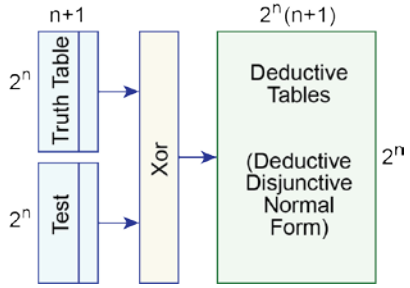

Рис. 3. Секвенсор синтеза дедуктивных таблиц для теста

Для понимания процедуры синтеза дедуктивной таблицы ниже предлагаются логические примитивы и результаты, которые можно использовать для записи дедуктивных выражений в форме ДНФ по единичным значениям выходов логических элементов (and, or, xor, not-xor, not, rep):

C			$T_0 = 000$	$T_1 = 010$	$T_2 = 100$	$T_3 = 111$
X_1	X_2	Y_{and}	$L_0 = T_0 \oplus C$	$L_1 = T_1 \oplus C$	$L_2 = T_2 \oplus C$	$L_3 = T_3 \oplus C$
0	0	0	000	010	100	111
0	1	0	010	000	110	101
1	0	0	100	110	000	011
1	1	1	111	101	011	000

C			$T_0 = 000$	$T_1 = 011$	$T_2 = 101$	$T_3 = 111$
X_1	X_2	Y_{or}	$L_0 = T_0 \oplus C$	$L_1 = T_1 \oplus C$	$L_2 = T_2 \oplus C$	$L_3 = T_3 \oplus C$
0	0	0	000	011	101	111
0	1	1	011	000	110	100
1	0	1	101	110	000	010
1	1	1	111	100	010	000

C			$T_0 = 000$	$T_1 = 011$	$T_2 = 101$	$T_3 = 110$
X_1	X_2	Y_{xor}	$L_0 = T_0 \oplus C$	$L_1 = T_1 \oplus C$	$L_2 = T_2 \oplus C$	$L_3 = T_3 \oplus C$
0	0	0	000	011	101	110
0	1	1	011	000	110	101
1	0	1	101	110	000	011
1	1	0	110	101	011	000

C			$T_0 = 001$	$T_1 = 010$	$T_2 = 100$	$T_3 = 111$
X_1	X_2	Y_{nxr}	$L_0 = T_0 \oplus C$	$L_1 = T_1 \oplus C$	$L_2 = T_2 \oplus C$	$L_3 = T_3 \oplus C$
0	0	1	000	011	101	110
0	1	0	011	000	110	101
1	0	0	101	110	000	011
1	1	1	110	101	011	000

C		$T_0 = 01$	$T_1 = 10$
X_1	Y_{not}	$L_0 = T_0 \oplus C$	$L_1 = T_1 \oplus C$
0	1	00	11
1	0	11	00

C		$T_0 = 00$	$T_1 = 11$
X_1	Y_{rep}	$L_0 = T_0 \oplus C$	$L_1 = T_1 \oplus C$
0	0	00	11
1	1	11	00

Представленные таблицы истинности не являются упорядоченными по возрастанию двоично-десятичных кодов входных наборов. Поскольку таблица истинности представляет собой множество отношений, то такого упорядочения не требуется, чтобы записать аналитическую форму полученной функциональности. Для каждой функции и каждого входного набора ниже записаны дедуктивные функции примитивных элементов (and, or, xor, not-xor, not, rep) в форме СДНФ, которые можно использовать как для аппаратного синтеза логических схем моделирования неисправностей, так и для создания облачного программного сервиса, выполняющего дедуктивный анализ цифровых проектов:

$$L_{and}(000) = X_1X_2;$$
$$L_{and}(010) = X_1\bar{X}_2;$$
$$L_{and}(100) = \bar{X}_1X_2;$$
$$L_{and}(111) = \bar{X}_1X_2 \vee X_1\bar{X}_2 \vee X_1X_2 = X_1 \vee X_2;$$
$$L_{or}(000) = \bar{X}_1X_2 \vee X_1\bar{X}_2 \vee X_1X_2 = X_1 \vee X_2;$$
$$L_{or}(011) = \bar{X}_1X_2;$$
$$L_{or}(101) = X_1\bar{X}_2;$$
$$L_{or}(111) = X_1X_2.$$
$$L_{xor}(000) = \bar{X}_1X_2 \vee X_1\bar{X}_2;$$
$$L_{xor}(011) = \bar{X}_1X_2 \vee X_1\bar{X}_2;$$
$$L_{xor}(101) = \bar{X}_1X_2 \vee X_1\bar{X}_2;$$
$$L_{xor}(110) = \bar{X}_1X_2 \vee X_1\bar{X}_2.$$
$$L_{nxr}(001) = \bar{X}_1X_2 \vee X_1\bar{X}_2;$$
$$L_{nxr}(010) = \bar{X}_1X_2 \vee X_1\bar{X}_2;$$
$$L_{nxr}(100) = \bar{X}_1X_2 \vee X_1\bar{X}_2;$$
$$L_{nxr}(111) = \bar{X}_1X_2 \vee X_1\bar{X}_2.$$
$$L_{not}(01) = X_1;$$
$$L_{not}(10) = X_1.$$
$$L_{rep}(00) = X_1;$$
$$L_{rep}(11) = X_1.$$

Здесь взаимодействие логических переменных определяет пересечение, вычитание или объединение списков, или векторов неисправностей

$$(X_1X_2,\ X_1\bar{X}_2,\ X_1 \vee X_2)$$

соответственно, принадлежащих внешним входам примитивов.

