Д. т. н. В. И. ХАХАНОВ, ХАК Х. М. ДЖАХИРУЛ, МАСУД М. Д. МЕХЕДИ

Украина, Харьковский гос. технич. ун-т радиоэлектроники

Дата поступления в редакцию
12.07 2000 г.
Оппонент *д. т. н. Г. Ф. КРИВУЛЯ*

МОДЕЛИ АНАЛИЗА НЕИСПРАВНОСТЕЙ ЦИФРОВЫХ СИСТЕМ НА ОСНОВЕ FPGA, CPLD

Предлагаются модели и методы анализа цифровых проектов для генерации тестов и моделирования одиночных константных неисправностей.

Программируемые логические интегральные схемы (ПЛИС) — Field Programable Gate Array (FPGA), Complex Programable Logic Device (CPLD) — достойно конкурируют с базовыми матричными кристаллами (БМК), сигнальными процессорами. Такой успех определяется использованием субмикронных технологий изготовления кристаллов, применением Hardware-Software Cooperation-Design, минимальным временем проектирования цифровой системы (4—5 месяцев), высоким быстродействием выполнения операций (до 500 МГц), большой степенью интеграции элементов на кристалле (до 3,5 млн.).

Наряду с преимуществами ПЛИС существуют и проблемы их тестирования, стимулирующие развитие таких методов, алгоритмов и программ, которые должны обеспечивать:

— тестирование цифровых проектов большой размерности, вентильного, функционального алгоритмического уровней описания, заданных в форме графов переходов конечных автоматов, булевых уравнений, многоуровневых иерархических структур;

— проектирование тестов в виде покрытия одномерных путей активизации для проверки одиночных константных неисправностей с полнотой, близкой к 100%;

— приемлемое быстродействие алгоритмов моделирования неисправностей для оценки качества тестов; построение алгоритмов поиска дефектов;

— верификацию и диагностирование синтезированных цифровых устройств на основе FPGA, CPLD;

— возможность параллельного выполнения векторных операций логического анализа для генерации тестов и оценки их качества;

— поддержку стандарта VHDL для описания цифрового устройства и полученного теста;

— возможность интегрирования в существующие системы автоматизированного проектирования ведущих фирм мира.

Теоретические источники настоящей работы: многозначная алгебра [1], двухтактная кубическая алгебра [2], дедуктивный метод моделирования неисправностей [3—6], методы генерации тестов [7].

Предпочтительным по быстродействию является дедуктивный метод моделирования неисправностей. Он позволяет за одну итерацию обработки схемы определить все константные дефекты, проверяемые на входном тестовом векторе. Но данный метод ориентирован на вентильный уровень описания цифровых схем. Это связано со сложностью решения проблемы генерирования выходных списков неисправностей (output fault list generation) для невентильных примитивов.

Предлагаемый в работе метод кубического моделирования неисправностей позволяет обрабатывать цифровые схемы, описанные на вентильном, функциональном и алгоритмическом уровнях. Обратной стороной решения упомянутой проблемы является метод генерации тестов для константных неисправностей, использующий кубические покрытия примитивов для построения одномерного пути активизации.

Математический аппарат анализа примитива

Автоматная модель последовательностного примитива представляется в виде

$$M=<X, Y, Z, f, g>,$$

где $X=(X_1, X_2, ..., X_i, ..., X_m)$, $Y=(Y_1, Y_2, ..., Y_i, ..., Y_h)$, $Z=(Z_1, Z_2, ..., Z_i, ..., Z_k)$ — множества входных, внутренних и выходных автоматных переменных, отношения между которыми описываются уравнениями

$$Y(t)=f[X(t-1), X(t), Y(t-1), Z(t-1)];$$

$$Z(t)=g[X(t-1), X(t), Y(t-1), Y(t), Z(t-1)]. \quad (1)$$

Переменные $Z(t)$ отличаются от $Y(t)$ тем, что первые наблюдаемы по выходным линиям, а $Y(t)$ в этом смысле есть внутренние. Формат автоматных переменных для записи кубического покрытия, соответствующий (1), имеет вид

X(t – 1)	Y(t – 1)	Z(t – 1)
X(t)	Y(t)	Z(t)

,

которому соответствует автоматная модель, изображенная на **рис. 1**.

Функциональный последовательностный примитив задается компонентами

$$F^2=<(t-1,t),(X,Z,Y),\{A^2\}>,$$

где (t−1,t) — два автоматных соседних такта в описании функции; (X,Z,Y) — векторы входных, внутренних и выходных переменных; $\{A^2\}$ – двухтактный алфавит описания состояний (переходов) автоматных переменных [1, 2] —

A^2={Q=00,E=01,H=10,J=11,O={Q,H},I={E,J},A={Q,E};
B={H,J},S={Q,J},P={E,H},C={E,H,J},F={Q,H,J},L={Q,E,J};
V={Q,E,H},Y={Q,E,H,J}, A^1={0,1,X={0,1}}, ∅(U)}.

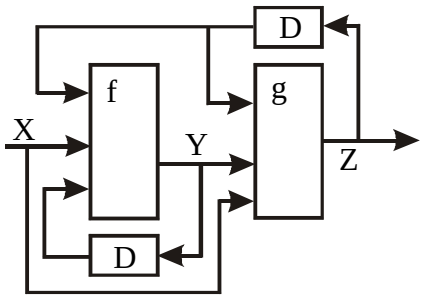

Рис. 1. Автоматная модель примитива

Примитив описывается кубическим покрытием

$C = (C_1, C_2, ..., C_i, ..., C_n)$,

где $C_i = (C_{i1}, C_{i2}, ..., C_{ij}, ..., C_{iq})$ – куб, включающий входные, внутренние, выходные координаты $C_i = (C_i^X, C_i^Y, C_i^Z)$, q=m+h+k.

Для комбинационного автомата формат описания кубического покрытия

$F^1 = <(t),(X,Z),\{A^1\}>$

определяется отношениями на (q=m+k)-мерном векторе переменных $C_i=(C_i^X, C_i^Z)$. Формат задает многовыходовый комбинационный примитивный элемент с m входами и k выходами. Двоичная булева функция от m переменных $Z=f(X_1,X_2,..., X_m)$ определяется при k=1.

