СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ
CAD-SYSTEMS

УДК 004.272.2

А. Л. Стемпковский, академик РАН, директор, **Д. В. Тельпухов,** канд. техн. наук, зав. отд.,
Р. А. Соловьев, канд. техн. наук, вед. науч. сотр., e-mail: ZF-Turbo@yandex.ru
Институт проблем проектирования в микроэлектронике РАН (ИППМ РАН)

Повышение сбоеустойчивости логических схем на основе частичного ресинтеза схемы

Ряд разнородных факторов из области микроэлектронной промышленности в последнее время существенно актуализировал разработки в области повышения сбоеустойчивости комбинационных схем. В настоящее время в современных системах автоматизированного проектирования отсутствуют средства для оценки или повышения сбоеустойчивости комбинационных схем на логическом уровне. В данной работе была сделана попытка создания методологии для реализации подобных программных средств. Предложен маршрут ресинтеза, позволяющий без внедрения существенной избыточности повысить логическую устойчивость комбинационных схем к одиночным сбоям. Были проведены вычислительные эксперименты, демонстрирующие высокую эффективность предложенного метода.

Ключевые слова: *комбинационные схемы, наблюдаемость вентиля, коэффициент чувствительности, ресинтез*

Введение

Современные тенденции, заключающиеся в увеличении тактовых частот, росте степени интеграции, уменьшении порогового напряжения и напряжения питания, приводят к экспоненциальному росту одиночных сбоев в комбинационных схемах, обусловливая острую необходимость в обеспечении сбоеустойчивости не только модулей памяти, но и комбинационных схем. Для процессоров и иных микроэлектронных устройств, где последовательные элементы уже имеют встроенные механизмы защиты, комбинационные схемы становятся одним из основных источников одиночных сбоев [1].

Существующие методы повышения отказоустойчивости комбинационных схем эксплуатируют одно из маскирующих свойств логических схем: логическое, электрическое, временное.

- *Логическое маскирование:* сбой возникает в участках схемы, не оказывающих влияния на выход схемы в условиях заданных входных воздействий.
- *Электрическое маскирование:* сбой затухает и не распространяется до выхода схемы благодаря передаточным характеристикам электрических цепей логических вентилей.
- *Временное маскирование:* сбой не проявляется ввиду того, что его длительность не позволяет зафиксировать ошибку на выходном регистре.

В настоящем исследовании учитывалось лишь логическое маскирование. Это обусловлено тем, что данное свойство не зависит от технологического базиса и вносит наибольший вклад в общую интенсивность сбоев схемы [2]. Кроме того, логическое маскирование является наиболее трудоемким свойством для моделирования и характеризации [3].

В настоящее время за основу чаще всего берется модель независимых вентильных сбоев в трактовке Фон Неймана [4], в рамках которой считается, что все вентили имеют одинаковую независимую вероятность сбоя, в то время как число ошибок в схеме не ограничено. В рамках такой постановки задачи было разработано большое число различных мажоритарных подходов, защищающих схемы на архитектурном уровне, таких как каскадное тройное резервирование (*cascade triple modular redundancy* (*CTMR*)), логические схемы с четырехкратным резервированием (*quadded logic*), случайно-переплетенная логика (*random interwoven redundancy*) и т.д. Однако несмотря на большое количество научных публикаций, на практике для защиты комбинационных схем до сих пор используют архаичные методы тройного резервирования.

В настоящей работе для повышения сбоеустойчивости комбинационной схемы предлагается использовать метод ресинтеза, который подразумевает итерационное замещение уязвимых участков схем на более устойчивые аналоги.

Методы оценки логической устойчивости схемы к одиночным сбоям

Дестабилизирующие воздействия и помехи на выходе логических вентилей моделируют как двоичный симметричный канал связи. Это означает, что

каждый вентиль имеет постоянную независимую вероятность сбоя $p < 0{,}5$. Эта вероятность интерпретируется как совокупность различных дестабилизирующих эффектов и источников помех, включая перекрестные помехи, наземное космическое излучение, электромагнитные помехи и т.д. В общем случае свойства логического маскирования полностью определяются функцией вероятности ошибки на выходе схемы $F(p)$ от вероятности сбоя вентиля p [5].

Введем некоторые обозначения. Пусть Ω обозначает набор всех вентилей в схеме, в то время как N и M — число первичных входов и число вентилей соответственно. Будем обозначать ошибки, возникающие на элементах как e_i, причем $e_i = 1$, если на i-м элементе возникла ошибка. В случае отсутствия ошибки $e_i = 0$. Вектор $\mathbf{e} = (e_1, e_2, ..., e_M)$ будем называть вектором ошибки, по аналогии с вектором входных значений $\mathbf{X} = (x_1, x_2, ..., x_N)$.

В качестве примера рассмотрим схему в базисе элементов NAND, реализующую функцию прямой импликации (рис. 1), и на основе его расширенной таблицы истинности получим основные формулы для нахождения $F(p)$. Построим расширенную таблицу истинности (табл. 1) для всех комбинаций входных векторов и векторов ошибок ($\mathbf{X}$, $\mathbf{e}$).