Таким образом, представленный метод для синтеза дедуктивных покрытий и ДНФ, ориентированных на дедуктивный анализ цифровых систем, значительно отличается от существующих аналогов быстродействием Q, которое определяется параллелизмом выполнения регистровых операций между очередным входным воздействием и n кубами покрытия: $Q = 2^{2^n}$.

3. Метод синтеза дедуктивного кубитного покрытия и ДНФ по кубитным покрытиям функциональностей

Еще более высокое быстродействие вычислительных процедур для синтеза дедуктивных функций можно получить, если использовать вместо кубических покрытий (таблиц) кубитные векторы. Для этого необходимо упорядочить полученные ранее кубические формы дедуктивных функций в соответствии с возрастанием двоично-десятичных кодов входных наборов:

C			$T_0=000$	$T_1=010$	$T_2=100$	$T_3=111$
X_1	X_2	Y_{and}	$L_0=T_0\oplus C$	$L_1=T_1\oplus C$	$L_2=T_2\oplus C$	$L_3=T_3\oplus C$
0	0	0	000	000	000	000
0	1	0	010	010	011	011
1	0	0	100	101	100	101
1	1	1	111	110	110	111

C			$T_0=000$	$T_1=011$	$T_2=101$	$T_3=111$
X_1	X_2	Y_{or}	$L_0=T_0\oplus C$	$L_1=T_1\oplus C$	$L_2=T_2\oplus C$	$L_3=T_3\oplus C$
0	0	0	000	000	000	000
0	1	1	011	011	010	010
1	0	1	101	100	101	100
1	1	1	111	110	110	111

C			$T_0=000$	$T_1=011$	$T_2=101$	$T_3=110$
X_1	X_2	Y_{xor}	$L_0=T_0\oplus C$	$L_1=T_1\oplus C$	$L_2=T_2\oplus C$	$L_3=T_3\oplus C$
0	0	0	000	000	000	000
0	1	1	011	011	011	011
1	0	1	101	101	101	101
1	1	0	110	110	110	110

C			$T_0=001$	$T_1=010$	$T_2=100$	$T_3=111$
X_1	X_2	Y_{nxr}	$L_0=T_0\oplus C$	$L_1=T_1\oplus C$	$L_2=T_2\oplus C$	$L_3=T_3\oplus C$
0	0	1	000	000	000	000
0	1	0	011	011	011	011
1	0	0	101	101	101	101
1	1	1	110	110	110	110

C		$T_0=01$	$T_1=10$
X_1	Y_{not}	$L_0=T_0\oplus C$	$L_1=T_1\oplus C$
0	1	00	00
1	0	11	11

C		$T_0=00$	$T_1=11$
X_1	Y_{rep}	$L_0=T_0\oplus C$	$L_1=T_1\oplus C$
0	0	00	00
1	1	11	11

Вместо упорядоченных по возрастанию двоично-десятичных кодов входных наборов таблиц истинности можно записывать кубитные покрытия, которые представляют собой дедуктивную матрицу размерностью m**2:

and	$Q_1=0$	$Q_2=0$	$Q_3=0$	$Q_4=1$
Q	$L_1=Q_1\oplus Q$	$L_2=Q_2\oplus Q$	$L_3=Q_3\oplus Q$	$L_4=Q_4\oplus Q$
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
1	1	0	0	1

or	$Q_1=0$	$Q_2=1$	$Q_3=1$	$Q_4=1$
Q	$L_1=Q_1\oplus Q$	$L_2=Q_2\oplus Q$	$L_3=Q_3\oplus Q$	$L_4=Q_4\oplus Q$
0	0	0	0	0
1	1	1	0	0
1	1	0	1	0
1	1	0	0	1

xor	$Q_1=0$	$Q_2=1$	$Q_3=1$	$Q_4=0$
Q	$L_1=Q_1\oplus Q$	$L_2=Q_2\oplus Q$	$L_3=Q_3\oplus Q$	$L_4=Q_4\oplus Q$
0	0	0	0	0
1	1	1	1	1
1	1	1	1	1
0	0	0	0	0

nxr	$Q_1=1$	$Q_2=0$	$Q_3=0$	$Q_4=1$
Q	$L_1=Q_1\oplus Q$	$L_2=Q_2\oplus Q$	$L_3=Q_3\oplus Q$	$L_4=Q_4\oplus Q$
1	1	1	1	1
0	0	0	0	0
0	0	0	0	0
1	1	1	1	1

not	$Q_1=1$	$Q_2=0$
Q	$L_1=Q_1\oplus Q$	$L_2=Q_2\oplus Q$
1	0	0
0	1	1

rep	$Q_1=0$	$Q_2=1$
Q	$L_1=Q_1\oplus Q$	$L_2=Q_2\oplus Q$
0	0	0
1	1	1

Далее предлагается более простой путь получения матрицы кубитных дедуктивных векторов без использования таблиц истинности на основе только кубитного покрытия функциональности.