Пример 1. Преимущества двухтактного алфавита A^2 демонстрируется на примере проектирования кубического покрытия CD-триггера. Исходное описание определено в виде таблицы переходов:

C_{CD} =

C_{t-1}	C_t	D_{t-1}	D_t	Q_{t-1}	Q_t
1	0	1	1	X	1
0	0	X	X	1	1
0	1	X	X	1	1
1	1	X	X	1	1
1	0	0	0	X	0
0	0	X	X	0	0
0	1	X	X	0	0
1	1	X	X	0	0

.

Далее применяется *С-процедура* получения двухтактного кубического покрытия:

1. Выполнение координатной операции конкатенации (#) для преобразования пары соседних во времени однотактных символов в один двухтактный:

$C_{t-1}\#C_t$ =

#	0	1	X
0	Q	E	A
1	H	J	B
X	O	I	Y

.

2. Итеративная минимизация полученного покрытия по правилу «два куба объединяются в один, если они отличаются по одной переменной».

3. Дополнение полученной модели примитива кубами описания возможных состязаний.

Применение данной процедуры к покрытию CD-триггера дает следующий результат:

C_{CD} =

C	D	Q
H	J	I
Q	Y	J
E	Y	J
J	Y	J
H	Q	O
Q	Y	Q
E	Y	Q
J	Y	Q

=

C	D	Q
H	J	I
L	Y	J
H	Q	O
L	Y	Q

=

C	D	Q
H	J	I
L	Y	S
H	Q	O

=

C	D	Q
H	J	I
L	Y	S
H	Q	O
H	P	X

.

В правое покрытие добавлен куб (HPX), который идентифицирует состязания при одновременном изменении сигналов на входах C и D. Полученное покрытие имеет только 12 символов, в то время как исходное содержит 48 координат.

Применение С-процедуры к таблице переходов двухразрядного счетчика еще более эффективно, чем для триггера. Процесс получения двухтактного покрытия счетчика представлен следующим выражением:

V	C	A_{t-1}	B_{t-1}	A_t	B_t
0	X	X	X	0	0
1	E	0	0	0	1
1	E	0	1	0	0
1	E	1	0	1	1
1	E	1	1	1	0
1	F	0	0	0	0
1	F	0	1	0	1
1	F	1	0	1	0
1	F	1	1	1	1

=

V	C	A	B
0	X	0	H
1	E	Q	E
1	E	E	H
1	E	J	E
1	E	H	H
1	F	Q	Q
1	F	Q	J
1	F	J	Q
1	F	J	J

=

V	C	A	B
0	X	0	0
1	E	S	E
1	E	P	N
1	E	S	S

.

Минимальное покрытие счетчика, записанное в двухтактном алфавите, имеет всего 16 координат. В общем случае размерность таблиц описания функции счета для n разрядов в одно- и двухтактном покрытии определяется отношением $h=2^n/n$.

Формулировка проблем тестирования

Проблемы тестирования цифрового устройства (примитива) формулируются в условиях неопределенности одного из компонентов модели:

W=(M,L,T),

где M – модель, представленная кубическим покрытием $C=(C_1, C_2, ..., C_i, ..., C_n)$; L – кубическое покрытие списков неисправностей (КПСН) (Fault Lists Cubic Covering – FLCC); T – тест проверки неисправностей.

Покрытие неисправностей для примитива или цифровой схемы задается в виде

$L= (L_1, L_2, ..., L_i, ..., L_n)$,

где $L_i = (L_{i1}, L_{i2}, ..., L_{ij}, ..., L_{iq})$ – куб, включающий входные, внутренние, выходные координаты: $L_i=(L_i^X,L_i^Y,L_i^Z)$, q=m+h+k; $(L_{ij}^Y,L_{ij}^Z)=\{0,1,X\}$; 0 – определяет вычитание (дополнение) списка L_j;

1 – задает пересечение L_j; X={0,1} – идентифицирует несущественность списка неисправностей L_j.

Если L_{ir}^Z – выходная наблюдаемая переменная, то (0) 1 – есть идентификатор (не-) проверки неисправностей куба L_i на выходе r, X={0,1} – задает неопределенное состояние выходной координаты L_{ir}^Z, которое можно интерпретировать и как 0, и как 1.

Проблема 1. Кубическое покрытие списков неисправностей L для вектора T и покрытия примитива C вычисляется по линейному уравнению

$$T \oplus C = L, \qquad (2)$$

где ⊕ – бинарная координатная операция XOR, определяющая взаимодействие компонентов T, C (L) в троичном алфавите:

$$T_j \oplus C_{ij} = \begin{array}{|c|c|c|c|}\hline \oplus & 0 & 1 & X \\ \hline 0 & 0 & 1 & X \\ \hline 1 & 1 & 0 & X \\ \hline X & X & X & X \\ \hline \end{array}. \qquad (3)$$

Универсальная формула анализа КПСН, полученного в результате применения (3) к тест-вектору T и покрытию многовыходового примитива C для определения по выходу r списка проверяемых неисправностей L_r, имеет следующий вид:

$$L = \bigcup_{\forall i(T_r \oplus C_{ir}=1)} \bigcap_{j=1}^{k} L_j^{T_j \oplus C_{ij}}, \qquad (4)$$

где $L_j^{T_j \oplus C_{ij}} = \begin{cases} L_j \leftarrow T_j \oplus C_{ij} = 1; \\ \overline{L_j} \leftarrow T_j \oplus C_{ij} = 0; \end{cases}$

$\overline{L_j}$ – рассматривается как список дефектов, относящийся к линии j, который следует вычитать из неисправностей, проверяемых по невыходным линиям примитива; L_j – неисправности, которые необходимо пересекать с невыходными списками.

Замечание 1. Если тест-вектор определен в троичном алфавите T_j={0,1,X}, то после получения покрытия списков неисправностей каждый куб L_j следует верифицировать по правилу

$$L = L - L_i \Leftarrow (L_i \oplus T \neq C_i),\ i = \overline{1,r}. \qquad (5)$$

Здесь и далее знак «–» эквивалентен теоретико-множественному вычитанию.

