ИЗВЕСТИЯ РАН. ТЕОРИЯ И СИСТЕМЫ УПРАВЛЕНИЯ, 2017, № 3, с. 97–117

СИСТЕМНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ

УДК 621.382

ОЦЕНКА ЭНЕРГОПОТРЕБЛЕНИЯ ЦИФРОВЫХ УСТРОЙСТВ, ПРЕДСТАВЛЕННЫХ В ВИДЕ КОМПОЗИЦИИ УПРАВЛЯЮЩЕГО И ОПЕРАЦИОННОГО АВТОМАТОВ

© 2017 г. П. Н. Бибило

Белоруссия, Минск, Объединенный ин-т проблем информатики НАН Белоруссии

e-mail: bibilo@newman.bas-net.by

Поступила в редакцию 15.09.16 г.
После доработки 17.01.17 г.

Предлагается проводить оценку энергопотребления логических схем, реализующих цифровые устройства, используя результаты быстрого моделирования как структурных описаний логических схем, так и исходных алгоритмических описаний устройств, по которым строятся логические схемы. Оценка энергопотребления сводится к нахождению “энергоемких” тестов, вызывающих повышенное энергопотребление. Предлагается два способа алгоритмических описаний цифровых устройства рассматриваемого класса, по которым синтезируются логические схемы, различающиеся по параметрам энергопотребления, площади и быстродействия. Эксперименты показывают, что добиться существенного снижения энергопотребления можно, правильно описывая функционирование устройств на алгоритмическом уровне.

DOI: 10.7868/S0002338817030064

Введение. Снижение энергопотребления цифровых блоков является одной из важнейших проблем, стоящих в настоящее время перед разработчиками *систем автоматизированного проектирования* (САПР) сверхбольших интегральных *комплементарных металл-оксид-полупроводниковых схем* (КМОП-схем) и систем на кристалле [1–3]. Одним из эффективных подходов к снижению энергопотребления является создание исходного алгоритмического описания проекта на языке высокого уровня, в котором предусматривается отключение тех блоков, функционирование которых не требуется. Другими словами, надо написать алгоритм поведения, по которому в результате синтеза будет построена логическая схема, отличающаяся тем, что в некоторые интервалы времени в ней не будут функционировать определенные подсхемы. Такое поведение выбранных блоков и надо заранее предусмотреть в алгоритмическом описании. В данной работе предлагается именно такой способ алгоритмического описания операционных устройств на языке ***V**ery high speed integrated circuits **H**ardware **D**escription **L**anguage* (VHDL) – *язык описания аппаратуры сверхскоростных интегральных схем*, являющемся одним из базовых языков проектирования цифровых систем на основе *сверхбольших интегральных схем* (СБИС), выполняемых как в виде *программируемых логических интегральных схем* (ПЛИС), так и в виде заказных СБИС. Данный способ алгоритмического описания сравнивается с традиционным способом VHDL-описания поведения цифрового устройства. Традиционный стиль описания, как правило, не учитывает аспект энергопотребления и ориентирован только на правильную функциональность проектируемой логической схемы.

Оценка энергопотребления получаемых вариантов логических схем из библиотечных КМОП-элементов является одним из аспектов решения проблемы снижения энергопотребления. Достаточно точная оценка энергопотребления (например, оценка потребляемого тока) получается, если перейти от структурных логических описаний схем к схемотехническим (транзисторным) описаниям, однако для моделирования транзисторных описаний требуется гораздо большее время по сравнению с быстродействующим моделированием на логическом уровне.

В литературе для оценки энергопотребления логических схем предлагаются различные подходы. Одним из них является подход, основанный на расчете переключательной активности [4], однако здесь не принимаются в расчет кратковременные изменения внутренних сигналов схемы, связанные с гонками сигналов. Известны также подходы, базирующиеся на расчете вероят-

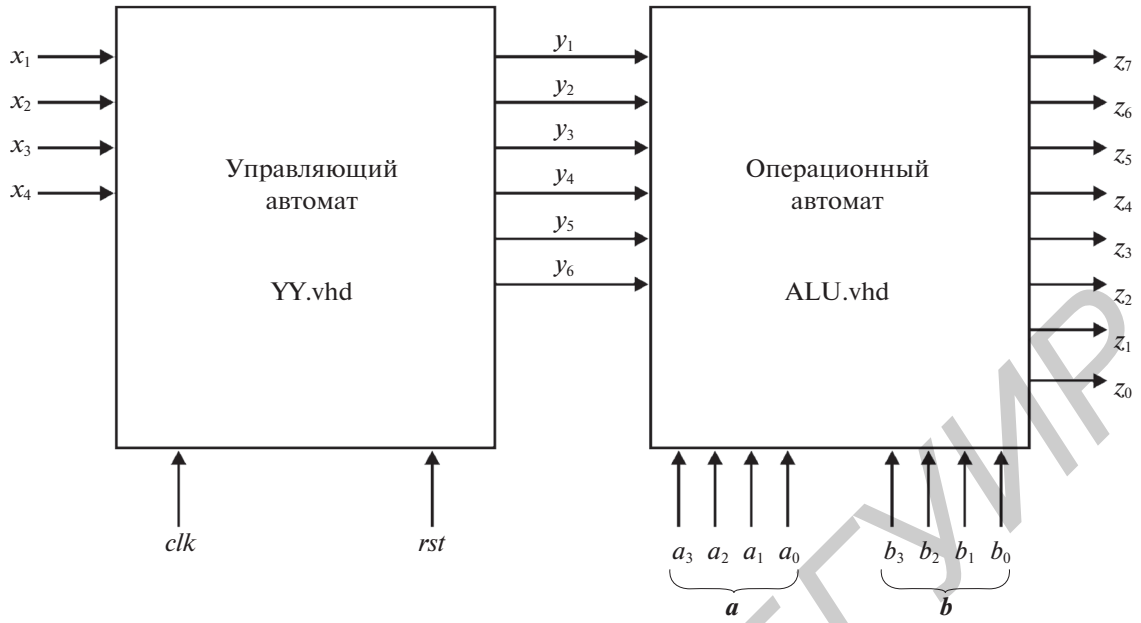

Рис. 1. Цифровое устройство *system* ($n = 4$)

ностей изменения состояний внутренних сигналов (выходных полюсов элементов) схемы по вероятностям изменения входных сигналов [5, 6]. Данные подходы являются абстрактными и малоприменимыми на практике, когда логические схемы состоят из десятков и сотен тысяч логических элементов. Более практичным считается подход, использующий логическое моделирование структурных описаний, при этом функциональность логических элементов расширяется путем введения дополнительных средств, позволяющих учитывать энергопотребление элементов при изменении состояний входных сигналов элементов [7]. Как показали эксперименты, данный подход является быстродействующим, погрешности в оценке энергопотребления не превосходят 5% для комбинационных схем и 10% для схем, реализующих конечные автоматы. В данной работе этот подход применяется для операционных цифровых устройств, представимых в виде композиции (соединения) управляющего и операционного автоматов. Такие цифровые устройства давно нашли широкое применение в практике проектирования [8] и развиваются в настоящее время [9, 10]. Исходные описания таких устройств представляются на языках VHDL и Verilog [11]. Далее будут использоваться только VHDL-описания операционных устройств. По исходным алгоритмическим описаниям строятся логические схемы в заданном базисе (целевой библиотеке) КМОП-элементов.

Предлагаемый в данной работе подход использует специфику описания цифровых устройств рассматриваемого класса и базируется на VHDL-моделировании как исходных алгоритмических описаний операционных цифровых устройств, так и структурных описаний соответствующих синхронных логических схем [12, 13]. По результатам моделирования строятся *энергоемкие* тесты, т.е. тесты, вызывающие повышенное энергопотребление. Моделирование логических схем на таких тестах позволяет найти верхнюю оценку энергопотребления в системах схемотехнического моделирования, при этом результаты моделирования энергопотребления в системах схемотехнического Spice-моделирования считаются эталонными.

1. Структура и функционирование операционного цифрового устройства. Операционное цифровое устройство состоит из управляющего и операционного автоматов [8]. Операционный автомат будем называть также операционным блоком. Простой пример такого устройства *system* представлен на рис. 1. Описание устройства на языке VHDL приведено в Приложении: VHDL-описание устройства *system* в виде соединения двух подсхем YY, ALU – в листинге 1, описание управляющего автомата YY – в листинге 2, операционного блока ALU – в листинге 3. Управляющий автомат задан графом G переходов (рис. 2), функции переходов даны в табл. 1. На рис. 2 не показаны асинхронные переходы (рис. 3) из любого состояния s_i в начальное состояние s_1 при единичном значении сигнала *rst*. Двоичные входные векторы (порты) ***a***, ***b*** называются *операнда-*

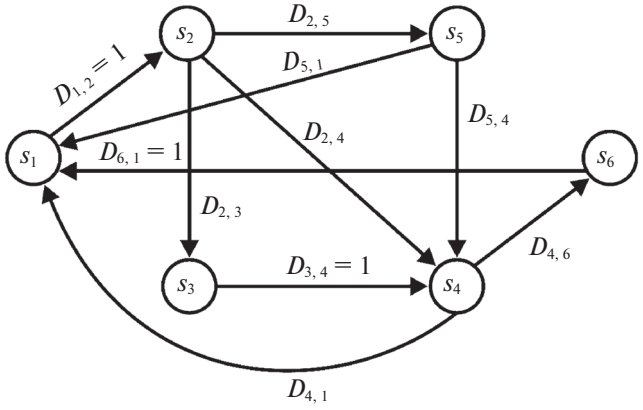

Рис. 2. Граф G переходов управляющего автомата YY

ми операционного блока. В примере число n разрядов каждого из операндов равно четырем. Функции операционного автомата заданы в табл. 2. В зависимости от признака операции функция операционного автомата может быть либо арифметической (сложение или умножение операндов $\boldsymbol{a}$, $\boldsymbol{b}$, понимаемых как двоичные коды чисел без знака), либо логической, в этом случае логическая операция выполняется над соответствующими разрядами двоичных векторов $\boldsymbol{a}$, $\boldsymbol{b}$.