Метод синтеза дедуктивной матрицы для анализа неисправностей на основе Q-покрытия функциональности содержит две операции (рис. 4):

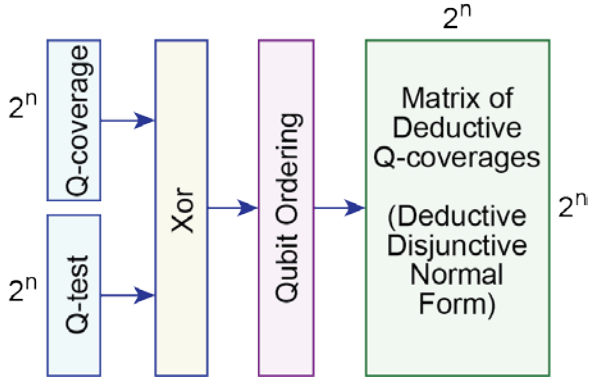

Рис. 4. Секвенсор синтеза дедуктивных покрытий для Q-теста

1) Синтез матрицы кубитных покрытий дедуктивных функций, зависящей от двоично-десятичного номера входного набора и координат Q-вектора функциональности от n переменных:

$$L^{*}_{ij}=(Q_i \overset{m}{\underset{i=1}{\oplus}} Q_j)^{m}_{j=1},\ m=2^{2^n}.$$

Фактически берется первая координата кубитного покрытия, которая xor-складывается со всеми координатами Q-вектора для формирования де-

дуктивного вектора на первой входной последовательности. Затем берется вторая координата Q-вектора, которая также xor-складывается со всеми координатами. Процедура заканчивается после того, как все координаты Q-вектора были xor-сложены с кубитным вектором. Вычислительная сложность данной процедуры равна m**2, которая может быть уменьшена до m путем аппаратной параллельной реализации xor-операции.

2) После получения дедуктивной матрицы на всех входных наборах по кубитному покрытию функциональности необходимо выполнить процедуру перестановки битов в столбцах по правилу

$$L_{ij} = [L^{*}(H_{ij})]_{\{i,j\}=1}^{m}$$

в соответствии с номерами, представленными в матрице перестановки, которая по размерности равна матрице кубитных покрытий дедуктивных функций. Далее представлены примеры получения дедуктивных функций на основе использования только кубитных покрытий функциональностей:

$$\begin{array}{|c|cccc|}\hline Q_{and} & L_1 & L_2 & L_3 & L_4 \\ \hline 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ \hline \end{array} \rightarrow \begin{array}{|c|c|c|c|}\hline 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \\ \hline \end{array} \rightarrow \begin{array}{|c|cccc|}\hline X_1X_2 & L_1 & L_2 & L_3 & L_4 \\ \hline 00 & 0 & 0 & 0 & 0 \\ 01 & 0 & 0 & 1 & 1 \\ 10 & 0 & 1 & 0 & 1 \\ 11 & 1 & 0 & 0 & 1 \\ \hline \end{array} \rightarrow$$

$$\rightarrow L = (00)(X_1X_2) \vee (01)(X_1\bar{X}_2) \vee (10)(\bar{X}_1X_2) \vee (11)(X_1 \vee X_2).$$

$$\begin{array}{|c|cccc|}\hline Q_{or} & L_1 & L_2 & L_3 & L_4 \\ \hline 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ \hline \end{array} \rightarrow \begin{array}{|c|c|c|c|}\hline 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \\ \hline \end{array} \rightarrow \begin{array}{|c|cccc|}\hline X_1X_2 & L_1 & L_2 & L_3 & L_4 \\ \hline 00 & 0 & 0 & 0 & 0 \\ 01 & 1 & 1 & 0 & 0 \\ 10 & 1 & 0 & 1 & 0 \\ 11 & 1 & 0 & 0 & 1 \\ \hline \end{array} \rightarrow$$

$$\rightarrow L = (00)(X_1 \vee X_2) \vee (01)(\bar{X}_1X_2) \vee (10)(X_1\bar{X}_2) \vee (11)(X_1X_2).$$

$$\begin{array}{|c|cccc|}\hline Q_{xor} & L_1 & L_2 & L_3 & L_4 \\ \hline 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ \hline \end{array} \rightarrow \begin{array}{|c|c|c|c|}\hline 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \\ \hline \end{array} \rightarrow \begin{array}{|c|cccc|}\hline X_1X_2 & L_1 & L_2 & L_3 & L_4 \\ \hline 00 & 0 & 0 & 0 & 0 \\ 01 & 1 & 1 & 1 & 1 \\ 10 & 1 & 1 & 1 & 1 \\ 11 & 0 & 0 & 0 & 0 \\ \hline \end{array} \rightarrow$$

$$\rightarrow L = (00 \vee 01 \vee 10 \vee 11)(\bar{X}_1X_2 \vee X_1\bar{X}_2) = (xx)(\bar{X}_1X_2 \vee X_1\bar{X}_2).$$

$$\begin{array}{|c|cccc|}\hline Q_{nxr} & L_1 & L_2 & L_3 & L_4 \\ \hline 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ \hline \end{array} \rightarrow \begin{array}{|c|c|c|c|}\hline 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \\ \hline \end{array} \rightarrow \begin{array}{|c|cccc|}\hline X_1X_2 & L_1 & L_2 & L_3 & L_4 \\ \hline 00 & 0 & 0 & 0 & 0 \\ 01 & 1 & 1 & 1 & 1 \\ 10 & 1 & 1 & 1 & 1 \\ 11 & 0 & 0 & 0 & 0 \\ \hline \end{array}$$

$$\rightarrow L = (00 \vee 01 \vee 10 \vee 11)(\bar{X}_1X_2 \vee X_1\bar{X}_2) = (xx)(\bar{X}_1X_2 \vee X_1\bar{X}_2).$$