Здесь $E(\mathbf{X}, \mathbf{e})$ обозначает характеристическую функцию от пар векторов (входной вектор $\mathbf{X}$ и вектор ошибки $\mathbf{e}$):

$$E(\mathbf{X}, \mathbf{e}) = \begin{cases} 1, \text{ если набор } (\mathbf{X}, \mathbf{e}) \text{ приводит к ошибке} \\ 0, \text{ иначе.} \end{cases}$$

В последнем столбце указаны вероятности появления вектора ошибки при условии того, что вероятность возникновения ошибки на каждом вентиле равна p. Например, вероятность появления вектора ошибки $\mathbf{e} = (1, 0)$ равна $p(1 - p)$, вследствие

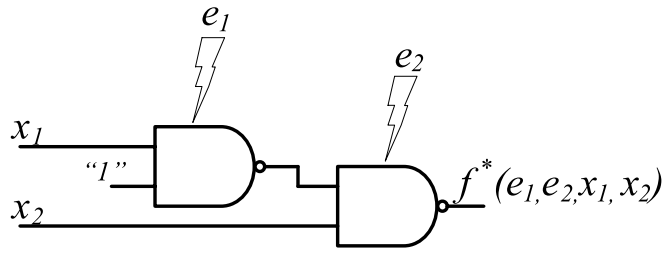

Рис. 1. Прямая импликация в базисе {NAND}

независимости двух случайных событий: возникновение ошибки на первом вентиле (p) и отсутствие ошибки на втором вентиле ($1 - p$).

Нас интересуют строки, в которых значение функции не совпадает с эталонным, а иными словами — строки, в которых характеристическая функция $E(\mathbf{X}, \mathbf{e})$ равна единице. Обратившись к строчке под номером 5 в табл. 1 и учитывая тот факт, что вероятность любой комбинации двух входов равна $\frac{1}{4}$, получим вероятность наступления этого события: $\frac{1}{4}(1 - p)p$. Если для каждой строчки, в которой $E(\mathbf{X}, \mathbf{e}) = 1$ найти вероятность ее возникновения, то просуммировав все эти выражения, получим полином ошибки, который характеризует вероятность несовпадения результата работы схемы с эталонным при вероятности ошибки на вентиле, равной p. Для рассмотренного примера полином будет выглядеть следующим образом:

$$F(p) = \frac{1}{4}(0 \cdot (1 - p)^2) + \frac{1}{4}(4 \cdot (1 - p)p) + \frac{1}{4}(2 \cdot p(1 - p)) + \frac{1}{4}(2 \cdot p^2);$$

$$F(p) = 1{,}5p - p^2. \quad (1)$$

Таблица 1

Таблица истинности по всем (X, e) для схемы прямой импликации

№	e_1	e_2	x_1	x_2	$f(x_1, x_2)$	$f^*(x_1, x_2, e_1, e_2)$	$E(\mathbf{X}, \mathbf{e})$	Вероятность появления вектора ошибки
0	0	0	0	0	1	1	0	$(1 - p)^2$
1	0	0	0	1	0	0	0	
2	0	0	1	0	1	1	0	
3	0	0	1	1	1	0	0	
4	0	1	0	0	1	0	1	$(1 - p)p$
5	0	1	0	1	0	1	1	
6	0	1	1	0	1	0	1	
7	0	1	1	1	1	0	1	
8	1	0	0	0	1	1	0	$p(1 - p)$
9	1	0	0	1	0	1	1	
10	1	0	1	0	1	1	0	
11	1	0	1	1	1	0	1	
12	1	1	0	0	1	0	1	p^2
13	1	1	0	1	0	0	0	
14	1	1	1	0	1	0	1	
15	1	1	1	1	1	1	0	

Теперь выведем формулу для расчета полинома ошибки в общем виде для произвольной логической схемы. Учитывая, что вероятность появления на входе конкретного вектора входных сигналов **X** длины N (в предположении равновероятности всех таких наборов) равна $\frac{1}{2^N}$, а вероятность возникновения вектора ошибки **e** длины M и веса $|\mathbf{e}|$ равна $p^{|\mathbf{e}|}(1-p)^{M-|\mathbf{e}|}$, получаем вероятность ошибки на выходе схемы (вес вектора равен числу его ненулевых элементов):

$$F(p) = \frac{1}{2^N} \sum_{\mathbf{X},\,\mathbf{e}} E(\mathbf{X},\,\mathbf{e}) p^{|\mathbf{e}|}(1-p)^{M-|\mathbf{e}|}. \qquad (2)$$

Из формулы (2) видно, что вычислительная сложность аналитического расчета полинома ошибки экспоненциально зависит от числа входов и числа элементов, что делает этот метод неприменимым даже для сравнительно небольших схем. В настоящее время все методы оценки маскирующих свойств комбинационной логики балансируют между высокой вычислительной сложностью и точностью результатов характеристик сбоеустойчивости [6].

