Сведения о ходе выполнения проекта

«Тестирование и контролепригодное проектирование логических схем высокой производительности»,

поддержанного Российским научным фондом

Соглашение № 14-19-00218

Руководитель д-р техн. наук Матросова Анжела Юрьевна

2014 год

1. Схемы высокой производительности, как правило, строятся из вентилей AND, OR, NAND, NOR, NOT, XOR. Их надежность обеспечивается тестированием на всех этапах проектирования. Тем не менее, возможно обнаружение неисправностей на заключительных этапах производства логической схемы. В таких случаях, как правило, обнаруживаются неисправности, имеющие мало тестовых наборов, так называемые трудно обнаружимые неисправности. Предложен метод, позволяющий маскировать любую логическую неисправность вентиля проектируемой схемы за счет замены подсхемы с неисправным вентилем подсхемой из программируемых блоков [Marosova A., Ostanin S., Kirienko I., Singh V. Partially programmable Circuit Design // Proc. of the IEEE East-West Design&Test Symposium. –Kiev: IEEE. 2014].

В предложенном методе маскирующая подсхема подключается к входам проектируемой схемы и к выходу неисправной подсхемы. Последнее осуществляется благодаря включению мультиплексоров в точки ветвления проектируемой подсхемы из вентилей. Известные за рубежом подходы к синтезу частично программируемых схем используют мультиплексоры аналогичным образом, однако, предлагаемые ими подходы позволяют маскировать 80% одиночных константных неисправностей вентилей, причем, обнаруженная неисправность может не попасть в множество маскируемых.

Предлагаемый нами подход основан на замещении подсхемы, выход которой является ближайшей к неисправному элементу точкой ветвления, подсхемой из программируемых блоков. Замещение выполняется при условии сохранения функционирования всех вентилей замещаемой подсхемы, связанных с точками ветвления этой подсхемы, не совпадающими с выходом рассматриваемой подсхемы.

Преимущество предлагаемого нами подхода заключается в возможности маскирования любой логической неисправности вентиля, а также в компактном представлении частичной функции двумя ROBDD-графами, один для нулевых наборов частичной функции, другой – для ее единичных наборов. Если один из графов пуст, вход мультиплексора для замещающей подсхемы соединяется либо с источником питания, либо заземляется. Иначе предлагается строить по возможности более простую подсхему из программируемых блоков, реализующую частичную функцию, как правило, слабо определенную.

2. В современных высокопроизводительных схемах наряду с тестированием константных неисправностей требуется тестировать неисправности задержек путей. К сожалению, в произвольных логических схемах неисправности задержек большинства путей обнаруживаются только в условиях отсутствия задержек остальных путей, что затрудняет тестирование задержек в схеме в целом. Разработанные за рубежом методы синтеза схем, в которых задержка каждого пути обнаруживается независимо от задержек других путей, требуют введения

дополнительных входов. Это неприемлемо на практике. Ранее нами был разработан метод синтеза последовательностных схем, позволяющий сократить длины путей в них и не требующий дополнительных входов. С этой целью при использовании State Transition Graph (STG) описания поведения асинхронного автомата в качестве задания на синтез его состояния кодируются равновесными (m, n)-кодами, где m - число единичных компонент кода, а n его длина. Далее нулевые значения компонент кодов заменяются неопределенными значениями. Структура последовательностной схемы представляется композицией ROBDD – графов и монотонных конъюнкций[Matrosova A., Mitrofanov E., Roymjantseva E. Combinational Part Structure Simplification // Proc. of the IEEE East-West Design&Test Symposium. –Kiev: IEEE. 2014]

Проведенные на контрольных примерах (LGSynth93) эксперименты показали, что в получаемых предложенным методом последовательностных схемах длины путей комбинационных составляющих в 2-9 раз меньше длин путей комбинационных составляющих, полученных разработанным нами ранее методом (также не требующим дополнительных входов) синтеза контролепригодных комбинационных схем. Оказалось также, что синтезируемые нами контролепригодные последовательностные схемы и схемы, построенные без учета контролепригодных свойств, реализующие одинаковое поведение, имеют один и тот же порядок сложности.

Предложено упрощение реализации монотонных конъюнкций, зависящих от переменных состояния, в предположении, что они реализуются двухвходовыми элементами AND. Использовался метод выделения конъюнктивных факторов. Предполагалось рассматривать все возможные факторы ранга два (ранги монотонных конъюнкций реальных схем, как правило, не велики), упорядочивая их по убыванию весов. Весом фактора является число его вхождений в монотонные конъюнкции. С целью сохранения контролепригодных свойств последовательностной схемы покрытие конъюнкций выполнялось без пересечения покрывающих факторов. Эксперименты на контрольных примерах (LGSynth93) показали, что оборудование, необходимое для реализации монотонных конъюнкций, сокращается от 28% до 58%.

3. Сжатие тестовых последовательностей, представленных троичными векторами-одна из актуальных задач тестирования. Ранее был предложен оригинальный подход, основанный на построении специального дерева декомпозиции. В рамках данного проекта разработана его модификация.