Пример 2. Для мультиплексора, описанного функцией $f(a,b,c) = a\bar{c} \vee bc$, КПСН для тестового набора T=(1010) определяется по формуле (2):

$$T(1010) \oplus C_{MUX} \begin{array}{|c|c|c|c|}\hline a & b & c & d \\ \hline 0 & X & 0 & 0 \\ \hline 1 & X & 0 & 1 \\ \hline X & 0 & 1 & 0 \\ \hline X & 1 & 1 & 1 \\ \hline \end{array} = L \begin{array}{|c|c|c|c|}\hline a & b & c & d \\ \hline 1 & X & 1 & 0 \\ \hline 0 & X & 1 & 1 \\ \hline X & 0 & 0 & 0 \\ \hline X & 1 & 0 & 1 \\ \hline \end{array}. \qquad (6)$$

Здесь L – кубическое покрытие списков неисправностей, по которому можно записать дизъюнктивную нормальную форму (ДНФ) или теоретико-множественное уравнение для вычисления выходного списка дефектов на векторе (1010):

$$l = \bar{a}c \vee b\bar{c} = (c - a) \cup (b - c).$$

Из (6) следует, что собственный список проверяемых входных одиночных константных неисправностей мультиплексора представлен множеством

$$L = \{b^1, c^0).$$

В общем случае для вычисления собственных неисправностей примитива применяется F-процедура:

1) терм дизъюнктивной нормальной формы, имеющий более одной переменной без инверсии или все переменные с инверсией, вычеркивается;

2) оставшиеся термы должны иметь одну переменную без инверсии, которая формирует собственную проверяемую неисправность $j^{\overline{T}_j}$, инверсную состоянию координаты тест-вектора T_j.

Пример 3. Задано кубическое покрытие функции

$$C(\bar{X}_2\bar{X}_3 \vee X_1X_2X_3) = \begin{array}{|c|c|c|c|}\hline X_1 & X_2 & X_3 & Y \\ \hline X & 0 & 0 & 1 \\ \hline 1 & 1 & 1 & 1 \\ \hline X & 0 & 1 & 0 \\ \hline X & 1 & 0 & 0 \\ \hline 0 & 1 & X & 0 \\ \hline \end{array}. \qquad (7)$$

Списки идентификаторов неисправностей для трех входов определены подмножествами

$$L_1 = \{1,2,4,5\};\ L_2 = \{1,2,3,6\};\ L_3 = \{1,3,4,7\}. \qquad (8)$$

Определить выходной список неисправностей для тест-вектора 1111.

Решение сводится к выполнению ⊕-операции между вектором T и покрытием C:

$$T(1111) \oplus C \begin{array}{|c|c|c|c|}\hline X_1 & X_2 & X_3 & Y \\ \hline X & 0 & 0 & 1 \\ \hline 1 & 1 & 1 & 1 \\ \hline X & 0 & 1 & 0 \\ \hline X & 1 & 0 & 0 \\ \hline 0 & 1 & X & 0 \\ \hline \end{array} = L \begin{array}{|c|c|c|c|}\hline X_1 & X_2 & X_3 & Y \\ \hline X & 1 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 \\ \hline X & 1 & 0 & 1 \\ \hline X & 0 & 1 & 1 \\ \hline 1 & 0 & X & 1 \\ \hline \end{array}.$$

Далее по единичным значениям выхода Y в покрытии L записывается дизъюнктивная нормальная форма (теоретико-множественное уравнение) проверяемых входных списков:

$$L = X_2\bar{X}_3 \vee \bar{X}_2X_3 \vee X_1\bar{X}_2 =$$

$$= (X_2 - X_3) \cup (X_3 - X_2) \cup (X_1 - X_2). \qquad (9)$$

Подстановкой вместо переменных X_i соответствующих списков L_i получается следующий результат:

$$L = (\{1,2,3,6\} - \{1,3,4,7\}) \cup (\{1,3,4,7\} - \{1,2,3,6\}) \cup$$
$$\cup (\{1,2,4,5\} - \{1,2,3,6\}) = \{2,4,5,6,7\}. \qquad (10)$$

Проблема 2. Тест проверки дефектов примитива, задаваемых L_i^1-кубом покрытия списка неисправностей $L = (L_i^0, L_i^1)$, определяется по уравнению

$$L_i^1 \oplus C = T^k, \qquad (11)$$

где L_i^0, L_i^1 – кубы, имеющие нулевые или единичные значения на выходной координате r.

При этом определяются векторы-кандидаты в тест $T_t^k \in T^k$. Из множества T^k формируется тест T, где в качестве элемента рассматривается набор $T_t \in T$. Каждый $T_t \in T$ должен удовлетворять условиям

$$T_t = C_i \cap T_t^k \Leftarrow \exists i (C_i \cap T_t^k \neq \varnothing). \qquad (12)$$

Пример 4. Для функции AND-NOT построить тест, проверяющий списки неисправностей, заданные кубом L(011).

Ниже приведена процедура получения теста на основе последовательного применения формул (11) и (12).

$$L(011)\oplus C_{AND}\begin{vmatrix} a & b & c \\ 0 & X & 1 \\ X & 0 & 1 \\ 1 & 1 & 0 \end{vmatrix} = T^k\begin{vmatrix} a & b & c \\ 0 & X & 0 \\ X & 1 & 0 \\ 1 & 0 & 1 \end{vmatrix} = T\begin{vmatrix} a & b & c \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{vmatrix}. \quad (13)$$

В результате получены 2 набора, которые одномерно активизируют входную переменную b. Следовательно, тест является полным относительно проверки одиночных константных дефектов на линиях b, c.

Определение. Тест проверки одиночных константных неисправностей одновыходового примитива, задаваемых L^1-покрытием списка дефектов

$$L^1 = \begin{vmatrix} X_1 & X_2 & ...X_j... & X_m & Z \\ 1 & 0 & ...0... & 0 & 1 \\ 0 & 1 & ...0... & 0 & 1 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ...1... & 1 & 1 \\ ... & ... & ... & ... & ... \\ 0 & 0 & ...0... & 1 & 1 \end{vmatrix}, \quad (14)$$

определяется Т-процедурой:

1. Формирование кандидатов в тест $T_t^k \in T^k$ путем выполнения операций над векторами из L^1,C:

$$L_i^1 \oplus C = T^k. \quad (15)$$

2. Получение теста $T(T_t \in T)$ из множества кандидатов T^k на основе применения выражения (12).

3. Минимизация множества тест-векторов Т путем выполнения операции поглощения:

$$T = T - T_r \Leftarrow T_i \cap T_r = T_r.$$

Пример 5. Построить тест проверки всех одиночных константных неисправностей для функции, заданной покрытием (7).