Функционирование исходной VHDL-модели и реализующей ее синхронной логической схемы осуществляется по тактам. Смена состояния управляющего автомата выполняется по переднему фронту синхросигнала *clk*. В текущем такте вырабатывается *признак* $y_i = 1$ операции y_i, в следующем такте для операндов $\boldsymbol{a}$, $\boldsymbol{b}$ операционный автомат выполняет операцию y_i. Управляющий автомат начинает функционирование из начального состояния s_1, меняет свои состояния и всегда возвращается в начальное состояние. Заметим, что для других цифровых устройств такого рода в графе переходов могут быть петли, что означает, что управляющий автомат не выходит из текущего состояния, а ожидает требуемую комбинацию управляющих входных сигналов x_1, x_2, x_3,x_4, чтобы перейти в другое состояние. В процессе функционирования цепочки состояний управляющего автомата образуют на графе G различные циклы. В данном простом примере каждый переход из одного состояния в другое вызывает смену выполнения соответствующей операции в операционном блоке.

Таблица 1. Таблица переходов управляющего автомата YY

s_i	s_j	Условия перехода	Выходные сигналы (признак операции)
s_1	s_2	$D_{1,2} = 1;$	y_2
s_2	s_3	$D_{2,3} = x_1\bar{x}_2\bar{x}_3 \vee x_1x_2$	y_3
	s_4	$D_{2,4} = x_1\bar{x}_2x_3$	y_4
	s_5	$D_{2,5} = \bar{x}_1$	y_5
s_3	s_4	$D_{3,4} = 1$	y_4
s_4	s_1	$D_{4,1} = \bar{x}_2$	y_1
	s_6	$D_{4,6} = x_2$	y_6
s_5	s_1	$D_{5,1} = \bar{x}_1x_4$	y_1
	s_4	$D_{5,4} = \bar{x}_4 \vee x_1x_4$	y_4
s_6	s_1	$D_{6,1} = 1$	y_1

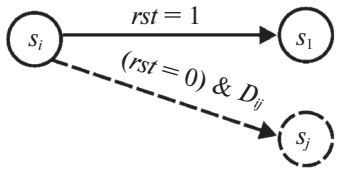

Рис. 3. Асинхронные переходы в начальное состояние s_1

2. Способы алгоритмического описания операционных устройств. 2.1. Способ A1 алгоритмического описания цифровой системы (без учета энергопотребления). Рассмотрим пример цифрового устройства для операндов длины $n = 4$. Построим по алгоритмическому описанию (Приложение, листинги 1–3) логическую схему, используя синтезатор *LeonardoSpectrum* [14]. Обратим внимание на то, что в листинге 3 приведен пример первого способа A1 алгоритмического описания операционного блока. Если синтез всего цифрового устройства ведется с использованием этого способа, то и все описание цифрового устройства будем считать описанием по способу A1.

Автоматический синтез логической схемы по способу A1 алгоритмического описания характеризуется тем, что для каждого из операндов каждая из операций в каждом такте выполняется в операционном блоке, но на выход устройства подается лишь результат той операции, выполнение которой требуется в данном такте (рис. 4).

Рассмотрим более практический случай, когда операнды $\boldsymbol{a}$, $\boldsymbol{b}$ являются 16-разрядными векторами, а логическая схема C^{A1} была построена по VHDL-описанию (листинги 1–3), когда

```
generic (
        byte_in : natural := 16;
        byte_out : natural := 32).
```

Результаты выполнения логического синтеза для $n = 16$ даны в табл. 3. Проанализировав табл. 3, можно прийти к выводу, что схемная реализация операционного блока ALU и управляющего автомата YY осуществляется независимо, при этом основная доля (1941 логический элемент) аппаратной сложности схемы C^{A1} приходится на однотактный 16-разрядный умножитель. Следующая по сложности подсхема – 16-разрядный сумматор, состоящий из 124 логических элементов. Заметим, что в схеме имеются три триггера для хранения закодированных по коду Грэя состояний $s_1, \ldots, s_6$ управляющего автомата. Элементы, реализующие логические константы 0, 1, в подсхемах повторяются, поэтому общее число 2237 элементов в логической схеме C^{A1}, реализующей все устройство *system* по способу A1 алгоритмического описания, немного меньше, чем суммарное число 2246 элементов при независимой схемной реализации операций.

2.2. Способ A2 алгоритмического описания операционного блока (с учетом энергопотребления). Методика составления VHDL-описания операционного блока по способу A2 (для снижения энергопотребления) предусматривает наличие двух процессов (Приложение, листинг 4), обеспечивающих для каждого такта работу только той подсхемы (операции), которую требуется выполнить в данном такте. Говоря более точно, сокращение

Таблица 2. Операции блока ALU

Признак операции	Операция	Тип операции
$y_1 = 1$	$z = a \,\&\, b$	Логическая
$y_2 = 1$	$z = a \vee b$	«»
$y_3 = 1$	$z = a \oplus b$	«»
$y_4 = 1$	$z = a \sim b$	«»
$y_5 = 1$	$z = a + b$	Арифметическая
$y_6 = 1$	$z = a \times b$	«»

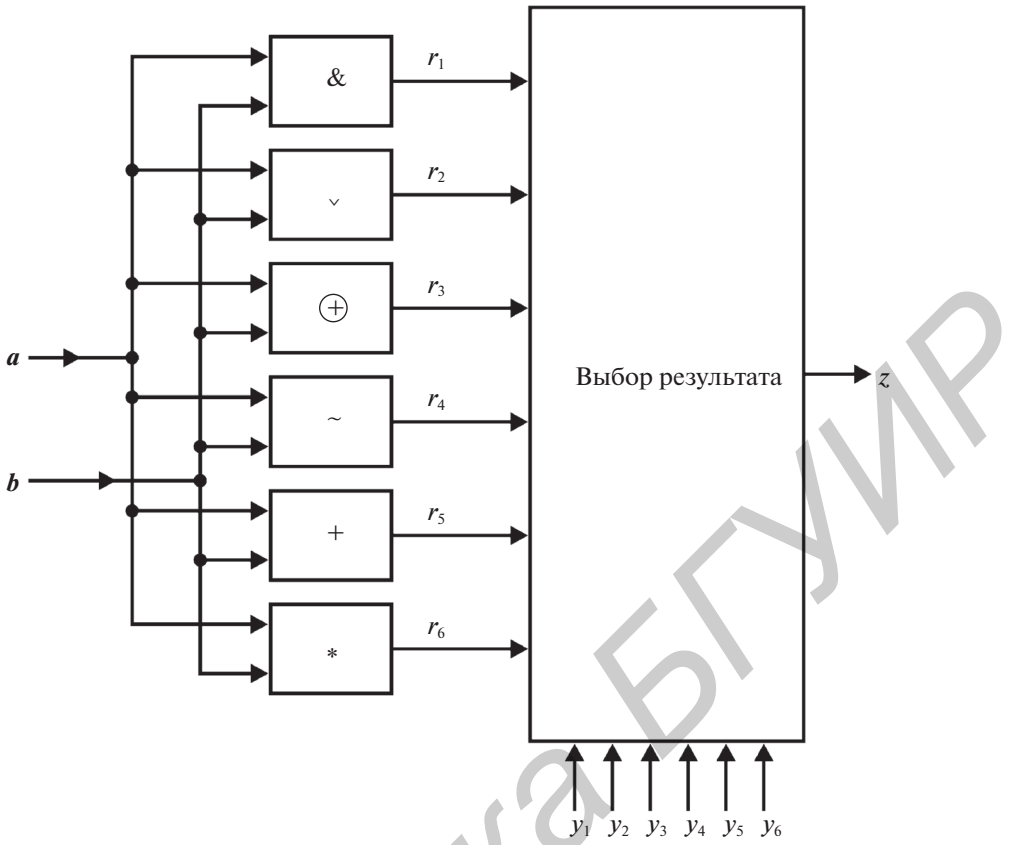

Рис. 4. Структура схемы операционного автомата, получаемая по способу A1

энергопотребления основывается на том, что операнды ***a***, ***b*** для невыполняемой операции обнуляются, и если операция не выполняется несколько тактов, то в соответствующей подсхеме в этих тактах нет переключений транзисторов в логических элементах, составляющих подсхему. Для каждой операции декларируется своя пара внутренних сигналов – векторов ***ap*(*i*)**, ***bp*(*i*)** $p = 1, ..., 6$. Первый процесс *comm* выполняет функции мультиплексора – в зависимости от значений компонент y_i передает операнды ***a***, ***b*** на вход процессов, выполняющих соответствующие операции.