$$\begin{array}{|c|cc|}\hline Q_{not} & L_1 & L_2 \\ \hline 1 & 0 & 1 \\ 0 & 1 & 0 \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline 1 & 2 \\ 2 & 1 \\ \hline \end{array} \rightarrow \begin{array}{|c|cc|}\hline X & L_1 & L_2 \\ \hline 0 & 0 & 0 \\ 1 & 1 & 1 \\ \hline \end{array} \rightarrow$$

$$\rightarrow L = (0 \vee 1)(X \vee X) = (xx)(X).$$

$$\begin{array}{|c|cc|}\hline Q_{rep} & L_1 & L_2 \\ \hline 0 & 0 & 1 \\ 1 & 1 & 0 \\ \hline \end{array} \rightarrow \begin{array}{|c|c|}\hline 1 & 2 \\ 2 & 1 \\ \hline \end{array} \rightarrow \begin{array}{|c|cc|}\hline X & L_1 & L_2 \\ \hline 0 & 0 & 0 \\ 1 & 1 & 1 \\ \hline \end{array} \rightarrow$$

$$\rightarrow L = (0 \vee 1)(X \vee X) = (xx)(X).$$

4. Метод синтеза матрицы перестановки битов для формирования дедуктивных функций

В общем случае метод синтеза матрицы перестановки битов в столбцах для формирования дедуктивных функций от n переменных может быть представлен в следующем виде:

$$H_{ij}(n = 1,2,3) = \begin{array}{|cc|}\hline 1 & 2 \\ 2 & 1 \\ \hline \end{array} = \begin{array}{|cccc|}\hline 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \\ \hline \end{array} = \begin{array}{|cccccccc|}\hline 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 1 & 4 & 3 & 6 & 5 & 8 & 7 \\ 3 & 4 & 1 & 2 & 7 & 8 & 5 & 6 \\ 4 & 3 & 2 & 1 & 8 & 7 & 6 & 5 \\ 5 & 6 & 7 & 8 & 1 & 2 & 3 & 4 \\ 6 & 5 & 8 & 7 & 2 & 1 & 4 & 3 \\ 7 & 8 & 5 & 6 & 3 & 4 & 1 & 2 \\ 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \\ \hline \end{array}$$

Каждая матрица обладает симметрией относительно диагоналей, которые представлены минимальными и максимальными числами. Существует также вертикальная и горизонтальная симметрия, где симметричные пары чисел находятся в отношении $H + \bar{H} = 2^n + 1$. Если нумеровать координаты столбцов матрицы начиная с нуля, то предыдущее отношение будет иметь вид $H + \bar{H} = 2^n$. Квадратичная по размеру матрица чисел в своем синтезе имеет следующие закономерности: первый столбец является вектором упорядоченных по возрастанию чисел; второй столбец представляет собой перестановку двух соседних чисел в каждой паре относительно предыдущего столбца; третий создает перестановку соседних пар чисел; четвертый столбец определяется перестановкой соседних тетрад чисел, идущих в обратном порядке. Остальные столбцы для функции от 3-х переменных генерируются как зеркальные отображения синтеза четырех предыдущих столбцов, начиная с последнего столбца, который представляет собой первый столбец, занумерованный в обратном порядке.

Синтез матрицы перестановки битов основан на всех видах симметрии (вертикальная, горизонтальная, диагональная), которая дает возможность, имея один известный квадрант, получить все остальные путем применения формулы:

$$H + \bar{H} = 2^n + 1 \rightarrow \bar{H} = (2^n + 1) - H.$$

Следующая структура матриц объясняет тривиальность построения сколь угодно сложной матрицы перестановки битов от n=1,2,3, … переменных:

$$\begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \rightarrow \begin{matrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \\ \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \end{matrix} \rightarrow \begin{matrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} & \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \\ \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} & \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \\ \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} & \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \\ \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} & \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix} \end{matrix}$$

Фактически, зная любой квадрант от n-1 переменной, за 4 автоматных такта можно получить матрицу перестановки битов для дедуктивной функциональности от n переменных:

$$H^1 \in H^2 \in H^3 \in ... \in H^n.$$

Масштабируемость квадрантов можно продемонстрировать следующей структурой, которая инварианта к количеству переменных анализируемой логической функциональности:

$$\overset{H^1}{\begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix}} = H \in H^2 \rightarrow \overset{H^2}{\begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix}} = H \in H^3 \rightarrow \overset{H^3}{\begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix}} ... = H \in H^n \rightarrow \overset{H^n}{\begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix}}.$$

Примером рекурсивной генерации матриц перестановки битов для переменных n=1,2,3 может служить следующая структура:

$$H_{ij}(n=1,2,3)=\begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} & \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix} \\ \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix} & \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} \end{bmatrix} \rightarrow \begin{bmatrix} \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} & \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix} & \begin{bmatrix} 5 & 6 \\ 6 & 5 \end{bmatrix} & \begin{bmatrix} 7 & 8 \\ 8 & 7 \end{bmatrix} \\ \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix} & \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} & \begin{bmatrix} 7 & 8 \\ 8 & 7 \end{bmatrix} & \begin{bmatrix} 5 & 6 \\ 6 & 5 \end{bmatrix} \\ \begin{bmatrix} 5 & 6 \\ 6 & 5 \end{bmatrix} & \begin{bmatrix} 7 & 8 \\ 8 & 7 \end{bmatrix} & \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} & \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix} \\ \begin{bmatrix} 7 & 8 \\ 8 & 7 \end{bmatrix} & \begin{bmatrix} 5 & 6 \\ 6 & 5 \end{bmatrix} & \begin{bmatrix} 3 & 4 \\ 4 & 3 \end{bmatrix} & \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} \end{bmatrix}$$

Рекурсивное взаимодействие матриц перестановки битов показывает: для получения сколь угодно сложной матрицы от n переменных необходимо знать только один элемент (в данном случае единицу) в матрице для функции от одной переменной.