Генерируется L^1-покрытие списка дефектов по аналогии с (14) и для строки L_i^1 выполняется ⊕-операция по (11):

$$L^1\begin{vmatrix} X_1 & X_2 & X_3 & Y \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{vmatrix}\oplus C\begin{vmatrix} X_1 & X_2 & X_3 & Y \\ X & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ X & 0 & 1 & 0 \\ X & 1 & 0 & 0 \\ 0 & 1 & X & 0 \end{vmatrix} = T^k(L_i^1), i=\overline{1,3}. \quad (16)$$

К полученным подмножествам кандидатов в тест

$T^k(L_1^1)$

X_1	X_2	X_3	Y
X	0	0	0
0	1	1	0
X	0	1	1
X	1	0	1
1	1	X	1

;

$T^k(L_2^1)$

X_1	X_2	X_3	Y
X	1	0	0
1	0	1	0
X	1	1	1
X	0	0	1
0	0	X	1

;

$T^k(L_3^1)$

X_1	X_2	X_3	Y
X	0	1	0
1	1	0	0
X	0	0	1
X	1	1	1
0	1	X	1

применяется процедура (12), которая определяет в данном случае уже минимальный тест, гарантированно проверяющий схемную структуру, которая соответствует двухуровневой реализации булевой функции (7):

$$T = \begin{vmatrix} X_1 & X_2 & X_3 & Y \\ 0 & 1 & 1 & 0 \\ 1 & 1 & X & 1 \\ X & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ X & 1 & 1 & 1 \\ X & 0 & 0 & 1 \\ 0 & 0 & X & 1 \\ X & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ X & 0 & 0 & 1 \\ X & 1 & 1 & 1 \end{vmatrix}. \quad (17)$$

Проблема 3. Определяется возможность верификации результатов, полученных при решении первых двух проблем, —

$$T \oplus L = C. \quad (18)$$

Необходимость данной процедуры связана с многочисленными ошибками, возникающими в процессе ручного или автоматизированного проектирования тестов. Кроме того, согласно замечанию 1, решение проблемы 3 обязательно для верификации покрытия неисправностей при наличии в тесте символов X. Для полученных наборов (17) верификация путем выполнения операций $L_i^1 \oplus T(L_i^1)$, $i=\overline{1,3}$ дает положительный результат:

$$L^1\begin{vmatrix} X_1 & X_2 & X_3 & Y \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{vmatrix}\oplus T\begin{vmatrix} X_1 & X_2 & X_3 & Y \\ 0 & 1 & 1 & 0 \\ 1 & 1 & X & 1 \\ X & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ X & 1 & 1 & 1 \\ X & 0 & 0 & 1 \\ 0 & 0 & X & 1 \\ X & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ X & 0 & 0 & 1 \\ X & 1 & 1 & 1 \end{vmatrix} = C\begin{vmatrix} X_1 & X_2 & X_3 & Y \\ X & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ X & 0 & 1 & 0 \\ X & 1 & 0 & 0 \\ 0 & 1 & X & 0 \end{vmatrix}.$$

Это означает, что любой вектор $L_i^1 \oplus T(L_i^1) \in C$ является кубом исходного покрытия функции. Следовательно, тест проверки неисправностей не имеет ошибок проектирования.

Моделирование неисправностей в последовательностных примитивных элементах

Функциональная зависимость списка дефектов по наблюдаемому выходу Z_r согласно (1) определяется уравнением

$$L_r^t = f\left[(T),(C),(L_X^{t-1},L_X^t,L_Y^{t-1},L_Z^{t-1})\right], \quad (19)$$

где $T_t \in T=(T_1,...,T_t,...,T_p)$ — пара соседних входных воздействий, где каждая координата определена в следующих сочетаниях:

$$T_t = \left|\frac{0}{0},\frac{0}{1},\frac{1}{0},\frac{1}{1},\frac{0}{X},\frac{1}{X},\frac{X}{0},\frac{X}{1},\frac{X}{X}\right|. \quad (20)$$

Двухфреймовый формат входного набора ориентирован на анализ последовательностного примитива, поскольку его покрытие в общем случае задано в двухтактном алфавите A^2. Для технологичного выполнения операции XOR между координатами тест-вектора и кубического покрытия $T_{ij} \oplus C_{ij}$ используется **табл. 1**.

Каждая координата таблицы есть сокращенная форма записи списков неисправностей $L_j = T_{ij} \oplus C_{ij}$. Например, если $T_{ij} = (01)$, $C_{ij} = P$, то по входной координате j в соответствии с табл. 1 получается:

$$T_{tj}(01) \oplus P\{(01),(10)\} = (\bar{L}_j^{t-1} \bar{L}_j^{t}) \vee (L_j^{t-1} L_j^{t}).$$

Таблица 1

⊕	0	1	X	Z	G	T	K	N	Q	E	H	J	O	I	A	B	S	P	C	F	L	V	Y
0 0	0	1	X	Z	G	T	K	N	Q	E	H	J	O	I	A	B	S	P	C	F	L	V	Y
0 1	1	0	X	Z	T	G	K	N	E	Q	J	H	I	O	A	B	P	S	F	C	V	L	Y
1 0	0	1	X	Z	G	T	K	N	H	J	Q	E	O	I	B	A	P	S	L	V	C	F	Y
1 1	1	0	X	Z	T	G	K	N	J	H	E	Q	I	O	B	A	S	P	V	L	F	C	Y
0 X	X	X	X	X	G	T	K	N	A	A	B	B	Y	Y	A	B	Y	Y	B	B	A	A	Y
1 X	X	X	X	X	T	G	K	N	B	B	A	A	Y	Y	B	A	Y	Y	A	A	B	B	Y
X 0	0	1	X	Z	X	X	X	X	O	I	O	I	O	I	Y	Y	Y	Y	I	O	I	O	Y
X 1	1	0	X	Z	X	X	X	X	I	O	I	O	I	O	Y	Y	Y	Y	O	I	O	I	Y
X X	X	X	X	X	X	X	X	X	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

Табл. 2 предназначена для доопределения выходных координат куба покрытия списков неисправностей в моменты (t−1, t). Например, для координаты L_{ij}=V ее значение в такте t−1 равно 0, а в момент t равно 1, что в табл. 2 определяется буквой E. Исключение составляют преобразования символов S, P. Здесь существует различие интерпретации в зависимости от того, является ли выходная переменная *функцией* или *аргументом* к выходу, для которого строится список проверяемых неисправностей. В первом случае доопределение упомянутых символов дает (J, E), во втором − (X, X), что свидетельствует об отсутствии списков неисправностей для данного выхода в момент t−1.