В рамках настоящей работы для оценки сбоеустойчивости логических схем предлагается использовать некоторый обобщенный коэффициент логической чувствительности схемы:

$$\alpha = \frac{1}{2^N} \sum_{\mathbf{X},\,\mathbf{e},\,|\mathbf{e}|=1} E(\mathbf{X},\,\mathbf{e}), \qquad (3)$$

где $|\mathbf{e}| = 1$ говорит о том, что суммирование идет только по векторам ошибки с весом, равным единице.

Эта метрика сбоеустойчивости имеет ряд преимуществ по сравнению с предложенными ранее метриками [6]. Во-первых, вычислительная сложность метода линейная относительно числа элементов, что вкупе с методами бит-параллельного моделирования и методами Монте-Карло позволяет использовать эту метрику для сравнительно больших схем. Во-вторых, предлагаемый коэффициент не зависит от вероятности сбоя вентиля, что позволяет использовать его на ранних этапах проектирования сбоеустойчивых схем, а также методов повышения сбоеустойчивости, когда не определена элементная база и условия эксплуатации схемы. В-третьих, для большинства практических применений, в условиях, когда вероятность сбоя вентиля стремится к нулю — эта аппроксимация является наиболее точной, являясь касательной к графику полинома ошибки в точке ноль. Кроме того, предлагаемая метрика может быть модифицирована для оценки участков подсхем в составе больших комбинационных схем. Для этого достаточно в формуле (2) учесть два дополнительных коэффициента, характеризующих вероятность возникновения входной комбинации и вероятность наблюдаемости конкретной ошибки подсхемы на выходах основной схемы.

Разработка маршрута ресинтеза комбинационных схем в целях повышения логической устойчивости к одиночным сбоям

Для больших логических схем, для которых невозможно в разумные сроки посчитать параметры сбоеустойчивости и сгенерировать более устойчивую к сбоям схему, был разработан подход, который заключается в локальной перезаписи небольших участков схемы. Суть метода заключается в итерационном изменении схемы, где каждая итерация гарантированно увеличивает уровень логического маскирования схемы. Предлагаемый подход не гарантирует нахождение оптимума по параметру сбоеустойчивости, однако позволяет за разумное время повысить устойчивость схемы к единичным сбоям. Общая схема процесса ресинтеза представлена на рис. 2.

Как видно из рис. 2, в контексте задачи ресинтеза можно выделить три больших подзадачи:

1) подсчет характеристик;
2) выбор уязвимой подсхемы;
3) генерация набора эквивалентных подсхем и оценка их отказоустойчивости.

Рассмотрим последовательно пути решения обозначенных задач.

Задача предлагаемого итеративного подхода к ресинтезу заключается в том, чтобы после замены подсхемы на более надежную сбоеустойчивость всей схемы гарантированно увеличилась. Если для исход-

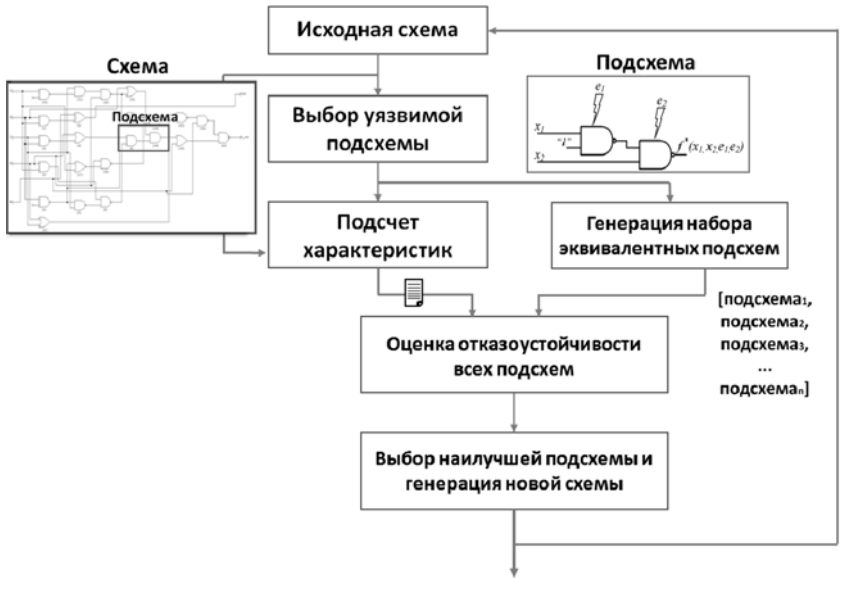

Рис. 2. Общий маршрут ресинтеза логической схемы

ной схемы и для подсхем применять разработанную метрику, то в общем случае это не всегда так. Эксперименты показали, что зачастую замена подсхемы на более надежный эквивалент приводит к деградации сбоеустойчивости исходной схемы. В результате исследований было выявлено два основных фактора, которые влияют на возникновение подобных ситуаций.