Если первым элементом нумерации входных двоично-десятичных кодов (начальный адрес) принять ноль, то взаимодействие матриц перестановки битов будет иметь следующий вид:

$$H_{ij}(n=1,2,3)=\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & \begin{bmatrix} 2 & 3 \\ 3 & 2 \end{bmatrix} \\ \begin{bmatrix} 2 & 3 \\ 3 & 2 \end{bmatrix} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{bmatrix} \rightarrow \begin{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & \begin{bmatrix} 2 & 3 \\ 3 & 2 \end{bmatrix} & \begin{bmatrix} 4 & 5 \\ 5 & 4 \end{bmatrix} & \begin{bmatrix} 6 & 7 \\ 7 & 6 \end{bmatrix} \\ \begin{bmatrix} 2 & 3 \\ 3 & 2 \end{bmatrix} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & \begin{bmatrix} 6 & 7 \\ 7 & 6 \end{bmatrix} & \begin{bmatrix} 4 & 5 \\ 5 & 4 \end{bmatrix} \\ \begin{bmatrix} 4 & 5 \\ 5 & 4 \end{bmatrix} & \begin{bmatrix} 6 & 7 \\ 7 & 6 \end{bmatrix} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} & \begin{bmatrix} 2 & 3 \\ 3 & 2 \end{bmatrix} \\ \begin{bmatrix} 6 & 7 \\ 7 & 6 \end{bmatrix} & \begin{bmatrix} 4 & 5 \\ 5 & 4 \end{bmatrix} & \begin{bmatrix} 2 & 3 \\ 3 & 2 \end{bmatrix} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{bmatrix}$$

Из приведенной структуры, иллюстрирующей синтез матриц перестановки битов для получения кубитных покрытий дедуктивного анализа, вытекают следующие полезные свойства, характеризующие метод:

1) Матрица любого уровня иерархии содержит 4 квадранта, которые диагонально равны друг другу (рис. 5):

$$H = \begin{bmatrix} H_1 & H_2 \\ H_3 & H_4 \end{bmatrix} = \begin{bmatrix} H & \bar{H} \\ \bar{H} & H \end{bmatrix}, \; H_1 = H_4; \;\; H_1 = \bar{H}_2; \\ H_2 = H_3; \;\; H_3 = \bar{H}_4.$$

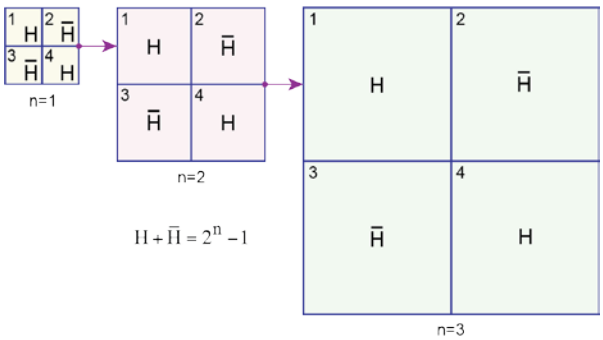

Рис. 5. Структурная иерархия матриц для получения кубитных покрытий

2) Горизонтальные и вертикальные отношения между квадрантами определяются дополнениями (при нулевом начальном элементе или адресе):

$$H_1 + \bar{H}_2 = 2^n - 1 \rightarrow \bar{H}_2 = (2^n - 1) - H_1; \\ \bar{H}_3 + H_4 = 2^n - 1 \rightarrow \bar{H}_3 = (2^n - 1) - H_4; \\ H_1 + \bar{H}_3 = 2^n - 1 \rightarrow \bar{H}_3 = (2^n - 1) - H_1; \\ \bar{H}_2 + H_4 = 2^n - 1 \rightarrow \bar{H}_2 = (2^n - 1) - H_4.$$

Сумма двух чисел, симметричных относительно горизонтальной или вертикальной оси, делящей матрицу от n переменных на две одинаковые части, равна $h + \bar{h} = 2^n - 1$. Четверку компонентов примитивной матрицы, не содержащей других матриц в качестве составных частей, всегда формируют два соседних числа или цифры.

3) Рекурсивная формула получения матрицы перестановки битов для синтеза дедуктивной функции от i=1,2,3,… переменных имеет следующий вид:

$$H^i = \begin{bmatrix} H^i & \bar{H}^i \\ \bar{H}^i & H^i \end{bmatrix}, \; H^1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \\ H_1^i = \begin{bmatrix} H^{i-1} & \bar{H}^{i-1} \\ \bar{H}^{i-1} & H^{i-1} \end{bmatrix}; \\ H_4^i = H_1^i = \begin{bmatrix} H^{i-1} & \bar{H}^{i-1} \\ \bar{H}^{i-1} & H^{i-1} \end{bmatrix}; \\ H_2^i = \bar{H}_1^i = (2^n - 1) - \begin{bmatrix} H^{i-1} & \bar{H}^{i-1} \\ \bar{H}^{i-1} & H^{i-1} \end{bmatrix}; \\ H_3^i = \bar{H}_2^i = (2^n - 1) - \begin{bmatrix} H^{i-1} & \bar{H}^{i-1} \\ \bar{H}^{i-1} & H^{i-1} \end{bmatrix}.$$

На каждом шаге рекурсии формируются всегда четыре компонента, где два из них, расположенных по диагонали, переносятся из матрицы предыдущего шага, а два других генерируются путем дополнения каждого нового элемента в квадранте к соответствующим, относительно вертикальной или горизонтальной оси симметрии, битам перенесенных диагональных компонентов:

$$\bar{H}^i = (2^n - 1) - H^i.$$

Следует напомнить, что синтезированные столбцы матрицы используются для окончательного формирования кубитного покрытия дедуктивной функции путем перестановки битов, полученного для транспортирования входных векторов (списков) неисправностей на внешний выход функциональности при заданном входном тестовом наборе.