Таблица 2

0	1	X	Z	G	T	K	N	Q	E	H	J	O	I	A	B	S	P	C	F	L	V	Y
Z	1	1	Z	G	T	Z	Z	G	E	T	J	Z	1	E	J	J/X	E/X	1	J	1	E	1

Пример 6. Определить списки проверяемых неисправностей счетчика на двух тест-векторах 1101, 1001 при исходном состоянии переменных 1000.

Моделирование первого набора дает следующий результат:

$$T_1 \left| \begin{matrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \end{matrix} \right| \oplus C_{count} \left| \begin{matrix} V & C & A & B \\ 0 & X & 0 & 0 \\ 1 & E & S & E \\ 1 & E & P & H \\ 1 & F & S & S \end{matrix} \right| = L \left| \begin{matrix} V & C & A & B \\ 1 & X & 0 & 1 \\ 0 & Q & S & Q \\ 0 & Q & P & J \\ 0 & C & S & P \end{matrix} \right|.$$

$$L_A(T_1) = \bar{V}_t \bar{C}_{t-1} \bar{C}_t A_{t-1} \bar{B}_{t-1} \vee \bar{V}_t \bar{C}_{t-1} \bar{C}_t A_{t-1} \bar{B}_{t-1} \vee$$
$$\vee \bar{V}_t (C_{t-1} \vee C_t) A_{t-1} \vee A_t = A_{t-1} \vee B_{t-1} \vee A_t;$$

$$L_B(T_1) = V_t \vee \bar{V}_t \bar{C}_{t-1} \bar{C}_t \bar{B}_{t-1} \vee \bar{V}_t (\bar{C}_{t-1} \vee \bar{C}_t) \bar{B}_{t-1} \vee$$
$$\vee B_t = V_t \vee B_{t-1} \vee C_{t-1} \vee C_t \vee B_t;$$

$$L(T_1) = V_t^0 \vee C_{t-1}^1 \vee C_t^0 \vee A_{t-1}^1 \vee B_{t-1}^1 \vee A_t^1 \vee B_t^0.$$

Результат моделирования второго набора имеет вид

$$T_2 \left| \begin{matrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \end{matrix} \right| \oplus C_{count} \left| \begin{matrix} V & C & A & B \\ 0 & X & 0 & 0 \\ 1 & E & S & E \\ 1 & E & P & H \\ 1 & F & S & S \end{matrix} \right| = L \left| \begin{matrix} V & C & A & B \\ 1 & X & 0 & 1 \\ 0 & J & S & H \\ 0 & J & P & E \\ 0 & V & S & S \end{matrix} \right|.$$

$$L_A(T_2) = \bar{V}_t C_{t-1} C_t A_{t-1} B_{t-1} \vee \bar{V}_t C_{t-1} C_t \bar{A}_{t-1} \bar{B}_{t-1} \vee$$
$$\vee \bar{V}_t (\bar{C}_{t-1} \vee \bar{C}_t) A_{t-1} \vee A_t = A_{t-1} \vee A_t;$$

$$L_B(T_2) = V_t \vee \bar{V}_t C_{t-1} C_t \bar{B}_{t-1} \vee \bar{V}_t (\bar{C}_{t-1} \vee \bar{C}_t) B_{t-1} \vee B_t = V_t \vee B_t;$$

$$L(T_2) = V_t^0 \vee A_{t-1}^1 \vee A_t^1 \vee B_t^0.$$

Таким образом, проверяемые на выходах счетчика неисправности составляют множества $L(T_1)$ и $L(T_2)$, где знак константной неисправности отмечен верхним индексом.

⊕-алгоритм кубического моделирования неисправностей цифровых схем

На основе применения формулы (4) определяется ⊕-алгоритм кубического дедуктивного моделирования неисправностей цифрового устройства.

1. Моделирование исправного поведения очередного примитива P_i ($i = \overline{1, M}$) на тест-векторе T_t ($t = \overline{1, N}$). Если t=N − формируется список L(T) проверенных неисправностей на тесте T. Конец моделирования. Иначе, t< N − переход к п. 2.

2. Если все элементы схемы обработаны (i−M), выполняется сравнение векторов исправных состояний линий в двух соседних итерациях. Если векторы идентичны $T_t^r = T_t^{r-1}$, конец моделирования T_t и переход к п. 3. Иначе − переход к п. 1.

3. Определение списков неисправностей внешних входов в виде дополнения к их исправному состоянию $L_j = \left\{ j^{\bar{T}_j} \right\}$. Для невходных линий устройства выполняется операция присвоения

$$Lj = \varnothing (j = \overline{m+1, q}).$$

4. Моделирование неисправностей примитива P_i ($i = \overline{1, M}$) по процедуре (4). Дополнение полученного списка неисправностью выходной линии примитива, идентифицируемой в виде $\left\{ j^{\bar{T}_j} \right\}$.

5. Если после обработки всех примитивов на двух соседних итерациях выполняется условие $L_j^{r-1}=L_i^r$, ($j=\overline{1,q}$), осуществляется переход к п. 6. Иначе — к п. 4.

6. Формирование списка проверяемых дефектов

$$L(T_t)=\bigcup_{j=m+h+1}^{q} L_j$$

по всем наблюдаемым выходам устройства и переход к п. 1.

Быстродействие описанного выше ⊕-алгоритма сквозного интерпретативного синхронного моделирования неисправностей для одной итерации имеет оценку

$$W=\sum_{i=1}^{M}\{q_i\times n_i\times[(0{,}01\times Q)^2+3]+2\},$$

где n_i, q_i – число кубов и переменных в покрытии C_i; $(0{,}01\times Q)$ – средняя длина списка неисправностей для каждой линии схемы; Q – общее число линий в цифровом устройстве.

Пример 7. Выполнить моделирование последовательностной схемы, представленной на **рис. 2** [8, с. 177].