Первый фактор связан с предположением о равновероятности входных комбинаций, которое было принято при выводе формулы полинома ошибки. И если для исходной схемы это допущение является оправданным, то в случае подсхемы оно становится источником существенных погрешностей. Решением этой проблемы является модификация метрики для оценки отказоустойчивости подсхем таким образом, чтобы там учитывалось входное распределение входных узлов. Для этого необходимо вычислить таблицу вероятностей входных комбинаций для подсхемы, что приведет к появлению дополнительного коэффициента в формуле

$$\alpha = \sum_{\mathbf{X},\,\mathbf{e},\,|\mathbf{e}|\,=\,1} p(\mathbf{X})\,E(\mathbf{X},\,\mathbf{e}).$$

Второй фактор связан с учетом маскирующих свойств исходной схемы. Дело в том, что разные ошибки на выходах подсхемы имеют разные вероятности распространения по схеме и влияния на ее выходы. Этот факт также не отражен в предложенной метрике. Для того чтобы учесть наблюдаемость ошибок на выходных узлах подсхемы, необходимо построить двумерную квадратную матрицу наблюдаемости ошибок размерностью $2^n \times 2^n$, где n — число выходов подсхемы. В соответствующих ячейках хранятся вероятности того, что ошибка вида *abcd -> abce* на выходах подсхемы даст ошибку на выходах всей схемы. С учетом наблюдаемости возможных ошибок на выходе схемы формула для подсчета сбоеустойчивости подсхемы примет следующий вид:

$$\alpha = \sum_{\mathbf{X},\,\mathbf{e},\,|\mathbf{e}|\,=\,1} E(\mathbf{X},\,\mathbf{e})p(\mathbf{X})p(f(\mathbf{X}) \to f^*(\mathbf{X},\,\mathbf{e})),$$

где $p(\mathbf{X})$ — вероятность возникновения входного набора; $p(f(\mathbf{X}) \to f^*(\mathbf{X},\,\mathbf{e})$ — вероятность того, что ошибка, возникшая на выходе подсхемы, окажется наблюдаемой для исходной схемы.

Используя такую расширенную метрику, можно с большой вероятностью гарантировать, что если предлагаемый коэффициент подсхемы *A* больше коэффициента подсхемы *B*, то сбоеустойчивость всей схемы в случае использования подсхемы *A* будет больше, чем при использовании подсхемы *B*. Более того, количественные значения прироста предлагаемого коэффициента чувствительности при замене подсхемы примерно равны приросту коэффициента чувствительности всей схемы целиком. Точного соответствия принципиально невозможно достичь ввиду того, что мы считаем два вышеописанных фактора и соответствующие им вероятности независимыми, что не соответствует действительности. Для того чтобы учесть существующую зависимость потребуется выполнить такое количество вычислений, которое равно количеству вычислений, необходимых для подсчета коэффициента логической устойчивости исходной схемы. В этом случае теряется смысл оценки подсхемы, поскольку с такой же вычислительной нагрузкой можно после каждой замены оценивать сбоеустойчивость всей схемы целиком. С этих позиций данное допущение (о независимости вероятностей) является разумным компромиссом между быстродействием и точностью получаемых предсказаний.

Важно отметить, что рассматриваемые характеристики — вероятность возникновения входного набора и наблюдаемость ошибки на выходе подсхемы — не зависят от структуры подсхемы, а являются исключительно параметрами логической структуры "до" и "после" рассматриваемого участка. Следствием этого является возможность предварительного подсчета характеристик для дальнейшего использования в вычислениях коэффициента чувствительности для подсхем разной конфигурации. Это нашло отражение в разработанном маршруте ресинтеза комбинационных схем.

Рассмотрим конкретный пример. Рассмотрим подсхему с тремя входами и двумя выходами (рис. 3). В этом случае $P(\mathbf{X})$ будет состоять из восьми значений вероятностей для каждой из комбинаций входных воздействий.

В табл. 2 приведена матрица наблюдаемости ошибок. Здесь в помеченной ячейке хранится значение вероятности для события, что ошибка вида (1 0) → (0 1) окажется наблюдаемой на выходе схемы.

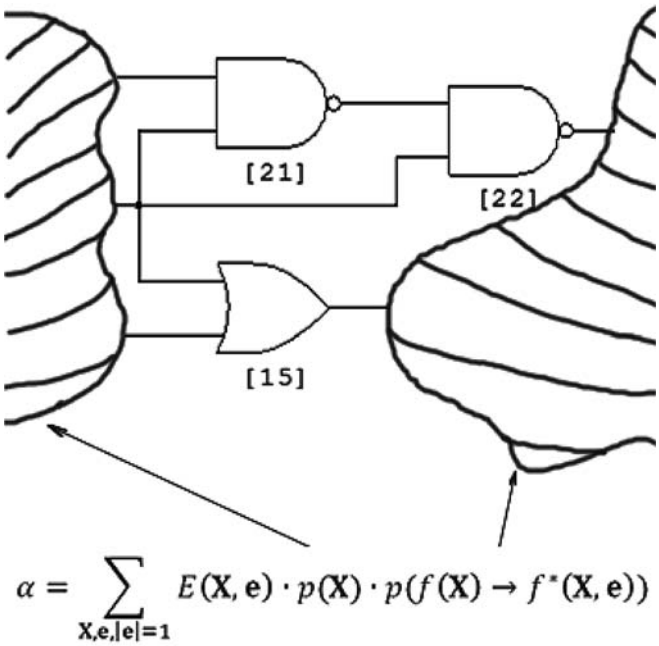

Рис. 3. Подсчет характеристик подсхемы

Таблица 2

Наблюдаемость ошибок подсхемы на выходах основной схемы

	0 0	0 1	1 0	1 1
0 0	q00	q01	q02	q03
0 1	q10	q11	q12	q13
1 0	q20	q21	q22	q23
1 1	q30	q31	q32	q33