Каждый из процессов *op1*,..., *op6* выполняет свою операцию в блоке ALU. Процесс *res* обеспечивает выбор результата выполненной операции и подачу его на выход операционного блока и устройства в целом. Если операция y_i не выполняется в данном такте, то в список чувствительности процесса *opi* ($i = 1, ..., 6$) подаются нулевые операнды – нуль-векторы ***a***, ***b***. Используемые

Таблица 3. Сложность схемной реализации устройства *system* ($n = 16$)

Операция (подсхема)	Число элементов подсхемы	Число элементов схемы
$z = a \,\&\, b$	33	2237
$z = a \vee b$	33	
$z = a \oplus b$	49	
$z = a \sim b$	33	
$z = a + b$	124	
$z = a \times b$	1941	
YY	33	

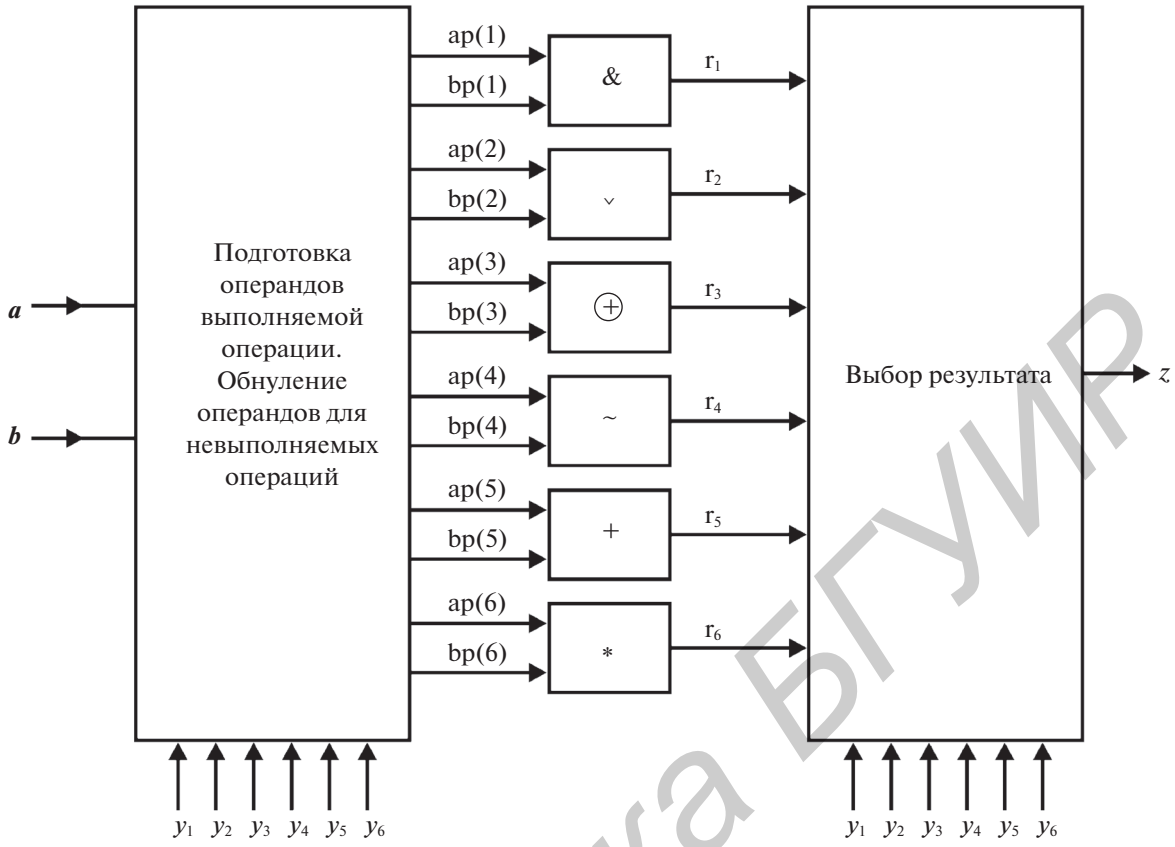

Рис. 5. Структура схемы операционного автомата, получаемая по способу A2

функции *RESIZE* и типы данных *std_logic*, *std_logic_vector*, *unsigned* определены в VHDL-пакетах, которые подробно описаны в [12].

Синтез логической схемы C^{A2} по способу A2 алгоритмического описания приводит к операционному блоку со структурой, изображенной на рис. 5. Сложность схемы C^{A2} всей цифрового устройства составляет 2825 элементов, в том числе в схеме появились дополнительно 32 элемента (триггеров) задержки LAT.

3. Формализация задачи нахождения энергоемкого теста. Управляющий автомат переходит из состояния в состояние и надо выбрать цикл на графе G переходов автомата, характеризуемый при его прохождении (в процессе моделирования) наибольшей энергоемкостью логической схемы, реализующей цифровое устройство. Переход из состояния в состояние в управляющем автомате каждый раз (в каждом такте) сопровождается выполнением операционным автоматом некоторой операции, для которой требуется подобрать операнды, обеспечивающие наибольшее энергопотребление. Таким образом, нахождение максимального энергопотребления логической схемы сводится к нахождению “энергоемкого” теста – набора тестирующих двоичных векторов, которые вызовут наибольшее энергопотребление логической схемы. В каждый набор входных сигналов данного теста будут входить как значения входных переменных для управляющего автомата, так и двоичные векторы – операнды операционного блока. Входные наборы для управляющего автомата должны обеспечить прохождение соответствующего цикла на графе G состояний управляющего автомата – назовем этот цикл “энергоемким” циклом.

4. Методы нахождения энергоемких тестов. 4.1. М е т о д М 1. Суть метода М1 заключается в моделировании VHDL-описаний на псевдослучайных тестах и выборе энергоемкого теста по циклу графа G, обладающего наибольшим энергопотреблением. Псевдослучайное моделирование позволяет проходить путь по состояниям управляющего автомата, при этом операндами для операционного автомата будут псевдослучайные двоичные векторы. Данный подход достаточно хорошо показал себя с практической стороны, так как является достаточно простым в реализа-

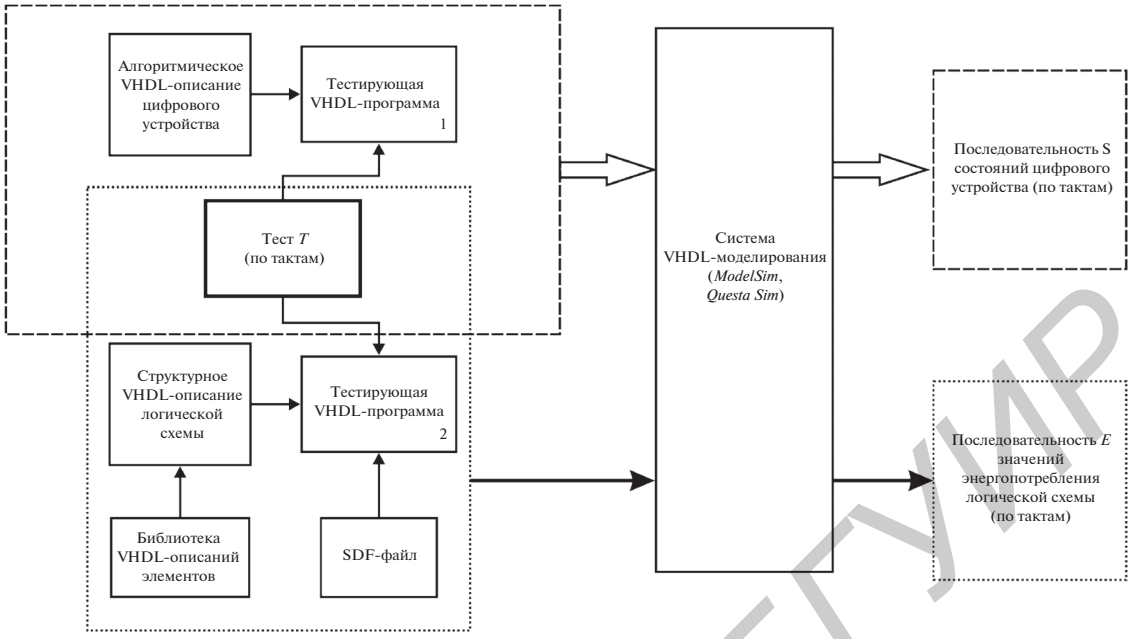

Рис. 6. Моделирование алгоритмического описания цифрового устройства и логической схемы на одном и том же тесте T

ции. Можно быстро провести VHDL-моделирование и выбрать энергоемкий тест. Достоинство данного подхода – не нужно подбирать операнды для операционного автомата, они определяются по результатам одного сеанса моделирования цифрового устройства.

Исходными данными для оценки энергопотребления по методу М1 являются два VHDL-описания: исходное алгоритмическое описание цифрового устройства и структурное описание (*net-list*) логической схемы из КМОП-элементов, функционально эквивалентное исходному описанию (рис. 6). Будем для краткости называть исходную VHDL-модель алгоритмической, а вторую VHDL-модель – структурной. Синхронные схемы синтезируются по исходным VHDL-описаниям операционных устройств в базисе библиотечных КМОП-элементов с помощью системы синтеза (синтезатора) *LeonardoSpectrum* [14]. Результатом синтеза является структурное описание логической схемы и SDF-файл о задержках элементов схемы [12].

Метод М1 оценки энергопотребления состоит из следующих семи этапов. Результатом является энергоемкий тест $T_э$.

Э т а п 1. Подготовка теста T_0 – последовательности входных тестирующих наборов-воздействий, подаваемых (при моделировании) на вход как алгоритмической модели, так и структурной модели автомата (рис. 1). Подготовку теста в форме текстового файла осуществляет обычно специальная программа – генератор тестов. Заметим, что далее в экспериментах будут использоваться только псевдослучайные тесты с равномерным распределением вероятностей появления нулей и единиц в тестирующих наборах.

Э т а п 2. Написание тестирующей программы для алгоритмической модели. Такая программа должна считывать тест и записывать (проходимые по тактам) состояния автомата в текстовый файл. Чтобы обеспечить выполнение таких функций, система моделирования должна иметь возможность работы со стандартом VHDL'2008 [13].

Э т а п 3. Выполнение VHDL-моделирования в системе *Questa Sim* для алгоритмической модели. Результатом такого моделирования является последовательность S (текстовый файл) внутренних состояний, соответствующих каждому из тактов функционирования автомата. В нулевом такте моделирования осуществляется подача на вход модели сигнала асинхронного сброса, который устанавливает VHDL-модель в начальное состояние. Каждый последующий такт моделирования включает: 1) подачу входного тестового набора, 2) переключение сигнала синхронизации в единичное состояние (формирование переднего фронта синхросигнала), 3) переключение синхросигнала в нулевое состояние (формирование заднего фронта синхросигнала).

Э т а п 4. Расширение функциональности для оценки энергопотребления VHDL-описаний логических элементов, составляющих целевую библиотеку проектирования.

Заметим, что выполнение этапа 4 осуществляется только один раз и касается библиотеки элементов, а не конкретной логической схемы, реализующей автомат. Применять модифицированную библиотеку VHDL-описаний с использованием пакетов VITAL [15] можно для оценки логических схем, полученных по произвольным синтезируемым VHDL-описаниям цифрового блока СБИС. Далее предполагается, что выполнена модификация элементов библиотеки в соответствии с подходом, предложенным в [7].

Э т а п 5. Выполнение моделирования в системе *Questa Sim* для структурной VHDL-модели с тестом T_0. Результатом такого моделирования (с подключением SDF-файла) является текстовый файл, содержащий последовательность значений энергопотребления E, каждое из которых соответствует одному такту моделирования VHDL-описания. По последовательности E легко получить среднее энергопотребление схемы на начальном тесте T_0.

Заметим, что энергопотребление в каждом такте складывается из потребления элементами схемы как при подаче тестового набора, так и при изменении синхросигнала из 0 в 1 (и обратно) (см. этап 3). В текстовый файл E для каждой пары (такта) сменяемых значений входных воздействий записывается суммарное значение энергии, потребляемой схемой в данном такте.

Таким образом, начальное моделирование позволяет получить для каждой пары тестовых воздействий (тестовых векторов – двоичных входных наборов) некоторое число, определяющее общее (динамическое и статическое) потребление, возникающее в результате переключений транзисторов, входящих в логические КМОП-элементы синхронной схемы, реализующей конечный автомат.

Э т а п 6. Формирование энергоемкого теста T_{M1} по последовательностям T_0, S, E, полученным в результате начального моделирования.

Алгоритм является простым и будет пояснен далее на примере.

Э т а п 7. Выполнение моделирования полученного структурного описания для энергоемкого теста T_{M1} с оценкой среднего энергопотребления.

Для проверки правильности функционирования логической схемы автомата на этапах 3, 5 можно формировать и записывать в соответствующие текстовые файлы значения выходов (реакций) автомата в каждом такте. Совпадение (в соответствующих тактах) выходных значений сигналов алгоритмической модели и структурной модели (логической схемы) будет свидетельствовать о том, что в результате модификации структурной модели поведение схемы не изменилось, кроме того, так может быть проверена правильность выполнения синтеза схемы.

4.2. М е т о д М 2. Исходя из оценок аппаратной сложности, примем следующее предположение: основная доля энергопотребления приходится на операционный блок. Поэтому метод М2 нахождения энергоемкого теста состоит в нахождении энергоемкого цикла с подбором соответствующих операндов для операционного блока. Подбор операндов для операционного блока заключается в отдельном моделировании для каждой команды операционного блока и нахождении пары векторов $\langle \boldsymbol{a}_i, \boldsymbol{b}_i \rangle, \langle \boldsymbol{a}_{i+1}, \boldsymbol{b}_{i+1} \rangle$, вызывающей наибольшее энергопотребление логической схемы. Если число различных команд превышает десяток, то проведение отдельного моделирования для каждой из команд является трудоемким. Поэтому может быть применен более простой и достаточно эффективный подход, когда выбирается *одна команда* (в нашем примере это арифметическая команда умножения), которая имеет наибольшую сложность схемной реализации, и для этой команды подбирается *одна пара векторов*, вызывающая наибольшее энергопотребление, назовем такую пару входных векторов *приоритетной* для операционного блока. В таком случае на графе G требуется найти цикл из начального состояния, который вызовет частое выполнение этой энергоемкой команды в операционном блоке, при этом значениями операндов будут служить по очереди векторы выбранной приоритетной пары.

Метод М2 включает следующие этапы.

Э т а п 1. Нахождение приоритетной пары операндов для операционного блока.

Э т а п 2. Выполнение метода М1 построения энергоемкого теста T_{M1} для всего цифрового устройства.

Э т а п 3. Модификация теста T_{M1} – замена операндов $\boldsymbol{a}$, $\boldsymbol{b}$ приоритетными наборами, получение теста T_{M2}.

Таблица 4. Результаты начального моделирования алгоритмического описания устройства *system* ($n = 4$) и логической схемы на 20 наборах

Такт	S	T_0			E, мкА	Цикл				
		$\boldsymbol{x}$	$\boldsymbol{a}$	$\boldsymbol{b}$		1	2	3	4	5
0	s_1					s_1				
1	s_2	1111	0101	1101	68	s_2				
2	s_5	0001	1110	0101	170	s_5				
3	s_1	0001	0101	1100	116	s_1	s_1			
4	s_2	1110	0100	1101	82		s_2			
5	s_3	1000	0000	1100	109		s_3			
6	s_4	1000	1110	0110	130		s_4			
7	s_1	0000	1101	1000	174		s_1	s_1		
8	s_2	1100	0110	0001	118			s_2		
9	s_5	0011	1111	1101	168			s_5		
10	s_1	0001	1100	1111	150			s_1	s_1	
11	s_2	0100	0010	1111	168				s_2	
12	s_3	1110	0110	0010	130				s_3	
13	s_4	1010	1100	0000	107				s_4	
14	s_1	1001	1110	0111	173				s_1	s_1
15	s_2	**0100**	**0001**	**1011**	181					s_2
16	s_5	**0111**	**1010**	**1010**	152					s_5
17	s_4	**0100**	**0000**	**1110**	151					s_4
18	s_6	**0111**	**1101**	**0101**	169					s_6
19	s_1	**1101**	**0101**	**1110**	140					s_1
20	s_2	1011	0011	1011	153					
VHDL-оценка среднего потребляемого тока, мкА						118	124	145	145	**159**

Э т а п 4. Моделирование логической схемы на полученном энергоемком тесте T_{M2} и оценка среднего энергопотребления.

4.3. М е т о д М 3. Э т а п 1. Выбор проектировщиком цикла в графе переходов, по которому предполагается наибольшая энергоемкость.

Э т а п 2. Построение теста T для управляющего автомата, позволяющего проходить заранее выбранный цикл в графе переходов.

Э т а п 3. Дополнение теста T значениями операндов $\boldsymbol{a}$, $\boldsymbol{b}$, в роли которых будут выступать наборы из приоритетной пары, полученной таким же образом, как в методе М2. Результат – тест T_{M3}.

Э т а п 4. Моделирование логической схемы на полученном энергоемком тесте T_{M3} и оценка среднего энергопотребления.

5. Пример для $n = 4$. Оценим энергопотребление схемы, полученной по способу А1, для $n = 4$ с помощью методов М1 и М2 нахождения энергоемких тестов.

5.1. М е т о д М 1. Проведем моделирование на псевдослучайном тесте T_0 из 20 наборов и получим энергоемкий цикл $\langle s_1, s_2, s_5, s_4, s_6, s_1\rangle$. Результаты оценки первых пяти циклов, которые проходятся в графе переходов G управляющего автомата, представлены в табл. 4. В данной таблице из 16000 приведены только 20 первых наборов теста T_0. Первый, второй и пятый циклы показаны на рис. 7–9 соответственно. В табл. 4 через $\boldsymbol{x} = (x_1, x_2, x_3, x_4)$ обозначен вектор входных переменных управляющего автомата, через $\boldsymbol{a} = (a_3, a_2, a_1, a_0)$, $\boldsymbol{b} = (b_3, b_2, b_1, b_0)$ – операнды, по-

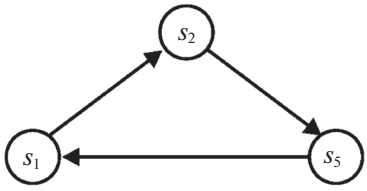

Рис. 7. Цикл 1 в графе переходов управляющего автомата

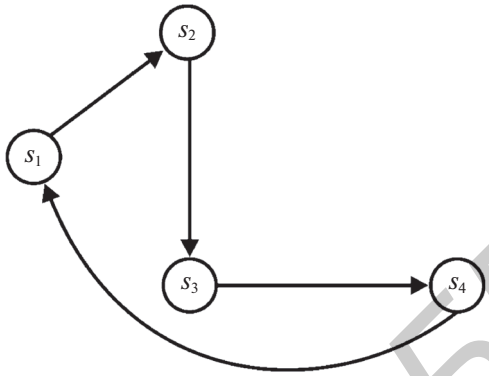

Рис. 8. Цикл 2 в графе переходов управляющего автомата

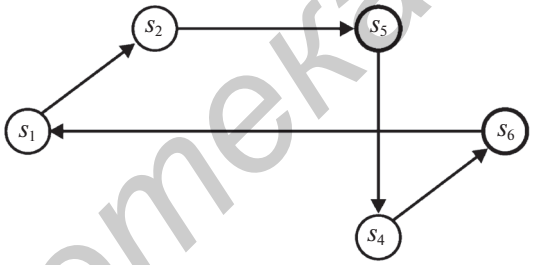

Рис. 9. Энергоемкий цикл 5 в графе переходов управляющего автомата

даваемые в операционный блок. Энергопотребление будем оценивать по среднему значению потребляемого тока, измеряемого в микроамперах (мкА). В системе схемотехнического моделирования *Accusim II*, данный параметр обозначается как *Average* [16].

Алгоритм формирования энергоемкого теста T_{M1} состоит в подсчете средней энергоемкости каждого из циклов, начинающегося в начальном состоянии s_1 (и заканчивающегося в этом же состоянии s_1), и нахождении цикла с максимальной средней энергоемкостью. Средняя энергоемкость цикла подсчитывается как сумма соответствующих элементов из последовательности E (потактового энергопотребления), деленная на число вершин в цикле. Например, цикл 5, включающий пять вершин, является наиболее энергоемким, так как его средняя энергоемкость $(181 + 152 + 151 + 169 + 140)/5 = 159$ является наибольшей среди пяти циклов, представленных в табл. 4. Соответствующие входные наборы

```
0100 0001 1011
0111 1010 1010
0100 0000 1110
0111 1101 0101
1101 0101 1110
```

Таблица 5. Сравнение VHDL-оценок среднего потребляемого тока на различных тестах для устройства *system* ($n = 4$)

Начальный тест T_0 (16000 наборов)			Тест T_{M1} (метод M1)			Тест T_{M2} (метод M2)		
x	***a***	***b***	***x***	***a***	***b***	***x***	***a***	***b***
1111	0101	1101	0000	0111	1101	0000	0100	1010
0001	1110	0101	0011	0001	1000	0011	1111	1111
0001	0101	1100	0010	1111	1111	0010	0100	1010
1110	0100	1101	0101	1011	0010	0101	1111	1111
1000	0000	1100	1000	0111	1111	1000	0100	1010
1000	1110	0110	1010	0010	1011	1010	1111	1111
0000	1101	1000	0011	0001	1000	0011	0100	1010
1100	0110	0001	0010	1111	1111	0010	1111	1111
0011	1111	1101	0101	1011	0010	0101	0100	1010
0001	1100	1111	1000	0111	1111	1000	1111	1111
0100	0010	1111	1010	0010	1011	1010	0100	1010
1110	0110	0010	0011	0001	1000	0011	1111	1111
1010	1100	0000	0010	1111	1111	0010	0100	1010
1001	1110	0111	0101	1011	0010	0101	1111	1111
0100	0001	1011	1000	0111	1111	1000	0100	1010
0111	1010	1010	1010	0010	1011	1010	1111	1111
0100	0000	1110	0011	0001	1000	0011	0100	1010
0111	1101	0101	0010	1111	1111	0010	1111	1111
1101	0101	1110	0101	1011	0010	0101	0100	1010
1011	0011	1011	1000	0111	1111	1000	1111	1111
150 мкА			221 мкА			250 мкА		

последовательности T_0, подаваемые в тактах 15–19, образуют энергоемкий тест T_{M1} для начального моделирования, выполненного на 20 тестовых наборах. Этот тест состоит из пяти входных наборов, которые можно повторять несколько раз, чтобы получить энергоемкий тест заданной длины. Если же провести моделирование на 16000 наборов (табл. 5), то получим тот же энергоемкий цикл $\langle s_1, s_2, s_5, s_4, s_6, s_1\rangle$, однако тестовые наборы будут другими. Заметим, что в табл. 5 тестовые наборы энергоемкого цикла повторяются 4 раза и составляют 20 наборов. Это сделано для того, чтобы сравнить среднюю энергоемкость логической схемы на начальной последовательности из 20 наборов с энергоемкостью функционирования той же схемы в режиме повышенного энергопотребления также на 20 наборах. Метод M2 позволяет построить тест T_{M2}, приводящий к среднему потребляемому току в 250 мкА, что значительно больше среднего тока 150 мкА на тесте T_0.

5.2. М е т о д M2. Проведем оценку энергопотребления отдельно операционного блока ALU на 2048 псевдослучайных наборах и получим приоритетную пару операндов для операционного блока:

$$\langle \boldsymbol{a}_i, \boldsymbol{b}_i\rangle = \langle 0100, 1010\rangle, \quad \langle \boldsymbol{a}_{i+1}, \boldsymbol{b}_{i+1}\rangle = \langle 1111, 1111\rangle.$$

В тесте T_{M1} заменим операнды, используя по очереди наборы из приоритетной пары, и сформируем тест T_{M2} (табл. 5). Как следует из табл. 5, метод M2 построения энергоемкого теста позволил найти лучший энергоемкий тест для логической схемы, реализующей цифровое устройство размерности $n = 4$.

6. Пример для $n = 16$. Рассмотрим построение энергоемких тестов для цифрового устройства *system* при размерности $n = 16$ операндов $\boldsymbol{a}$, $\boldsymbol{b}$. Будем строить различные энергоемкие тесты для схем, полученных по способам A1, A2 исходного алгоритмического описания цифрового устрой-

Таблица 6. Результаты эксперимента для схем C^{A1}, C^{A2}

Параметр	Логическая схема	
	C^{A1}	C^{A2}
Число элементов логической схемы	2237	**2825**
Задержка, *ns*	20.42	**31.67**
VHDL-оценка среднего потребляемого тока на тесте T_0 (16000), мкА	**5574**	2264
VHDL-оценка пикового значения потребляемого тока на тесте T_0 (16000), мкА	9413	**13600**
VHDL-оценка минимального (*Rst* = 1) потребляемого тока, мкА	16.250	**16.259**

ства для $n = 16$. Введем следующие компактные обозначения тестов, используя следующую запись:

$$T_{Mj}^{Ai}(K1, K2),$$

Ai – тест *T* предназначен для логической схемы C^{Ai}, построенной для способа *Ai* алгоритмического описания цифрового устройства, $i = 1, 2$;

Mj – тест *T* строится с помощью метода *Mj*, $j = 1, 2, 3$;

K1 – длина *начальной* тестовой последовательности для метода 1;

*K*2 – длина тестовой последовательности для выбора операндов в методах 2 и 3. Если в скобках из двух параметров *K*1, *K*2 указан только один параметр, то предполагается, что это *K*1.

Начальный псевдослучайный тест с равновероятным появлением нулей и единиц в тестовом векторе (включая и операнды) будем обозначать через $T_0(K1)$.

6.1. Построение энергоемких тестов для схемы C^{A1}. Моделирование логической схемы C^{A1} на начальном тесте $T_0(16000)$ дает средний потребляемый ток 5574 мкА, пиковое значение составляет 9413 мкА (табл. 6). На рис. 10 приведен график потребляемого схемой C^{A1} тока для 20 первых наборов начального теста $T_0(16000)$, последовательность пройденных состояний управляющим автоматом имеет вид $\langle s_2, s_5, s_1, s_2, s_3, s_4, s_1, s_2, s_5, s_1, s_2, s_3, s_4, s_1, s_2, s_5, s_4, s_6, s_1, s_2\rangle$. Вся схема операционного блока схемы C^{A1} срабатывает в каждом такте, операнды ***a***, ***b*** меняют свои значения случайным образом, что приводит к нерегулярности энергопотребления.

6.1.1. Метод М1. Данный метод позволяет получить тест $T_{M1}^{A1}(16000)$, на котором оценка среднего тока составляет 7652 мкА (табл. 7).

6.1.2. Метод М2. Моделирование блока ALU на 16000 псевдослучайных наборах позволило выбрать приоритетную пару

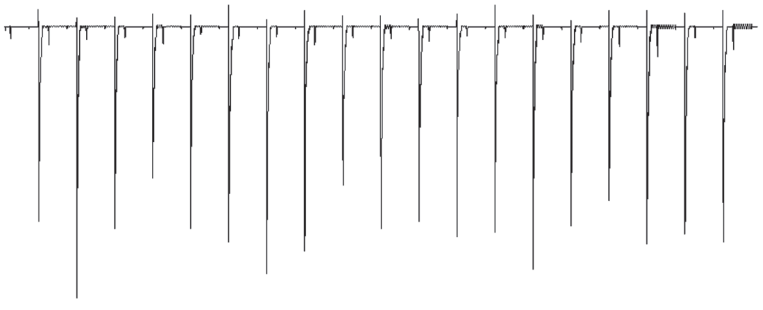

Рис. 10. График потребляемого тока схемой C^{A1} на 20 первых наборах теста $T_0(16000)$

Таблица 7. Энергопотребление на энергоемких тестах

Энергоемкий тест	Способ алгоритмического VHDL-описания цифрового устройства, логическая схема	
	C^{A1}	C^{A2}
	Оценка среднего значения потребляемого тока, мкА	
$T_{M1}^{A1}(16000)$	7652	3213
$T_{M2}^{A1}(16000, 16000)$	8262	3380
$T_{M3}^{A1}(16000, 64000)$	8303	652
$T_{M3}^{A1}(64000)$	**8444**	3987
$T_{M1}^{A2}(16000)$	6382	**5444**

$$\langle \boldsymbol{a}_i, \boldsymbol{b}_i \rangle = <1110110110011001, 0110100011010000\rangle,$$

$$\langle \boldsymbol{a}_{i+1}, \boldsymbol{b}_{i+1} \rangle = <1011110011100111, 1011111111111110\rangle$$

операндов и построить тест $T_{M2}^{A1}(16000, 16000)$ со средним потребляемым током 8262 мкА (табл. 7). Результаты схемотехнического моделирования схемы C^{A1} на тесте $T_{M2}^{A1}(16000, 16000)$ приведены на рис. 11, последовательность пройденных состояний управляющим автоматом имеет вид $\langle s_2, s_5, s_4, s_6, s_1, s_2, s_5, s_4, s_6, s_1, s_2, s_5, s_4, s_6, s_1, s_2, s_5, s_4, s_6, s_1\rangle$. График потребления тока имеет регулярный характер, так как операнды $\boldsymbol{a}$, $\boldsymbol{b}$ повторяются и повторяются состояния согласно циклу $\langle s_1, s_2, s_5, s_4, s_6, s_1\rangle$.

Проведем начальное моделирование для теста $T_0(64000)$ с целью проверки влияния длины начальной последовательности на результативность методов М2, М3.

Моделирование операционного блока ALU на тесте из 64000 псевдослучайных наборов позволило найти другую приоритетную пару

$$\langle \boldsymbol{a}_i, \boldsymbol{b}_i \rangle = \langle 1010111100110010, 1101101101011101\rangle,$$

$$\langle \boldsymbol{a}_{i+1}, \boldsymbol{b}_{i+1} \rangle = \langle 1101010110100111, 0111101111111010\rangle$$

и создать тест $T_{M2}^{A1}(16000, 64000)$, который привел к циклу $\langle s_1, s_2, s_3, s_4, s_1\rangle$. Оценка среднего потребляемого тока схемой C^{A1} на тесте $T_{M2}^{A1}(16000, 64000)$ составила 8303 мкА (табл. 7). Заметим, что на данном цикле устройство не попадает в “арифметические” состояния s_5, s_6, однако арифметические операции в схеме C^{A1} выполняются всегда, так как первый способ алгоритмического описания не заботится об энергопотреблении. Данный факт говорит о том, что увеличение дли-

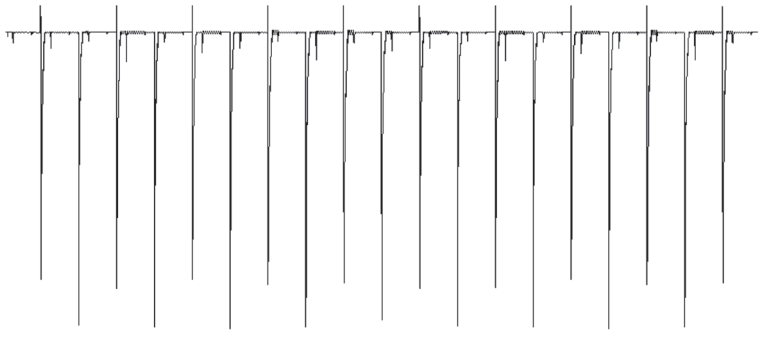

Рис. 11. График потребляемого тока схемой C^{A1} на тесте $T_{M2}^{A1}(16000, 16000)$

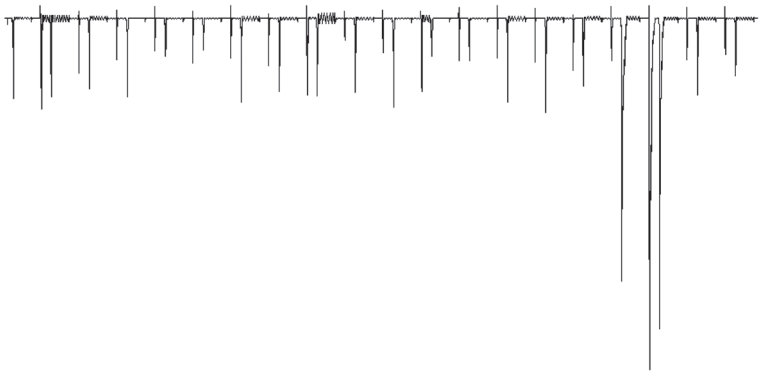

Рис. 12. График потребляемого тока схемой C^{A2} на 20 первых наборах теста $T_0(16000)$

ны начального теста позволяет выбрать лучше операнды и получить более качественный энергоемкий тест, т.е. тест с еще большим энергопотреблением моделируемой схемы.

Если составить тест $T_{M3}^{A1}(64000)$, позволяющий обходить цикл $\langle s_1, s_2, s_4, s_6, s_1\rangle$ и использовать приоритетную пару, выбранную для теста $T_{M2}^{A1}(16000,64000)$, то по результатам заключительного моделирования можно установить, что $T_{M3}^{A1}(64000)$ – наилучший энергоемкий тест (с VHDL-оценкой 8444 мкА) для логической схемы C^{A1}, построенной по способу A1 алгоритмического VHDL-описания.

Чтобы закончить рассмотрение способов конструирования тестов для схемы C^{A1}, приведем информацию о минимальном потреблении энергии этой схемой: минимальное значение потребляемого схемой C^{A1} тока 16.250 мкА (табл. 6) получается при постоянном единичном значении ($Rst = 1$) сигнала *Rst* асинхронной установки устройства в начальное состояние s_1 и нулевых значениях всех остальных входных сигналов.

6.2. Построение энергоемких тестов для схемы C^{A2}. Начальное моделирование схемы C^{A2} на 16000 наборах дает средний потребляемый ток 2264 мкА, это более чем в 2 раза лучше, чем для схемы C^{A1} на том же начальном тесте $T_0(16000)$ (табл. 6). Пиковый ток в схеме C^{A2} на отдельном такте составляет 13600 мкА (это больше пикового 9413 мкА для схемы C^{A1}). На рис. 12 показаны результаты схемотехнического моделирования схемы C^{A2} на первых 20 наборах начального теста $T_0(16000)$, последовательность пройденных состояний управляющим автоматом имеет вид $\langle s_2, s_5, s_1, s_2, s_3, s_4, s_1, s_2, s_5, s_1, s_2, s_3, s_4, s_1, s_2, s_5, s_4, s_6, s_1, s_2\rangle$. Как видно на рис. 12, выполнение трудоемкой арифметической операции умножения вызывает значительное увеличение потребляемого тока.

6.2.1. Метод М1. Построенный по этому методу тест $T_{M1}^{A2}(16000)$ (табл. 8) приводит к циклу $\langle s_1, s_2, s_3, s_4, s_6, s_1\rangle$ и дает оценку 5362 мкА среднего потребляемого тока. Результат схемотехнического моделирования схемы C^{A2} на тесте $T_{M1}^{A2}(16000)$ показан на рис. 13, последовательность пройденных состояний управляющим автоматом имеет вид $\langle s_2, s_3, s_4, s_6, s_1, s_2, s_3, s_4, s_6, s_1, s_2, s_3, s_4, s_6, s_1, s_2, s_3, s_4, s_6, s_1\rangle$. Проходимые энергоемкие состояния вызывают повышенный ток (повышенное энергопотребление).

Минимальный ток для логической схемы C^{A2} составляет 16.259 мкА (табл. 6).

7. Результаты эксперимента. Подведем первые итоги проведенного эксперимента для схем C^{A1}, C^{A2}, синтезированных по различным алгоритмическим описаниям цифрового устройства *system* для разрядности $n = 16$ операндов ***a***, ***b***. Рассмотрим табл. 6, в которой дана информация о параметрах схем C^{A1}, C^{A2}, жирным шрифтом выделены максимальные значения для каждого из параметров.

Таблица 8. Тест $T_{M1}^{A2}(16\,000)$

x	*a*	*b*
0011	1011101001110010	0111000010100111
1000	0011001010111010	0001110010111000
1100	1101111010101111	0011111111011110
0110	1110111001111101	1110011011001100
1111	1111000101110010	0100000100010110
0011	1011101001110010	0111000010100111
1000	0011001010111010	0001110010111000
1100	1101111010101111	0011111111011110
0110	1110111001111101	1110011011001100
1111	1111000101110010	0100000100010110
0011	1011101001110010	0111000010100111
1000	0011001010111010	0001110010111000
1100	1101111010101111	0011111111011110
0110	1110111001111101	1110011011001100
1111	1111000101110010	0100000100010110
0011	1011101001110010	0111000010100111
1000	0011001010111010	0001110010111000
1100	1101111010101111	0011111111011110
0110	1110111001111101	1110011011001100
1111	1111000101110010	0100000100010110

Наименьшую площадь и наибольшее быстродействие схемы C^{A1} обеспечивает способ А1 алгоритмического описания операционного блока, однако за это приходится “платить” наибольшим энергопотреблением. Способ А2, по сути, является противоположным – среднее энергопотребление схемы C^{A2} меньше в 2.5 раза, однако площадь схемы C^{A2} возросла на 26%, задержка – на 55% по сравнению со схемой C^{A1}.

Продолжим эксперимент, выполнив моделирование схемы C^{A2} на построенных энергоемких тестах.

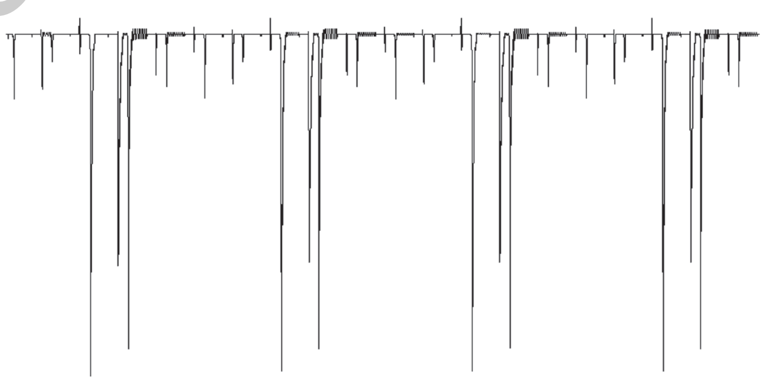

Рис. 13. График потребляемого тока схемой C^{A2} на тесте $T_{M1}^{A2}(16\,000)$

Таблица 9. Увеличение энергопотребления

Схема	Средний потребляемый ток на начальном тесте $T_0(16000)$, мкА	Средний потребляемый ток на лучшем энергоемком тесте, мкА	Увеличение потребляемого тока на лучшем энергоемком тесте, %
C^{A1}	5574	8444	51.4
C^{A2}	2264	5444	140.4

Результаты представлены в табл. 7, для каждой схемы жирным шрифтом выделены максимальные значения потребляемого тока. Анализ результатов позволяет сделать следующие выводы.

1. Для способа A1 алгоритмического описания операционного блока решающее значение имеет выбор приоритетной пары операндов, увеличение длины начального теста с 16000 до 64000 позволило выбрать более энергоемкую пару операндов. Выбор цикла (метод M3), в котором наиболее часто повторяется операция умножения, привел к построению лучшего энергоемкого теста, на котором схема C^{A1} имеет наибольший средний потребляемый ток 8444 мкА. Эффективность методов M1 и M2 определяется длиной и видом начального теста.

2. Хорошие энергоемкие тесты для схемы C^{A1} не являются хорошими энергоемкими тестами для схем C^{A2}.

3. В процессе моделирования схемы C^{A1}, не ориентированной на снижение энергопотребления, на тесте $T_{M3}^{A1}(16000, 64000)$ не выполняются трудоемкие арифметические операции (только логические), поэтому для схемы C^{A2}, ориентированной на снижение энергопотребления, средний потребляемый ток (652 мкА) незначителен.

4. В табл. 9 приведено повышение среднего потребляемого схемами тока на начальной последовательности и на лучшем энергоемком (для данной схемы) тесте.

5. Актуальной проблемой является решение задачи подбора приоритетных пар операндов для построения энергетических тестов для схем, построенных по способу A2 описания операционных блоков.

8. Методика оценки энергопотребления. Предлагаемая практическая методика оценки энергопотребления синхронного цифрового устройства состоит в выполнении следующих шагов.

1. Разработать VHDL-модели логических КМОП-элементов, позволяющие учитывать потактовую переключательную активность либо потребляемый ток при моделировании структурного VHDL-описания логической схемы.

2. Провести оценку энергопотребления структурного VHDL-описания логической схемы, используя быстродействующее VHDL-моделирование, и получить энергоемкие тесты с помощью предложенных в данной статье методов.

3. По структурному VHDL-описанию схемы получить транзисторное Spice-описание схемы и выполнить схемотехническое Spice-моделирование на энергоемких тестах.

В результате применения методики будет получена оценка среднего энергопотребления транзисторного описания КМОП-схемы, функционирующей *в режиме повышенного энергопотребления* на энергоемком тесте, и сокращено время моделирования. Погрешность оценки энергопотребления проверяется путем сравнения результатов эталонного схемотехнического моделирования и VHDL-моделирования на энергоемком тесте.

Результаты эксперимента по сокращению времени моделирования и точности (погрешности) оценки потребляемого тока приведены в табл. 10, в которой через $I_{Quesa\ Sim}$ обозначены средние значения потребляемого тока, полученные VHDL-моделированием в системе *Questa Sim*; через $I_{Accusim\ II}$ – средние значения тока, полученные в системе *Accusim II* схемотехнического моделирования [16]. Функциональные описания схем Mealy, Bar_157, Bar_204, Bar_38 взяты из [17].

Для промышленных примеров схем сокращение времени является весьма значительным. Например, моделирование транзисторного описания схемы C^{A1} в системе *Accusim II* схемотехнического Spice-моделирования на первых 100 наборах теста $T_0(16000)$ потребовало 55 мин, следова-

Таблица 10. Сокращение времени при замене схемотехнического моделирования логическим моделированием

Имя схемы	Число элементов схемы	Число наборов теста T_0	VHDL-моделирование		Схемотехническое моделирование		Погрешность оценки среднего тока, %
			Время, с	$I_{Questa\ Sim}$, мкА	Время, с	$I_{Accusim\ II}$, мкА	
Mealy	59	1000	1.5	50.72	104	48.50	4.5
Bar_157	103	1000	2.0	68.27	196	62.18	9.7
Bar_204	198	1000	5.3	102.78	350	98.12	4.7
Bar_38	200	1000	6.1	130.82	331	118.41	10.4

тельно, моделирование на всем тесте $T_0(16000)$ потребует, скорее всего, не менее 8400 мин (140 ч). VHDL-моделирование схемы C^{A1} на том же тесте $T_0(16000)$ заняло 21 мин работы того же компьютера, что в 400 раз меньше.

Заключение. Используя предложенные способы алгоритмического описания операционного блока, можно оценивать варианты схем по параметрам энергопотребления, площади, быстродействия и выбирать приемлемый вариант схемы, реализующей цифровое устройство в целом. Существенного снижения энергопотребления синхронных цифровых устройств, представимых в виде композиции управляющего и операционного автоматов, можно добиться, используя соответствующее описание их функционирования на алгоритмическом уровне. Изменение нескольких операторов в алгоритмическом описании может значительно изменить параметры схемы. Важное значение для сокращения времени моделирования при построении энергоемких тестов имеет замена трудоемкого схемотехнического моделирования логическим VHDL-моделированием при решении задач практической размерности. Предлагаемые методы построения энергоемких тестов могут быть использованы и при наличии быстродействующих систем схемотехнического моделирования транзисторных описаний логических схем, позволяющих получать потактовые значения энергопотребления моделируемых схем.

ПРИЛОЖЕНИЕ

Л и с т и н г 1. VHDL-описание цифрового устройства *system* ($n = 4$) в виде соединения двух подсхем:

```
library ieee;
use ieee.std_logic_1164.all;
entity system is
 generic (
     byte_in : natural := 4;
    byte_out : natural := 8);
     port(x : in std_logic_vector(1 to 4);
    clk, rst : in std_logic;
a : in std_logic_vector(byte_in-1 downto 0);
b : in std_logic_vector(byte_in-1 downto 0);
z : out std_logic_vector(byte_out-1 downto 0));
end;
architecture str of system is
component YY is
port( x : in std_logic_vector(1 to 4);
 clk, rst : in std_logic;
        y : out std_logic_vector(1 to 6));
end component;
component ALU
```

```
generic (
    byte_in : natural := 4;
    byte_out : natural := 8);
 port (
    a : in std_logic_vector(byte_in-1 downto 0);
    b : in std_logic_vector(byte_in-1 downto 0);
    y : in std_logic_vector(1 to 6);
    z : out std_logic_vector(byte_out-1 downto 0));
end component;
 signal y : std_logic_vector(1 to 6);
 begin
m1: YY port map (x => x, clk => clk, rst => rst, y => y);
m2: ALU port map (a => a, b => b, y => y, z => z);
end;
```

Л и с т и н г 2. Алгоритмическое VHDL-описание управляющего автомата YY:

```
library ieee;
use ieee.std_logic_1164.all;
entity YY is
port( x : in std_logic_vector(1 to 4);
 clk, rst : in std_logic;
     y : out std_logic_vector(1 to 6));
end YY;

architecture rtl of YY is
type T_state is (s1, s2, s3, s4, s5, s6);
signal NEXT_state, state: T_state;
begin
NS : process (state, x)
begin
case state is
when s1 =>
          NEXT_state <= <= s2; y <= "010000";
when s2 =>
          if ((x(1) and not x(2) and not x(3)) or (x(1) and x(2))) = '1'
             then NEXT_state <= s3; y <= "001000";
            elsif ((x(1) and not x(2) and x(3)) = '1')
          then NEXT_state <= s4; y <= "000100";
          elsif (not x(1) = '1')
          then NEXT_state <= s5; y <= "000010";
          end if;
when s3 => NEXT_state <= s4; y <= "000100";
when s4 => if (not x(2) = '1')
          then NEXT_state <= s1 ; y <= "100000";
          elsif ( x(2) = '1')
          then NEXT_state <= s6 ; y <= "000001";
          end if;
when s5 =>
if ((not x(1) and x(4) ) = '1')
then NEXT_state <= s1; y <= "100000";
elsif ( (not x(4)) or (x(1) and x(4)) ) = '1'
```

```
            then NEXT_state <= s4; y <= "000100";
            end if;
                 when s6 => NEXT_state <= s1; y <= "100000";
            end case;
  end process NS;
  state <= s1 when rst = '1' else
  NEXT_state when (clk'event and clk = '1') else state;

  end;
```

Листинг 3. Способ A1 алгоритмического VHDL-описания операционного блока ALU:

```
  library ieee;
  use ieee.std_logic_1164.all;
  use ieee.numeric_std.all;
  entity ALU is
    generic (
      byte_in : natural := 4;
      byte_out : natural := 8);
    port (
      a : in std_logic_vector(byte_in-1 downto 0);
      b : in std_logic_vector(byte_in-1 downto 0);
      y : in std_logic_vector(1 to 6);
      z : out std_logic_vector(byte_out-1 downto 0));
  end alu;
  architecture beh of ALU is
  signal zz : unsigned(byte_out-1 downto 0);
  begin
    z <= std_logic_vector(zz);
    zz <=
  RESIZE(unsigned(a and b),byte_out) when y="100000" else
  RESIZE(unsigned(a or b),byte_out) when y="010000" else
  RESIZE(unsigned(a xor b),byte_out) when y="001000" else
  RESIZE(unsigned(a xnor b),byte_out) when y="000100" else
  RESIZE((unsigned(a) + ('0'& unsigned(b)) ),byte_out) when y="000010"
else
  RESIZE(unsigned(a) * unsigned(b),byte_out) when y="000001" else
        (others => 'X');
  end beh;
```

Листинг 4. Способ A2 алгоритмического VHDL-описания операционного блока ALU:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity ALU is
  generic (
    byte_in : natural := 4;
    byte_out : natural := 8);
  port (
    a : in std_logic_vector(byte_in-1 downto 0);
    b : in std_logic_vector(byte_in-1 downto 0);
    y : in std_logic_vector(1 to 6);
    z : out std_logic_vector(byte_out-1 downto 0));
end alu;
architecture beh of ALU is
```

```
signal r1, r2, r3, r4, r5, r6 : unsigned(byte_out-1 downto 0);
constant zero : std_logic_vector(3 downto 0):= "0000";
type prom is array (1 to 6) of std_logic_vector(byte_in-1 downto 0);
signal ap, bp : prom;
function operation (
signal a: std_logic_vector(byte_in-1 downto 0);
signal y : std_logic_vector(1 to 6)) return prom is
variable result : prom;
  constant zero : std_logic_vector(byte_in-1 downto 0):= (others => '0');
begin
lp1: for i in 1 to 6 loop
if (y(i)= '1') then result (i) := a;
             else result (i) := zero;
end if;
end loop lp1;
return result;
end operation;
begin
comm: process (y)
begin
  ap <= operation (a => a, y => y);
  bp <= operation (a => b, y => y);
end process;
op1: process (ap(1), bp(1))
begin
  r1 <= RESIZE(unsigned(ap(1) and bp(1)),byte_out);
end process;
op2: process (ap(2), bp(2))
begin
  r2 <= RESIZE(unsigned(ap(2) or bp(2)),byte_out);
end process;
op3: process (ap(3), bp(3))
begin
  r3 <= RESIZE(unsigned(ap(3) xor bp(3)),byte_out);
end process;
op4: process (ap(4), bp(4))
begin
  r4 <= RESIZE(unsigned(ap(4) xnor bp(4)),byte_out);
  end process;
op5: process (ap(5), bp(5))
begin
  r5 <= RESIZE((unsigned(ap(5)) + ('0'& unsigned(bp(5))) ),byte_out);
  end process;
op6: process (ap(6), bp(6))
begin
  r6 <= RESIZE(unsigned(ap(6)) * unsigned(bp(6)),byte_out);
  end process;
res: process (r1, r2, r3, r4, r5, r6)
begin
  if y="100000" then z <= std_logic_vector(r1);
  elsif y="010000" then z <= std_logic_vector(r2);
  elsif y="001000" then z <= std_logic_vector(r3);
```

```
   elsif y="000100" then z <= std_logic_vector(r4);
   elsif y="000010" then z <= std_logic_vector(r5);
   elsif y="000001" then z <= std_logic_vector(r6);
else null;
end if;
end process;
end;
```

СПИСОК ЛИТЕРАТУРЫ

1. *Рабаи Ж.М., Чандракасан А., Николич Б.* Цифровые интегральные схемы, 2-е изд.: Пер. с англ. М.: ООО “И.Д. Вильямс”, 2007.
2. *Белоус А.И., Мурашко И.А., Сякерский В.С.* Методы минимизации энергопотребления при проектировании КМОП БИС // Технология и конструирование в электронной аппаратуре. 2008. № 2.
3. *Немудров В., Мартин Г.* Системы-на-кристалле. Проектирование и развитие. М.: Техносфера, 2004.
4. *Ghosh A., Devadas S., Keutzer K., White J.* Estimation of Average Switching Activity in Combinational and Sequential Circuits. // Proc. 29th ACM/IEEE Design Automation Conf. Anaheim, California, USA, 1992.
5. *Roy K., Prasad S.C.* Low Power CMOS VLSI Circuit Design. N. Y.: John Wiley and Sons, Inc., 2000.
6. *Грэсь Т.Н., Соловьев В.В.* Минимизация потребляемой мощности конечных автоматов путем расщепления внутренних состояний // Изв. РАН. ТиСУ. 2015. № 3.
7. *Бибило П.Н., Соловьев А.Л.* Оценка энергопотребления комбинационных КМОП схем на основе логического моделирования с учетом временных задержек элементов // Управляющие системы и машины. 2014. № 6.
8. *Майоров С.А., Новиков Г.И.* Структура электронных вычислительных машин. Л.: Машиностроение, 1979.
9. *Skliarova I., Sklyarov V., Sudnison A.* Design of FPGA-based Circuits Using Hierarchical Finite State Machines. Tallinn: TUT Press, 2012.
10. *Иванюк А.А.* Проектирование встраиваемых цифровых устройств и систем. Минск: Бестпринт, 2012.
11. *Поляков А.К.* Языки VHDL и VERILOG в проектировании цифровой аппаратуры. М.: СОЛОН-Пресс, 2003.
12. *Бибило П.Н., Авдеев Н.А.* VHDL. Эффективное использование при проектировании цифровых систем. М.: СОЛОН-Пресс, 2006.
13. *Ashenden P.J., Lewis J.* VHDL-2008. Just the New Stuff. Burlington, MA, USA: Morgan Kaufman Publishers, 2008.
14. *Бибило П.Н.* Системы проектирования интегральных схем на основе языка VHDL. StateCAD, ModelSim, LeonardoSpectrum. М.: СОЛОН-Пресс, 2005.
15. *Munden R.* ASIC and FPGA Verification: A Guide to Component Modeling. San Francisco: Morgan Kaufmann Publ., 2005.
16. *Авдеев Н.А., Бибило П.Н.* Оценка энергопотребления цифрового блока СБИС // Современная электроника. 2009. № 9.
17. *Баранов С.И., Скляров В.А.* Цифровые устройства на программируемых БИС с матричной структурой. М.: Радио и связь, 1986.