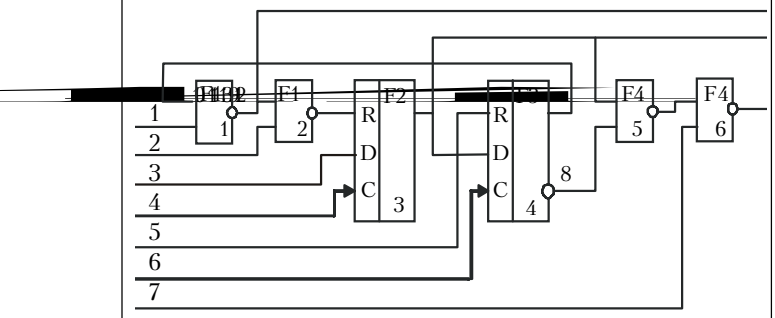

Рис. 2. Схема с обратной связью

Функции четырех типов примитивов схемы описаны кубическими покрытиями:

C(F1)

11	1	12
12	2	9
1	X	0
X	1	0
0	0	1

;

C(F2)

9	3	4	13
1	X	X	0
0	0	E	0
0	1	E	1
0	X	1	S

;

C(F3)

5	13	6	11	8
1	X	X	0	1
0	0	E	0	1
0	1	E	1	0
0	X	F	Q	J
0	X	F	J	Q

;

C(F4)

13	8	10
10	7	14
0	X	1
X	0	1
1	1	0

.

Моделирование исправного поведения на восьми двоичных входных наборах в соответствии с пунктами 1 и 2 ⊕-алгоритма кубического моделирования дает следующий результат:

1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	1	0	1	0	0	1	1	1	0	0	0	1
1	1	1	1	0	1	0	0	0	1	1	0	1	1
1	0	1	0	0	1	1	0	1	1	1	0	0	0
1	1	1	1	1	0	1	1	0	1	1	0	1	1
0	1	1	0	0	1	1	0	0	1	1	0	1	0
0	0	1	0	0	0	1	0	1	1	1	0	0	0
1	1	0	1	1	1	1	1	0	1	0	0	0	0
0	1	1	0	1	0	1	1	0	1	0	1	0	0

.

Далее выполняется моделирование одиночных константных неисправностей для первого вектора.

Исходные списки проверяемых неисправностей всех линий схемы, определяемые в соответствии с п. 3, имеют следующий вид:

$L_1=\{1^0\}$, $L_2=\{2^1\}$, $L_3=\{3^0\}$, $L_4=\{4^1\}$, $L_5=\{5^0\}$, $L_6=\{6^1\}$, $L_7=\{7^1\}$,

$L_8=L_9=L_{10}=L_{11}=L_{12}=L_{13}=\varnothing$.

Если очередное кубическое покрытие является однотактным, то формат тест-вектора можно задавать в троичном алфавите, который определяет состояния линий в момент времени t: | X(t) | Y(t) |.

Далее выполняется первая итерация обработки всех шести примитивов устройства (п. 4 алгоритма), которая представлена следующими вычислениями:

$$T_1\left|\begin{array}{cc|c}11&1&12\\\hline 0&1&1\end{array}\right|\oplus C_1\left|\begin{array}{cc|c}11&1&12\\\hline 1&X&0\\X&1&0\\0&0&1\end{array}\right|=L^1\left|\begin{array}{cc|c}11&1&12\\\hline 1&X&0\\X&0&0\\0&1&1\end{array}\right|\Rightarrow$$

$\Rightarrow L_{12}=L_1\bar{L}_{11}\vee 12^1=\{1^0, 12^1\}$.

$$T_1\left|\begin{array}{cc|c}12&2&9\\\hline 0&0&1\end{array}\right|\oplus C_2\left|\begin{array}{cc|c}12&2&9\\\hline 1&X&0\\X&1&0\\0&0&1\end{array}\right|=L^2\left|\begin{array}{cc|c}12&2&9\\\hline 1&X&1\\X&1&1\\0&0&0\end{array}\right|\Rightarrow$$

$\Rightarrow L_9=L_{12}\vee L_2\vee 9^0=\{1^0, 12^1, 2^1, 9^0\}$.

$$T_1\left|\begin{array}{ccc|c}9&3&4&13\\\hline X&X&X&X\\1&1&0&0\end{array}\right|\oplus C_3\left|\begin{array}{ccc|c}9&3&4&13\\\hline 1&X&X&0\\0&0&E&0\\0&1&E&1\\0&X&F&S\end{array}\right|=L^4\left|\begin{array}{ccc|c}9&3&4&13\\\hline 1&X&X&0\\0&0&E&0\\1&0&E&1\\1&X&O&Y\end{array}\right|\Rightarrow$$

$\Rightarrow L_{13}=L_9\bar{L}_3\bar{L}'_4L_4\vee 13^1=\{13^1\}$.

Здесь и далее штрих (′) идентифицирует момент времени t−1.

$$T_1\left|\begin{array}{ccc|cc}5&13&6&11&8\\\hline X&X&X&X&X\\1&0&0&0&1\end{array}\right|\oplus C_4\left|\begin{array}{ccc|cc}5&13&6&11&8\\\hline 1&X&X&0&1\\0&0&E&0&1\\0&1&E&1&0\\0&X&F&Q&J\\0&X&F&J&Q\end{array}\right|=L^4\left|\begin{array}{ccc|cc}5&13&6&11&8\\\hline 1&X&X&0&0\\0&0&E&0&0\\1&1&E&1&1\\1&X&O&O&I\\1&X&O&I&O\end{array}\right|\Rightarrow$$

$$\Rightarrow\begin{cases}L_{11}=L_5L_{13}\bar{L}'_6L_6\vee L_5\bar{L}_6\vee 11^1=\{5^0,11^1\};\\ L_8=L_5L_{13}\bar{L}'_6L_6\vee L_5\bar{L}_6\vee 8^0=\{5^0,8^0\}.\end{cases}$$

$$T_1\left|\begin{array}{cc|c}13&8&10\\\hline 0&0&1\end{array}\right|\oplus C_5\left|\begin{array}{cc|c}13&8&10\\\hline 0&X&1\\X&0&1\\1&1&0\end{array}\right|=L^5\left|\begin{array}{cc|c}13&8&10\\\hline 0&X&0\\X&0&0\\1&1&1\end{array}\right|\Rightarrow$$

$\Rightarrow L_{10}=L_{13}L_8\vee 10^0=\{10^0\}$.

$$T_1\left|\begin{array}{ccc|c}10&7&14\\\hline 1&0&1\end{array}\right| \oplus C_5\left|\begin{array}{ccc}10&7&14\\0&X&1\\X&0&1\\1&1&0\end{array}\right| = L^6\left|\begin{array}{ccc}10&7&14\\1&X&0\\X&0&0\\0&1&1\end{array}\right| \Rightarrow$$

$\Rightarrow L_{14}=L_7\bar{L}_{10}\vee 14^0=\{7^1, 14^0\}.$

Вторая итерация моделирования неисправностей дает следующие множества:

$L_1=\{1^0\}, L_2=\{2^1\}, L_3=\{3^0\}, L_4=\{4^1\}, L_5=\{5^0\},$

$L_6=\{6^1\}, L_7=\{7^1\}, L_8=\{5^0, 8^0\}, L_9=\{1^0, 12^1, 2^1, 9^0\},$

$L_{10}=\{10^0\}, L_{11}=\{5^0, 11^1\}, L_{12}=\{1^0, 12^1\}, L_{13}=\{13^1\}.$

На основании пункта 5 алгоритма делается вывод о прекращении моделирования входного вектора, поскольку списки дефектов, полученные в двух соседних итерациях, тождественно равны.