Следующим важным этапом в разработке маршрута ресинтеза комбинационных схем является задача выбора уязвимых участков схемы. В общем случае можно выбирать произвольные подсхемы и пытаться отыскать более устойчивые аналоги. Однако такой подход может негативно отразиться на эффективности алгоритма ресинтеза с точки зрения скорости его сходимости. Но даже в такой упрощенной постановке задачи возникает множество вопросов относительно того, как именно и какие именно подсхемы следует выбирать. Во-первых, необходимо определиться с размером подсхем таким образом, чтобы можно было точно посчитать все необходимые характеристики: распределение логических значений на входах, матрицу наблюдаемости ошибок для выходов и коэффициент чувствительности схемы. Эксперименты показали, что на заданные подсхемы необходимо наложить следующие ограничения: число входов не более 10 и число выходов не более 12. В этом случае размер логических характеристик на входах не превосходит 1024, а размерность матрицы ошибок на выходах подсхемы будет не более 4096×4096.

Для генерации случайной подсхемы используется следующий эвристический алгоритм:

Шаг 0. Создаем пустую подсхему.

Шаг 1. Выбираем случайный элемент в схеме и добавляем его в список элементов подсхемы.

Шаг 2. Случайно выбираем произвольный узел, являющийся входом подсхемы. Добавляем к подсхеме все элементы, которые подсоединены к этому узлу.

Шаг 3. Считаем число входов в схеме. Если оно незначительно превысило или равно заданному числу входов, возвращаем подсхему, иначе переходим к Шагу 2. Если число входов и выходов превысило лимиты в 10 входов или 12 выходов, переходим к Шагу 0.

Рассмотрим этот процесс на детальном примере. Пусть нам требуется получить произвольную подсхему с пятью входами из заданной на рис. 3.

На первом шаге выбирается случайный вентиль N19 (рис. 4). Далее выбираем случайный вход, который соединен с вентилем N16. Подсхема теперь состоит из двух вентилей N16 и N19, трех входов и одного выхода. Положим дальше из трех входов подсхемы, мы выбрали узел, ведущий к вентилю N12. В этом случае в подсхему также требуется добавить элемент N13, который подключен к тому же узлу. Подсхема теперь состоит из четырех элементов N12, N13, N16 и N19, пяти входов и двух выходов. Мы достигли лимита в пять входов, следовательно, требуемая подсхема получена.

Кроме того, было разработано несколько модификаций данного метода, которые позволяют получать наиболее уязвимые подсхемы с точки зрения их наблюдаемости на основных выходах исходной схемы. Для этого перед началом алгоритма генерации случайной подсхемы выполняем построение карты уязвимости схемы. Иными словами, определяем значения наблюдаемости для каждого вентиля в схеме. Имея такую информацию, можно выбирать первоначальный вентиль для ветвления с учетом его уязвимости, а на втором шаге алгоритма ветвиться в сторону более уязвимых элементов. Для этого был использован метод асимметричного колеса рулетки. Предлагаемый метод направленного поиска уязвимой подсхемы позволяет ускорить сходимость алгоритма за счет того, что каждая успешная итерация вносит больший прирост для сбоеустойчивости схемы. Стоит также отметить, что задача построения "карты уязвимости" схемы является базовой процедурой при подсчете коэффициента чувствительности исходной схемы, так как он, по сути, является суммой наблюдаемостей всех элементов схемы. Таким образом, предложенный метод направленного поиска уязвимой подсхемы практически не привносит дополнительных вычислительных затрат, так как вся необходимая для него информация в неявном виде рассчитывается на каждой итерации алгоритма ресинтеза.

Наиболее сложной является задача синтеза эквивалентной подсхемы, имеющей наименьшую наблюдаемость для исходной схемы. Задача осложняется тем, что при подсчете коэффициента чувствительности используются параметры, связанные с ис-