Идея сокращения тестовой последовательности заключается в выделении максимально совместимых подмножеств троичных векторов, которые затем используются для сжатия исходной тестовой последовательности. Построение максимально совместимых подмножеств предлагается выполнять с помощью дерева разложения, по немонотонным переменным. Из вершины дерева декомпозиции проводится три дуги – по одной на каждое значение немонотонной переменной из множества {0,1,X}. В результате такого разложения строится дерево, в котором концевым вершинами соответствуют совместимые подмножества, а ветвям со значением X- множества, элементы которых добавляются в совместимые подмножества с целью построения максимально совместимых подмножеств. Предлагается модификация разработанного ранее алгоритма , заключающаяся в том, что в каждой вершине дерева декомпозиции выполняется разложение в общем случае по подмножеству немонотонных переменных, характеризующихся максимальным числом определенных компонент в соответствующем вершине множестве троичных векторов.

В результате реализации предложенной модификации [Andreeva V., Sorudeykin K.A. Decomposition Tree - based Compaction Tree Procedure with Iteration Steps for Interconvensional Layouts of Tasks // Proc. of the IEEE East-West Design&Test Symposium. –Kiev: IEEE. 2014] существенно уменьшается число шагов декомпозиции, что делает разработанный алгоритм более эффективным по времени при сохранении качества сжатия тестовой последовательности.

4. На выходах самопроверяемой схемы, к которой подключен детектор равновесных кодовых слов, не всегда достигаются всевозможные кодовые слова. Число l достигаемых кодовых слов может быть меньше числа всевозможных кодовых слов (меньше числа сочетаний из n по m). Из сказанного следует, что некоторые неисправности самотестируемого детектора, подключенного к выходам самопроверяемой схемы, могут оказаться необнаружимыми, так как обнаруживающие неисправности кодовые слова не поступают на входы детектора во время работы самопроверяемой схемы.

Предлагается метод синтеза самотестируемого детектора из программируемых логических блоков на подмножестве кодовых слов, мощность которого задана. Самотестируемость обеспечивается для кратных константных неисправностей на полюсах блока. Только один блок может быть неисправным. Будем иметь в виду, что обеспечение самотестируемости на подмножестве кодовых слов не возможно при синтезе самотестируемых детекторов на базе пороговых элементов или полусумматоров.

Разработанные ранее методы синтеза самотестируемых детекторов равновесных кодов из программируемых блоков основаны на представлении множества всех кодовых слов скобочной формой над множеством {V, ^,¬ }. Ее построение обеспечивается многократным применением специальной формулы разложения (1). Различные ДНФ D формулы представляют всевозможные кодовые слова, длины которых задаются нижним индексом, а число единичных компонент - верхним индексом. Скобочная форма покрывается программируемыми блоками. Для покрытия скобочной формы используется представление ее в виде И, ИЛИ дерева.

В работе [Butorina N. Self-Testing Checker Design for Incomplete m-out-of-n Codes // Proc. of the IEEE East-West Design&Test Symposium. –Kiev: IEEE. 2014] ведено понятие существенного поддерева, которое строится при движении от корня И, ИЛИ дерева к его листьям в соответствие со следующими правилами. При попадании в вершину, помеченную символом V, в существенное поддерево включается одно исходящее из нее ребро. При попадании в вершину, помеченную символом ^, в существенное поддерево включаются все ребра. Концевая вершина в И, ИЛИ дереве помечается ДНФ D, задающей всевозможные кодовые слова на подмножестве переменных детектора. Каждое существенное поддерево представляет подмножество кодовых слов, задаваемое логическим произведением ДНФ, сопоставляемых его листьям. Весом листа называется число кодовых слов ДНФ, соответствующей листу, а весом существенного поддерева - число кодовых слов (m, n)-кода, сопоставляемых поддереву. Вес поддерева определяется произведением весов составляющих его листьев.

Рассматривается ситуация, когда n=2m. Предложен способ построения скобочной формы (И, ИЛИ дерева), при котором каждое применение формулы разложения связано с выбором k=n/2, ближайшего сверху. На каждом шаге разложения n – число переменных разлагаемой ДНФ. Листьям такого дерева сопоставляются ДНФ D, зависящие либо от одной либо от двух переменных.

Доказана теорема о том что любое число l, 1 <= l< (число сочетаний из n по m), может быть представлено суммой весов соответствующих поддеревьев рассматриваемого И, ИЛИ дерева. Предложен алгоритм выбора существенных поддеревьев для заданного значения l. Алгоритм сводится к включению на очередном шаге допустимого существенного поддерева максимально возможного веса и ориентирован на упрощение схемы детектора. Схема получается покрытием найденного подмножества существенных поддеревьев программируемыми логическими блоками.

Доказано, что построенный детектор является самотестируемым на подмножестве кодовых слов мощности l. Кодовые слова представлены покрываемой программируемыми логическими блоками совокупностью существенных поддеревьев. Получаемые схемы самотестируемых детекторов не сложнее схемы детектора всевозможных кодовых слов (m, n)-кода, что существенно лучше дублирования последнего.