

Программная инженерия



Пр **2**
ИН **2021**
Том 12

31 МАЯ – 4 ИЮНЯ 2021 Г., ДГТУ, РОСТОВ-НА-ДОНУ

МЕЖДУНАРОДНАЯ НАУЧНАЯ МУЛЬТИКОНФЕРЕНЦИЯ

Кибер-физические системы: проектирование и моделирование
Cyber-physical systems design and modelling (CyberPhy-2021)

СЕКЦИИ

1. Cyber-Physical Systems: digital technologies and applications (Кибер-физические системы: цифровые технологии и приложения)
2. Cyber-physical systems: design and application for Industry 4.0 (Кибер-физические системы: проектирование и применение для Индустрии 4.0)
3. Cyber-Physical Systems: Modelling and Intelligent Control (Кибер-физические системы: моделирование и интеллектуальное управление)
4. Society 5.0: Cyberspace for advanced human-centered society (Общество 5.0: киберпространство для развитого общества, ориентированного на человека)

XXXIV Международная научная конференция МАТЕМАТИЧЕСКИЕ МЕТОДЫ В ТЕХНИКЕ И ТЕХНОЛОГИЯХ - ММТТ-34

СЕКЦИИ

1. Качественные и численные методы исследования дифференциальных и интегральных уравнений
2. Оптимизация, автоматизация и оптимальное управление технологическими процессами
3. Математическое моделирование технологических и социальных процессов
4. Математическое моделирование и оптимизация в задачах САПР, аддитивных технологий, цифрового производства
5. Математические методы в задачах радиотехники, радиоэлектроники и телекоммуникаций, геоинформатики, авионики и космонавтики
6. Математические методы и интеллектуальные системы в робототехнике и мехатронике
7. Математические методы в медицине, биотехнологии и экологии
8. Математические методы в экономике и гуманитарных науках
9. Информационные и интеллектуальные технологии в технике и образовании
10. Математические и инструментальные методы технологий Индустрии 4.0
11. Обсуждение квалификационных работ

Подробная информация о конференции и условиях участия в ней размещается на сайте
<http://mmtt.sstu.ru/>



Программная инженерия

Том 12
№ 2
2021
Пр
ИН

Учредитель: Издательство "НОВЫЕ ТЕХНОЛОГИИ"

Издается с сентября 2010 г.

DOI 10.17587/issn.2220-3397

ISSN 2220-3397

Редакционный совет

Садовничий В.А., акад. РАН
(председатель)
Бетелин В.Б., акад. РАН
Васильев В.Н., чл.-корр. РАН
Жижченко А.Б., акад. РАН
Макаров В.Л., акад. РАН
Панченко В.Я., акад. РАН
Стемпковский А.Л., акад. РАН
Ухлинов Л.М., д.т.н.
Федоров И.Б., акад. РАН
Четверушкин Б.Н., акад. РАН

Главный редактор

Васенин В.А., д.ф.-м.н., проф.

Редколлегия

Антонов Б.И.
Афонин С.А., к.ф.-м.н.
Бурдонов И.Б., д.ф.-м.н., проф.
Борзовс Ю., проф. (Латвия)
Гаврилов А.В., к.т.н.
Галатенко А.В., к.ф.-м.н.
Корнеев В.В., д.т.н., проф.
Костюхин К.А., к.ф.-м.н.
Махортов С.Д., д.ф.-м.н., доц.
Манцивода А.В., д.ф.-м.н., доц.
Назирова Р.Р., д.т.н., проф.
Нечаев В.В., д.т.н., проф.
Новиков Б.А., д.ф.-м.н., проф.
Павлов В.Л. (США)
Пальчунов Д.Е., д.ф.-м.н., доц.
Петренко А.К., д.ф.-м.н., проф.
Позднеев Б.М., д.т.н., проф.
Позин Б.А., д.т.н., проф.
Серебряков В.А., д.ф.-м.н., проф.
Сорокин А.В., к.т.н., доц.
Терехов А.Н., д.ф.-м.н., проф.
Филимонов Н.Б., д.т.н., проф.
Шапченко К.А., к.ф.-м.н.
Шундеев А.С., к.ф.-м.н.
Щур Л.Н., д.ф.-м.н., проф.
Язов Ю.К., д.т.н., проф.
Якобсон И., проф. (Швейцария)

Редакция

Лысенко А.В., Чугунова А.В.

Журнал издается при поддержке Отделения математических наук РАН, Отделения нанотехнологий и информационных технологий РАН, МГУ имени М.В. Ломоносова, МГТУ имени Н.Э. Баумана

СОДЕРЖАНИЕ

- Авдеев Н. А., Бибило П. Н.** Автоматизированное проектирование цифровых операционных устройств с пониженным энергопотреблением 63
- Орлов Д. А., Карелова Р. А.** Архитектура связанных JavaScript-микрокомпонентов для крупных веб-проектов 74
- Шампаров В. Е., Маркин А. Л.** Механизм оптимизации Structure Splitting в составе компилятора для микропроцессоров Эльбрус 82
- Луцник В. Г., Макарова М. С., Решмин А. И.** Применение трехпараметрической дифференциальной модели турбулентности для решения задач течения и теплообмена в каналах переменного сечения. Часть 2 89
- Харлампов И. Е., Ощепков А. Ю.** Преобразование и отображение расчетных данных оптико-электронных данных дистанционного зондирования Земли на основе расширений GeoServer и распределенных технологий хранения 107

Журнал зарегистрирован
в Федеральной службе
по надзору в сфере связи,
информационных технологий
и массовых коммуникаций.

Свидетельство о регистрации

ПИ № ФС77-38590 от 24 декабря 2009 г.

Журнал распространяется по подписке, которую можно оформить в любом почтовом отделении (индекс по Объединенному каталогу "Пресса России" — 22765) или непосредственно в редакции.

Тел.: (499) 269-53-97. Факс: (499) 269-55-10.

Http://novtex.ru/prin/rus E-mail: prin@novtex.ru

Журнал включен в систему Российского индекса научного цитирования и базу данных RSCI на платформе Web of Science.

Журнал входит в Перечень научных журналов, в которых по рекомендации ВАК РФ должны быть опубликованы научные результаты диссертаций на соискание ученой степени доктора и кандидата наук.

© Издательство "Новые технологии", "Программная инженерия", 2021

SOFTWARE ENGINEERING

PROGRAMMNAYA INGENERIA

Vol. 12

N 2

2021

Published since September 2010

DOI 10.17587/issn.2220-3397

ISSN 2220-3397

Editorial Council:

SADOVNICHY V. A., Dr. Sci. (Phys.-Math.),
Acad. RAS (*Head*)
BETELIN V. B., Dr. Sci. (Phys.-Math.), Acad. RAS
VASIL'EV V. N., Dr. Sci. (Tech.), Cor.-Mem. RAS
ZHIZHCENKO A. B., Dr. Sci. (Phys.-Math.),
Acad. RAS
MAKAROV V. L., Dr. Sci. (Phys.-Math.), Acad.
RAS
PANCHENKO V. YA., Dr. Sci. (Phys.-Math.),
Acad. RAS
STEMPKOVSKY A. L., Dr. Sci. (Tech.), Acad. RAS
UKHLINOV L. M., Dr. Sci. (Tech.)
FEDOROV I. B., Dr. Sci. (Tech.), Acad. RAS
CHETVERTUSHKIN B. N., Dr. Sci. (Phys.-Math.),
Acad. RAS

Editor-in-Chief:

VASENIN V. A., Dr. Sci. (Phys.-Math.)

Editorial Board:

ANTONOV B.I.
AFONIN S.A., Cand. Sci. (Phys.-Math)
BURDONOV I.B., Dr. Sci. (Phys.-Math)
BORZOV JURIS, Dr. Sci. (Comp. Sci), Latvia
GALATENKO A.V., Cand. Sci. (Phys.-Math)
GAVRILOV A.V., Cand. Sci. (Tech)
JACOBSON IVAR, Dr. Sci. (Philos., Comp. Sci.),
Switzerland
KORNEEV V.V., Dr. Sci. (Tech)
KOSTYUKHIN K.A., Cand. Sci. (Phys.-Math)
MAKHORTOV S.D., Dr. Sci. (Phys.-Math)
MANCIVODA A.V., Dr. Sci. (Phys.-Math)
NAZIROV R.R., Dr. Sci. (Tech)
NECHAEV V.V., Cand. Sci. (Tech)
NOVIKOV B.A., Dr. Sci. (Phys.-Math)
PAVLOV V.L., USA
PAL'CHUNOV D.E., Dr. Sci. (Phys.-Math)
PETRENKO A.K., Dr. Sci. (Phys.-Math)
POZDNEEV B.M., Dr. Sci. (Tech)
POZIN B.A., Dr. Sci. (Tech)
SEREBRJAKOV V.A., Dr. Sci. (Phys.-Math)
SOROKIN A.V., Cand. Sci. (Tech)
TEREKHOV A.N., Dr. Sci. (Phys.-Math)
FILIMONOV N.B., Dr. Sci. (Tech)
SHAPCHENKO K.A., Cand. Sci. (Phys.-Math)
SHUNDEEV A.S., Cand. Sci. (Phys.-Math)
SHCHUR L.N., Dr. Sci. (Phys.-Math)
YAZOV Yu. K., Dr. Sci. (Tech)

Editors: LYSENKO A.V., CHUGUNOVA A.V.

CONTENTS

- Avdeev N. A., Bibilo P. N.** Design of Digital Operational Units with Low Power Consumption 63
- Orlov D. A., Karelova R. A.** Connected JavaScript-Microcomponents Architecture for Large Web-Projects 74
- Shamparov V. E., Markin A. L.** Structure Splitting for Elbrus Processor Compiler 82
- Lushchik V. G., Makarova M. S., Reshmin A. I.** Application of the Three-Parameter Differential Model of Turbulence for Solving Problems of Flow and Heat Transfer in Channels of Variable Cross-Section. Part 2 89
- Kharlampenkov I. E., Oshchepkov A. U.** Conversion and Display of the Calculated Data of Spectral Remote Sensing Data on the Basis of GeoServer Extensions and Distributed Storage Technologies . . . 107

Н. А. Авдеев, канд. техн. наук., ст. науч. сотр., avdeev_n@newman.bas-net.by,
П. Н. Бибило, д-р техн. наук, проф., зав. лаб., bibilo@newman.bas-net.by,
Объединенный институт проблем информатики Национальной академии наук Беларуси,
г. Минск

Автоматизированное проектирование цифровых операционных устройств с пониженным энергопотреблением

Предложены способы описания на языке VHDL синтезируемых моделей цифровых операционных устройств, называемых также конечными автоматами с трактом данных. После выполнения синтеза логических схем в библиотеке проектирования заказных цифровых КМОП СБИС и моделирования результаты сравниваются по площади и энергопотреблению полученных схем. Существенного снижения энергопотребления логических схем таких устройств можно добиться, используя соответствующие описания их функционирования на алгоритмическом уровне. Общий подход к сокращению энергопотребления состоит в том, чтобы уменьшить число переключений сигналов в логической схеме и тем самым уменьшить потребляемую динамическую мощность.

Ключевые слова: цифровое устройство, конечный автомат с трактом данных, синтез логической схемы, моделирование, VHDL, СБИС, КМОП-технология

Введение

Снижение энергопотребления цифровых систем, реализуемых на элементной базе заказных комбинированных металл-оксид-полупроводниковых схем (КМОП-схем) и систем-на-кристалле, является одной из важнейших проблем, стоящих в настоящее время перед разработчиками как интегральных схем, так и систем автоматизированного проектирования (САПР) [1, 2]. На практике проблема снижения энергопотребления сверхбольших интегральных схем (СБИС) решается практически на всех этапах проектирования — от алгоритмического до топологического, на котором принимаются во внимание длины соединений, габаритные размеры (формы) и размеры логических и схемотехнических элементов. Для снижения потребляемой мощности выбирают соответствующую синхронизацию, создают специальные библиотеки логических элементов, предлагают новые конструктивные и схемотехнические решения [3], например, доменные системы питания, для описания которых используют [4] специальный формат UPF (*Unified Power Format*).

Исходные описания функционирования цифровых СБИС осуществляются на языках VHDL (*Very high speed integrated circuits hardware description language*) [5] и Verilog [6]. По HDL-описаниям проектов устройств (HDL — *Hardware description language*) системы синтеза (синтезаторы) автоматически строят синхронные логические схемы в том или ином базисе логических элементов, называемом технологи-

ческим (целевым) базисом либо целевой библиотекой логических элементов, в качестве которой рассматривается библиотека КМОП-элементов. При этом современные синтезаторы осуществляют синтез логических схем заменой каждой VHDL-конструкции соответствующей логической подсхемой (далее будут использоваться описания на языке VHDL). Результаты синтеза — логические схемы из библиотечных КМОП-элементов, их площадь, быстродействие и энергопотребление в значительной мере зависят от вида исходного VHDL-описания, подаваемого на вход синтезатора [5].

Операционные цифровые устройства давно нашли широкое применение в практике проектирования [7] и развиваются в настоящее время [8, 9] с использованием для их описания HDL-модели. В зарубежной литературе [10] операционные устройства называют конечными автоматами с каналом данных (*Finite state machine (FSM) with datapath (FSM-D)*). Канал данных называют также "трактом данных" [11].

Одним из эффективных подходов к снижению динамического энергопотребления цифровых устройств является создание такого алгоритмического описания VHDL-проекта, в котором предусматривается отключение тех блоков, функционирование которых не требуется в одном либо нескольких (многих) тактах функционирования синхронной схемы. В данной работе исследованы именно такие способы алгоритмического описания операционных устройств, состоящих из управляющего автомата и операционного блока (тракта данных). Проведено сравнение эффек-

тивности предложенных способов с традиционным способом VHDL-описания поведения цифрового устройства, который не учитывает аспект энергопотребления и ориентирован только на правильную функциональность проектируемой логической схемы, и предложена методика алгоритмического описания операционных устройств. Для оценки энергопотребления логических схем использован подход, предложенный в работе [12] и позволяющий использовать быстродействующее логическое VHDL-моделирование структурных описаний (*netlists*) логических схем вместо трудоемкого схемотехнического моделирования Spice-описаний схем.

1. Постановка проблемы

Пусть в качестве исходных спецификаций на проектирование операционного устройства заданы: 1) таблица (либо граф) переходов управляющего автомата; 2) состав и разрядность логических и арифметических операций, которые должны быть представлены синтезируемыми конструкциями языка VHDL и выполняться в операционном блоке; 3) целевая библиотека КМОП-элементов. Требуется составить синтезируемое VHDL-описание, по которому с помощью синтезатора *LeonardoSpectrum* [13] будет получена логическая схема из элементов заданной библиотеки, характеризующаясь возможно меньшим энергопотреблением и относительно небольшой сложностью.