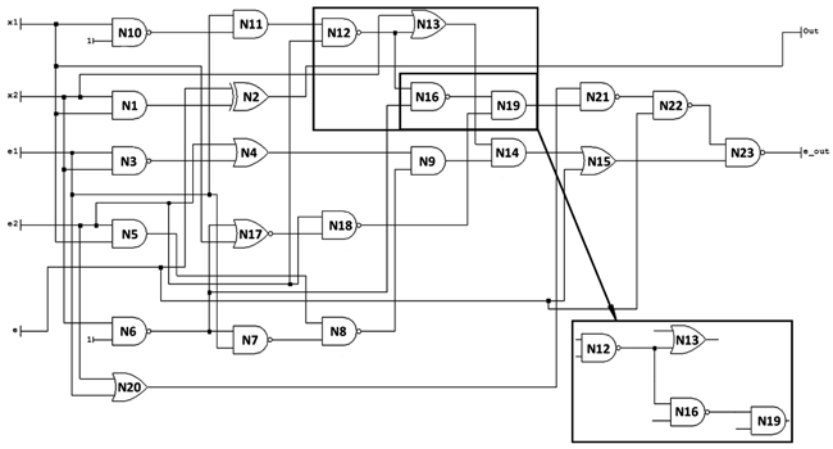

Рис. 4. Тестовая схема для задачи ресинтеза

ходной схемой. Это приводит к тому, что одна и та же подсхема может иметь абсолютно разные коэффициенты чувствительности, будучи помещенной в разные части схемы. Этот факт не позволяет использовать традиционные методы с использованием заранее просчитанной библиотеки стандартных замен. На текущий момент задача решается с помощью генерации большого набора подсхем различной конфигурации с последующим выбором подсхемы с наименьшим параметром коэффициента чувствительности. В данной работе использовано несколько подходов по автоматической генерации подсхем, эквивалентных заданной:

- несколько эвристических методов генерации подсхемы на базе минимизации логической функции с помощью Espresso [7];
- синтез оптимизированной схемы с помощью свободного ПО для синтеза схем Yosys-ABC [8];
- тройное резервирование (TMR) для подсхемы.

Отсутствие методов направленного синтеза подходящей подсхемы выливается в очень большое число холостых прогонов алгоритма: доля успешных итераций в текущей реализации составляет всего 1—2 %. Задача по увеличению этой доли остается в качестве основного направления дальнейших исследований.

Результаты вычислительных экспериментов

В ходе исследований был разработан программный пакет на языке *Python*, с критическими частями, написанными на *C* в целях увеличения скорости вычислений. Программа свободно доступна по ссылке [9].

Для первого вычислительного эксперимента использовали схемы ISCAS85 [10], предварительно синтезированные в базе библиотеки, состоящей из семи базовых элементов: INV, NAND, NOR, AND, OR, XOR, XNOR. Синтез проводили с помощью программы *Synopsys Design Compiler* (с оптимизацией по задержке). В табл. 3 приведены значения коэффициента чувствительности для схемы до и после ресинтеза. Итерация считается успешной, если удалось найти более надежный аналог выбранной случайной подсхемы. Если не было успешных замен за последние 1000 итераций, работа алгоритма завершалась.

Замечание: в алгоритме не был использован метод TMR для подсхем ввиду серьезного увеличения площади схемы при относительно незначительном уменьшении коэффициента чувствительности. Метод TMR имеет смысл использовать для замены сразу больших участков схем, при условии, что нельзя выполнить TMR сразу над всей схемой ввиду жестких ограничений по площади. Все сгенерированные подсхемы имели площадь, сравнимую с площадью изначальной подсхемы.

Как видно из табл. 3, коэффициент чувствительности для схемы уменьшается иногда более чем на 18 %. При этом площадь изменяется незначительно и иногда даже уменьшается. То есть маскирование ошибок происходит не за счет увеличения избыточности, а за счет оптимальной логики работы схемы.

Длина критического пути для схемы после ресинтеза в рамках данного вычислительного эксперимента увеличивается, что снижает скорость работы схемы. Это связано с тем, что для моделирования использовали схемы, максимально оптимизированные по скорости работы (*compile_ultra*), и любое изменение логической структуры с большой вероятностью приводит к увеличению критического пути. Вторая вероятная причина — на более длинном пути прохождения сигнала маскируется больше сбоев. Дополнительный эксперимент на c432 показал, что задержка для подсхемы и ее замены в среднем уменьшается на 5 %, однако общая задержка схемы после всех замен выросла на 31 %.

В следующем эксперименте была сделана попытка проанализировать, как влияет начальный синтез схемы на эффективность последующего ресинтеза. Для этого использовали различные варианты синтеза c432: структурная версия до синтеза, синтез в Synopsys с оптимизацией по задержке,

Таблица 3

Результаты работы алгоритма ресинтеза на схемах ISCAS85

Схема	c432	c1355	c1908	c3540	c5315	c6288	c7552
Число входов	36	41	33	50	178	32	207
Коэффициент чувствительности (до)	63,36	91,08	140,71	299,63	641,38	1476,69	566,43
Коэффициент чувствительности (после)	51,76	90,94	121,19	278,88	612,55	1376,68	514,22
Улучшение, %	18,31	0,2	13,9	6,9	4,5	6,8	9,2
Число логических элементов (до)	204	234	287	958	1351	1776	1369
Число логических элементов (после)	192	238	284	1000	1421	1700	1486
Процент от начального, %	94,12	101,7	99,0	104,4	105,2	95,7	108,6
Задержка на критическом пути, нс (до)	19	9	17	24	16	55	17
Задержка на критическом пути, нс (после)	25	9	20	29	18	66	26
Изменение задержки, %	31,58	0	17,7	20,8	12,5	20	52,9
Общее число итераций	5128	5000	4540	7805	11 841	11 572	16 534
Число успешных замен	46	6	32	101	110	256	165
Успешные замены, %	0,9	0,1	0,7	1,3	0,9	2,2	1

Таблица 4

Результаты ресинтеза разных версий схемы c432

Схема	c432 Initial	c432 Min Path	c432 Min Area	c432 ABC
Коэффициент чувствительности (до)	71,71	63,36	54,49	107,66
Коэффициент чувствительности (после)	54,53	51,76	50,82	56,23
Улучшение, %	23,96	18,31	6,74	47,77
Число логических элементов (до)	216	204	166	301
Число логических элементов (после)	197	192	171	400
Процент от начального, %	91,2	94,12	103,01	132,89
Задержка на критическом пути, нс (до)	44	19	28	43
Задержка на критическом пути, нс (после)	34	25	29	58
Изменение задержки, %	−22,73	31,58	3,57	34,88
Общее число итераций	4575	5128	2483	9759
Число успешных замен	62	46	20	156
Успешные замены, %	1,36	0,9	0,81	1,6

Таблица 5

Результаты работы алгоритма ресинтеза на схемах с малым числом входов из набора LGSynth89

Схема	5xp1	9sym	alu2_synth	alu4_synth	bw	f51m_synth	ldd_synth	misex3
Число входов	7	9	10	14	5	8	9	14
Коэффициент чувствительности (до)	54,8	18,7	145,8	281,8	107,8	61,8	73,1	155,2
Коэффициент чувствительности (после)	50,5	17,4	78,5	163,6	98,5	43,7	51,2	104,5
Улучшение, %	7,85	6,95	46,16	41,94	8,63	29,29	29,96	32,67
Число логических элементов (до)	135	252	386	741	195	111	85	1509
Число логических элементов (после)	132	255	752	1490	189	103	81	2142
Процент от начального, %	97,8	101,2	194,8	201,1	96,9	92,8	95,3	141,9
Задержка на критическом пути, нс (до)	11	16	30	37	10	26	14	22
Задержка на критическом пути, нс (после)	11	15	45	53	9	27	12	26
Изменение, %	0	−6,25	50	43,24	−10	3,9	−14,3	18,2
Общее число итераций	4151	2945	26 026	29 846	3176	6618	3004	18 077
Число успешных замен	23	12	316	589	22	60	46	267
Успешные замены, %	0,55	0,41	1,21	1,97	0,69	0,91	1,53	1,48

синтез в Synopsys с оптимизацией по площади, и стандартный синтез в системе *Yosys-ABC*. Результаты работы ресинтеза для всех четырех схем приведены в табл. 4.

Еще один эксперимент был проведен для схем из набора LGSynth89 [11]. Были выбраны схемы с небольшим числом входов и выходов, чтобы статистические параметры для схемы считались точно, а не на базе метода Монте-Карло, как в эксперименте со схемами ISCAS85. Результаты приведены в сводной табл. 5.

Заключение

В результате проведенного исследования были получены практические результаты, свидетельствующие о том, что предложенный метод ресинтеза может быть эффективно использован в любом маршруте проектирования на этапе постсинтеза. Его можно применять как самостоятельно, так и в дополнение к традиционным методам кратного резервирования, поскольку он сам незначительно влияет на площадь конечного устройства. К основному недостатку метода можно отнести маленький процент успешных итераций (1—2 %), что ведет к относительно медленной работе метода. Однако для устранения этого и других недостатков остались широкие возможности для исследований, которые не были охвачены в данной работе.

1. Генерация схем на замену проводится случайным образом, если же генерировать схему сразу с учетом входного распределения и матрицы ошибок, то можно увеличить частоту успешных итераций, а также повысить общий процент улучшения коэффициента чувствительности.

2. Выбор случайной подсхемы в данный момент, очевидно, не является оптимальным.

3. Скорость работы методов, основанных на методе Монте-Карло, довольно низкая — требуется изучить возможности использования более быстрых методов.

4. Необходима разработка методов предварительной оценки возможности улучшения показателей чувствительности схемы методом ресинтеза еще до начала моделирования.

5. В настоящее время на схемах с малым числом логических уровней (например, c1355) метод дает маленький процент улучшения. Необходимо исследовать фундаментальные причины этого явления.

6. Необходима доработка алгоритма для учета ограничений на общую площадь схемы, а также длину критического пути.

Для удобства разработчиков ресинтез лучше использовать в составе пакета логического синтеза. Среди свободных решений есть открытый пакет для логического синтеза — Yosys. Планируется реализация модуля ресинтеза для Yosys, после отработки маршрута на Python.