Таким образом, предложенный метод синтеза дедуктивных кубитных покрытий для моделирования неисправностей отличается от известных в мире аналогов оригинальностью математических решений, высоким уровнем параллелизма и компактностью структур данных, что дает возможность использовать его программную (аппаратную) реализацию для синтеза, анализа, тестирования, верификации и диагностирования цифровых систем на кристаллах. Высокое быстродействие синтеза дедуктивных кубитных покрытий на заданном входном наборе, определяемое битовыми операциями $Q=2x2^n=2^{n+1}$, является основанием для его использования в целях тестирования. моделирования и диагностирования цифровых систем в режиме online. Регистровая реализация фактически двух операций: xor-сравнения и перестановки битов позволит свести упомянутую оценку к двум (нескольким) автоматным тактам. Кроме того, две упомянутые операции можно объединить в одну процедуру, исполняемую в автоматном такте.

5. Синтез тестов для логических Х-функций

Сущность или уникальная особенность теста для любой Х-функции заключается в его непостроении, поскольку существуют две Т-аксиомы, объясняющие отсутствие синтеза: 1) Логическая Х-функция, записанная в виде единичных термов СДНФ

$$T_i \in T,\ f(T_i)=1,$$

представляет собой тестовые наборы для проверки одиночных константных 0-неисправностей входных, внутренних переменных и выхода:

$$T^1(\equiv 0)=\bigvee_{\forall i[f(T_i)=1]} T_i,$$

$$T=\{T_0, T_1,..., T_i,..., T_k\},$$

$$k=2^{2^n}-1;\ f(T_i)=\{0,1\}.$$

2) Тест в форме СДНФ логической Х-функции всегда дополняется единственной входной последовательностью – любым термом обратной СДНФ:

$$T_i \in T^0,\ f(T_i)=0,$$

который проверяет все константные 1-неисправности внутренних линий и выхода:

$$T^0(\equiv 1)=T_i \leftarrow \forall i: f(T_i)=0.$$

В частности, дополнение к 1-тесту Х-функции (Q=01101001) определяется нулевым тестовым набором (000) по всем входным координатам; дополнение к 1-тесту для Х-функции (Q=10010110) задается первым тестовым набором (001), содержащим в последнем разряде единицу на фоне всех остальных нулей. На самом деле дополнением к 1-тесту является любая входная последовательность, которая отсутствует в 1-тесте. Поэтому полным проверяющим тестом для Х-функции всегда является тест размерностью

$$Q=1+\frac{1}{2}\times 2^{2^n}.$$

Следовательно, тестом для логической Х-функции от n переменных является ее СДНФ, дополненная любым термом обратной СДНФ данной функции:

$$T=T^1\vee T_i^0,$$

$$T^1=\forall T_i: f(T_i)=1$$

$$T_i^0 \in T^0=\forall T_i: f(T_i)=0;$$

$$T(01101001)=(001\vee 010\vee 100\vee 111)\vee 000;$$

$$T(10010110)=(000\vee 011\vee 101\vee 110)\vee 001.$$

Другое уникальное свойство Х-функции определяется как покоординатная xor-сумма всех кодов, соответствующих СДНФ (обратной СДНФ), равна нулевому (по всем двоичным разрядам) вектору:

$$\bigoplus_{i=1}^{n_1} C(T_i^1)=0;$$

$$\bigoplus_{i=1}^{n_0} C(T_i^0)=0;$$

$$n_0+n_1=2^{2^n}.$$

Иллюстрация факта конволюции пространства СДНФ или обратной СДНФ в нуль-вектор представлена двумя Х-функциями от трех переменных:

$$Y(01101001)=\bar{X}_1\bar{X}_2X_3\vee\bar{X}_1X_2\bar{X}_3\vee X_1\bar{X}_2\bar{X}_3\vee X_1X_2X_3\rightarrow$$
$$\rightarrow 001\oplus 010\oplus 100\oplus 111=000;$$

$$Y(10010110)=\bar{X}_1\bar{X}_2\bar{X}_3\vee\bar{X}_1X_2X_3\vee X_1\bar{X}_2X_3\vee X_1X_2\bar{X}_3\rightarrow$$
$$\rightarrow 000\oplus 011\oplus 101\oplus 110=000.$$

Свойство конволюции дает возможность вычислять неизвестные термы теста или СДНФ на основе известных компонентов путем применения, например, следующего равенства для Х-функции от трех переменных:

$$T_1\oplus T_2\oplus T_3\oplus T_4=0;$$
$$001\oplus 010\oplus 100\oplus 111=0(000).$$
$$T_2\oplus T_3\oplus T_4=T_1;$$
$$010\oplus 100\oplus 111=001.$$

Для логических Х-функций от двух переменных (рис. 6), которые известны как xor, not-xor примитивы, ниже представлено моделирование исправного поведения всех входных наборов (таблица Т), анализ неисправностей (таблица D) и кубитная форма четырех вариантов минимальных тестов – таблица T(Q):