Список проверяемых на тест-векторе дефектов по наблюдаемым выходам (12, 13, 14) определяется как объединение множеств:

$L(T_1)=L_{12}\cup L_{13}\cup L_{14}=\{1^0, 12^1, 13^1, 7^1, 14^0\}.$

Для второго тест-вектора выполнение кубического ⊕-алгоритма дает следующие результаты:

$$T_1\left|\begin{array}{ccc|c}11&1&12\\\hline 1&1&0\end{array}\right| \oplus C_1\left|\begin{array}{ccc}11&1&12\\1&X&0\\X&1&0\\0&0&1\end{array}\right| = L^1\left|\begin{array}{ccc}11&1&12\\0&X&0\\X&0&0\\1&1&1\end{array}\right| \Rightarrow$$

$\Rightarrow L_{12}=L_{11}L_1\vee 12^1=\{12^1\}.$

$$T_1\left|\begin{array}{ccc|c}12&2&9\\\hline 0&1&0\end{array}\right| \oplus C_2\left|\begin{array}{ccc}12&2&9\\1&X&0\\X&1&0\\0&0&1\end{array}\right| = L^2\left|\begin{array}{ccc}12&2&9\\1&X&0\\X&0&0\\0&1&1\end{array}\right| \Rightarrow$$

$\Rightarrow L_9=\bar{L}_{12}L_2\vee 9^1=\{2^0, 9^1\}.$

$$T_1\left|\begin{array}{ccc|c}9&3&4&13\\\hline 1&1&0&0\\0&1&1&1\end{array}\right| \oplus C_3\left|\begin{array}{ccc}9&3&4&13\\1&X&X&0\\0&0&E&0\\0&1&E&1\\0&X&F&S\end{array}\right| = L^4\left|\begin{array}{ccc}9&3&4&13\\1&X&X&1\\0&1&Q&1\\0&0&Q&0\\0&X&C&P\end{array}\right| \Rightarrow$$

$\Rightarrow L_{13}=L_9\vee\bar{L}_9L_3\bar{L}'_4\bar{L}_4\vee\bar{L}_9L_4\vee 13^0=\{2^0, 9^1, 3^0, 4^0, 13^0\}.$

$$T_1\left|\begin{array}{ccccc|cc}5&13&6&11&8\\\hline 1&0&0&0&1\\0&1&1&1&0\end{array}\right| \oplus C_4\left|\begin{array}{ccccc}5&13&6&11&8\\1&X&X&0&1\\0&0&E&0&1\\0&1&E&1&0\\0&X&F&Q&J\\0&X&F&J&Q\end{array}\right| = L^4\left|\begin{array}{ccccc}5&13&6&11&8\\1&X&X&1&1\\0&1&Q&1&1\\0&0&Q&0&0\\0&X&C&E&E\\0&X&C&H&H\end{array}\right| \Rightarrow$$

$$\Rightarrow \begin{cases} L_{11} = L_5 \vee \bar{L}_5L_{13}\bar{L}'_6\,\bar{L}_6 \vee \bar{L}_5L_6\bar{L}_{11}\bar{L}_8 \vee 11^0 = \\ = \{5^1, 12^1, 9^1, 3^0, 4^0, 13^0, 6^0, 11^0\}; \\ L_8 = L_5 \vee \bar{L}_5L_{13}\bar{L}'_6\,\bar{L}_6 \vee \bar{L}_5L_6\bar{L}_{11}\bar{L}_8 \vee 8^1 = \\ = \{5^1, 12^1, 9^1, 3^0, 4^0, 13^0, 6^0, 8^1\}. \end{cases}$$

$$T_1\left|\begin{array}{ccc|c}13&8&10\\\hline 0&0&1\end{array}\right| \oplus C_5\left|\begin{array}{ccc}13&8&10\\0&X&1\\X&0&1\\1&1&0\end{array}\right| = L^5\left|\begin{array}{ccc}13&8&10\\0&X&0\\X&0&0\\1&1&1\end{array}\right| \Rightarrow$$

$\Rightarrow L_{10}=\bar{L}_{13}L_8\vee 10^0=L_8-L_{13}\cup 10^0=$

$=\{5^1, 2^1, 9^1, 3^0, 4^0, 13^0, 6^0, 8^1\}-\{12^1, 9^1, 3^0, 4^0, 13^0\}\cup 10^0=\{5^1, 6^0, 8^1, 10^0\}.$

$$T_1\left|\begin{array}{ccc|c}10&7&14\\\hline 1&0&1\end{array}\right| \oplus C_5\left|\begin{array}{ccc}10&7&14\\0&X&1\\X&0&1\\1&1&0\end{array}\right| = L^6\left|\begin{array}{ccc}10&7&14\\1&X&0\\X&0&0\\0&1&1\end{array}\right| \Rightarrow$$

$\Rightarrow L_{14}=L_7\bar{L}_{10}\vee 14^0=\{7^1, 14^0\}.$

Повторение итерации вычисления списков неисправностей не изменяет полученные для линий множества неисправностей.