Список литературы

1. **Mahatme N. N., Jagannathan S., Loveless T. D., Massengill L. W., Bhuva B. L., Wen S. J.** et al. Comparison of combinational and sequential error rates for a deep submicron process // IEEE Trans Nucl Sci (TNS). 2011. Vol. 58 (6): 27. P. 19—25.
2. **Asadi H., Tahoori M. B., Fazeli M., Miremadi S. G.** Efficient algorithms to accurately compute derating factors of digital circuits // Microelectronics Reliability. 2012. Vol. 52 (6): 12. P. 15—26.
3. **George N., Lach J.** Characterization of logical masking and error propagation in combinational circuits and effects on system vulnerability // Dependable Systems Networks (DSN). 2011 IEEE/IFIP 41st International Conference. 2011. P. 323—334.
4. **Von Neumann J.** Probabilistic logics and the synthesis of reliable organisms from unreliable components // Automata Studies / C. E. Shannon, J. McCarthy, Eds. Princeton, NJ: Princeton Univ. Press, 1956. P. 43—98.
5. **Стемпковский А. Л., Тельпухов Д. В., Соловьев Р. А., Соловьев А. Н., Мячиков М. В.** Моделирование возникновения неисправностей для оценки надежностных характеристик логических схем // Информационные технологии. 2014. № 11. С. 30—36.
6. **Xiao R., Chen C.** Gate-level circuit reliability analysis: A survey // VLSI Design. 2014. P. 1—12.
7. **Espresso** — a Multi-valued PLA minimization. URL: http://embedded.eecs.berkeley.edu /pubs/downloads/espresso/
8. **Wolf C., Glaser J.** Yosys—A Free Verilog Synthesis Suite // Proceedings of the 21st Austrian Workshop on Microelectronics (Austrochip), Linz, Austria, 2013. Vol. 10. 6 p.
9. **Reliability-aware** resynthesis. URL: https://github.com/ID-MIPPM/reliability-aware-resynthesis
10. **Bryan D.** The ISCAS'85 benchmark circuits and netlist format // North Carolina State University, 1985. 4 p.
11. **Yang S.** Logic synthesis and optimization benchmarks user guide: version 3.0 // Microelectronics Center of North Carolina (MCNC), 1991. 45 p.

A. L. Stempkovskiy, Academician, Director, **D. V. Telpukhov,** Ph. D., Head of the Department,
R. A. Solovyev, Ph. D., Chief Researcher, ZF-Turbo@yandex.ru
Institute for Design Problems in Microelectronics, Moscow (IPPM RAS)

Enchancing Reliability of Logic Circuits with Partial Resynthesis Method

A number of diverse factors in the area of microelectronics industry significantly actualized the developments in the field of fault-tolerant combinational circuits design recently. Currently in modern CAD systems there are no means to assess and improve logic masking effects in combinational circuits. In this paper, an attempt was made to create a methodology for the implementation of such software. This paper proposes re-synthesis design flow, providing a logical masking capability for the combinational circuits resistance to a single event upsets without the introduction of substantial redundancy. The details of the method including the development of reliability metrics, the selection of vulnerable sub-schemes, as well as methods of generating reliable circuits are presented. Computational experiments were conducted to demonstrate the high efficiency of the proposed method.

Keywords: *combinational circuit, gate observability, sensitivity factor, resynthesis*

References

1. **Mahatme N. N., Jagannathan S., Loveless T. D., Massengill L. W., Bhuva B. L., Wen S. J.** et al. Comparison of combinational and sequential error rates for a deep submicron process, *IEEE Trans Nucl Sci (TNS)*, 2011, vol. 58 (6): 27, pp. 19—25.
2. **Asadi H., Tahoori M. B., Fazeli M., Miremadi S. G.** Efficient algorithms to accurately compute derating factors of digital circuits, *Microelectronics Reliability*, 2012, vol. 52 (6): 12, pp. 15—26.
3. **George N., Lach J.** Characterization of logical masking and error propagation in combinational circuits and effects on system vulnerability, *Dependable Systems Networks (DSN), 2011, IEEE/IFIP 41st International Conference*, 2011, pp. 323—334.
4. **Von Neumann J.** Probabilistic logics and the synthesis of reliable organisms from unreliable components, *Automata Studies*, C. E. Shannon, J. McCarthy, Eds. Princeton, NJ: Princeton Univ. Press, 1956, pp. 43—98.
5. **Stempkovskij A. L., Tel'puhov D. V., Solov'ev R. A., Solov'ev A. N., Mjachikov M. V.** Modelirovanie vozniknovenija neispravnostej dlja ocenki nadezhnostnyh harakteristik logicheskih shem, *Informacionnye tehnologii*, 2014, no. 11, pp. 30—36 (in Russian).
6. **Xiao R., Chen C.** Gate-level circuit reliability analysis: A survey, *VLSI Design*, 2014, pp. 1—12.
7. **Espresso** — a *Multi-valued PLA minimization*, URL: http://embedded.eecs.berkeley.edu /pubs/downloads/espresso/
8. **Wolf C., Glaser J.** Yosys — A Free Verilog Synthesis Suite, *Proceedings of the 21st Austrian Workshop on Microelectronics (Austrochip)*, Linz, Austria, 2013, vol. 10, 6 p.
9. **Reliability-aware resynthesis,** URL: https://github.com/ID-MIPPM/reliability-aware-resynthesis
10. **Bryan D.** *The ISCAS'85 benchmark circuits and netlist format*, North Carolina State University, 1985, 4 p.
11. **Yang S.** *Logic synthesis and optimization benchmarks user guide: version 3.0*, Microelectronics Center of North Carolina (MCNC), 1991, 45 p.