T(xor)	1	2	3	4	5
0	0	0	0	0	0
1	0	1	0	1	1
2	1	0	1	0	1
3	1	1	0	0	0

→

D	1	2	3	4	5
0	1	1	1	1	1
1	1	0	.	0	0
2	0	1	0	.	0
3	0	0	1	1	1

→

T(Q)	1	2	3	4
0	0	1	1	1
1	1	0	1	1
2	1	1	0	1
3	1	1	1	0

T(nxr)	1	2	3	4	5
0	0	0	1	0	1
1	0	1	0	0	0
2	1	0	0	0	0
3	1	1	0	1	1

→

D	1	2	3	4	5
0	1	1	0	.	1
1	1	0	1	1	0
2	0	1	1	1	0
3	0	0	.	0	1

→

T(Q)	1	2	3	4
0	0	1	1	1
1	1	0	1	1
2	1	1	0	1
3	1	1	1	0

Для обеих функций здесь получены минимальные тесты, состоящие из трех входных наборов. Это связано с тем, что противоположные входные векторы имеют одинаковые состояния выходной переменной.

Тест, использующий Т-аксиомы для записи входных последовательностей, проверяющих все одиночные константные неисправности цифровых схем (xor, nxr), имеет следующий вид:

$$T(\text{xor},\text{nxr}) = T^1 \vee T_i^0,$$
$$T(Q = 0110)(\text{xor}) = (01 \vee 10) \vee 00;$$
$$T(Q = 1001)(\text{nxr}) = (00 \vee 11) \vee 01.$$

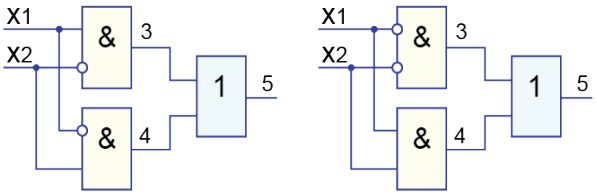

Рис. 6. Х-функции от двух переменных

Для двух логических Х-функций от одной переменной (рис. 7)

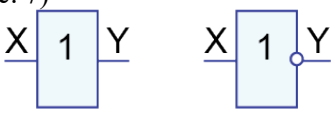

Рис. 7. Х-функции от одной переменной

ниже представлено моделирование исправного поведения всех входных наборов (таблица Т), анализ неисправностей (таблица D) и кубитная форма минимальных тестов – таблица T(Q), рассматриваются функциональные элементы (повторитель – rep, инвертор – not):

T(rep)	X	Y
0	0	0
1	1	1

→

D	X	Y
0	1	1
1	0	0

→

T(Q)
1
1

T(not)	X	Y
0	0	1
1	1	0

→

D	X	Y
0	1	0
1	0	1

→

T(Q)
1
1

Тест, использующий Т-аксиомы для записи входных последовательностей, проверяющих все одиночные константные неисправности цифровых примитивов (rep, not), имеет следующий вид:

$$T(\text{rep},\text{not}) = T^1 \vee T_i^0,$$
$$T(Q = 01)(\text{rep}) = 1 \vee 0;$$
$$T(Q = 10)(\text{not}) = 0 \vee 1.$$

Таким образом, применение двух упомянутых выше Т-аксиом дает возможность записывать без вычислений полный проверяющий тест для одиночных константных неисправностей входных, внутренних и выходных линий любой сколь угодно сложной логической Х-функции.

Выводы

1) Разработана структурная модель взаимодействия Х-функций и производных компонентов, ориентированных на синтез и анализ цифровых систем в целях получения тестопригодных решений, связанных с уменьшением времени проектирования и тестирования вычислительных устройств.

2) Впервые введено понятие простых Х-функций от конечного числа переменных, которые характеризуются отсутствием минимизации и наличием свойств тестопригодности, что дает возможность синтезировать цифровые устройства, технологичные для решения задач тестирования, моделирования и диагностирования.

3) Сформулированы метрические свойства Х-функций от конечного числа переменных, которые дают возможность использовать их в практике разработки тестопригодных цифровых устройств, генерирования проверяющих тестов и оценки их качества путем дедуктивного моделирования проверяемых константных неисправностей на кубитных структурах данных.

4) Предложено аналитическое выражение для синтеза кубитных покрытий Х-функций от конечного числа переменных, что дает возможность создавать тестопригодные логические схемы, не требующие экспоненциальных затрат на синтез и анализ тестов проверки и диагностирования неисправностей.

5) Получены дедуктивные формулы транспортирования входных списков неисправностей на внешние выходы для Х-функций от конечного числа переменных, которые характеризуются единичными векторами производных по всем переменным, что дает возможность построить секвенсор моделирования дефектов, инвариантный к входным тестовым наборам.

6) Предложен метод синтеза дедуктивных кубитных покрытий для моделирования неисправностей на основе использования Q-покрытий функциональностей, который отличается от известных в мире аналогов оригинальностью математических решений, высоким уровнем парал-

лелизма и компактностью структур данных, что дает возможность использовать его программную (аппаратную) реализацию для синтеза, анализа, тестирования, верификации и диагностирования цифровых систем на кристаллах.

7) Предложен метод синтеза тестов на основе кубитных покрытий Х-функций, который имеет линейную вычислительную сложность от числа переменных.

Дальнейшие исследования связаны с созданием технологий преобразования фрагментов логических схем к форме Х-функций, которые технологичны для решения задач тестирования и верификаци цифровых систем.

Литература:

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

2. *Hahanov V.I., Bani Amer T., Chumachenko S.V., Litvinova E.I.* Qubit technology for analysis and diagnosis of digital devices // Electronic modeling, J 2015. 37 (3). P. 17-40.

3. *Hahanov V., Gharibi W., Litvinova E., Liubarskyi M., Hahanova A.* Quantum memory-driven computing for test synthesis // IEEE East-West Design and Test Symposium, Novi Sad, Serbia. 2017. Pp. 123-128.

4. *Hahanov V.* Infrastructure intellectual property for SoC simulation and diagnosis service // Design of Digital Systems and Devices. Springer. 2011. P. 289-330.

5. *Abramovici M.* Digital System Testing and Testable Design, Comp. Sc. Press, 1998.

6. *Fujiwara H.* Fault Simulation // Logic Testing and Design for Testability. MIT Press, 1985. P.84-108.

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

8. *Hahanov V., Barkalov A., Adamsky M.* Design of Digital Systems and Devices. Infrastructure intellectual property for SoC simulation and diagnosis service. 2011. Springer. P. 289-330.

Поступила в редколлегию 11.02.2018

Рецензент: д-р техн. наук, проф. Меликян В.

Любарский Михаил Михайлович, соискатель кафедры АПВТ ХНУРЭ. Научные интересы: проектирование и тестирование цифровых систем. Хобби: путешествия. Адрес: Украина, 61166, Харьков, пр. Науки, 14.

Абдуллаев Вугар Гаджимахмудович, канд. техн. наук, доцент кафедры «Компьютерная инженерия технологии и программирование» Азербайджанской Государственной Нефтяной Академии (АГНА), Институт Кибернетики НАНА. Научные интересы: информационные технологии, веб-программирования, мобильные приложении. Увлечение: электронная коммерция, B2B, B2C проекты, научные книги, спорт. Адрес: Азербайджан, AZ1129, Баку, ул. М. Гади, 53, кв. 81, тел. (99412)5712428, (050)3325483, e-mail: abdulvugar@mail.com

Хаханов Владимир Иванович, д-р техн. наук, профессор, главный научный сотрудник кафедры АПВТ ХНУРЭ. Научные интересы: проектирование и тестирование цифровых систем. Хобби: футбол, горные лыжи. Адрес: Украина, 61166, Харьков, пр. Науки, 14, e-mail: hahanov@icloud.com.

Чумаченко Светлана Викторовна, д-р техн. наук, профессор, зав. кафедрой АПВТ ХНУРЭ. Научные интересы: математическое моделирование вычислительных процессов, теория рядов, методы дискретной оптимизации, инновационные формы обучения. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +380577021326, e-mail: svetlana.chumachenko@nure.ua

Литвинова Евгения Ивановна, д-р техн. наук, проф. кафедры АПВТ ХНУРЭ. Научные интересы: проектирование и тестирование цифровых систем. Хобби: музыка. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +380577021326, e-mail: litvinova_eugenia@icloud.com.

Хаханов Иван Владимирович, студент ХНУРЭ. Научные интересы: техническая диагностика цифровых систем, программирование. Хоби: горные лыжи, английский язык. Адреса: Украина, 61166, Харьков, пр. Науки, 14, тел. +3805770-21-326,e-mail: ivanhahanov@icloud.com.

Lyubarsky Mikhail Mikhailovich, PhD student, Design Automation Department, NURE. Scientific interests: project-bathing and testing digital systems. Scientific interests: design and testing of digital systems. Hobbies: traveling. Address: Ukraine, 61166, Kharkov, Nauki Ave, 14.

Abdullaev Vugar Gadzhimakhmudovich, Cand. tech. Sci., Associate Professor of Computer Engineering and Technology Programming at the Azerbaijan State Oil Academy (ASAN), Institute of Cybernetics of ANAS. Scientific interests: information technology, web programming, mobile application. Hobbies. e-commerce, B2B, B2C projects, science books, sports. Address: Azerbaijan, AZ1129, Baku, M. Gadi, 53, apt. 81, tel. (99412) 5712428, (050) 3325483, e-mail: abdulvugar@mail.com

Hahanov Vladimir Ivanovich, Dr., Prof., Chief Scientific Officer, Design Automation Department, NURE. Scientific interests: design and testing of digital systems. Hobby: football, downhill skiing. Address: Ukraine, 61166, Kharkov, Science, 14, e-mail: hahanov@icloud.com.

Chumachenko Svetlana Victorovna, Dr., Prof., Head of Design Automation Department, NURE. Scientific interests: mathematical modeling of computational processes, theory of series, methods of discrete optimization, educational innovations. Address: Ukraine, 61166, Kharkov, Nauki Ave, 14, phone + 3805770-21-326, e-mail: svetlana.chumachenko@nure.ua

Litvinova Evgenia Ivanovna, Dr., Prof., Design Automation Department, NURE. Scientific interests: design and testing of digital systems. Hobbies: music. Address: Ukraine, 61166, Kharkov, Nauki Ave, 14, e-mail: litvinova_eugenia@icloud.com.

Hahanov Ivan Vladimirovich, student, Design Automation Department, NURE. Scientific interests: technical diagnostics of digital systems, programming. Hobby: mountain skiing, English. Address: Ukraine, 61166, Kharkov, Nauki Ave., 14, ph. + 3805770-21-326, e-mail: ivanhahanov@icloud.com.