Список проверяемых дефектов на втором векторе по наблюдаемым выходам (12, 13, 14) равен

$L(T_2)=L_{12}\cup L_{13}\cup L_{14}=\{2^0, 9^1, 3^0, 4^0, 12^1, 13^0, 7^1, 14^0\}.$

Результат выполнения ⊕-кубического алгоритма моделирования неисправностей на восьми наборах полного проверяющего теста имеет вид

1	2	3	4	5	6	7	8	9	10	11	12	13	14	Q_t (%)	Q_Σ (%)
0	.	.	.	.	.	1	.	.	.	.	1	1	0	17	17
.	0	0	0	.	.	1	.	1	.	.	1	0	0	28	35
.	1	.	.	.	.	0	.	0	0	.	1	1	1	25	53
0	0	0	0	0	.	.	0	1	1	.	1	0	0	39	75
.	0	0	1	1	0	0	1	1	0	0	1	0	1	46	78
.	1	0	1	1	1	0	1	0	0	0	1	1	1	46	85
0	.	1	.	.	.	0	.	.	0	.	1	1	1	25	89
1	.	1	.	.	.	0	.	.	0	1	0	1	1	28	100

Здесь точка идентифицирует пустое множество проверяемых на линии неисправностей; 0, 1 – проверку константы нуля, единицы. Столбец Q_t, (Q_Σ) определяет процент проверяемых (проверенных) на векторе (тесте) дефектов.

Заключение

Метод кубического моделирования неисправностей является новой технологией обработки цифровых схем табличного (вентильного, функционального, алгоритмического) уровня представления. Он позволяет за одну итерацию моделировать все одиночные константные неисправности цифрового устройства, проверяемые тест-вектором. Условие применения заключается в наличии табличной формы описания примитивов цифрового устройства. Предложенный метод также эффективно обрабатывает и последовательностные примитивы цифровых автоматов, описанные двухтактными кубическими покрытиями [2, с. 33, 38]. Последние формализуют описания алгоритмов в виде примитивов, соответствующих графам переходов, граф-схемам, таблицам переходов цифровых автоматов.

Предложенная технология тестирования по уравнению T⊕C=L дает возможность:

— моделировать неисправности на основе анализа кубического покрытия;

— получать дедуктивные формулы для любых типовых функциональных элементов;

— проектировать компилятивные симуляторы для обработки цифровых устройств произвольного уровня описания;

— генерировать тесты для цифровых систем на основе использования КПСН;

— верифицировать результаты моделирования неисправностей и генерации тестов;

— проектировать аппаратурные быстродействующие симуляторы.

Модели и методы реализованы в виде программных приложений для системы проектирования VHDL-Active [9]. Они используются при генерации тестов для проектов на основе FPGA и CPLD. Класс обрабатываемых структур: конечные автоматы, описанные в виде графов переходов, а также булевы уравнения с триггерными схемами. Входной язык описания цифровых систем – VHDL, Verilog.

Время проектирования теста функционально зависит от квадрата суммарного объема кубических покрытий:

$$W(T) = \left[\sum_{i=1}^{M} (n_i \times q_i)\right]^2.$$

Объект диагностирования имеет следующие характеристики: число вентилей – до 20 тыс.; количество эквипотенциальных линий – до 5 тыс.; время проектирования теста активизации одномерных путей – до 2 часов.

Более полную картину о возможностях системы моделирования даст ее апробация на тест-задачах размерностью порядка миллиона вентилей.

ИСПОЛЬЗОВАННЫЕ ИСТОЧНИКИ

1. Hayes J. P. A systematic approach to multivalued digital simulation // ICCD-84: Proc. IEEE Int. Conf. Comput. — 1984. — N 4. — P. 177—182.

2. Хаханов В. И. Техническая диагностика элементов и узлов персональных компьютеров. — К. : ІЗМН, 1997.

3. Ермилов В. А. Метод отбора существенных неисправностей для диагностики цифровых схем. Общие выражения для неисправностей, возможных при эксперименте // Автоматика и телемеханика. — 1971. — № 1. — С. 159—167.

4. Armstrong D. B. A deductive method of simulating faults in logic circuits // IEEE Trans. on Computers. — 1972. — Vol. C-21, N 5. — P. 464—471.

5. Биргер А. Г. Многозначное дедуктивное моделирование цифровых устройств // Автоматика и вычислительная техника.– 1982.– № 4.– С. 77—82.

6. Levendel Y. H., Menon P. R. Comparison of fault simulation methods – Treatment of unknown signal values // Journal of Digital System. — 1980. — Vol. 4. — P. 443—459.

7. Abramovich M., Breuer M. A., Friedman A. D. Digital system testing and testable design. — Computer Science Press, 1998.

8. Автоматизация диагностирования электронных устройств / Ю. В. Малышенко, В. П. Чипулис, С. Г. Шаршунов. — М. : Энергоатомиздат, 1986.

9. Active-VHDL Series. Book 1—4. — Reference Guide. ALDEC Inc., 1998.

Д. т. н. В. П. МАЛАХОВ, к. т. н. В. С. СИТНИКОВ, П. В. СТУПЕНЬ, С. В. УЛЬЯШИН

Украина, Одесский гос. политехнический ун-т
E-mail: sitnv@promel.ospu.odessa.ua

Дата поступления в редакцию
29.11 2000 г.
Оппонент *д. т. н. В. А. АРБУЗНИКОВ*

ВЫБОР СТРУКТУРЫ ЦИФРОВОГО ФИЛЬТРА ПО УРОВНЮ ВЫХОДНОГО ШУМА ОКРУГЛЕНИЯ

Рассматривается возможность выбора наилучшей структуры цифрового фильтра из семейства широко используемых структур.

При проектировании и реализации цифровых фильтров (ЦФ) возникает проблема выбора его структуры по передаточной функции. Наилучшая структура определяется по критерию качества, который выбирается на основе требований, предъявляемых к фильтрам [1, с. 252]. Например, в локационных системах основными являются критерии быстродействия и минимального выходного шума округления, а в системах обработки биомедицинской информации — критерии минимального выходного шума округления и чувствительности к изменениям коэффициентов.

Следует отметить, что любая структурная реализация ЦФ основана на арифметических операциях с конечной точностью, поэтому сравнение различных структур целесообразно производить на основе характеристик шумов округления и чувствительности к изменениям коэффициентов.

На сегодня существует ряд подходов к синтезу новых структур ЦФ, однако это длительный процесс, не всегда приводящий к положительному результату [2, 3]. Анализ широко используемых структур упрощает эту задачу. Поэтому целью данной работы является исследование возможности выбора наилучшей структуры ЦФ второго порядка из семейства широко используемых структур на основе критерия минимального уровня выходного шума округления.

Для исследования структур ЦФ взято 17 широко используемых типовых структур (канонических — 8, лестничных — 5, мостовых — 4), приведенных на **рис. 1—3**. Исходная передаточная функция (ПФ) задана в виде аналогового фильтра – прототипа полиномиального типа [2]: