УДК 519.714.5
https://doi.org/10.37661/1816-0301-2024-21-3-23-38

Оригинальная статья
Original Article

Извлечение логических сетей при декомпиляции описаний КМОП-схем на уровне транзисторов

Д. И. Черемисинов, Л. Д. Черемисинова✉

Объединенный институт проблем информатики
Национальной академии наук Беларуси,
ул. Сурганова, 6, Минск, 220012, Беларусь
✉*E-mail: cld@newman.bas-net.by*

Аннотация

Цели. Рассматривается проблема восстановления функционального описания цифровых устройств СБИС, представленных на транзисторном уровне. Целью исследования является разработка метода и программных средств выделения блоков, представляющих логические сети, из двухуровневых описаний КМОП-схем на транзисторном уровне, которые были получены в результате распознавания (экстракции) подсхем, реализующих логические элементы.

Методы. Предлагаются графовые методы и программные средства извлечения связных блоков, представляющих логические сети, из двухуровневых описаний транзисторных схем в формате SPICE. В графовой интерпретации задача сводится к построению помеченного ориентированного графа логической сети, исходя из помеченного неориентированного двудольного графа, задающего двухуровневое описание транзисторной схемы.

Результаты. Предложенный метод позволяет выделить лексикографически ранжируемые логические сети, от которых производится переход к логическим уравнениям, задающим функции, реализуемые на выходных полюсах полученных сетей. Разработаны программные средства, которые обеспечивают генерацию иерархического описания в формате SPICE, реализующего исходную схему на транзисторном уровне, а также описания выделенных логических сетей на языке SF иерархических структурно-функциональных описаний дискретных устройств и на языках высокого уровня (VHDL и Verilog).

Заключение. Разработанные программные средства включены в программу декомпиляции транзисторных КМОП-схем и протестированы в ее составе на практических примерах схем транзисторного уровня. В работе приведены примеры обратного инжиниринга некоторых практических транзисторных схем.

Ключевые слова: экстракция транзисторных подсхем, КМОП-схемы, формат SPICE, распознавание логических вентилей, логическая сеть, обратный инжиниринг

Для цитирования. Черемисинов, Д. И. Извлечение логических сетей при декомпиляции описаний КМОП-схем на уровне транзисторов / Д. И. Черемисинов, Л. Д. Черемисинова // Информатика. – 2024. – Т. 21, № 3. – С. 23–38. https://doi.org/10.37661/1816-0301-2024-21-3-23-38

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Поступила в редакцию | Received 23.07.2024
Подписана в печать | Accepted 20.08.2024
Опубликована | Published 30.09.2024

Extraction of logical networks during decompiling transistor-level CMOS circuit descriptions

Dmitry I. Cheremisinov, Ljudmila D. Cheremisinova✉

The United Institute of Informatics Problems
of the National Academy of Sciences of Belarus,
st. Surganova, 6, Minsk, 220012, Belarus
✉*E-mail: cld@newman.bas-net.by*

Abstract

Objectives. The problem of restoring the functional description of digital VLSI devices presented at the transistor level is considered. The objective of the work is to develop means for extraction of blocks representing logical networks from two-level descriptions of CMOS circuits at the transistor level, which were obtained as a result of recognition (extraction) of subcircuits that implement logic elements.

Methods. Graph based methods and software tools are proposed for extracting a connected blocks representing a logical network from two-level descriptions of a transistor circuits in SPICE format. In the graph interpretation, the task is reduced to constructing a labeled directed graph of a logical network based on a labeled undirected bipartite graph specifying a two-level description of the transistor circuit.

Results. The proposed method makes it possible to identify lexicographically ranked logical networks, from which a transition is made to logical equations that specify the functions implemented at the outputs of the resulting networks. Software tools have been developed that provide the generation of a hierarchical description in SPICE format that implements the original circuit at the transistor level, as well as descriptions of found logical networks in the SF language of hierarchical structural and functional descriptions of discrete devices and in high-level languages (VHDL and Verilog).

Conclusion. The developed methods are implemented in C++, included in the program for decompiling transistor CMOS circuits and tested within it on practical examples of transistor-level circuits. The paper provides examples of reverse engineering of some practical transistor circuits.

Keywords: transistor subcircuit extraction, CMOS circuits, SPICE format, logical gates recognition, logical network, reverse engineering

For citation. Cheremisinov D. I., Cheremisinova L. D. *Extraction of logical networks during decompiling transistor-level CMOS circuit descriptions*. Informatika [*Informatics*], 2024, vol. 21, no. 3, pp. 23–38 (In Russ.). https://doi.org/10.37661/1816-0301-2024-21-3-23-38

Conflict of interest. The authors declare of no conflict of interest.

Введение. Мощным инструментом проектирования и тестирования топологии современных СБИС, содержащих миллионы транзисторов, является обратное проектирование, или обратный инжиниринг, СБИС (англ. hardware reverse engineering). Обратное проектирование широко используется для верификации [1, 2] и перепроектирования [3] интегральных схем на новый технологический базис. Кроме того, в последние годы появились работы, в которых предлагается использовать его и для обнаружения несанкционированных вкладок. Такие вкладки могут быть внесены в топологию устройства при его изготовлении [4–6].

В процессе обратного инжиниринга по плоскому (одноуровневому) структурному описанию интегральной схемы на транзисторном уровне восстанавливается ее описание на уровне логических элементов. Эта операция в смысле направления преобразований является инверсной задаче проектирования СБИС, и если перевод с уровня логических элементов на уровень транзисторов по аналогии с программированием можно считать компиляцией, то обратный процесс (извлечение описания уровня логических элементов) является декомпиляцией описания схемы на транзисторном уровне [7].

Структурный анализ цифровых схем путем их декомпиляции изучается в течение длительного времени. Обзор результатов решения этой задачи можно найти в работах [7–10]. Рассматривается наиболее распространенный стиль логики – КМОП – логических комплементарных МОП-структур (металл – оксид – полупроводник). Исходным объектом при декомпиляции является плоское описание схемы исследуемой СБИС. Для этой схемы может быть известна биб-

лиотека логических элементов, использованная при компиляции топологии СБИС, т. е. набор транзисторных подсхем, представляющих логические элементы. В этом случае декомпиляция состоит в распознавании (и замене) в описании декомпилируемой схемы библиотечных подсхем либо на функциональном уровне путем сравнения реализуемых подсхемами функций, либо на структурном уровне путем анализа изоморфизма соответствующих графов соединений транзисторов [11].

Библиотека логических элементов может быть и неизвестна, тогда в процессе декомпиляции выделяются часто встречающиеся фрагменты декомпилируемой схемы, которые выглядят как логические элементы. Те подсхемы, для которых распознаны реализуемые ими функции, являются логическими элементами, остальные относятся к классу псевдоэлементов [12]. Выделенные фрагменты составляют библиотеку элементов для анализируемой транзисторной схемы. В результате декомпиляции [7, 11, 12] плоское описание транзисторной схемы преобразуется в двухуровневое описание путем формирования уровня подсхем библиотечных элементов и генерации связей между экземплярами этих элементов в декомпилируемой схеме.

В настоящей работе исследуется полученное в результате декомпиляции СБИС двухуровневое описание транзисторной КМОП-схемы, представленной в формате SPICE (Simulation Program with Integrated Circuit Emphasis). Решается задача выделения блоков, представляющих логические сети и, соответственно, состоящих только из логических элементов. В процессе построения логической сети исходная транзисторная схема, в которой связи элементов рассматриваются как двунаправленные, преобразуется в логическую схему с направленными связями. Для схемы находятся входные и выходные порты и вычисляются функции, реализуемые схемой.

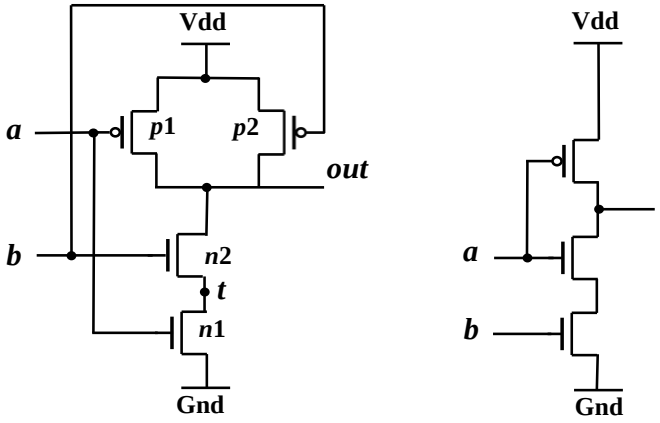
В графовой интерпретации задача извлечения логической сети сводится к построению помеченного ориентированного графа сети исходя из помеченного неориентированного графа, задающего двухуровневое описание транзисторной схемы. В общем случае могут быть выделены несколько логических сетей и могут оставаться элементы (ими будут, например, псевдоэлементы), не вошедшие ни в одну из сетей. После экстракции логических блоков строится иерархическое SPICE-описание, в которое эти блоки входят как элементы (наряду с оставшимися после экстракции элементами).

Для полученной логической сети (сетей) генерируется описание на языке SF иерархических структурно-функциональных описаний дискретных устройств [13], который является внутренним языком программных средств проектирования компонентов СБИС [14] в Объединенном институте проблем информатики НАН Беларуси. Возможна трансляция полученного SF-описания и на другие языки высокого уровня (VHDL и Verilog).

Разработанные методы реализованы на языке C++, включены в программу декомпиляции транзисторных КМОП-схем и протестированы в ее составе на практических примерах схем транзисторного уровня. Приведены примеры обратного инжиниринга некоторых практических транзисторных схем.

Постановка задачи декомпиляции транзисторных схем. Исходным объектом при декомпиляции является плоское описание схемы, которая состоит из полевых p-МОП- и n-МОП-транзисторов. МОП-транзистор имеет четыре вывода: сток (drain), затвор (gate), исток (source) и подложку (substrate). Доминирующим стилем логики при разработке современных цифровых СБИС является КМОП-структура [15], которая состоит из МОП-транзисторов двух упомянутых типов.

Элемент на основе стандартной (комплементарной) КМОП-логики включает две подсхемы, которые состоят из одинакового числа соответственно p-МОП- и n-МОП-транзисторов и реализуют взаимно инверсные булевы функции. Блок из p-МОП-транзисторов (Pull-Up network) размещен между шиной питания Vdd и выходом элемента, n-МОП-блок (Pull-Down network) – между шиной земли Gnd и выходом (рис. 1, *а*). Блоки обеспечивают связь выхода схемы элемента с источником питания Vdd, когда значение сигнала на выходе предполагается равным единице, или с землей Gnd в противном случае. Стандартные КМОП-схемы относятся к классу статических схем, в которых каждый выход в любой момент времени связан либо с источником питания Vdd, либо с шиной земли Gnd через тракт с малым сопротивлением.



a) *b)*

Рис. 1. Элементы КМОП-логики: *a)* вентиль И-НЕ на два входа; *b)* псевдоэлемент

Fig. 1. Elements of CMOS transistor logic: a) CMOS NAND gate; b) pseudo element

Предполагается, что значение сигнала на выходе КМОП-элемента представляется булевой функцией от сигналов, поступающих на вводы транзисторов соответствующей подсхемы (если игнорировать переходные эффекты во время переключения). Однако кроме таких (правильных) КМОП-вентилей при декомпиляции могут выделяться и псевдоэлементы, отличающиеся от КМОП-вентилей, например, тем, что p-МОП- и n-МОП-блоки имеют разную мощность или не реализуют взаимно инверсные булевы функции (рис. 1, *b*).

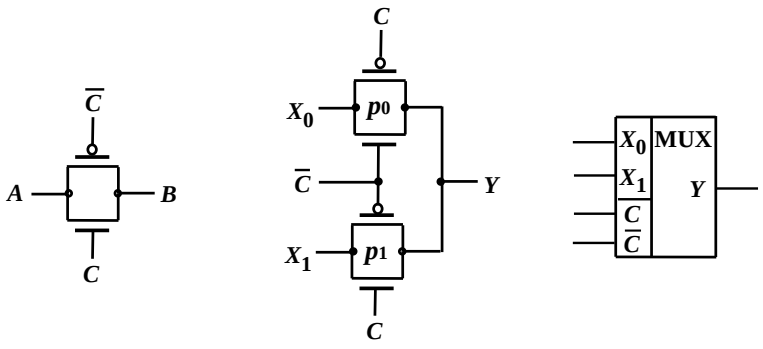
Наряду со стандартной КМОП-логикой при проектировании сложных регулярных структур интегральных схем широко используется логика на проходных транзисторах (pass transistor logic, PTL) [15], которая позволяет существенно сократить число транзисторов, необходимых для реализации логических функций, а также снизить энергопотребление схемы. В проходной логике сигналы с входных портов схемы допускается использовать для питания не только затворов транзисторов, но и выводов стока и истока n-МОП- и p-МОП-транзисторов. На эти выводы могут подаваться и внутренние сигналы схемы.

Однако транзисторы n-МОП-типа эффективно работают при передаче нулевого сигнала, но значительно понижают уровень сигнала логической единицы, а транзисторы p-МОП-типа, наоборот, эффективно работают при передаче сигнала логической единицы, но значительно понижают уровень сигнала логического нуля. В связи с этим транзисторы n-МОП- и p-МОП-типа используются, как правило, не по отдельности, а соединяются попарно, образуя передаточный вентиль. Такой вентиль состоит из пары транзисторов n-МОП и p-МОП-типа, которые связаны параллельно своими выводами стока и истока (рис. 2, *a*) [13]. Передаточные вентили могут использоваться в схемах как трехстабильные элементы, а также могут входить в состав схем сложных элементов, таких как мультиплексоры или вентили исключающее ИЛИ, которые реализуются более просто на основе передаточной логики, чем на основе стандартной статической КМОП-логики (рис. 2, *b*, *c*).

Подсхемы, представляющие передаточные вентили, находятся на этапе обработки плоского описания декомпилируемой схемы, которая предваряет экстракцию из него описаний стандартных КМОП-вентилей. После нахождения передаточных вентилей распознаются подсхемы мультиплексоров и вентилей исключающее ИЛИ [16].

На следующем этапе в транзисторной схеме распознаются подсхемы, реализующие статические КМОП-вентили. Структурный подход к декомпиляции КМОП-схем [12] позволяет разбить транзисторную схему на непересекающиеся подсхемы, представляющие группы транзисторов, связанных по постоянному току. Среди найденных групп транзисторов выделяются правильные подсхемы, представляющие собой статические КМОП-вентили, и определяются реализуемые ими функции. Каждая из подсхем, не распознанных как КМОП-вентиль, объявля-

ется псевдоэлементом, функциональное описание которого структурными методами в общем случае определить не удается.



a) *b)* *c)*

Рис. 2. Элементы проходной транзисторной логики: *a)* передаточный КМОП-вентиль; *b)* схемная реализация мультиплексора MUX 2×1 на основе передаточных вентилей; *c)* схематическое представление мультиплексора

Fig. 2. Elements of pass transistor logic: a) CMOS transmission gate; b) multiplexer MUX 2×1 circuit implementation based on transmission gates; c) schematic representation of the multiplexer

Далее множества всех распознанных подсхем, представляющих выделенные элементы (КМОП-вентили, передаточные вентили, псевдоэлементы, мультиплексоры, вентили исключающее ИЛИ, трехстабильные инверторы), разбиваются на классы функционально и топологически эквивалентных подсхем [11]. Каноническое описание представителя каждого класса эквивалентных подсхем вентилей представляет собой найденный библиотечный элемент. Им заменяются все экземпляры подсхемы этого класса в плоском описании декомпилируемой схемы, порождая двухуровневое описание декомпилируемой транзисторной схемы.

Задание транзисторных схем. Полученная при декомпиляции двухуровневая транзисторная схема задается в формате SPICE [1]. При распознавании подсхем, реализующих логические элементы, анализируется основная часть SPICE-описания транзисторной схемы Data Statements, которая описывает компоненты схемы и взаимосвязи между ними. Остальные части, если они присутствуют в исходном файле, игнорируются.

Раздел Data Statements описывает компоненты схемы в виде «моделей» элементов. В SPICE используются два основных типа моделей, которые и рассматриваются в настоящей работе: модели устройств (device models) и модели подсхем (subcircuit models). Модели устройств, по сути, представляют собой типы примитивных компонентов, таких как транзисторы, диоды и т. д. Модель подсхемы определяет схему непримитивного элемента и обычно (хотя и не обязательно) представляет собой схему. Модели других типов игнорируются. Модель подсхемы (.subckt в SPICE-описании) задается в виде блока в SPICE-файле и описывает структуру соединений элементов (примитивных и непримитивных) в этой подсхеме.

Тип модели используемого в SPICE-описании элемента задается первой буквой его имени. Имена примитивных элементов, являющихся МОП-транзисторами, должны начинаться с латинской буквы M. Имена непримитивных элементов, представляемых подсхемами, начинаются с латинской буквы X. Следует заметить, что описания на языке SPICE нечувствительны к регистру символов, т. е., в частности, допустимо указывать и M, и m (X и x).

Схема задается списком соединений ее элементов. В этом списке указаны связи между элементами, а точнее, между их выводами, которые осуществляются с помощью электрических цепей (nets). Каждая цепь представляет собой связанный набор выводов элементов, на которые подается один и тот же сигнал, и каждый вывод элемента подключен ровно к одной цепи. Для каждого элемента указываются связи всех его выводов путем задания имен связанных с ними цепей.

Описание транзистора начинается с его имени и содержит перечисление меток цепей, связанных с выводами стока, затвора, истока и подложки в заранее определенной последовательности. Общая форма описания связей униполярного транзистора в формате SPICE имеет вид

<name> <nd> <ng> <ns> <nb> <model-name>,

где name – имя транзистора в схеме (начиная с буквы M); nd, ng, ns и nb – идентификаторы цепей, связанных с выводами стока (drain), затвора (gate), истока (source) и подложки (substrate) соответственно; model-name – имя модели устройства (для *n*-МОП- и *p*-МОП-транзисторов это могут быть nmos и pmos).

Общая форма описания связей непримитивного элемента, модель которого представляется подсхемой с *n* выводами, в формате SPICE имеет вид

X<name> <P1> < P2> …< Pn> <model-name>,

где name – имя элемента в схеме; P1, P2… Pn – идентификаторы цепей, связанных с выводами элемента; model-name – имя модели элемента.

В программе декомпиляции принято соглашение, что имена моделей элементов разного типа различаются первой буквой. Например, имена моделей логических элементов начинаются с буквы G, псевдоэлементов – с P, передаточных вентилей – с CN, мультиплексоров – с `GM`. Логические элементы являются (*n*, 1)-полюсниками, т. е. имеют один выходной полюс. По соглашению последний параметр в описании моделей всех элементов всегда соответствует выходному полюсу и именуется как Y.

Пример представления транзисторной схемы в виде иерархического описания в формате SPICE, полученного в результате декомпиляции плоского описания схемы `fd`, приведен в листинге 1. В этом описании представлены четыре модели непримитивных элементов, задающих вентиль 2И-НЕ, инвертор, передаточный элемент и мультиплексор. Их модели обозначены соответственно через `G0`, `G1`, `CN2` и `GMAB_0`. Последний уровень иерархии схемы содержит девять экземпляров логических элементов: четыре вентиля 2И-НЕ, три инвертора и два мультиплексора MUX 2×1. Последний параметр в описании моделей соответствует выходному полюсу. Передаточные вентили являются двунаправленными, поэтому описание их модели `CN2` не имеет параметра, обозначаемого как `Y`. В описании используются четыре экземпляра элемента модели `CN2`, и все они входят в состав мультиплексоров.

Листинг 1. *Иерархическое SPICE-описание транзисторной схемы* `fd`

```
* SPICE deck for cell fd_gen

.GLOBAL  vcc  gnd
.SUBCKT G0 A B Y
* (A AND B)
M1 1 A gnd gnd nmos
M2 Y B 1 gnd nmos
M3 vcc A Y vcc pmos
M4 Y B vcc vcc pmos
.ENDS

.SUBCKT G1 A Y
* A
M1 Y A gnd gnd nmos
M2 Y A vcc vcc pmos
.ENDS

.SUBCKT CN2 A nC B C NB PB
M1 A nC B PB pmos
M2 A C B NB nmos
.ENDS

.SUBCKT GMAB_0 A B C nC Y
```

```
* (A AND C) OR (B AND nC)
XCN20 Y nC A C gnd vcc CN2
XCN21 B C Y nC gnd vcc CN2
.ENDS

.SUBCKT fd_gen rl sl c d q
XM0I1 rl 11 14 G0  Fets=nmos10+nmos9+pmos24+pmos25
XM0I2 sl 12 11 G0  Fets=nmos11+nmos12+pmos26+pmos27
XM0I3 8 rl 9 G0  Fets=nmos0+nmos1+pmos15+pmos16
XM0I4 sl 9 13 G0  Fets=nmos2+nmos3+pmos17+pmos18
XM1I1 11 qn G1  Fets=nmos13+pmos28
XM1I2 12 q G1  Fets=nmos14+pmos29
XM1I3 c 10 G1  Fets=nmos6+pmos21
XGMUAB_0 9 14 c 10 12 GMAB_0
XGMUAB_1 13 d c 10 8 GMAB_0
.ENDS
```

Графовая модель транзисторных схем и ее внутреннее представление. При декомпиляции удобной моделью схем на уровне транзисторов является помеченный неориентированный двудольный граф $G = (V_1, V_2, E)$, $V_1 \cap V_2 = \varnothing$. Этот граф задает структуру списка соединений элементов транзисторной схемы (транзисторов и подсхем из транзисторов). В нем вершины из первой доли V_1 графа соответствуют выводам экземпляров элементов (транзисторов или подсхем из транзисторов, представляющих элементы) и портам схемы (входам и выходам электрической схемы). Вершинам из V_2 ставятся в соответствие цепи, т. е. связи между выводами элементов. Примерами цепей являются цепи питания и земли, которые связаны с большим числом элементов схемы. Каждая из дуг $e \in E$ графа G связывает вершины из разных множеств V_1 и V_2.

Вершины графа из множества V_2 помечаются именами цепей, а вершины из V_1 имеют метки, идентифицирующие выводы экземпляров элементов. Эти метки состоят из имени экземпляра элемента и имени вывода модели этого элемента. Двудольный граф $G = (V_1, V_2, E)$, являющийся моделью схемы в формате SPICE, разреженный, и степени всех вершин в доле V_1 выводов элементов равны единице.

При решении задачи декомпиляции граф G задается в виде массива N списков смежности вершин из множества V_1. Такой массив обычно индексируется вершинами: каждой вершине из V_1 соответствует список ее соседей, в данном случае список связанных с ней цепей. Поскольку в графе G, задающем модель схемы в формате SPICE, степени всех вершин из множества V_1 равны единице, то списки соседей всех вершин такого графа являются одноэлементными и для задания графа G используется одномерный массив N.

Для того чтобы привязать задание графа к структуре транзисторной схемы, массив N смежности вершин графа G предлагается разбивать и, следовательно, индексировать на списки смежности вершин графа, соответствующих выводам отдельных элементов схемы (на начальном этапе декомпиляции элементами являются транзисторы). Для каждого элемента выделяется список связей его выводов, приведенных в том порядке, в котором они упоминаются в описании связей модели этого элемента в формате SPICE. Список связей i-го элемента схемы в массиве смежности вершин графа задается номером в N цепи, смежной вершине, которая соответствует первому выводу этого элемента (для транзистора это сток, для непримитивного элемента – вывод P1). Порядок следования списков цепей в массиве смежности определяется порядком следования элементов в SPICE-описании схемы.

Например, в табл. 1 приведен массив N смежности вершин из множества V_1 графа $G = (V_1, V_2, E)$, представляющего схему `fd` (листинг 1). Элементами этого массива являются вершины из V_2, соответствующие цепям схемы. Индексами 0, 3, 6, 9, 12, 14, 16, 18 и 23 массив делится на девять списков, задающих связи выводов отдельных элементов. Для удобства восприятия массив показан не в виде вектора, а разбит на списки, каждый из которых соответствует выводам одного элемента и приведен в отдельной строке. Каждая из строчек табл. 1 задает связи выводов одного элемента схемы. Например, первая строка соответствует элементу `M0I1`

(в листинге `xM0I1`), тип которого определяется моделью `G0`, а связи задаются парами `M0I1.P1 = rl`, `M0I1.P2 = l1` и `M0I1.Y = l4`.

Таблица 1
Массив смежности вершин графа, соответствующих выводам элементов схемы
Table 1
Adjacency array of graph vertices corresponding to the circuit element outputs

Элементы схемы *Elements of the diagram*	Связи выводов элементов *Connections of element pins*
M0I1	rl 11 14
M0I2	sl 12 11
M0I3	8 rl 9
M0I4	sl 9 13
M1I1	11 qn
M1I2	12 q
M1I3	c 10
GMUAB_0	9 14 c 10 12
GMUAB_1	13 d c 10 8

Упомянутое задание графа $G = (V_1, V_2, E)$, представляющего транзисторную схему, строится на этапе обработки исходного двухуровневого описания в формате SPICE. Табл. 1 полностью определяет граф $G = (V_1, V_2, E)$ схемы. Однако для сокращения вычислительных затрат при извлечении связных блоков, представляющих логические сети, из двухуровневых описаний транзисторных схем одновременно с построением списков смежности вершин из множества V_1 формируются также и списки вершин, смежных вершинам из множества V_2. Другими словами, связи цепей описания схемы задаются также в явном виде. Для рассматриваемой схемы эти связи показаны во втором столбце табл. 2, где через точку приведены имена элемента и его вывода.

Таблица 2
Массив смежности вершин графа, соответствующих цепям схемы
Table 2
Adjacency array of graph vertices corresponding to the circuit nets

Цепи *Chains*	Связи *Connections*
rl	M0I1.A M0I3.B
sl	M0I2.A M0I4.A
c	M1I3.A GMUAB_0.C GMUAB_1.C
d	GMUAB_1.B
q	M1I2.Y
qn	M1I1.Y
11	M0I1.B M0I2.Y M1I1.A
14	M0I1.Y GMUAB_0.B
12	M0I2.B M1I2.A GMUAB_0.Y
8	M0I3.A GMUAB_1.Y
9	M0I3.Y XM0I4.B GMUAB_0.A
13	M0I4.Y GMUAB_1.A
10	M1I3.A GMUAB_0.nC GMUAB_1.nC

Извлечение логической сети из двухуровневой транзисторной схемы. Логическая сеть дискретного устройства отражает его внутреннее строение с точностью до функций, реализуемых его элементами. В графовой интерпретации моделью логической сети является помеченный ориентированный граф $H = (W, A)$, где W и A – множества вершин и дуг графа. Множество W разбивается на три подмножества вершин, соответствующих входным и выходным портам сети и элементам. Каждая вершина из первых двух подмножеств помечена входной или выходной переменной сети. Вершины из третьего подмножеств помечены функциями, реализуемыми элементами сети.

Ориентированный граф $H = (W, A)$ логической сети строится исходя из неориентированного графа $G = (V_1, V_2, E)$, соответствующего объектной двухуровневой транзисторной схеме.

В множество W помимо вершин, соответствующих портам сети, входят вершины, соответствующие экземплярам логических элементов. Помимо таких элементов могут существовать (как результат декомпиляции) и элементы, которые не распознаны как логические. Например, они могут соответствовать псевдоэлементам. Наличие таких элементов приводит к тому, что может быть извлечена не логическая сеть, а несколько логических блоков. Эти блоки порождаются непересекающимися связными подграфами $H_i = (W_i, A_i)$ графа $G = (V_1, V_2, E)$.

Дуги $a \in A$ генерируемого графа $H = (W, A)$ логической сети порождаются цепями $x \in V_2$ графа G. Вершины $v \in W$ и $u \in W$ связываются дугой $a = (v, u)$, соответствующей цепи x, если эта цепь связывает выводы элементов, соответствующих вершинам v и u, причем для v цепь x указана в качестве выхода, а для u – в качестве одного из входов.

Выделение компонентов, представляющих логические сети, из двухуровневого описания транзисторной схемы включает решение следующих основных задач:

Задача 1. Поиск компонента связности графа $G = (V_1, V_2, E)$, описывающего логическую сеть $H^* = (W^*, A^*)$.

Задача 2. Определение входных и выходных портов извлеченной логической сети, описываемой графом $H^* = (W^*, A^*)$.

Задача 3. Генерация описания логической сети на языках высокого уровня.

Следует заметить, что, если декомпилируемая транзисторная схема получена путем автоматического синтеза с помощью какой-либо САПР с заданной библиотекой проектирования, результатом решения задачи 1 является одна логическая сеть, функционально эквивалентная исходной транзисторной схеме. Задача 2 представляет собой самостоятельный этап декомпиляции транзисторных схем, о которых ничего неизвестно, кроме описания структуры связей их транзисторов. Решение этой задачи нетривиально для сложных транзисторных схем, содержащих сотни миллионов транзисторов. Решение задачи 3 является завершающим этапом декомпиляции схемы, оно дает возможность использовать известные САПР для моделирования транзисторной схемы и перепроектирования ее на основе другого технологического базиса.

Задача 1. Поиск компонента связности графа $G = (V_1, V_2, E)$, описывающего логическую сеть. Поиск компонента связности графа $G = (V_1, V_2, E)$, описывающего логическую сеть $H^* = (W^*, A^*)$, осуществляется в процессе обхода графа G по входящим и исходящим путям, начиная от вершин из V_1, которые помечены как выводы некоторого произвольно взятого экземпляра логического элемента, не вошедшего ни в одну из ранее найденных логических сетей. При встрече вершины, не помеченной как логический элемент, просмотр соответствующего пути обрывается. Обход графа с целью построения компоненты связности, представляющей логическую сеть, осуществляется известным методом поиска в ширину (BFS, от англ. breadth-first search).Такой алгоритм имеет линейную временную сложность.

Метод поиска позволяет не только найти компоненту связности графа G и соответствующий граф логической сети $H^* = (W^*, A^*)$, но и получить лексикографическое упорядочение вершин графа H^*, учитывающее достижимость вершин друг из друга, и, следовательно, ранжировать вершины графа по уровням.

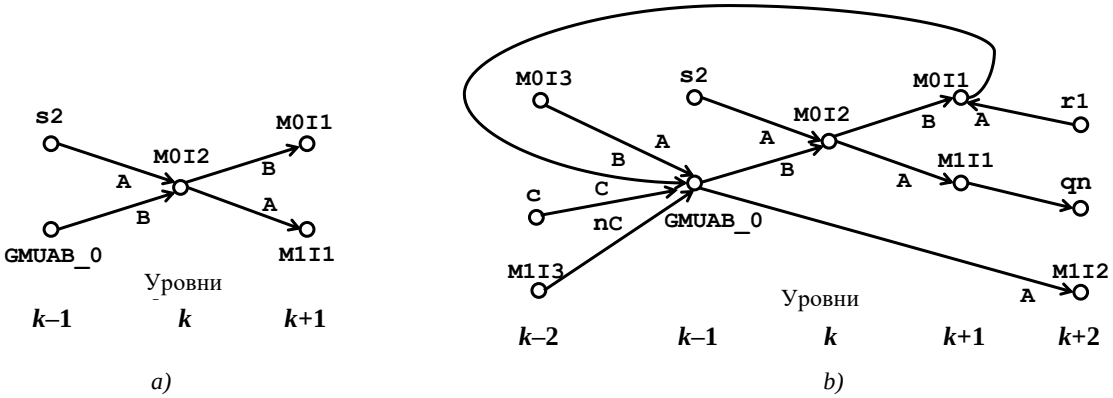
Поиск в ширину начинается с некоторой начальной вершины v графа, соответствующей некоторому логическому элементу. Рассмотренные в процессе просмотра вершины отмечаются как просмотренные. Анализируемая вершина v_0 включается в множество W^* и помечается как

вершина уровня k. Затем в множество W^* вносятся все вершины, которые соответствуют логическим элементам и входят в полуокрестности захода $\Gamma^+ v_0$ и исхода $\Gamma^- v_0$. Вершины из $\Gamma^+ v_0$ помечаются как вершины уровня (k–1), а вершины из $\Gamma^- v_0$ – уровня (k+1). После этого аналогично рассматриваются вершины из окрестностей всех вновь введенных в граф $H^* = (W^*, A^*)$ вершин и т. д. В общем случае все не помеченные ранее вершины из окрестностей уровня k помечаются как вершины уровня (k–1) или (k + 1). Просмотр вершин прекращается, когда все вершины будут помечены как просмотренные.

Процедура построения графа $H^* = (W^*, A^*)$ логической сети ведется, исходя из массивов смежности вершин из множеств V_1 и V_2 графа $G = (V_1, V_2, E)$ (см. табл. 1 и 2).

Для примера начнем с вершины v_0, соответствующей элементу `M0I2` (см. табл. 1), который не входит ни в одну ранее построенную логическую сеть и ранее не рассматривался. Выводы этого элемента связаны с цепями `sl`, `12` и `11`, причем полуокрестность захода $\Gamma^- v_0 = \{$`sl`, `12`$\}$, а полуокрестность исхода $\Gamma^+ v_0 = \{11\}$ (см. табл. 1). Вершину v_0 включим в множество W^* графа $H^* = (W^*, A^*)$ и отнесем ее к некоторому уровню k. Среди вершин из V_1 графа $G = (V_1, V_2, E)$, смежных вершине `sl` $\in V_2$ (их две: `M0I2.A` и `M0I4.A`), ищем вершину, соответствующую выводу `Y` какого-либо логического элемента. Таких вершин нет, значит, вершина `sl` является концевой. Среди вершин из V_1, смежных вершине `12` (их три: `M0I2.B`, `M1I2.A` и `GMUAB_0.Y`), находится вершина (для правильно построенной схемы она всегда одна), удовлетворяющая этому условию: `GMUAB_0.Y`. Она соответствует выходному полюсу элемента `GMUAB_0` и является драйвером второго входа элемента `M0I2`. Введем в сеть вершину для `GMUAB_0.Y` и отнесем ее к уровню (k–1). Аналогично рассматривается единственная вершина `11` из $\Gamma^+ v$, но в этом случае берутся все смежные ей вершины, соответствующие выводам элементов, отличных от `Y` (их две: `M0I1.B` и `M1I1.A`). Вершины, соответствующие элементам `M0I1` и `M1I1`, вносятся в множество W^* графа сети и относятся к уровню (k + 1). В множество A^* вносятся дуги (`sl`, `M0I2`), (`GMUAB_0`, `M0I2`), (`M0I2`, `M0I1`), (`M0I2`, `M1I1`).

Аналогичные действия повторяются для каждой из вновь введенных в граф вершин (`GMUAB_0`, `M0I1`, `M1I1`) уровней (k–1) и (k+1) до тех пор, пока они находятся. На рис. 3 показаны состояния графа логической сети после выполнения первой и второй итераций. Для каждой входящей дуги графов на рисунке приведено имя соответствующего вывода элемента (согласно описанию его модели на листинге 1).



a) *b)*

Рис. 3. Граф $H^* = (W^*, A^*)$ после выполнения первой итерации (*a*); второй итерации (*b*)

Fig. 3. Graph $H^* = (W^*, A^*)$ after the first iteration (a); the second iteration (b)

Элементы, входящие в иерархическое SPICE-описание `fd`, порождают единственный связный компонент графа $G = (V_1, V_2, E)$, порождающий граф $H^* = (W^*, A^*)$ (рис. 4). Вершин, не вошедших в этот компонент, не осталось, значит, построена единственная логическая сеть, реализующая анализируемую транзисторную схему.

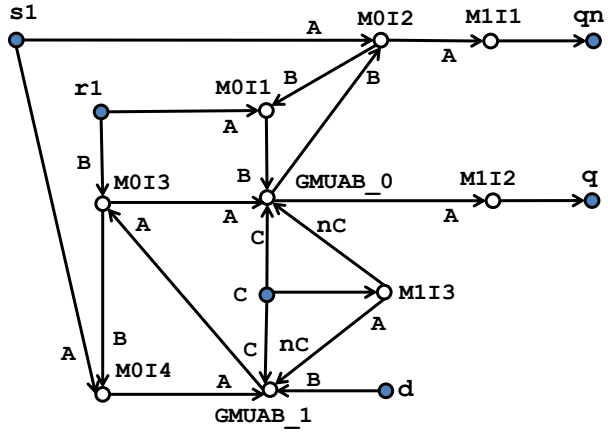


Рис. 4. Граф $H^* = (W^*, A^*)$ логической сети

Fig. 4. Graph $H^* = (W^*, A^*)$ of the logical network

Задача 2. Определение входных и выходных портов логической сети. Следующей задачей, связанной с выделением логической сети, является определение ее входных и выходных портов. Эта задача просто решается путем рассмотрения полуокрестностей исхода $\Gamma^+ v$ и захода $\Gamma^- v$ для всех вершин v графа $H^* = (W^*, A^*)$.

Если для некоторой вершины $v \in W^*$ все вершины из ее окрестностей $\Gamma^+ v$ и $\Gamma^- v$ помечены как элементы, то вершина v является внутренней. Вершины, не являющиеся внутренними, связаны с входами или выходами логической сети в зависимости от того, какое из множеств $\Gamma^+ v$ или $\Gamma^- v$ содержит каждую из этих не внутренних вершин.

Входы полученной логической сети соответствуют вершинам `rl`, `sl`, `c` и `d`, так как в графе $H^* = (W^*, A^*)$ полуокрестности захода этих вершин являются пустыми ($\Gamma^- v = \varnothing$). Выходы сети порождаются вершинами `q` и `qn`, так как полуокрестности исхода этих вершин $\Gamma^+ v = \varnothing$. На рис. 4 вершины графа $H^* = (W^*, A^*)$, для которых полуокрестности исхода или захода являются пустыми, помечены затемненным цветом.

В листинге 2 приведен фрагмент SPICE-описания, которое было получено после выделения компонента `C0`, описывающего логическую сеть. Приведенный фрагмент есть результат преобразования раздела `.SUBCKT fd_gen` (листинг 1). Остальные разделы листингов 1 и 2 совпадают. Здесь параметры модели `C0`, имена которых начинаются с символа `P` (`P0`, `P1`, `P2`, `P3`), задают входные полюсы, а параметры, имена которых начинаются с символа `O` (`O4`, `O5`), – выходные полюсы сети.

Листинг 2. *Фрагмент иерархического SPICE-описания транзисторной схемы* `fd`*, полученный после выделения логической сети*

```
.SUBCKT C0 P0 P1 P2 P3 O4 O5
XG0M0I1 P2 2 5 G0
XG0M0I2 P3 3 2 G0
XG0M0I3 6 P2 7 G0
XG0M0I4 P3 7 4 G0
XG1M1I1 P0 1 G1
XG1M1I2 2 O5 G1
XG1M1I3 3 O4 G1
XGMAB_0GMUAB_0 7 5 P0 1 3 GMAB_0
XGMAB_0GMUAB_1 4 P1 P0 1 6 GMAB_0
.ENDS
.SUBCKT fd_gen_ier rl sl c d q qn
XC0 c d rl sl q qn C0
.ENDS
```

Логическая сеть, построенная по SPICE-описанию транзисторной схемы, приведенному в листинге 2, показана на рис. 5.

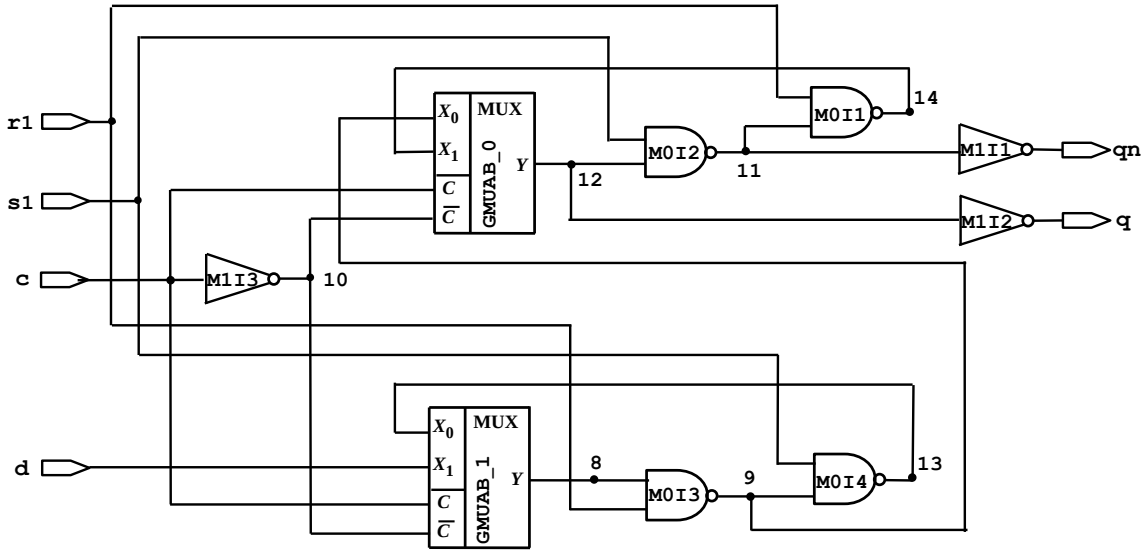


Рис. 5. Логическая сеть
Fig. 5. Logical network

Заметим, что задача определения входных и выходных портов логической сети имеет и самостоятельный интерес. Анализ сложных транзисторных схем, для которых известна только структура связей транзисторов, сильно затруднен без явного перечисления входных и выходных портов устройства. Нахождение внешних портов таких схем представляет собой трудоемкую задачу.

Для приводимого в статье демонстрационного примера список внешних портов (выделены жирным шрифтом) задан (хотя и без разделения на входы и выходы): `.SUBCKT fd_gen rl sl c d q` (листинг 1). В процессе декомпиляции после извлечения логической сети порты были разделены на входные и выходные. Однако оказалось, что приведенный в описании схемы список портов неполон: имеется еще один выходной порт – `qn`: `.SUBCKT fd_gen_ier rl sl c d q` **`qn`** и `XC0 c d rl sl q` **`qn`** (листинг 2).

Задача 3. Генерация описания логической сети на языках высокого уровня. Предлагаемый метод выделения логической сети ранжирует логические элементы по уровням. От такой сети несложно осуществить переход к логическим уравнениям, задающим функции, которые реализуются на ее внутренних и выходных полюсах. В работе реализован алгоритм построения представления логической сети в формате LOG [13] логических уравнений языка SF описания иерархических структурно-функциональных описаний дискретных устройств. Этот алгоритм исходит из иерархического SPICE-описания транзисторной схемы, полученного после выделения логической сети. Реализованы также программные средства перевода описаний на языке SF на другие языки проектирования, такие как VHDL и Verilog [17]. Для полученной логической сети LOG- и VHDL-описания приведены в листингах 3 и 4.

Листинг 3. *Описание логической* сети `C0` *в формате LOG языка SF*

```
TITLE C0
FORMAT SF
AUTHOR extractor
DATE 18-04-2024
PROJECT fd_gen_ier
DCL_PIN
EXT
```

```
INP
P0  P1  P2  P3
OUT
O4  O5
INTER
END_PIN
FUNCTION
LOG
4 2 4
O4  = ^ s3 ;
O5  = ^ s2 ;
s2  = ^ (P3  * s3 );
s3  = (s7  * P0 ) + ((^(P2  * s2 )) * s1 );
s1  = ^ P0 ;
s7  = ((^((^(P3  * s7 )) * P0 ) + (P1  * s1 )) * P2 );
END_LOG
END_FUNCTION
END_C0
```

Листинг 4. *Описание логической* сети `C0` *в формате языка VHDL*

```
library IEEE;
use ieee.Std_Logic_1164.all;

entity C0 is
  port (
    P0 : in std_logic;
    P1 : in std_logic;
    P2 : in std_logic;
    P3 : in std_logic;
    O4 : out std_logic;
    O5 : out std_logic
  );
end entity C0;

architecture sf of C0 is
signal s2: std_logic;
signal s3: std_logic;
signal s1: std_logic;
signal s7: std_logic;

begin
O4<=NOT s3;
O5<=NOT s2;
s2<=NOT (P3 AND s3);
s3<=((s7 AND P0) OR (NOT (P2 AND s2) AND s1));
s1<=NOT P0;
s7<=(NOT (NOT (P3 AND s7) AND P0) OR (P1 AND s1)) AND P2;

end architecture sf;
```

Обсуждение практических результатов. Предложенный графовый метод выделения компонентов следующего уровня иерархии SPICE-описания, которые представляют собой логические сети, позволяет переходить от описания схемы на уровне транзисторов к описанию на уровне логических элементов. Как правило, результатом такого преобразования является одна логическая сеть, поведение которой на функциональном уровне эквивалентно поведению исходной транзисторной схемы. Это гарантируется, если схема на уровне транзисторов получена с помощью систем автоматизации проектирования на основе некоторой заданной библиотеки логических элементов.

Произвольные транзисторные схемы, которые синтезированы некоторым иным способом, могут содержать выделяемые при декомпиляции подсхемы псевдоэлементы, для которых не удается определить функциональное описание. Соответственно, помимо выделенной логической сети в описании могут присутствовать и не вошедшие в нее элементы (в том числе туда могут попасть и КМОП-вентили). Плоские описания таких схем на транзисторном уровне декомпилируются в смешанные иерархические описания на транзисторно-логическом уровне. В них могут быть также несколько компонентов, представленных логическими сетями, и могут присутствовать компоненты, описанные на транзисторном уровне. Сведения о декомпиляции некоторых таких SPICE-описаний приведены в табл. 3, где к логическим элементам отнесены КМОП-вентили, мультиплексоры и КМОП-инверторы с тремя состояниями. Под остатком понимаются логические элементы, не вошедшие ни в одну логическую сеть. Время выполнения программы поиска логических сетей существенно зависит не только от сложности декомпилируемой схемы (измеряемой числом транзисторов), но от числа псевдоэлементов (которые могут увеличивать число порождаемых логических сетей) и числа передаточных элементов.

Таблица 3
Результаты выделения логических сетей

Table 3
Results of extracting logical networks

	Схемы *Circuits*					
	1	2	3	4	5	6
Число: транзисторов; логических элементов; псевдоэлементов; передаточных элементов; выделенных логических сетей	 243 806 50 524 1 6902 1	 11 935 2777 119 11 1	 39 424 4737 1178 766 13	 7884 378 341 – 4	 11 436 952 284 89 8	 247 41 4 – 4
Остаток логических элементов	–	34	228	36	3	3
Время выполнения, с	20,3460	0,0560	0,2410	0,0030	0,0140	0,0010

Заключение. Предложенные методы решения задачи извлечения логических сетей из двухуровневых SPICE-описаний реализованы как часть программы декомпиляции КМОП-схем на уровне транзисторов. Программа декомпиляции протестирована на практических схемах транзисторного уровня и имеет достаточное быстродействие, чтобы обрабатывать схемы более чем с 100 000 транзисторов за несколько минут на ПЭВМ. Декомпилированные схемы прошли проверку на соответствие исходной топологии транзисторной схемы с помощью разработанных средств верификации [18], а также с помощью Mentor Graphics Calibre nmLVS. Во всех случаях декомпилированные схемы успешно проходили проверку LVS топологии СБИС.

Вклад авторов. *Л. Д. Черемисинова* разработала метод решения задачи извлечения логических сетей из двухуровневых SPICE-описаний и подготовила текст статьи. *Д. И. Черемисинов* разработал алгоритмы и программные средства построения логических сетей и генерации их описаний на языках высокого уровня.

Список использованных источников

1. Baker, R. J. CMOS Circuit Design, Layout, and Simulation / R. J. Baker. – Third ed. – Wiley-IEEE Press, 2010. – 1214 p.

2. Abadir, M. S. An improved layout verification algorithm (LAVA) / M. S. Abadir, J. Ferguson // Proc. of the European Design Automation Conf., Glasgow, UK, 12–15 Mar. 1990. – Glasgow, 1990. – P. 391–395.

3. Hunt, V. D. Reengineering: Leveraging the Power of Integrated Product Development / V. D. Hunt. – Wiley, 1993. – 283 p.

4. Rostami, M. A primer on hardware security: Models, methods, and metrics / M. Rostami, F. Koushanfar, R. Karri // Proceedings of the IEEE. – 2014. – Vol. 102, no. 8. – P. 1283–1295.

5. Tehranipoor, M. A survey of hardware trojan taxonomy and detection / M. Tehranipoor, F. Koushanfar // IEEE Design & Test of Computers. – 2010. – Vol. 27, no. 1. – P. 10–25.

6. Белоус, А. И. Основы кибербезопасности. Стандарты, концепции, методы и средства обеспечения / А. И. Белоус, В. А. Солодуха. – М. : Техносфера, 2021. – 482 с.

7. Черемисинов, Д. И. Извлечение сети логических элементов из КМОП-схемы транзисторного уровня / Д. И. Черемисинов, Л. Д. Черемисинова // Микроэлектроника. – 2019. – Т. 48, № 3. – С. 224–234. https://doi.org/10.1134/S0544126919030037

8. Yang, L. FROSTY: A program for fast extraction of high-level structural representation from circuit description for industrial CMOS circuits / L. Yang, C.-J. R. Shi // Integration the VLSI J. – 2006. – Vol. 39, no 4. – P. 311–339.

9. Zhang, N. The subcircuit extraction problem / N. Zhang, D. C. Wunsch, F. Harary // Proc. IEEE Intern. Behavioral Modeling and Simulation Workshop. – 2005. – Vol. 33(3). – P. 22–25.

10. Han, M. Efficient Subgraph Matching: Harmonizing Dynamic Programming, Adaptive Matching Order, and Failing Set Together / M. Han, H. Kim, G. Gu // Proc. of Intern. Conf. on Management of Data (SIGMOD '19), Amsterdam, Netherlands, 30 June – 5 July 2019. – Amsterdam, 2019. – P. 1429–1446.

11. Черемисинов, Д. И. Канонизация графов при декомпиляции транзисторных схем / Д. И. Черемисинов, Л. Д. Черемисинова // Информатика. – 2022. − Т. 19, № 3. – С. 25–39. https://doi.org/10.37661/1816-0301-2022-19-3-25-39

12. Черемисинов, Д. И. Распознавание логических вентилей в плоской транзисторной схеме / Д. И. Черемисинов, Л. Д. Черемисинова // Информатика. – 2021. – Т. 18, № 4. – С. 96–107. https://doi.org/10.37661/1816-0301-2021-18-4-96-107.

13. Бибило, П. Н. Логическое проектирование дискретных устройств с использованием продукционно-фреймовой модели представления знаний / П. Н. Бибило, В. И. Романов. – Минск : Беларус. навука, 2011. – 279 с.

14. Система логического проектирования функциональных блоков заказных КМОП СБИС с пониженным энергопотреблением / П. Н. Бибило [и др.] // Микроэлектроника. – 2018. – Т. 47, № 1. – С. 72–88.

15. Рабаи, Ж. М. Цифровые интегральные схемы. Методология проектирования : пер с англ. / Ж. М. Рабаи, А. Чандраксан, Б. Николич. – Изд. 2-е. – М. : Вильямс, 2007. – 912 с.

16. Черемисинов, Д. И. Распознавание логических структур на основе проходных транзисторов в КМОП-схеме / Д. И. Черемисинов, Л. Д. Черемисинова // Информационные технологии и системы 2023 (ИТС 2023) = Information Technologies and Systems 2023 (ITS 2023) : материалы Междунар. науч. конф., Минск, Беларусь, 22 нояб. 2023 г. – Минск : БГУИР, 2023. – С. 113–114.

17. Черемисинов, Д. И. Анализ и преобразование структурных описаний СБИС / Д. И. Черемисинов. – Минск : Беларуская навука, 2006. – 275 с.

18. Черемисинов, Д. И. Верификация логических схем из КМОП-транзисторов / Д. И. Черемисинов, Л. Д. Черемисинова // Новые информационные технологии в исследовании сложных структур : материалы 13-й Междунар. конф., 7–9 сент. 2020 г. – Томск : Изд. дом Томского гос. ун-та, 2020. – С. 150–151.

References

1. Baker R. J. *CMOS Circuit Design, Layout, and Simulation*. Third edition. Wiley-IEEE Press, 2010, 1214 p.

2. Abadir M. S., Ferguson J. An improved layout verification algorithm (LAVA). *Proceedings of the European Design Automation Conference, Glasgow, UK, 12–15 March 1990*. Glasgow, 1990, pp. 391–395.

3. Hunt V. D. *Reengineering: Leveraging the Power of Integrated Product Development*. Wiley, 1993, 283 p.

4. Rostami M., Koushanfar F., Karri R. A primer on hardware security: Models, methods, and metrics. *Proceedings of the IEEE*, 2014, vol. 102, no. 8, pp. 1283–1295.

5. Tehranipoor M., Koushanfar F. A survey of hardware trojan taxonomy and detection. *IEEE Design & Test of Computers*, 2010, vol. 27, no. 1, pp. 10–25.

6. Belous A. I., Solodukha V. A. Osnovy kiberbezopasnosti. Standarty, kontseptsii, metody i sredstva obespecheniya. *Fundamentals of Cybersecurity. Standards, Concepts, Methods and Means of Support*. Moscow, Tekhnosfera, 2021, 482 p. (In Russ.).

7. Cheremisinov D. I., Cheremisinova L. D. *Extracting a logic gate network from a transistor-level CMOS circuit*. Mikrojelektronika [*Russian Microelectronics*], 2019, vol. 48, no. 3, pp. 224–234. https://doi.org/10.1134/S0544126919030037 (In Russ.).

8. Yang L., Shi C.-J. R. FROSTY: A program for fast extraction of high-level structural representation from circuit description for industrial CMOS circuits. *Integration the VLSI Journal*, 2006, vol. 39, no 4, pp. 311–339.

9. Zhang N., Wunsch D. C., Harary F. The subcircuit extraction problem. *Proceedings IEEE International Behavioral Modeling and Simulation Workshop*, 2005, vol. 33(3), pp. 22–25.

10. Han M., Kim H., Gu G. Efficient Subgraph Matching: Harmonizing Dynamic Programming, Adaptive Matching Order, and Failing Set Together. *Proceedings of International conference on Management of Data (SIGMOD '19), Amsterdam, Netherlands, 30 June – 5 July 2019.* Amsterdam, 2019, pp. 1429–1446.

11. Cheremisinov D. I., Cheremisinova L. D. *Canonization of graphs during transistor circuits decompilation*. Informatika [*Informatics*], 2022, vol. 19, no. 3, pp. 25−39 (In Russ.). https://doi.org/10.37661/1816-0301-2022-19-3-25-39 (In Russ.).

12. Cheremisinov D. I., Cheremisinova L. D. *Logical gates recognition in a flat transistor circuit.* Informatika [*Informatics*], 2021, vol. 18, no. 4, pp. 96−107. https://doi.org/10.37661/1816-0301-2021-18-4-96-107 (In Russ.).

13. Bibilo P. N., Romanov V. I. Logicheskoye proyektirovaniye diskretnykh ustroystv s ispol'zovaniyem produktsionno-freymovoy modeli predstavleniya znaniy. *Logical Design of Discrete Devices Using a Production-Frame Model of Knowledge Representation*. Minsk, Belaruskaja navuka, 2011, 279 p. (In Russ.).

14. Bibilo P. N., Avdeyev N. A., Kardash S. N., Kiriyenko N. A., Lankevich Yu. Yu., …, Cheremisinova L. D. *A System for Logical Design of Custom CMOS VLSI Functional Blocks with Reduced Power Consumption*. Mikrojelektronika [*Russian Microelectronics*], 2018, vol. 47, no. 1, pp. 72–88 (In Russ.).

15. Rabaev J. M., Chandrakasan A., Nikolic B. *Digital Integrated Circuits*, 2nd edition. Pearson, 2002, 800 p.

16. Cheremisinov D. I., Cheremisinova L. D. *Recognition of logical structures from pass transistors in a CMOS circuit.* Informacionnye tehnologii i sistemy 2023 (ITS 2023) : materialy Mezhdunarodnoj nauchnoj konferencii, Minsk, Belarus', 22 nojabrja 2023 g. [*Information Technologies and Systems 2023 (ITS 2023) : Proceedings of the International Scientific Conference, Minsk, Belarus, 22 November 2023*]. Minsk, Belorusskij gosudarstvennyj universitet informatiki i radiojelektroniki, 2023, pp. 113–114 (In Russ.).

17. Cheremisinov D. I. Analiz i preobrazovaniye strukturnykh opisaniy SBIS. *Analysis and Transformation of VLSI Structural Descriptions.* Minsk, Belaruskaja navuka, 2006, 275 p. (In Russ.).

18. Cheremisinov D. I., Cheremisinova L. D. *Verification of logic circuits from CMOS transistors*. Novyye informatsionnyye tekhnologii v issledovanii slozhnykh struktur : materialy 13-j Mezhdunarodnoj konferencii, 7–9 sentyabrya 2020 g. [*New Information Technologies in the Study of Complex Structures : Proceedings of the 13th International Conference, 7–9 September 2020*]. Tomsk, Izdatel'skij dom Tomskogo gosudarstvennogo universiteta, 2020, pp. 150–151 (In Russ.).

Информация об авторах

Черемисинов Дмитрий Иванович, кандидат технических наук, доцент, ведущий научный сотрудник, Объединенный институт проблем информатики Национальной академии наук Беларуси.
E-mail: cher@newman.bas-net.by

Черемисинова Людмила Дмитриевна, доктор технических наук, профессор, главный научный сотрудник, Объединенный институт проблем информатики Национальной академии наук Беларуси.
E-mail: cld@newman.bas-net.by

Information about the authors

Dmitry I. Cheremisinov, Ph. D. (Eng.), Assoc. Prof., Leading Researcher, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus.
E-mail: cher@newman.bas-net.by

Ljudmila D. Cheremisinova, D. Sc. (Eng.), Prof., Chief Researcher, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus.
E-mail: cld@newman.bas-net.by