Выбор синтезатора *LeonardoSpectrum* обосновывается тем, что он может быть настроен на пользовательскую библиотеку логических КМОП-элементов и имеет разнообразные опции управления синтезом. Вместо *LeonardoSpectrum* может быть использован и другой синтезатор логических схем, который имеет такие же возможности. Что касается целевой библиотеки, то предлагаемые далее приемы алгоритмического VHDL-описания цифровых устройств, ориентированные на снижение энергопотребления, экспериментально проверены на библиотеке логических КМОП-элементов, выполненных по тем нормам проектирования, для которых основную долю энергопотребления (рассеиваемой мощности) составляет динамическая мощность. Естественно, предлагаемые способы алгоритмического описания могут быть представлены и на языке Verilog.

2. Пример операционного устройства

Рассматривается класс цифровых операционных устройств [7–9]. Цифровое операционное устройство, пример которого приведен на рис. 1, состоит из композиции управляющего (*FSM*) и операционного (*ALU*) блоков, на входе схемы устройства имеется регистр (*REG*) синхронизируемых триггеров (элементов памяти). Штриховые связи на рис. 1 показывают, что не для всех далее рассматриваемых способов алгоритмического описания операционных устройств данные связи будут нужны.

В рассматриваемом примере цифрового устройства управляющий автомат *FSM* задан графом *G*

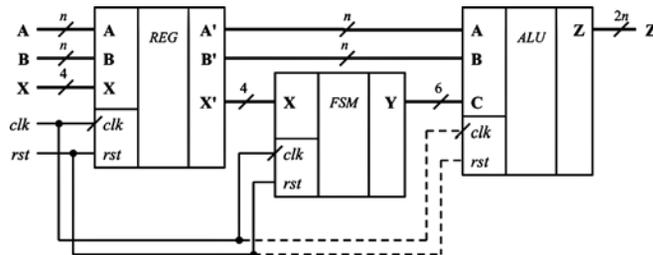


Рис. 1. Операционное устройство — композиция управляющего автомата *FSM* и операционного блока *ALU*

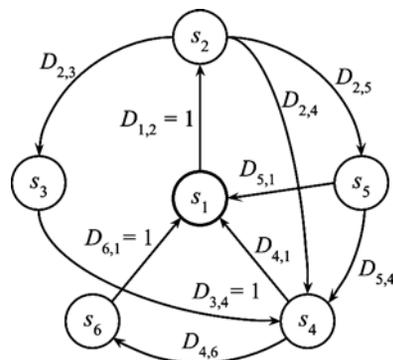


Рис. 2. Граф *G* переходов управляющего автомата *FSM*

переходов (рис. 2), функции переходов даны в табл. 1. На рис. 2 не показаны асинхронные переходы из любого состояния s_i в начальное состояние s_1 при единичном значении сигнала сброса *rst*. Двоичные входные векторы (порты) **A**, **B** назовем операндами операционного блока *ALU*. В рассматриваемом примере число n разрядов каждого из операндов **A**, **B** равно 16 либо 32. Функции операционного блока *ALU* заданы в табл. 2.

Признаки шести операций y_i поступают на входной порт **C** блока *ALU*. В зависимости от признака

Таблица 1

Таблица переходов управляющего автомата *FSM*

s_i	s_j	Условия перехода	Признак операции y_i
s_1	s_2	$D_{1,2} = 1$	y_2
s_2	s_3	$D_{2,3} = x_1 \bar{x}_2 \bar{x}_3 \vee x_1 x_2$	y_3
	s_4	$D_{2,4} = x_1 \bar{x}_2 x_3$	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	$D_{5,1} = \bar{x}_1 x_4$	y_1
	s_4	$D_{5,4} = \bar{x}_4 \vee x_1 x_4$	y_4
s_6	s_1	$D_{6,1} = 1$	y_1

Операции ALU

y_i	VHDL-операция	Тип операции
y_1	$Z = A \text{ and } B$	Логическая
y_2	$Z = A \text{ or } B$	
y_3	$Z = A \text{ xor } B$	
y_4	$Z = A \text{ xnor } B$	
y_5	$Z = A + B$	Арифметическая
y_6	$Z = A * B$	

операции (единичного значения сигнала y_i) функция операционного блока может быть либо арифметической (сложение или умножение операндов **A**, **B**, понимаемых как двоичные коды чисел без знака), либо логической, в этом случае логическая операция выполняется над соответствующими разрядами двоичных векторов **A**, **B**.

Функционирование 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 операции, и для операндов **A**, **B** блок *ALU* выполняет только одну из шести операций y_i . Управляющий автомат начинает функционирование из начального состояния s_1 , меняет свои состояния и всегда возвращается в начальное состояние. В процессе функционирования цепочки состояний управляющего автомата образуют на графе G различные циклы. В данном примере каждый переход из одного состояния в другое вызывает смену выполнения соответствующей операции в операционном блоке. Ниже приведен фрагмент VHDL-кода, описывающий условия переходов автомата *FSM* из состояния s_2 в другие соседние состояния:

```

when s2 =>
  if ((x1 and not x2 and not x3)
  or (x1 and x2))= '1' then
    NEXT_state <= s3; y <= "001000";
  elsif ((x1 and not x2 and x3)= '1') then
    NEXT_state <= s4; y <= "000100";
  elsif (not x1 = '1') then
    NEXT_state <= s5; y <= "000010";
  else
    NEXT_state <= s2; y <= "000000";
  end if;

```

На языке VHDL условие $D_{2,3} = x_1\bar{x}_4\bar{x}_3 \vee x_1x_2$ перехода из состояния s_2 в состояние s_3 записывается в виде

```
((x1 and not x2 and not x3) or (x1 and x2)) = '1',
```

где *and* — логическое И; *or* — логическое ИЛИ; *not* — отрицание (НЕ).

3. VHDL-модели операционных устройств

Ниже рассмотрены шесть способов создания VHDL-моделей операционных устройств, пять из которых (способы 2—6) ориентированы на снижение энергопотребления логической схемы, реализующей соответствующую VHDL-модель.

Способ 1 — традиционный. Алгоритмические VHDL-описания (VHDL-модели) операционных устройств (без учета энергопотребления) известны [8, 9] и заключаются в том, что каждая из операций выполняется в операционном блоке, однако на выход устройства подается лишь результат той операции, выполнение которой требуется в данном такте (рис. 3). Заметим, что основные аппаратные затраты приходится на схему, реализующую операцию арифметического умножения.

Для способов 2—6 структура *ALU* изображена на рис. 4, дополнительные блоки *BLK* для каждого из способов 2—6 имеют различную структуру и выполняют различные функции. Общий подход к сокращению энергопотребления состоит в том, чтобы уменьшить число переключений сигналов в логической схеме.

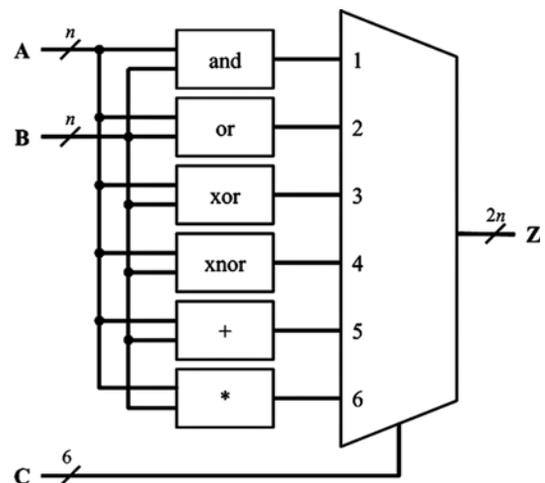
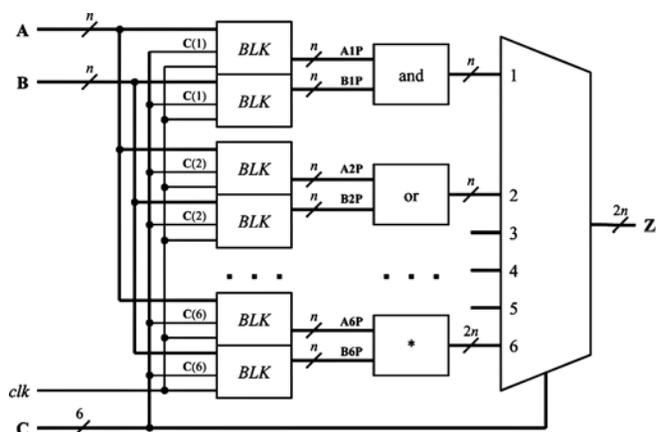


Рис. 3. Операционный блок (ALU)

Рис. 4. Структура ALU (способы 2—6); *BLK* — дополнительные блоки

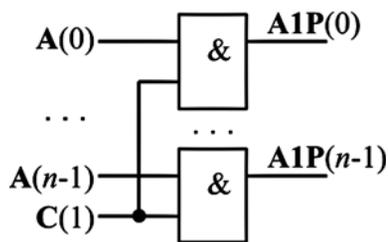


Рис. 5. Обнуление разрядов операнда А

Способ 2 — обнуление операндов. Общая структура *ALU* для второго способа приведена на рис. 4, каждый из блоков *BLK* выполняет функцию обнуления значений операндов **A**, **B** для каждой из неиспользуемых операций в текущем такте функционирования устройства (рис. 5).

Выражаясь более точно, сокращение энергопотребления в способе 2 основывается на том, что для операций, для которых $y_i = 0$, значения операндов **A**, **B** обнуляются и не меняются до тех пор, пока y_i не примет единичное значение. Таким образом, в соответствующей подсхеме, реализующей операцию y_i , не будет переключений транзисторов, пока на входах нулевые значения сигналов не изменятся. Для каждой операции используется своя пара внутренних сигналов — операндов **A**, **B**.

Ниже во фрагменте VHDL-кода **A1P**, **B1P** — векторы операндов, подаваемых на первую операцию *and* (рис. 4), **A2P**, **B2P** — на вторую операцию *or*, и т. д.

```
A1P <= A when C(1) = '1' else (others => '0');
B1P <= B when C(1) = '1' else (others => '0');
...
A6P <= A when C(6) = '1' else (others => '0');
B6P <= B when C(6) = '1' else (others => '0');
```

Разряды вектора **C(j)** в *ALU* соответствуют разрядам вектора y_j управляющего блока *FSM*. Блоки *BLK* (рис. 5) управляют передачей **A**, **B** к исполняемой операции и осуществляют обнуление операндов для неисполняемых в данном такте операций.

Блок мультиплексоров передает на выход устройства результат **Z** выполненной логической либо арифметической операции.

Способ 3 — единичные значения всех разрядов операндов для невыполняемых операций. Данный способ мало отличается от способа 2. По сути, будет проверяться следующее: что выгоднее с точки зрения минимизации энергопотребления — подавать на входы подсхем для невыполняемых операций нулевые, либо единичные значения всех разрядов операндов. Установка единичных значений операндов осуществляется следующим образом:

```
A1P <= A when C(1) = '1' else (others => '1');
B1P <= B when C(1) = '1' else (others => '1');
...
A6P <= A when C(6) = '1' else (others => '1');
B6P <= B when C(6) = '1' else (others => '1');
```

Способ 4 — сохранение значений операндов для невыполняемых операций. Общая структура *ALU* для

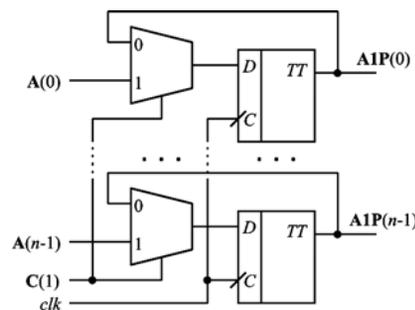


Рис. 6. Сохранение одного разряда $A(i)$ операнда А

четвертого способа VHDL-описания приведена на рис. 4, структура каждого из блоков *BLK* приведена на рис. 6. Для каждого из разрядов операнда **B** употребляются такие же схемы. В отличие от способов 2 и 3, такой подход минимизирует число переключений сигналов, так как первоначальное обнуление (установка в единичное значение) вызывает также переключения сигналов.

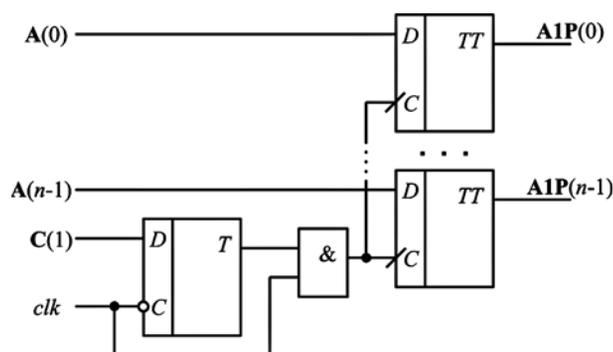
Фрагмент VHDL-кода, в котором подаются новые значения операндов для исполняемой операции (и тем самым сохраняются значения операндов для невыполняемых операций), выглядит следующим образом:

```
process(clk)
begin
    if(rising_edge(clk)) then
        if (C(1) = '1') then
            A1P <= A; B1P <= B;
        end if;
        if (C(2) = '1') then
            A2P <= A; B2P <= B;
        end if;
        if (C(3) = '1') then
            A3P <= A; B3P <= B;
        end if;
        if (C(4) = '1') then
            A4P <= A; B4P <= B;
        end if;
        if (C(5) = '1') then
            A5P <= A; B5P <= B;
        end if;
        if (C(6) = '1') then
            A6P <= A; B6P <= B;
        end if;
    end if;
end process;
```

VHDL-функция *rising_edge(clk)* определяет передний фронт синхросигнала *clk*. При схемной реализации данного VHDL-процесса в схему будет установлено 192 триггера (при $n = 16$). На рис. 6 показан триггер, сохраняющий значение разряда $A(i)$ операнда **A**.

Заметим, что данный способ требует значительного числа дополнительных регистров, в которых сохраняются значения операндов.

Способ 5 — сохранение значений операндов только для подсхемы, реализующей операцию арифмети-

Рис. 7. Схема *clock gating* для признака $C(j)$ операции *ALU*

ческого умножения (рис. 6) и обнуление для остальных операций (см. рис. 5). Это связано с тем, что нецелесообразно использовать способ 4 для простых логических операций, так как энергопотребление дополнительных регистров будет превышать энергопотребление схем, реализующих логические операции.

Способ 6 — использование приема *clock gating* [14, 15] для сохранения значений операндов и уменьшения числа переключений синхросигнала *clk*. Схема *BLK* для каждой из шести операций *ALU* приведена на рис. 7.

4. Экспериментальные исследования

В этом разделе описана организация экспериментов и приведены их результаты. Целью экспериментов являлось сравнение эффективности предложенных способов алгоритмического описания устройств по критериям энергопотребления и площади кристаллов СБИС. Логические схемы синтезировались в одной и той же библиотеке логических элементов. Эксперименты проводили для значений $n = 16$ и $n = 32$ — числа разрядов операндов **A**, **B**. В каждом из экспериментов синтез схем осуществлялся в синтезаторе *LeonardoSpectrum* с определенными опциями управления синтезом и построением дерева *clock tree* для синхросигнала, подаваемого на большое число регистров. В качестве таких опций (режимов синтеза) были выбраны опция *flatten* — синтез с устранением иерархии описания проекта и опция *preserve* — синтез с учетом иерархии. Так как алгоритмические описания синтезатор *LeonardoSpectrum* превращает в логические схемы с определенной структурой, то для реализации каждой операции в виде независимой подсхемы ее алгоритмическое описание должно быть представлено в виде отдельного модуля проекта (*entity*) и синтез должен быть выполнен с опцией *preserve* [13].

Библиотекой синтеза во всех экспериментах являлась библиотека проектирования заказных цифровых КМОП СБИС, состав библиотеки приведен в табл. 3.

Для элементов этой библиотеки основную долю потребляемой мощности составляет мощность переключений сигналов схемы (динамическая мощность). Тестовыми последовательностями при моделировании для проверки правильности функционирования

Элемент	Функция	Число транзисторов	Состав сложного элемента
GND	$y = 0$	1	—
VCC	$y = 1$	1	—
N	$y = \bar{A}$, инвертор	2	—
NX2	$y = \bar{A}$, 2-кратный инвертор	4	—
NX4	$y = \bar{A}$, 4-кратный инвертор	8	—
NA	$y = \overline{AB}$	4	—
NO	$y = \overline{A \vee B}$	4	—
NAO	$y = \overline{(A \vee B)C}$	6	—
NOA	$y = \overline{(AB) \vee C}$	6	—
NA3O	$y = \overline{(A \vee B)CD}$	8	—
NO3A	$y = \overline{(AB) \vee C \vee D}$	8	—
NA3	$y = \overline{ABC}$	6	—
NO3	$y = \overline{A \vee B \vee C}$	6	—
LAT	D-триггер (по уровню)	20	$4N + 2NOA$
DFP	D-триггер (по переднему фронту)	32	$4N + 4NOA$
DFFRS	D-триггер (по переднему фронту) с установкой и сбросом	44	$6N + 2NA3O + 2NO3A$
IBUF	$y = A$, входной буфер	4	$2N$
OBUF	$y = A$, выходной буфер	10	$N + NX4$

исходных описаний схем являлись псевдослучайные наборы, задающие операнды **A**, **B** и входные сигналы x_1, x_2, x_3, x_4 управляющего автомата. Для значения $n = 16$ во всех экспериментах по оценке энергопотребления (моделировании) была использована одна и та же последовательность из 16 000 псевдослучайных тестовых входных наборов (далее число тестовых наборов будем обозначать через L). Для $n = 32$ были использованы две последовательности $L = 2000$ и $L = 200$ наборов. Оценку энергопотребления (потребляемого тока) проводили с помощью средств логического моделирования, имеющихся в системе CMOSLD [16]. Используемая в данной системе методика оценки энергопотребления основана на характеристике библиотечных логических элементов по потребляемому току с помощью

Результаты эксперимента 1 ($n = 16$, операции ALU различные, $L = 16\ 000$ тестовых векторов)

Алгоритмическое описание	Число триггеров	Площадь триггеров	Режим синтеза: <i>flatten, Gray</i>			Режим синтеза: <i>preserve, Gray</i>		
			Средний ток, мА	Задержка, нс	Общая площадь схемы	Средний ток, мА	Задержка, нс	Общая площадь схемы
Способ 1	39	1716	6,296	23,00	14 943	6,512	23,01	14 728
Способ 2	39	1716	2,139	33,29	17 301	2,042	32,11	16 912
Способ 3	39	1716	3,356	35,55	18 181	3,733	33,59	17 954
Способ 4	237	8052	4,395	21,87	24 115	3,322	21,87	24 198
Способ 5	71	2740	2,718	34,66	17 389	2,498	32,97	17 606
Способ 6	243	8172	2,179	21,90	22 069	2,160	21,90	22 068

Spice-моделирования и составлении VHDL-моделей элементов, позволяющих подсчитывать значения потребляемых токов при логическом моделировании с учетом временных задержек распространения сигналов, полученных при синтезе схемы в *LeonardoSpectrum*. Временные задержки, в свою очередь, учитывают нагрузки элементов схемы и различия в энергопотреблении при переходах значений входных и выходных сигналов элементов из нулевых состояний в единичные, и наоборот, — из единичных в нулевые. Информация о задержках элементов схемы записывается при синтезе в специальный текстовый SDF-файл (*Standard delay format*), в котором для каждого элемента указываются задержки сигналов от входных полюсов к выходному полюсу.

Площадь схем оценивалась в суммарном числе транзисторов в логических элементах, составляющих схему. Энергопотребление схемы оценивалось по среднему потребляемому схемой току в миллиамперах (мА).

Эксперимент 1. Операции ALU различные (см. табл. 2), $n = 16$, синтез с устранением иерархии (режим *flatten*) и с учетом иерархии (режим *preserve*), длина теста $L = 16\ 000$ псевдослучайных тестовых

наборов. Результаты эксперимента 1 приведены в табл. 4. В данной таблице и других таблицах, содержащих результаты экспериментов, полужирным шрифтом выделены лучшие решения — наименьшие значения таких параметров логических схем, как средний ток, задержка, общая площадь схемы.

Эксперимент 2. Операции ALU различные (см. табл. 2), $n = 32$, синтез с устранением иерархии (режим *flatten*) и с учетом иерархии (режим *preserve*), длина теста $L = 2000$ наборов. Результаты эксперимента 2 приведены в табл. 5.

Особенность исследуемой схемы ALU состоит в том, что реализуемые операции не равнозначны по площади, а соответственно и по доле, вносимой в общее энергопотребление блоком. Поэтому был проведен следующий эксперимент, в котором были выбраны одинаковые по площади и энергопотреблению операции.

Эксперимент 3. Операции ALU одинаковые (арифметическое умножение), $n = 16$, $n = 32$, синтез с учетом иерархии (режим *preserve*), длина теста $L = 2000$ наборов для $n = 16$ и $L = 500$ для $n = 32$. Результаты эксперимента 3 приведены в табл. 6, они позволяют оценить эффективность использования

Таблица 5

Результаты эксперимента 2 ($n = 32$, операции ALU различные, $L = 2000$ тестовых векторов)

Алгоритмическое описание	Число триггеров	Площадь триггеров	Режим синтеза: <i>flatten, Gray</i>			Режим синтеза: <i>preserve, Gray</i>		
			Средний ток, мА	Задержка, нс	Общая площадь схемы	Средний ток, мА	Задержка, нс	Общая площадь схемы
Способ 1	74	3256	46,507	44,76	52 353	48,018	44,99	54 401
Способ 2	74	3256	8,683	56,24	58 593	8,454	55,91	55 861
Способ 4	458	15 484	13,680	56,73	74 355	16,544	56,59	75 445
Способ 5	138	5304	11,428	56,19	57 771	13,397	56,59	59 139
Способ 6	470	15 856	6,864	43,72	66 455	7,986	43,93	67 017

Результаты эксперимента 3 (операции ALU одинаковые – арифметическое умножение)

Алгоритмическое описание	$n = 16; L = 2000; \text{режим синтеза: } \textit{preserve}, \textit{onehot}$					$n = 32; L=200; \text{режим синтеза: } \textit{preserve}, \textit{onehot}$				
	Средний ток, мА	Число триггеров	Площадь триггеров	Задержка, нс	Общая площадь схемы	Средний ток, мА	Число триггеров	Площадь триггеров	Задержка, нс	Общая площадь схемы
Способ 1	34,395	42	1848	24,55	69 790	280,598	74	3256	46,09	277 522
Способ 2	11,079	42	1848	34,67	69 728	73,042	74	3256	55,33	278 702
Способ 4	19,321	234	7992	33,45	79 514	121,765	458	15 544	55,18	298 286
Способ 5	11,957	74	2872	34,65	71 380	77,723	138	5304	55,33	281 980
Способ 6	8,530	240	8304	22,93	75 440	52,629	470	15 586	44,69	289 896

Таблица 7

Результаты эксперимента 4 ($n = 16$, операции ALU различные, режим синтеза: *flatten*, *Gray*; $L = 16\ 000$ тестовых векторов)

Алгоритмическое описание	Число триггеров	Площадь триггеров	Арифметическое умножение в состоянии s_2 выполнялось 3818 раз			Арифметическое умножение в состоянии s_6 выполнялось 1698 раз		
			Средний ток, мА	Задержка, нс	Общая площадь схемы	Средний ток, мА	Задержка, нс	Общая площадь схемы
Способ 1	39	1716	6,357	23,00	14 969	6,512	23,01	14 728
Способ 2	39	1716	3,475	33,28	17 311	2,042	32,11	16 912
Способ 4	237	8052	4,137	21,87	24 115	3,322	21,87	24 198
Способ 5	71	2740	4,717	34,46	17 391	2,498	32,97	17 606
Способ 6	243	8172	2,948	21,90	22 069	2,160	21,90	22 068

способов сокращения энергопотребления для нескольких сложных операций, в качестве которых была выбрана операция арифметического умножения.

Эксперимент 4. Операции ALU различные (см. табл. 2), $n = 16$, синтез с устранением иерархии (режим *flatten*), длина теста $L = 16\ 000$ псевдослучайных тестовых наборов. Результаты эксперимента 4 приведены в табл. 7. Операция арифметического умножения повторяется различное число раз: в левой части таблицы арифметическое умножение выполняется (3818 раз) в состоянии s_2 , логическая дизъюнкция (ог) выполняется в состоянии s_6 . В правой части табл. 7 для сравнения повторяются данные из табл. 4, когда в эксперименте 1 операция арифметического умножения выполнялась 1698 раз в состоянии s_6 .

Эксперимент 5. Все операции ALU2 (табл. 8) являются арифметическими, $n = 32$, синтез в двух режимах — с устранением иерархии (режим *flatten*) и синтез без устранения иерархии (режим *preserve*), длина теста $L = 200$ псевдослучайных тестовых наборов. В способе 5 сохранение значений операндов проводится только для подсхемы, реализующей операцию y_6 арифметического умножения $Z = B*B$. Результаты эксперимента 5 приведены в табл. 9.

Таблица 8

Арифметические операции ALU2

y_i	VHDL-операция
y_1	$Z = A - B$
y_2	$Z = B - A$
y_3	$Z = 3*A*B$
y_4	$Z = 5*A*B$
y_5	$Z = A*A$
y_6	$Z = B*B$

5. Обсуждение результатов экспериментов

Как и следовало ожидать, традиционный способ 1 алгоритмического описания приводит к наименьшей площади схемы, однако является самым неэффективным по потребляемому току — соответствующие схемы потребляют больший ток по сравнению со схемами, в которых используются приемы сокращения энергопотребления. Увеличение разрядности операндов операционного блока в 2 раза (32 разряда

Результаты эксперимента 5 ($n = 32$, операции *ALU2* различные (см. табл. 8), $L = 200$ тестовых векторов)

Алгоритмическое описание	Число триггеров	Площадь триггеров	Режим синтеза: <i>onehot, flatten</i>			Режим синтеза: <i>onehot, preserve</i>		
			Средний ток, мА	Задержка, нс	Общая площадь схемы	Средний ток, мА	Задержка, нс	Общая площадь схемы
Способ 1	74	3256	220,126	48,98	221 776	209,445	45,60	216 267
Способ 2	74	3256	37,613	61,01	232 090	34,715	53,33	217 181
Способ 4	394	13 496	88,214	59,16	244 286	72,468	53,29	233 507
Способ 5	106	4280	46,383	61,49	241 752	39,622	53,33	218 817
Способ 6	406	13 808	34,311	45,74	243 470	30,839	44,47	226 557

операндов **A, B** вместо 16) приводит к тому, что средний потребляемый ток 6,296 мА (см. табл. 4, способ 1) вырос в 7,4 раза (46,507 мА, см. табл. 5), в то время как площадь выросла только в 3,5 раза.

Способ 2 обнуления операндов для невыполняемых операций оказался (по снижению потребляемого тока) в 1,5 раза более эффективной процедурой по сравнению с установкой единичных значений сигналов (способ 3), поэтому способ 3 был исключен из дальнейших экспериментов. Следует учесть, что это справедливо для рассматриваемой библиотеки логических КМОП-элементов, а для других технологических библиотек может не выполняться. Способы 2 и 3 не требуют дополнительных триггеров.

Сохранение значений операндов (способ 4) требует значительного увеличения числа дополнительных триггеров, его применение становится более эффективным с увеличением разрядности схемы операционного блока. Можно сделать вывод о том, что способ 4 алгоритмического описания может быть эффективен для тех операций, для которых сложность соответствующей подсхемы значительно превышает сложность вводимого регистра. Для простых логических операций сохранение значений операндов неэффективно, так как число транзисторов в регистрах значительно превышает сложность аппаратной реализации подсхем, реализующих простые логические операции. Поэтому более эффективным для рассматриваемого примера оказался способ 5 — сохранение значений операндов только для трудоемкой операции арифметического умножения.

Применение процедуры *clock gating* (способ 6) позволяет выигрывать в энергопотреблении, когда синтез проведен по иерархии описания (каждая операция реализуется независимо своей подсхемой) и сложность схемной реализации операции *ALU* значительно превосходит суммарную сложность вводимых дополнительных триггеров. Способ 6 оказался устойчивым (по показателю энергопотребления и быстрдействию) для различных режимов синтеза.

Другие эксперименты показали, что для кодирования состояний управляющего автомата для

минимизации энергопотребления целесообразно выбирать опцию *onehot* — кодирование каждого состояния автомата своей переменной. Использование опции *Gray* в синтезаторе *LeonardoSpectrum* приводит к кодированию состояний автомата кодом Грэя [13].

Эксперимент 4 показал, что увеличение частоты выполнения трудоемкой операции арифметического умножения (3818 раз вместо 1698 раз из всех 16 000 тестовых наборов) примерно в 1,5 раза увеличивает потребление для устройств с разрядностью $n = 16$ для способов 2, 4, 5, 6 и практически не увеличивает для способа 1.

Результаты эксперимента 5 свидетельствуют, что эффект снижения энергопотребления лишь увеличивается при увеличении разрядности операций для *ALU2* — обнуление операндов и *clock gating* являются эффективными приемами, при этом обнуление операндов не требует внесения в схему дополнительных регистров. Сравним способы 1 и 6 (см. табл. 9, режим *flatten*): увеличение площади на 9 % (способ 6) позволяет уменьшить энергопотребление в 6,4 раза (на 640 %) для устройств с разрядностью $n = 32$ по сравнению с традиционным способом 1.

Эксперимент 5 показал (см. табл. 9), что режим *flatten* не позволяет значительно сократить площадь схем (по сравнению с режимом *preserve*) при реализации подобных арифметических операций в случае большей размерности схем, однако для небольших размерностей задач синтеза режим *flatten* позволяет получать схемы меньшей площади.

Основные выводы по проведенным экспериментальным исследованиям: *clock gating* и сохранение значений операндов для сложных операций, а также обнуление операндов для простых операций являются эффективными процедурами для алгоритмического проектирования операционных устройств с пониженным энергопотреблением, реализуемых в базисе КМОП-элементов. При этом эффект применения таких приемов алгоритмического описания возрастает с увеличением сложности схем, реализующих вычислительные операции. Предложенные алгоритмические приемы снижения энергопотреб-

ления схем из КМОП-элементов целесообразно применять в тех случаях, когда имеются сложные вычислительные операции, т. е. когда потребление вводимых регистров значительно меньше потребления схем для соответствующих операций *ALU*. Если параметр площади является важным, то путем обнуления операндов, вызывающим незначительное увеличение площади, можно довольно значительно уменьшить энергопотребление. Если же снижение энергопотребления является самой важной целью, то применение *clock gating* является самым эффективным приемом для рассмотренной структуры операционных устройств, тем более что для больших размерностей задачи проектирования операционного устройства данный способ обеспечивает и лучшие показатели быстродействия схем.

Основными факторами, влияющими на принятие решений по составлению энергосберегающего VHDL-описания и превращению его в малопотребляющую логическую схему в рассматриваемой библиотеке КМОП-элементов являются:

- аппаратная сложность операций;
- частота выполнения операций, особенно тех, которые реализуются схемами большой сложности;
- режим (опция) синтеза;
- режим кодирования состояний управляющего автомата;
- сложность добавляемых регистров (триггеров), связанная с разрядностью операций;
- выбор одного из конкурирующих способов — обнуление операндов или их сохранение на основе *clock gating* для каждой из операций операционного блока;
- результаты оценки энергопотребления различных схем на одних и тех же тестовых последовательностях входных наборов.

6. Методика алгоритмического проектирования операционных устройств

Проведенные экспериментальные исследования по алгоритмическому описанию операционных устройств и анализ результатов экспериментов позволяют предложить следующую методику получения их алгоритмических описаний, по которым синтезируются КМОП-схемы с пониженным энергопотреблением.

1. Провести отдельно синтез и определить сложность схемной реализации (в заданной целевой библиотеке КМОП-элементов) каждой из операций *ALU* либо каждой совокупности параллельно выполняемых операций *ALU*.

2. Для управляющего автомата подобрать способ кодирования внутренних состояний по критерию минимизации энергопотребления логической схемы, реализующей автомат. Как правило, для этого в *LeonardoSpectrum* используется опция *onehot*, однако в работе [16] описаны и другие способы кодирования внутренних состояний управляющих автоматов.

3. Подсчитать число добавляемых элементов памяти для сохранения значений операндов.

4. Выбрать способ реализации операций *ALU*: обнуление (для простых операций), сохранение значений либо *clock gating* (для сложных операций).

5. Провести схемный синтез всего VHDL-описания устройства, используя выбранный способ кодирования состояний управляющего автомата для каждой из опций *preserve*, *flatten* синтеза, получить оценки сложности (площади) и быстродействия полученных схем.

6. Провести моделирование на псевдослучайных тестах и определить среднее энергопотребление схем.

7. Выбрать приемлемый вариант схемы (алгоритмическое VHDL-описание) по параметрам площади, быстродействия и энергопотребления.

Заключение

Используя различные способы алгоритмических VHDL-описаний цифровых устройств, состоящих из управляющего и операционного блоков, можно получать логические схемы различной площади, быстродействия и энергопотребления и выбирать, в зависимости от требований, приемлемый вариант. Существенного снижения энергопотребления синхронных цифровых устройств рассматриваемого класса можно добиться, используя соответствующее описание их функционирования на алгоритмическом уровне. Изменение алгоритмического описания и опций синтеза может значительно изменить сложность, быстродействие и характеристики энергопотребления схемы. Выполнять синтез для каждой операции *ALU* при минимизации энергопотребления можно независимо, чтобы была возможность отключения функционирования соответствующей подсхемы, когда это требуется. Вместе с тем минимизация числа элементов схемы, связанная с глобальной оптимизацией логического описания всей схемы и выполнением синтеза без учета иерархии, также во многих случаях позволяет уменьшать энергопотребление. Поэтому сочетание выбранного способа алгоритмического описания, ориентированного на снижение энергопотребления, и режима синтеза схемы является определяющим для характеристик получаемой логической схемы.

Предложенные приемы сокращения энергопотребления могут быть полезными и для других архитектур цифровых устройств, в которых имеются параллельные блоки, выполняющие каждый свою операцию (функционирующие не в каждом такте).

Важным в выполненном исследовании явилось также то, что замена трудоемкого схемотехнического моделирования быстродействующим логическим VHDL-моделированием позволила быстро оценивать энергопотребление сложных логических КМОП-схем и существенно сокращать время проведения экспериментов.

Полученные результаты могут быть полезными и для других технологий производства интегральных схем, для которых основную долю потребляемой мощности составляет динамическая мощность, связанная с переключениями сигналов.

Список литературы

1. Рабаи Ж. М., Чандракасан А., Николитч Б. Цифровые интегральные схемы, 2-е изд.: Пер. с англ. — М.: ООО "И. Д. Вильямс", 2007. — 912 с.
2. Немудров В., Мартин Г. Системы-на-кристалле. Проектирование и развитие. — М.: Техносфера, 2004. — 216 с.
3. Белоус А. И., Солодуха В. А., Шведов С. В. Космическая электроника. — Кн. 1. — М.: Техносфера, 2015. — 696 с.
4. Шашков А. С. Проектирование цифровых систем с пониженным энергопотреблением с применением технологии UPF-описания подсистемы питания // Информатика. — 2015. — № 3. — С. 90—104.
5. Rushton A. VHDL for logic synthesis. Third Edition. — John Wiley & Sons, Ltd. — 2011. — 466 p.
6. Поляков А. К. Языки VHDL и VERILOG в проектировании цифровой аппаратуры. — М.: СОЛОН-Пресс, 2003. — 320 с.
7. Майоров С. А., Новиков Г. И. Структура электронных вычислительных машин. — Л.: Машиностроение, 1979. — 384 с.
8. Skliarova I., Sklyarov V., Sudnison A. Design of FPGA-based Circuits using Hierarchical Finite State Machines. — Tallinn: TUT Press, 2012. — 242 p.
9. Иванюк А. А. Проектирование встраиваемых цифровых устройств и систем. — Минск: Бестпринт, 2012. — 337 с.
10. Schaumont P. R. Finite State Machine with Datapath // A Practical Introduction to Hardware. Software Codesign. Springer, Boston, MA. — 2010. — P. 95—132.
11. Томас Д. Логическое проектирование и верификация систем на SystemVerilog / пер. с англ. А. А. Слинкина, А. С. Камкина, М. М. Чупилко; науч. ред. А. С. Камкин, М. М. Чупилко. — М.: ДМК Пресс, 2019. — 384 с.
12. Бибило П. Н. Оценка энергопотребления цифровых устройств, представленных в виде композиции управляющего и операционного автоматов // Известия РАН. Теория и системы управления. — 2017. — № 3. — С. 97—117.
13. Бибило П. Н. Системы проектирования интегральных схем на основе языка VHDL. StateCAD, ModelSim, Leonardo-Spectrum. — М.: СОЛОН-Пресс, 2005. — 384 с.
14. Hai L., Bhunia S., Yiran C., Roy K., Vijaykumar T. N. DCG: Deterministic clock-gating for low-power microprocessor design // IEEE Trans. VLSI Systems. — 2004. — Vol. 12, N 3. — P. 245—254.
15. Kaxiras S., Martonosi M. Idle unit switching activity: clock gating // Architectural Techniques for Low Power. — Morgan & Claypool Publishers, 2008. — 207 p.
16. Бибило П. Н., Авдеев Н. А., Кардаш С. Н., Кириенко Н. А., Ланкевич Ю. Ю., Логинова И. П., Романов В. И., Черемисинов Д. И., Черемисинова Л. Д. Система логического проектирования функциональных блоков заказных КМОП СБИС с пониженным энергопотреблением // Микроэлектроника. — 2017. — Т. 46. — № 1. — С. 72—88.

Design of Digital Operational Units with Low Power Consumption

N. A. Avdeev, avdeev_n@newman.bas-net.by, P. N. Bibilo, bibilo@newman.bas-net.by,
The United Institute of Informatics Problems of the National Academy of Sciences of Belarus,
Minsk, 220012, Belarus

Corresponding author:

Bibilo Petr N., Head of Laboratory, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Minsk, 220012, Belarus
E-mail: bibilo@newman.bas-net.by

Received on October 13, 2020
Accepted on December 07, 2020

The lowering of power consumption in CMOS VLSI digital systems is one of the most important problems that appear now for developers of CAD systems. One of the effective approaches to lowering the dynamic power consumption is creation of an algorithmic description of the VHDL project, which provides for the deactivation of some functional blocks which are not necessary in particular moments. Contemporary synthesizers fulfill the high level synthesis of logic circuits by substitution of description of each VHDL construction with functionally structural description of a proper logic subcircuit. The results of digital logic circuit synthesis (the number of logic elements and power consumption) depend significantly on initial VHDL code. During initial VHDL code development it is possible to use different approaches to improve some parameters of synthesized logic circuit. At the algorithmic level of the digital design, it is necessary to provide for disconnection of the units, which cause the higher power consumption. In this paper such methods of algorithmic VHDL description of logic circuit are studied. The efficiency of the proposed methods is compared with the traditional method of VHDL-description which does not take the aspect of power consumption into account and is oriented only to the correct functionality of the developed logic circuit. To estimate the power consumption of logic circuits the approach is used which allows applying high-speed logical VHDL-simulation of structural descriptions (netlists) of logic circuits instead of slow SPICE simulation. The main conclusion of the provided study is the following: the clock gating and the storage of operand values for complex operations as well as zero value setting for simple ones are effective methods for the VHDL description of operational units with low power consumption implemented in the CMOS basis.

Keywords: digital device, finite state machine with datapath, digital logic synthesis, simulation, power consumption, VHDL, VLSI, CMOS

For citation:

Avdeev N. A., Bibilo P. N. Design of Digital Operational Units with Low Power Consumption, *Programmная Inzheneriya*, 2021, vol. 12, no. 2, pp. 63–73

DOI: 10.17587/prin12.63-73

References

1. Rabai M., Chandrakasan A., Nolic B. *Digital Integrated Circuits: A Design Perspective*, Moscow, Williams, 2007, 912 p. (in Russian).
2. Nemudrov V., Martin G. *System on chip. Design and development*, Moscow, Tehnosfera, 2004, 216 p. (in Russian).
3. Belous A. I., Soloduha V. A., Shvedov S. V. *Space electronics*, Kn. 1. Moscow, Tehnosfera, 2015, 696 p. (in Russian).
4. Shashkov A. S. Design of low-power electronic systems using UPF power intent specification technology, *Informatika*, 2015, no. 3, pp. 90–104. (in Russian).
5. Rushton A. *VHDL for logic synthesis*, Third Edition, John Wiley & Sons, Ltd, 2011, 466 p.
6. Polyakov A. K. *The VHDL and VERILOG languages in design of the digital equipment*, Moscow, SOLON Press, 2003, 320 p. (in Russian).
7. Majorov S. A., Novikov G. I. *The structure of the electronic computing machines*, Leningrad, Mashinostroenie, 1979, 384 p. (in Russian).
8. Skliarova I., Sklyarov V., Sudnison A. *Design of FPGA-based Circuits using Hierarchical Finite State Machines*, Tallinn, TUT Press, 2012, 242 p.
9. Ivanyuk A. A. *Design of the embedded digital devices and systems*, Minsk, Belsprint, 2012, 337 p. (in Russian).
10. Schaumont P. R. *Finite State Machine with Datapath, A Practical Introduction to Hardware. Software Codesign*. Springer, Boston, MA., 2010, pp. 95–132.
11. Tomas D. *Logical design and verification using SystemVerilog*, Moscow, DMK Press, 2019, 384 p. (in Russian).
12. Bibilo P. N. Estimation of the Energy Consumption of Digital Devices Represented by a Composition of a Control FSM and an Arithmetic and Logic Unit, *Izv. RAN. Teoriya i Sistemy Upravleniya*, 2017, no. 3, pp. 97–117. (in Russian).
13. Bibilo P. N. *CAD of integrated circuits based on the VHDL. StateCAD, ModelSim*, LeonardoSpectrum, Moscow, SOLON-Press, 2005, 384 p. (in Russian).
14. Hai L., Bhunia S., Yiran C., Roy K., Vijaykumar T. N. DCG: Deterministic clock-gating for low-power microprocessor design, *IEEE Trans. VLSI Systems*, 2004, vol. 12, no. 3, pp. 245–254.
15. Kaxiras S., Martonosi M. Idle unit switching activity: clock gating, *Architectural Techniques for Low Power*, Morgan & Claypool Publishers, 2008, 207 p.
16. Bibilo P. N., Avdeev N. A., Kardash S. N., Kirienko N. A., Lankevich Ju. Ju., Loginova I. P., Romanov V. I., Cheremisinov D. I., Cheremisinova L. D. A System for Logical Design of Custom CMOS VLSI Functional Blocks with Reduced Power Consumption, *Mikroelektronika*, 2017, vol. 46, no. 1, pp. 72–88. (in Russian).

ИНФОРМАЦИЯ

Продолжается подписка на журнал "Программная инженерия" на первое полугодие 2021 г.

Оформить подписку можно в любом отделении Почты России, через подписные агентства или непосредственно в редакции журнала.

Подписной индекс по Объединенному каталогу

"Пресса России" — 22765

Сообщаем, что с 2020 г. возможна подписка на электронную версию нашего журнала через:

ООО "ИВИС": тел. (495) 777-65-57, 777-65-58; e-mail: sales@ivis.ru,
ООО "УП Урал-Пресс". Для оформления подписки (индекс 013312) следует обратиться в филиал по месту жительства — <http://ural-press.ru>

Адрес редакции: 107076, Москва, Стромынский пер., д. 4,
Издательство "Новые технологии",
редакция журнала "Программная инженерия"

Тел.: (499) 269-53-97. Факс: (499) 269-55-10. E-mail: prin@novtex.ru

Д. А. Орлов, студент, cvaize@gmail.com, **Р. А. Карелова**, канд. пед. наук, доц., riya2003@mail.ru, Уральский федеральный университет имени первого Президента России Б. Н. Ельцина, Нижнетагильский технологический институт, Нижний Тагил, Россия

Архитектура связанных JavaScript-микрокомпонентов для крупных веб-проектов

Приведено описание архитектуры связанных JavaScript-компонентов, которая позволяет реализовать взаимодействие множества скриптов, подключенных в веб-проекте асинхронно и исполняемых в любое время и в любом порядке. Архитектура основана на идее разделения кода на компоненты, JavaScript-событиях и их прослушивании. Данный подход призван избавить разработчика от необходимости организации очередности подключения скриптов в HTML-документе, отслеживания времени их загрузки и выбора метода загрузки, что дает большие возможности для расширения и гибкости кода.

Ключевые слова: HTML, JavaScript, Event, CustomEvent, микрокомпоненты, асинхронное подключение скриптов, компоненты

Введение

В статье представлено описание архитектуры связанных JavaScript-компонентов (далее — архитектура) [1] для реализации взаимодействия множества скриптов, подключенных в веб-проекте асинхронно и исполняемых в любое время и в любом порядке.

При разработке клиентской части веб-сайтов можно обрабатывать CSS-, JS- и HTML-компоненты вручную или использовать некоторые инструменты — сборщики модулей. При этом современные сборщики модулей, основанные на Node.js (например, Webpack, Browserify и др.), не могут работать с PHP, Python, .NET-языками [2], которые формируют ответ клиенту на запрос в виде HTML-файла. Для веб-разработки на языках программирования, в которых Node.js не имеет доступа к контексту и не может формировать HTML-страницу, было предложено архитектурное решение, позволяющее делегировать функции подключения скриптов этим языкам. В рамках предлагаемого решения с помощью сборщиков модулей можно проводить обработку скриптов и стилей компонентов, размещаемых впоследствии в определенных директориях и используемых оттуда языками программирования для подключения на HTML-страницу. Речь идет о любых языках, применяемых в веб-разработке.

Архитектура основана на идее разделения кода на компоненты, JavaScript-событиях и их прослушивании. JavaScript-компонентами (далее — компонентами) названы участки JavaScript-кода, объединенные по общему признаку, их действию и роли в приложении [3]. Компонентам присваивается уникальное название для их отличия от других компонентов.

Подход, описанный в статье, призван избавить разработчика от необходимости организации очеред-

ности подключения скриптов в HTML-документе, от отслеживания времени их загрузки и выбора метода загрузки, что дает большие возможности для расширения и гибкости кода. Для понимания сути решаемой проблемы представим два подхода к реализации веб-сайтов.

Первый подход к реализации веб-сайтов

В современной веб-разработке часто встречаются проекты, в которых HTML-, CSS- и JavaScript-код используется без жесткой связи [4]. Этот факт означает, если разработчику будет необходимо удалить или подключить часть кода на сайте (например, удалить или подключить слайдер), то понадобится отредактировать HTML-, CSS- и JavaScript-код. Такое можно встретить в проектах на PHP, Python, HTML.

В таких проектах с JavaScript-кодом, как правило, осуществляют перечисленные далее действия.

- Объединяют в 1...3 файлах (в зависимости от задумки) и получают JavaScript-файлы, содержащие код для всех страниц сайта. Такие файлы могут иметь размер 0,5 Кбайт...5 Мбайт.
- Разбивают код на части и подключают только на страницах, где это нужно. Но тут возникает вопрос со связью разбитого кода между собой. Чтобы не возникало ошибок, разделенный код должен идти в строгом порядке, это сложно отслеживать вручную, когда над проектом работает несколько разработчиков и связанных компонентов становится много. В случаях объединения этих компонентов в кластеры сложность увеличивается еще больше, так как нужно учитывать связи целого кластера компонентов [5, 6]. Архитектура, которая представлена в данной статье, решает этот вопрос путем ожидания событий инициализации от зависимых компонентов.

• Разбивают по страницам и получают несколько JavaScript-файлов с размерами от 0,3 Кбайт до 2 Мбайт. Такая оптимизация позволяет не загружать код, который не используется на странице. Это решение имеет свои недостатки, а именно большое время сборки JavaScript-файлов, необходимость загружать один и тот же код, дублирующийся в этих файлах, для нескольких страниц сайта.

Второй подход к реализации веб-сайтов

В противовес первому подходу выступают проекты, основанные на JavaScript-фреймворках [7]. Они создаются с помощью программной платформы Node.js и собирают HTML/CSS/JavaScript-код на сервере [8]. Благодаря этому у JavaScript есть доступ к страницам и логике отображения контента, что дает возможность автоматически разбить код по страницам. Этот подход позволяет автоматически дробить код на части и подгружать только необходимый код. Однако у таких проектов есть недостатки:

- сложность в SEO-продвижении [9];
- к обычной верстке по макету и описанию логики на JavaScript добавляется необходимость знаний фреймворка и его особенностей, необходимость в описании сборки всего кода и проработки его доставки на страницу пользователю;
- увеличение сложности влечет за собой увеличение временных и финансовых затрат на реализацию, поддержку и изменения.

Постановка задачи

При разделении JavaScript-кода на отдельные файлы и их подключении на страницах сайта существует вопрос, связанный с наличием связей в этом коде. Подключение скриптов на странице должно осуществляться таким образом, чтобы зависимости исполняемого JavaScript-кода уже были загружены, исполнены и записаны в заранее известное хранилище, например, в `window` — глобальный объект, который предоставляет переменные и функции, доступные в любом месте программы.

Для разъяснения сути предлагаемой в статье архитектуры, позволяющей решить вопросы связей в JavaScript-коде, были поставлены следующие задачи:

- рассмотреть существующие методы решения задачи связанности разделенного JavaScript-кода;
- объяснить суть работы предлагаемого архитектурного решения и показать примеры кода;
- провести ряд тестов, чтобы сравнить показатели производительности.

Одно из существующих альтернативных рассматриваемой задачи решений основано на использовании атрибута `defer`. Чтобы запустить разделенные скрипты последовательно, можно использовать атрибут `defer` тега `script`. Этот атрибут позволяет загрузить скрипты параллельно, а запустить их последовательно. Тем самым это решает задачу связи между разделенными скриптами.

Однако использование этого метода вызывает сложность в контроле очереди подключаемых скриптов на странице, что, в свою очередь, создает трудности в подключении приоритетных скриптов. Данный метод не решает проблему отложенного исполнения JavaScript-кода.

Вывод: в маленьких проектах можно ограничиться атрибутом `defer`, в крупных и/или проектах со сложной логикой необходимо использовать иные решения, одним из которых является представленный алгоритм.

Суть предлагаемого архитектурного решения

Для соблюдения очереди выполнения разбитого на части JavaScript-кода предлагается архитектуру приложения основывать на ожидании зависимостей. При этом каждая часть кода при исполнении генерирует событие, которое перехватывают другие зависимые участки кода, а затем исполняются, когда все зависимости были выполнены (исполнены/инициализированы).

Разберем подробнее, как связаны между собой группы скриптов в HTML-документе.

Допустим, что имеется два скрипта А и Б, загружаемые в случайное время. Скрипт Б нуждается в скрипте А. Чтобы сработал скрипт Б, он должен проверить наличие результата исполнения скрипта А в объекте `window`. Если результата исполнения скрипта А в объекте `window` нет, то скрипт Б дожидается события инициализации от скрипта А через прослушивание событий (сигналов от браузера о том, что что-то произошло) [10], передаваемых через объект `document` — основную "входную точку" DOM (*Document Object Model* — объектная модель документа, которая представляет все содержимое страницы в виде объектов, которые можно менять). Только после перехвата события инициализации от скрипта А исполнится скрипт Б и запишет результат своей работы в объект `window`. Таким образом работает пример реализации алгоритма на JavaScript (рис. 1).

Как место для сохранения компонентов и проверки их наличия используется глобальный объект `window`. При этом как объект для перехвата сигналов и генерации событий был выбран глобальный объект `document`.

Сначала была составлена первая версия реализации алгоритма, она понятнее всего раскрывает механизм работы (см. листинг 1). Компоненту `autocomplete` нужна зависимость `trigger` (см. листинг 2), которая находится в компоненте `app`, если эта зависимость есть, то компонент сразу инициализируется и отправляет событие `autocomplete:init` в `document`, остальные компоненты, зависимые от этого компонента, прослушивают это событие и инициализируются следом. Если зависимости `trigger` нет, то компонент `autocomplete` прослушивает событие `app:init` и инициализируется следом.

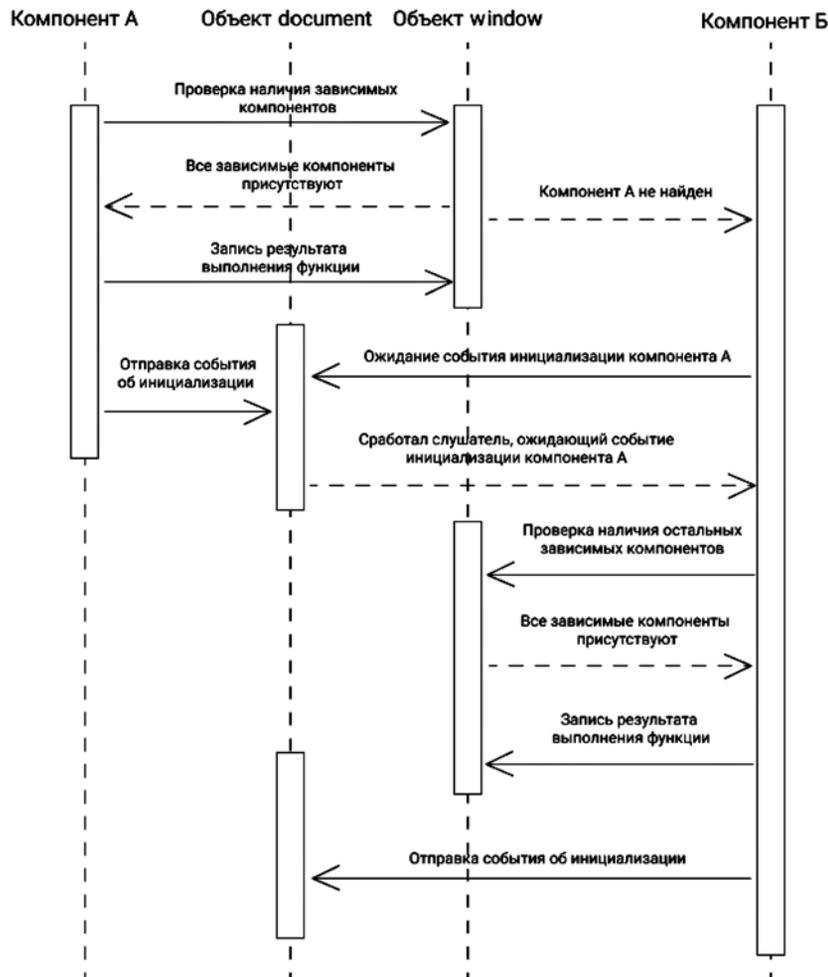


Рис. 1. Схема взаимодействия компонентов

```

//./components/autocomplete.js
// импортируется autocomplete
import autocomplete from "autocomplete";

// autocomplete помещается в объект window
window.autocomplete = autocomplete;

// Функция добавления слушателя
function on (el, e, fn) {
  el.removeEventListener(e, fn)
  el.addEventListener(e, fn)
}

// Функция, передающая событие об инициализации в объект document
function init () {
  window.trigger(document,
  'autocomplete:init')
}

// Проверка наличия функции trigger, если ее нет,
// то выполняется подписка на событие
// инициализации главного компонента, в котором
// есть функция trigger
if(!window.trigger){

```

```

on(document, 'app:init', init)
}else{
  init()
}

```

Листинг 1. Первая версия реализации алгоритма

```

//./components/app.js
window.trigger = function (element, eventName,
---перенос строки---
bubbles = true, cancelable = true) {
  if ("createEvent" in document) {
    let evt = document.createEvent("HTMLEvents");
    evt.initEvent(eventName, bubbles, cancelable);
    element.dispatchEvent(evt);
  } else {
    // IE 8
    let e = document.createEventObject();
    e.eventType = eventName;
    element.fireEvent('on' + e.eventType, e);
  }
}
//... other code

```

Листинг 2. Функция, которая генерирует событие и передает его в переданный DOM-элемент

Компонент `cart.js` использует компонент `autocomplete.js`: сначала идет проверка наличия компонента. Если его нет, то происходит подписка на событие инициализации, как только событие всплывает, вызывается функция `autocomplete` (см. листинг 3).

```

//./components/cart.js

//... other code
const actions = {
  autocomplete () {
    let input = document.querySelector
    ---перенос строки---
    (S_CHECKOUT_ADDRESS);
    if (!input) {
      return true
    }
    // если нет компонента autocomplete,
    // то происходит подписка на событие его
    // инициализации

    // после отлова этого события данная функ-
    //ция исполнится снова
    if (!window.autocomplete) {
      on(document, 'autocomplete:init',
      ---перенос строки---
      actions.autocomplete);
      return true
    }

    if (!window.fetchAddresses) {
      on(document, 'app:init', actions.autocom-
      plete);
      return true
    }

    //... other code
    window.autocomplete({
      input,
      minLength: 2,
      fetch,
      emptyMsg: 'Адрес не найден:',
      render: function (item, search) {
        //... other code
      },
      onSelect: function (data) {
        //... other code
      }
    });
    //... other code
  }
};
//... other code

```

Листинг 3. Компонент `cart.js`

В описанной выше версии решения (первой версии) для каждого компонента необходимо прописывать логику прослушивания и генерации событий вручную, кроме этого разработчики не защищены от ошибок в названиях компонентов и событий.

Чтобы исправить эти недочеты, было принято решение выделить названия компонентов в отдельные файлы, чтобы их можно было импортировать и не допускать ошибок в названиях.

Вид компонентов после внесенных изменений компоненты приведен на листингах 4 и 5.

```

//./components/autocomplete.js

// основной код компонента 'autocomplete'
import autocomplete from 'autocomplete';

// функция, обрабатывающая логику связи
// с другими компонентами
import init from '../helpers/init';

// уникальное название текущего компонента
import NAME from '../names/autocomplete';

init(function () {
  // Вывод в консоль для проверки
  console.log('Hello from autocomplete.js');
  return autocomplete;
}, NAME)

```

Листинг 4. Компонент `autocomplete.js`

```

//./components/cart.js

import init from '../helpers/init';

// уникальное название текущего компонента
import NAME from '../names/cart';

// уникальное название зависимого компонента
import AUTOCOMPLETER_NAME from
'../names/autocomplete';

init(function () {
  console.log('Hello from autocomplete.js');
  const autocomplete = window [AUTOCOMPLET-
  ER_NAME];
  //... other code
  return {
    actions: {...}
  };
}, NAME, [AUTOCOMPLETER_NAME])

```

Листинг 5. Компонент `cart.js`

Для инициализации компонентов и ожидания их зависимостей был написан небольшой скрипт (см. листинг 6).

```

//./helpers/init.js

// функция, возвращающая название события
// инициализации для названия компонента

import getInitEvent from './events/init';
// функция, генерирующая переданное событие
// на переданном элементе
import trigger from './trigger';

```

```
// функция, обрабатывающая логику связи с
// другими компонентами
export default function (callback, name, depends = []) {

  function forEachDependencies(event){
    let active = true;

    depends.forEach(depend => {
      if(!window[depend]){
        active = false;
      }
    })

    if(active){
      depends.forEach(depend => {
        document.removeEventListener
---перенос строки---
(getInitEvent(depend), forEachDependencies);
      });
      window[name] = callback() || {};
      trigger(document, getInitEvent(name));
    }
    return active;
  }

  if(!forEachDependencies()){
    depends.forEach(depend => {
      document.addEventListener
---перенос строки---
(getInitEvent(depend), forEachDependencies);
    });
  }
}
```

Листинг 6. Функция, проверяющая и ожидающая зависимости компонентов, а также вызывающая событие инициализации в document и записывающая результат исполнения в window

После выполнения функции `./helpers/init.js` ее результат помещается в объект `window`, если все зависимости присутствуют в этом объекте. Если нет хотя бы одной зависимости, то функция ждет событий инициализации и только после инициализации всех компонентов выполняет функцию `callback`. Названия всех событий также выделены в отдельную папку `./events/`. Основное событие инициализации выделено в функцию `./events/init.js` (см. листинг 7). В примере показано, как целый компонент `cart.js` ждет компонент `autocomplete.js`, но это не обязательно. Можно сделать проверку как в первой версии, только названия компонентов и событий берутся из файлов, что исключает ошибки.

Также очень важна очередь подключения. Первыми будут загружены первые скрипты, а так как они небольшие, то могут быстро загрузиться и работать без простоя.

```
// './events/init.js'
export default function (name) {
  return name + ':init';
}
```

Листинг 7. Функция, возвращающая название события инициализации в зависимости от названия компонента

Тесты и производительность

Сравним производительность алгоритма на примере 22 JavaScript-компонентов, созданных для тестов. В первом случае они будут подключены отдельно (рис. 2). Во втором случае они собраны

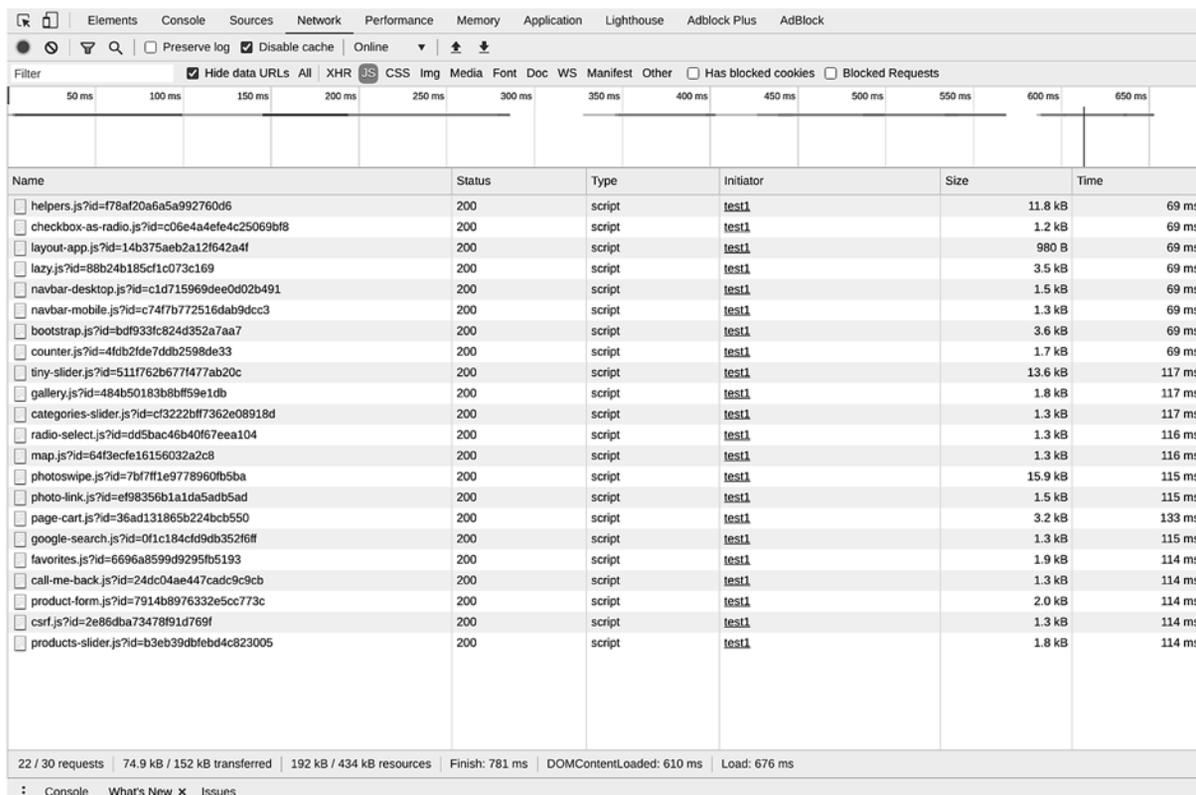


Рис. 2. Результаты работы алгоритма для JavaScript-компонентов, разбитых на файлы

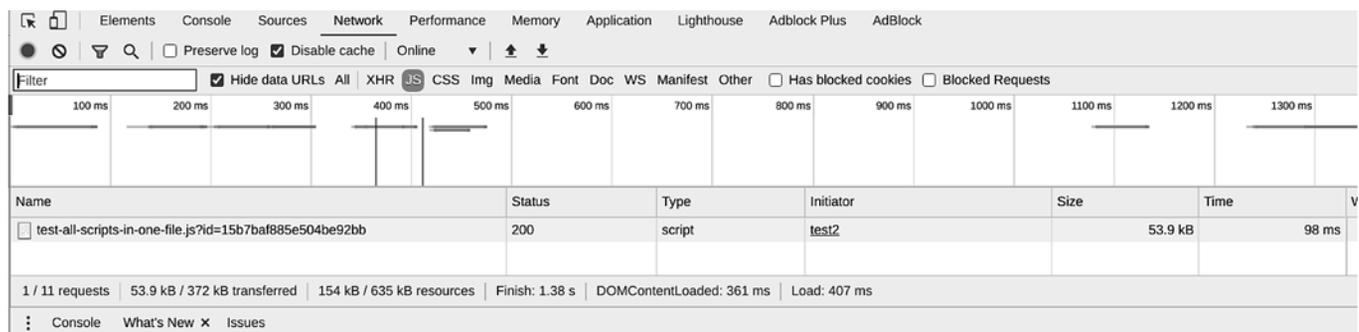


Рис. 3. Результаты работы алгоритма для JavaScript-компонентов, объединенных в один файл

в один файл (рис. 3). В качестве сервера, отдающего JavaScript-файлы, выступает nginx с использованием протокола HTTP/2.

На рис. 2 можно заметить, что 8 скриптов загрузились параллельно, остальные 14 загрузились тоже параллельно, но с отставанием от первых 8. Это произошло вследствие ограничений в браузере на число одновременных запросов.

Сравнение размеров

Очевидно, что размеры скриптов отличаются. Размер всех 22 скриптов, разделенных на компоненты, равен 192 Кбайта, а скриптов, объединенных в один файл, — 154 Кбайта. Разница в размерах составляет 38 Кбайт. Если поделить эту цифру на число компонентов (22), то получится 1,7272 Кбайта. И, если учесть, что в объединенном файле скриптов есть один скрипт, обрабатывающий цепочку вызовов, такой же, как в разделенных компонентах, то можно принять вес этого скрипта равным 1,6 Кбайта.

Вывод: скрипт обработки вызовов `../helpers/init.js` добавляет примерно 1,6 Кбайт веса каждому файлу.

Сравнение времени загрузки

Следует обратить внимание на время загрузки последних скриптов в обоих случаях. В случае разделения на 22 компонента время загрузки последнего скрипта `page-cart.js` составляет 133 мс (см. рис. 2). В случае объединения время загрузки скрипта — 98 мс (см. рис. 3). Разница в 35 мс является недостатком метода разбиения на компоненты, но только в случае загрузки всех скриптов.

Если сравнить минимальное время загрузки разделенных скриптов, равное 69 мс, и время загрузки объединенного файла, равное 98 мс, то разница в 29 мс показывает, что алгоритм быстрее загружает отдельные первые компоненты. В тесте было загружено 8 компонентов за время 69 мс, и эти компоненты начали свою работу раньше, чем все остальные компоненты и раньше, чем объединенный файл.

Общее время обработки, которое также включает в себя ожидание в очереди, отличается. Для разбитых скриптов оно равно 676 мс, а для объединенного скрипта — 407 мс.

Вывод: скрипты, разделенные на компоненты, все вместе загружаются медленнее, однако часть

скриптов, разделенных на компоненты, загружается быстрее, чем все скрипты, собранные в один файл. Это обстоятельство предоставляет возможность применять этот алгоритм для приоритетной загрузки скриптов. Разделив скрипты по приоритету и поместив более приоритетные в начало, можно добиться ускорения загрузки для этих скриптов и немного отложить не приоритетные скрипты.

Проверка алгоритма на задержку инициализации и повторные инициализации

Чтобы проверить алгоритм на задержку инициализации и повторные инициализации модифицируем его, добавив небольшой код счетчика (функция `counter` в листинге 8), а также добавим скрипт напрямую на страницу. Этот счетчик покажет, есть ли отставания в работе скриптов.

```
import getInitEvent from './events/init'
import trigger from './trigger';

function counter(){
  if(window.appCounterIndex === undefined){
    window.appCounterIndex = 0;
  }
  window.appCounterIndex++;
  console.log(new Date(),
  ---перенос строки---
  `Index = ` +window.appCounterIndex)
}

export default function (callback, name, depends = []){
  function forEachDependencies(event){
    let active = true;

    depends.forEach(depend => {
      if(!window[depend]){
        active = false;
      }
    })

    if(active){
      depends.forEach(depend => {
        document.removeEventListener
        ---перенос строки---
        (getInitEvent(depend), forEachDependencies);
      })
    }
  }
}
```

```

});
window[name] = callback() || {};
counter()
trigger(document, getInitEvent(name));
}
return active;
}
if(!forEachDependencies()){
depends.forEach(depend => {
document.addEventListener
---перенос строки---
(getInitEvent(depend), forEachDependencies);
});
}
}

```

Листинг 8. Модифицированная версия алгоритма для проверки наличия задержек

Результат работы алгоритма представлен на рис. 4 и 5.

В консоли (см. рис. 4 и 5) можно увидеть, что после загрузки страницы сразу обработалась строка с выводом начала теста — Start, затем пошла загрузка скриптов и их инициализация, после инициализации каждый скрипт отправлял в консоль сообщение с порядковым номером инициализации (Index, от 1 до 22). Если сравнить время начала загрузки скриптов и их инициализации, то может показаться, что время не изменилось, на самом деле загрузка и инициализация скриптов заняла меньше 1 с времени, а если точнее, то для разбитых скриптов — 676 мс, а для объединенного скрипта — 407 мс.

Вывод: скрипт не тормозит естественную инициализацию и на его исполнение требуется минимум времени.

Заключение

Представленный в статье алгоритм прост в работе, не влияет на производительность и позволяет гиб-

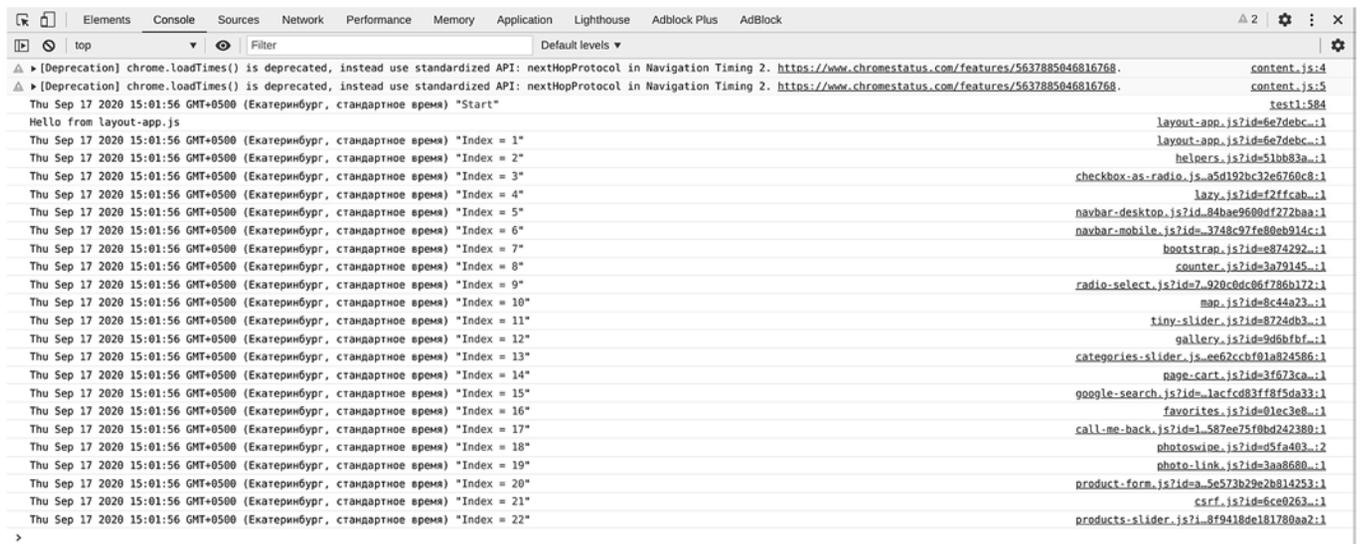


Рис. 4. Вывод результата работы алгоритма при подключении скриптов, разбитых на файлы

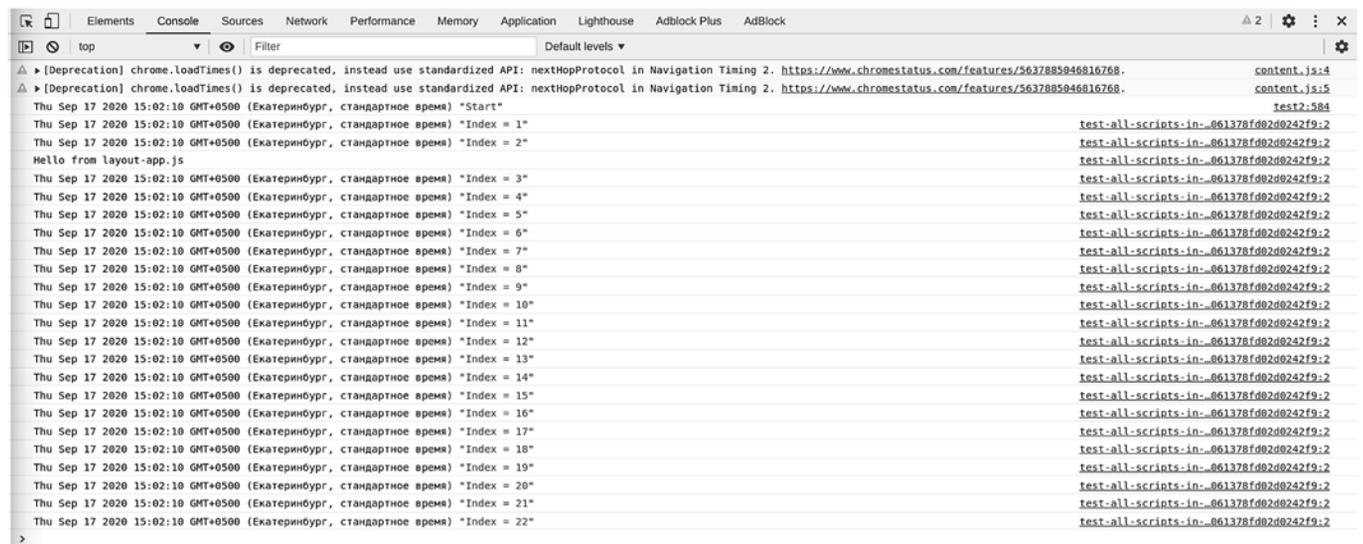


Рис. 5. Вывод результата работы алгоритма при подключении скриптов одним файлом

ко управлять частями JavaScript-кода, выделенными в компоненты. Кроме этого, данный подход стандартизирует общение между компонентами путем выделения названий и событий в отдельные файлы, предоставляет возможность откладывать инициализацию целых цепочек компонентов. Также он способен упростить работу с динамически подгружаемыми компонентами.

Представленное решение разбиения кода на компоненты имеет преимущества перед объединением JavaScript-кода в один файл: при разбиении появляется возможность использовать приоритетные скрипты раньше остальных, а более тяжелую и/или не приоритетную часть подгружать в самом конце.

В дальнейшем в рамках описанного решения можно уменьшить число подключенных в HTML-документ скриптов путем объединения компонентов в кластеры — JavaScript-файлы с множеством компонентов. Это позволит уменьшить время загрузки и сохранить связанность компонентов.

Список литературы

1. **Горюнова М. П.** Архитектурные стили в разработке web-приложений и область их применения // Проблемы современной науки и образования. — 2017. — № 12 (94). — С. 32–34.
2. **Programming** languages market share. URL: <https://www.wappalyzer.com/technologies/programming-languages/>
3. **С орбитальной** высоты. URL: <https://learn.javascript.ru/webcomponents-intro>
4. **Кириченко А. В.** JavaScript для FrontEnd-разработчиков. Написание. Тестирование. Развертывание. — СПб.: Наука и Техника, 2020. — 320 с.
5. **Lee B.** Build components in JavaScript without a framework. URL: <https://developer.okta.com/blog/2019/12/18/vanilla-javascript-components>
6. **Пьюривал С.** Основы разработки веб-приложений. — СПб.: Питер, 2015. — 272 с.
7. **Vue.js vs Vanilla.js** — what are the differences? URL: <https://vue-view.com/vue-js-vs-vanilla-js-what-are-the-differences/>
8. **Стефанов С.** JavaScript. Шаблоны. — Пер. с англ. — СПб.: Символ-Плюс, 2011. — 272 с.
9. **Сайты** на JavaScript-движке: стоит ли игра свеч? URL: <https://serpstat.com/ru/blog/sajti-na-javascript-dvizhkah/>
10. **JavaScript.** HTML DOM EventListener. URL: https://www.w3schools.com/js/js_htmlDOM_eventlistener.asp

Connected JavaScript-Microcomponents Architecture for Large Web-Projects

D. A. Orlov, cvaize@gmail.com, **R. A. Karel'ova**, riya2003@mail.ru, Nizhny Tagil Technological Institute (branch) of the Ural Federal University named after the first President of Russia B. N. Yeltsin, Nizhny Tagil, 622031, Russian Federation

Corresponding author:

Karel'ova Riya A., PhD, Associate Professor, Nizhny Tagil Technological Institute (branch) of the Ural Federal University named after the first President of Russia B. N. Yeltsin, Nizhny Tagil, 622031, Russian Federation
E-mail: riya2003@mail.ru

*Received on October 15, 2020
Accepted on December 09, 2020*

The article describes the architecture of connected JavaScript components, which allows you to implement the interaction of many scripts connected in a web project asynchronously and executable at any time and in any order. The architecture is based on the idea of dividing code into components, JavaScript events and listening to them. This approach is intended to relieve the developer from the need to organize the sequence of connecting scripts in an HTML-document, track their loading time and select the loading method, which gives great opportunities for expanding and flexibility of the code.

Keywords: HTML, JavaScript, Event, CustomEvent, micro-components, asynchronous script connection, components

For citation:

Orlov D. A., Karel'ova R. A. Connected JavaScript-Microcomponents Architecture for Large Web-Projects, *Programm'naya Ingeneria*, 2021, vol. 12, no. 2, pp. 74–81

DOI: 10.17587/prin.12.74-81

References

1. **Goryunova M. P.** Architectural styles in the development of web applications and their area of application, *Problemy sovremennoj nauki i obrazovaniya*, 2017, no. 12, pp. 32–34 (in Russian).
2. **Programming** languages market share, available at: <https://www.wappalyzer.com/technologies/programming-languages/>
3. **From** orbital altitude, available at: <https://learn.javascript.ru/webcomponents-intro> (in Russian).
4. **Kirichenko A. V.** *Javascript for Front-End Developers. Writing. Testing. Deployment*, Saint Petersburg, Nauka i Tekhnika, 2020, 320 p. (in Russian).

5. **Lee B.** Build components in JavaScript without a framework, available at: <https://developer.okta.com/blog/2019/12/18/vanilla-javascript-components>
6. **P'yurival S.** *Fundamentals of web application development*, Saint Petersburg, Piter, 2015, 272 p. (in Russian).
7. **Vue.js vs Vanilla.js** — what are the differences?, available at: <https://vue-view.com/vue-js-vs-vanilla-js-what-are-the-differences/>
8. **Stefanov S.** *JavaScript. Shablony*, — Per. s angl. Saint Petersburg, Simvol-Plyus, 2011, 272 p. (in Russian).
9. **Sajty** na JavaScript-dvizhkah: stoit li igra svech? — mnenie ekspertov, available at: <https://serpstat.com/ru/blog/sajti-na-javascript-dvizhkah/> (in Russian).
10. **JavaScript** HTML DOM EventListener, available at: https://www.w3schools.com/js/js_htmlDOM_eventlistener.asp

В. Е. Шампаров, инженер-программист, Victor.E.Shamparov@mcst.ru,
А. Л. Маркин, начальник сектора, Alex.L.Markin@mcst.ru, АО "МЦСТ",
 Московский физико-технический институт

Механизм оптимизации Structure Splitting в составе компилятора для микропроцессоров Эльбрус

Предложена новая версия оптимизации Structure Splitting, реализованная в составе компилятора для микропроцессоров с архитектурами Эльбрус и SPARC. Structure Splitting предназначена для улучшения локальности данных с помощью преобразования массивов структур в массивы из структур меньшего размера. Оптимизация была применена к варианту массива структур, вложенного в другую структуру, для которого память может быть перевыделена. При применении разработанного механизма оптимизации скорость исполнения двух тестов из наборов SPEC CPU2000 и SPEC CPU2006 увеличилась на 19 и 12 % соответственно.

Ключевые слова: компилятор, оптимизация, Structure Splitting, Эльбрус, SPARC

Введение

Улучшение работы программы с памятью, особенно с кешем, является одной из ключевых задач оптимизирующего компилятора. Для этого в компиляторы включают различные методы, среди которых оптимизации расположения данных в памяти (*data layout optimizations*). Эти механизмы увеличивают степень локальности данных с помощью их перераспределения в памяти. Благодаря таким механизмам уменьшается вероятность промахов кеша и, как следствие, уменьшается время исполнения программы.

Подмножеством механизмов оптимизации расположения данных в памяти являются механизмы реорганизации структур, увеличивающие локальность расположения их полей. Среди таких способов оптимизаций можно отметить Structure Peeling [1] и Structure Splitting, преобразующие массив структур в несколько массивов из структур меньшего размера. В Structure Splitting, в отличие от Structure Peeling, учтена возможность наличия указателей на структуры преобразуемых типов среди изменяемых структур.

В данной работе представлен реализованный авторами механизм оптимизации (далее для краткости — оптимизация) Structure Splitting, обобщенный на случай вложенного в структуру динамически выделенного массива структур, размер которого может изменяться во время исполнения программы. В разд. 1 приведена решаемая созданной оптимизацией проблема. Созданный для решения проблемы алгоритм описан в разд. 2. В разд. 3 приведены полученные результаты. Связанные с темой данной работы публикации описаны в разд. 4. В Заключении подведены итоги проделанной работы и описаны планы дальнейших исследований над оптимизациями реорганизации структур.

1. Решаемая задача

В ходе исследования пакета тестирования SPEC CPU2000 [2] в одной из задач (181.mcf) была обнаружена ситуация, в которой в структуру было вложено несколько массивов структур. Пример на рис. 1 показывает упрощенное описание иерархии структур из задачи 181.mcf. Массивы структур `node_t* nodes`, `arc_t* arcs` и `arc_t* dummy_arcs`

```
typedef struct node
{
    ...
    struct node *pred, *child, *sibling, *sibling_prev;
    ...
    struct arc *basic_arc;
    struct arc *firstout, *firstin;
    ...
} node_t;

typedef struct arc
{
    node_t *tail, *head;
    struct arc *nextout, *nextin;
    int32_t cost;
    ...
} arc_t;

typedef struct network
{
    ...
    node_t *nodes, *stop_nodes;
    arc_t *arcs, *stop_arcs;
    arc_t *dummy_arcs, *stop_dummy;
    ...
} network_t;
```

Рис. 1. Упрощенная иерархия структур из задачи 181.mcf

```

network_t *net;
arc_t *arc;
arc_t *stop;
int new_cost;

arc = net->arcs;
for ( stop = (void*) net->stop_arcs; arc < stop; arc++ )
    arc->cost = new_cost;

```

Рис. 2. Упрощенный пример неэффективного цикла для иерархии структур из задачи 181.mcf

вложены в структуру `network_t`, а все остальные указатели типов `node_t*` и `arc_t*` внутри структур `node_t` размером 32 байта, `arc_t` размером 60 байт и `network_t` являются указателями на элементы трех перечисленных массивов. Хранение данных в виде массива структур может быть неэффективно, если в коде программы встречаются циклы, в которых обращаются только к небольшой части полей элемента массива. Пример подобного цикла с использованием иерархии структур из задачи 181.mcf приведен на рис. 2. При обращении к полю `cost` (поле имеет сдвиг 16 байт относительно начала структуры `arc_t`) элемента массива в кеш-строку загружается вся структура размером 32 байта, из которых 28 байт в данном цикле не нужны. Пример кеш-строки на первой итерации приведенного цикла показан на рис. 3. Вследствие загрузки в кеш-память ненужных данных частота промахов кеша выше, чем может быть.

Схожий шаблон иерархии структур был также найден в задаче 429.mcf из пакета SPEC CPU2006. Для улучшения локальности данных в этой иерархии требуется:

1) разделить поля каждой из структур `node_t` и `arc_t` на часто и редко используемые и затем объединить часто используемые поля (далее будем называть их горячими) в структуры `node_hot` и `arc_hot`, а редко используемые поля (далее будем называть их холодными) — в структуры `node_cold` и `arc_cold`;

2) в структуры `node_hot` и `arc_hot` добавить по указателю на соответствующие структуры `node_cold` и `arc_cold`;

3) создать из каждого массива структур, вложенного в `network_t`, массив горячих структур и массив холодных структур;

4) все указатели на элементы преобразованных массивов структур преобразовать в указатели на соответствующие элементы соответствующего массива горячих структур.

Преобразованные структуры показаны на рис. 4.

Подобная оптимизация называется Structure Splitting и реализована в компиляторах GCC [3], Open64 [4] и LLVM [5]. Следует однако отметить, что



Рис. 3. Содержимое кеш-строки на первой итерации цикла из рис. 2

```

typedef struct node_hot
{
    ...
    struct node_hot *pred;
    ...
    struct arc_hot *basic_arc;
    ...
    struct node_cold *cold_node_ptr;
} node_hot;
typedef struct node_cold
{
    ...
    struct node_hot *child, *sibling, *sibling_prev;
    ...
    struct arc_hot *firstout, *firstin;
    ...
} node_cold;

typedef struct arc_hot
{
    node_hot *head;
    struct arc_hot *nextin;
    ...
    struct arc_cold* cold_arc_ptr;
} arc_hot;
typedef struct arc_cold
{
    node_hot *tail;
    struct arc_hot *nextout;
    ...
} arc_cold;

typedef struct network
{
    ...
    node_hot *hot_nodes;
    node_hot *stop_nodes;
    arc_hot *hot_arcs;
    arc_hot *stop_arcs;
    arc_hot *hot_dummy_arcs;
    arc_hot *stop_dummy;
    ...
    node_cold *cold_nodes;
    arc_cold *cold_arcs;
    arc_cold *cold_dummy_arcs;
} network_t;

```

Рис. 4. Преобразованная иерархия структур в задаче 181.mcf

в задаче 181.mcf в функции `resize_prob` обнаружено изменение размера одного из массивов с помощью стандартной функции `realloc` и алгоритма изменения указателей на элементы внутри этого массива. Упрощенный пример указанного алгоритма приведен на рис. 5. В силу наличия данного кода указанные реализации оптимизации Structure Splitting неприменимы. Поэтому появляется необходимость разработать новый алгоритм преобразования компилируемой программы с учетом применения процедуры `realloc`.

```

network_t *net;
arc_t *arc;
node_t *node, *stop, *root;
long off;

arc = (arc_t *) realloc( net->arcs, new_size * sizeof(arc_t) );
off = (long)arc - (long)net->arcs;

net->arcs = arc;
node = net->nodes;
for( node++, stop = (void*)net->stop_nodes; node < stop; node++ )
    node->basic_arc = (arc_t *)((long)node->basic_arc + off);

```

Рис. 5. Упрощенный пример алгоритма изменения указателей после `realloc` в задаче 181.mcf

2. Реализация

В рамках работы, результаты которой представлены в настоящей статье, авторами реализована новая версия оптимизации Structure Splitting в оптимизирующем компиляторе lcc для процессорных архитектур Эльбрус и SPARC. Алгоритм подобно описанным в работах [3, 4] состоит из двух этапов:

- 1) анализ применимости оптимизации — поиск подходящих для оптимизации массивов структур;
- 2) применение оптимизации.

2.1. Анализ применимости

Задача этапа 1 состоит в выявлении всех массивов структур и определении того, к каким из них возможно применить оптимизацию. Для возможности выявления всех искомым структур необходимо, чтобы компилятор работал в режиме сборки "вся программа", при котором все модули компилируемой программы объединяются в один.

В работе [6] описан алгоритм анализа применимости на основе профильной информации. Этот алгоритм выбирает структуры для оптимизации на основе счетчика всех обращений ко всем полям структур, суммарных счетчиков обращений к полям конкретных структур и счетчиков обращений к каждому полю. Решение по каждой структуре принимается на основе формулы, включающей все перечисленные счетчики. В ходе проделанной работы указанный алгоритм модернизирован и состоит из трех этапов.

Первый этап заключается в сборе данных обо всех структурах и полях структур в программе:

- для каждой структуры — число обращений к объектам типа этой структуры (назовем это число счетчиком структуры);
- для каждого поля каждой структуры — число обращений к этому полю (назовем это число счетчиком поля).

Полученные данные записываются в ориентированный граф *Struct Nesting Graph* (SNG). Узлы графа SNG — все найденные структуры. Если структура *A* или указатель на нее является полем *f* другой структуры *B*, то между ними проводится ребро $A \xrightarrow{f} B$.

На рис. 6 показан упрощенный пример иерархии структур. В данном примере поля `data0` и `data1` в структуре `BigStruct` — массивы структур, а все остальные поля типов `InnerStruct0*` и `InnerStruct1*` —

```

typedef struct _InnerStruct0 InnerStruct0;
typedef struct _InnerStruct1 InnerStruct1;

struct _InnerStruct0 {
    int hot0_0;
    float cold0_0;
    InnerStruct1 *hot0_1;
};

struct _InnerStruct1 {
    int hot1_0;
    float cold1_0;
    InnerStruct1 *cold1_1;
};

typedef struct _BigStruct {
    int b0;
    InnerStruct0 *data0;
    InnerStruct1 *data1;
    int b1;
} BigStruct;

typedef struct _OtherStruct {
    int o0;
    InnerStruct0 *ptr0;
    int o1;
} OtherStruct;

```

Рис. 6. Пример иерархии структур

указатели на элементы массивов `data0` и `data1`. Пример графа SNG для описанной в этом примере иерархии изображен на рис. 7.

Кроме графа, на первом этапе формируются таблицы *Field Counters Table* (FCT) и *Struct Counters Table* (SCT). FCT содержит для каждого поля его счетчик, а SCT для каждой структуры — ее счетчик.

Второй этап заключается в определении подходящих для применения оптимизации массивов структур. Для этого в графе *Struct Nesting Graph* проводится поиск всех ребер, поля на которых являются указателями. Таким образом находят поля, которые

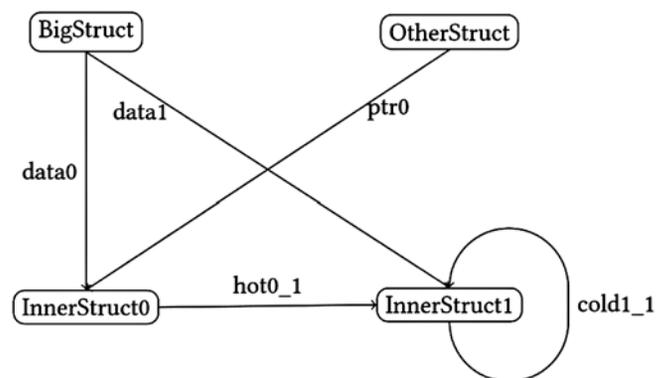


Рис. 7. Граф SNG к рис. 6

могут оказаться вложенными массивами структур. Эти поля записываются в таблицу *Fields for Optimization Table* (FOT).

Далее для каждого поля F_i в таблице FOT, соединяющего структуры A_i и B_i ($A_i \xrightarrow{F_i} B_i$) проводятся следующие действия:

- 1) определение счетчиков $c_{i,j}$ для каждого поля $f^{i,j}$ структуры B_i ;
- 2) определение максимального счетчика $C_i = \max_j \{c_{i,j}\}$ среди счетчиков полей структуры B_i ;
- 3) определение полей $f_{i,j}$, удовлетворяющих условию $c_{i,j} > \frac{C_i}{N}$ (N — некая константа), такие поля считаются горячими.

Из таблицы FOT исключаются все поля F_i ($A_i \xrightarrow{F_i} B_i$), для которых не выполнено хотя бы одно из условий:

- в программе есть вызов функции выделения памяти для поля F_i ;
- среди полей структуры B_i есть и горячие, и холодные.

Результатом анализа применимости является таблица FOT, в которой после описанных выше действий остались только те вложенные массивы структур, к которым надо применить оптимизацию Structure Splitting.

2.2. Применение оптимизации

Задача этапа 2 оптимизации Structure Splitting состоит в изменении определенных ранее типов структур, предназначенных для оптимизации, и промежуточного представления программы. Алгоритм применения оптимизации по составу этапов похож на описанные в работах [4, 6]. Он состоит из следующих стадий:

- 1) создание новой иерархии полей структур;
- 2) изменение типов переменных компилируемой программы согласно новой иерархии;
- 3) изменение доступа к переменным в компилируемой программе.

2.2.1. Создание новой иерархии полей структур

В ходе данного этапа для каждой записи в таблице FOT (поля F_i , в графе SNG, являющегося ребром $A_i \xrightarrow{F_i} B_i$), выполняются следующие действия:

- 1) структура B_i (элемент массива) разрезается на структуру с горячими полями H_i и структуру с холодными полями C_i ; в H_i добавляется указатель на C_i ; данное действие назовем декомпозицией;
- 2) в структуре A_i (т. е. охватывающей) тип поля F_i изменяется с указателя на B_i на указатель на H_i ; в конец структуры A_i добавляется указатель на C_i , имеющий смысл указателя на начало массива холодных структур;
- 3) во всех структурах N_j , для которых в графе SNG существуют ребра $N_j \xrightarrow{E_j} B_i$, типы полей E_j изменяются с указателя на B_i на указатель на H_i .

На рис. 8 показан набор структур, представленных на рис. 6, преобразованный в ходе данного этапа.

```
typedef struct _InnerStruct0_Hot InnerStruct0_Hot;
typedef struct _InnerStruct1_Hot InnerStruct1_Hot;
typedef struct _InnerStruct0_Cold InnerStruct0_Cold;
typedef struct _InnerStruct1_Cold InnerStruct1_Cold;

struct _InnerStruct0_Hot {
    int hot0_0;
    InnerStruct1_Hot *hot0_1;
    InnerStruct0_Cold *cold0_ptr;
};
struct _InnerStruct0_Cold {
    float cold0_0;
};
struct _InnerStruct1_Hot {
    int hot1_0;
    InnerStruct0_Cold *cold1_ptr;
};
struct _InnerStruct1_Cold {
    float cold1_0;
    InnerStruct1_Hot *cold1_1;
};

typedef struct _BigStruct {
    int b0;
    InnerStruct0_Hot *data0_hot;
    InnerStruct1_Hot *data1_hot;
    int b1;
    InnerStruct0_Cold *data0_cold;
    InnerStruct1_Cold *data1_cold;
} BigStruct;

typedef struct _OtherStruct {
    int o0;
    InnerStruct0_Hot *ptr0;
    int o1;
} OtherStruct;
```

Рис. 8. Преобразованный набор структур из рис. 6

2.2.2. Изменение типов переменных компилируемой программы

В ходе данного этапа все типы переменных — указателей на декомпозированные структуры — заменяются на указатели на созданные при декомпозиции горячие структуры.

2.2.3. Изменение доступа к переменным в компилируемой программе

В ходе данного этапа выполняется преобразование промежуточного представления программы с изменением всех обращений к декомпозированным массивам и их элементам. Обращения бывают нескольких категорий. Для каждой категории ниже приведен пример оригинального и преобразованного обращения. В представленных примерах:

- `data` — указатель на начало массива до преобразования иерархии структур; `data_hot` и `data_cold` — указатели на начала горячего и холодного массивов после преобразования иерархии структур (для краткости будем называть эти указатели горячим и холодным соответственно);
- `ptr` — некий указатель на элемент массива до преобразования переменных программы; `ptr_hot` —

тот же указатель после преобразования переменных программы;

- `field_hot` и `field_cold` — горячее и холодное поля элемента массива, к которому осуществляется доступ;
- `data_elem_cold` — созданный внутри горячей структуры в ходе преобразования иерархии структур указатель на соответствующую холодную структуру;
- `s` — число элементов в массиве; `s_new` — число элементов в массиве после вызова `realloc`;
- `ArrElem` — структура элемента массива; `ArrElemHot` и `ArrElemCold` — структуры элементов горячего и холодного массивов, созданные из `ArrElem` в ходе преобразования иерархии структур.

Отметим категории обращений и соответствующие им преобразования.

1. Работа с полем элемента массива:

1.1) доступ к горячему полю элемента массива с помощью указателя на начало массива и индекса `data[index].field_hot`; преобразование состоит в изменении указателя на начало массива: `data_hot[index].field_hot`;

1.2) доступ к горячему полю элемента массива с помощью указателя на элемент массива `ptr->field_hot`; преобразование состоит в изменении указателя на элемент массива: `ptr_hot`;

1.3) доступ к холодному полю элемента массива с помощью указателя на начало массива и индекса `data[index].field_cold`; преобразование состоит в изменении указателя на начало массива: `data_cold[index].field_cold`;

1.4) доступ к холодному полю элемента массива с помощью указателя на элемент массива `ptr->field_cold`; преобразование состоит в изменении указателя на элемент массива и получении указателя на холодный элемент массива из указателя на горячий элемент массива: `ptr->data_elem_cold->field_cold`.

2. Работа с самим массивом:

2.1) выделение или изменение размера выделенной для массива памяти: `data = (ArrElem*) malloc(s*sizeof(ArrElem))`; преобразование состоит в изменении размера выделяемой памяти для двух созданных массивов `data_hot` и `data_cold`, создании операции вычисления указателя на начало холодного массива и создании цикла с вычислением указателей на холодные элементы массива для записи в горячие; для `realloc` добавляется перемещение данных холодного массива по новому адресу; пример преобразованного выделения памяти показан на рис. 9 и 10; для сохранения корректности указателей во время алгоритма после `realloc`, описанного на рис. 5, вместо отдельного выделения памяти для обоих созданных массивов требуется выделить для них одну область памяти и разметить одну часть для горячего массива, а другую — для холодного;

```
size_t new_sizeof = sizeof(ArrElemHot) + sizeof(ArrElemCold);
data_hot = (ArrElemHot*) malloc(s * new_sizeof);
data_cold = (ArrElemCold*)( data_hot + s );
for ( i = 0; i < s; i++ ) {
    data_hot[i]->data_elem_cold = &data_cold[i];
}
```

Рис. 9. Преобразованное выделение памяти для массива

```
size_t new_sizeof = sizeof(ArrElemHot) + sizeof(ArrElemCold);
data_hot = (ArrElemHot*) realloc(data_hot, s_new * new_sizeof);
ArrElemCold* old_data_cold = (ArrElemCold*)( data_hot + s );
data_cold = (ArrElemCold*)( data_hot + s_new );
memmove( data_cold, old_data_cold, s*sizeof(ArrElemCold) );
for ( i = 0; i < s_new; i++ ) {
    data_hot[i]->data_elem_cold = &data_cold[i];
}
```

Рис. 10. Преобразованное перевыделение памяти для массива

2.2) освобождение памяти для массива: `free(data)`; преобразование состоит в изменении подаваемого указателя `free` на указатель на начало горячего массива: `free(data_hot)`; так как для обоих массивов выделяется одна область памяти, достаточно вызвать функцию освобождения памяти для начала этой области, в данном случае начало области совпадает с началом горячего массива;

2.3) сравнение указателя на начало массива с нулем: `data == 0`; преобразование состоит в изменении сравниваемого указателя на горячий: `data_hot == 0`;

2.4) присвоение нуля указателю на начало массива: `data = 0`; преобразование состоит в присвоении нуля и горячему, и холодному указателям на начала массивов: `data_hot = 0`; `data_cold = 0`.

3. Результаты измерений

Разработана оптимизация Structure Splitting. Измерения были проведены на компьютере с процессором Эльбрус-4С частотой 750 МГц с архитектурой системы команд типа VLIW [7].

Реализованную оптимизацию удалось применить к нескольким задачам.

1. Задачи 181.mcf (SPEC CPU2000) и 429.mcf (SPEC CPU2006), представляющие собой реализации симплекс-метода оптимизации расписания движения транспорта, созданные А. Лёбелем (институт Цузе) для оптимизации транспортных систем Берлина и иных городов [8].

2. GeoNoise&Water — авторская реализация алгоритма генерации случайных карт "diamond-square" и наивного алгоритма моделирования водной эрозии на сгенерированной карте.

При применении реализованной оптимизации к задаче 181.mcf удалось получить ускорение выполнения на 26 %. Ускорение произошло вследствие уменьшения числа блокировок конвейера процессора для ожидания загрузки данных из памяти на 27 %.

Задача 429.mcf после применения оптимизации ускорилась на 14 %. Ускорение произошло в силу уменьшения числа блокировок конвейера процессора для ожидания загрузки данных из памяти на 19 %.

Задача GeoNoise&Water после применения оптимизации ускорилась на 4 %. Ускорение произошло вследствие уменьшения числа блокировок конвейера процессора для ожидания загрузки данных из памяти на 54 %.

4. Работы по смежной тематике

В работе [3] описан иной алгоритм оптимизации Structure Splitting, выполненной для компилятора GCC.

Отличия целей разработки указанной оптимизации от целей данной работы состоят в следующем:

- перед версией для GCC не стоит задачи обеспечения работы при изменении размера массива, поэтому в ходе оптимизации обрабатываются только стандартные функции выделения и освобождения памяти для массива;

- не обрабатываются случаи, когда указатели на элементы массива структур могут лежать в других структурах.

Описанный в работе [3] алгоритм состоит из двух этапов:

- 1) определение необходимости применения оптимизации;

- 2) применение оптимизации.

Определение необходимости проведения оптимизации проводится по составленным для каждой процедуры графам *Field Reference Graph* (FRG) и *Close Proximity Graph* (CPG). FRG — ориентированный граф, узлами которого являются операции доступа к полям исследуемой структуры, а дуги определяют порядок доступа к полям. Каждой дуге соответствует счетчик, взятый из профилированного графа потока управления процедуры, и дистанция — число доступов к памяти между доступом к полям. Второй граф является упрощенным графом FRG, в котором рассматриваются только дуги с дистанцией, меньшей некоторой определенной заранее. Также в CPG по сравнению с FRG множественные дуги между узлами сведены к единичным. На основе анализа графа CPG выносится решение о необходимости применения оптимизации.

Применение Structure Splitting в версии GCC состоит из следующих этапов:

- 1) создание новой иерархии полей структур;

- 2) изменение типов переменных компилируемой программы согласно новой иерархии;

- 3) изменение доступа к полям согласно новой иерархии;

- 4) изменение обращений к функциям выделения и освобождения памяти;

- 5) изменение доступа к элементам преобразованного массива согласно новой иерархии.

Способ оптимизации Structure Splitting в указанном выше варианте применен в компиляторе GCC версии 4.3 [8]. В версии 4.8 эта оптимизация была убрана на доработку из-за вопросов, связанных с надежностью.

В работе [6] предлагается другой способ анализа применимости оптимизации для программ на языке Java. Он состоит в подсчете полных счетчиков обращений к полям и структурам и определении горячих полей на основе:

- S — счетчика обращений к структуре элемента массива;

- N — числа полей в элементе массива;

- s_i — счетчиков обращений к полю i элемента массива.

Горячими считаются поля, для которых $s_i > \frac{S}{2N}$.

Немного отличается от описанного в статье [6] анализ применимости, описанный в работе [4]. В нем

применены профили каждого узла графа потока управления программы.

Предложенный алгоритм анализа состоит из следующих этапов для каждой структуры.

1. Для каждого горячего (т. е. с большим счетчиком выполнения) узла выяснение, к каким полям исследуемой структуры проводятся обращения и сколько раз.

2. Разделение полей исследуемой структуры на группу горячих и группу холодных согласно определенным на предыдущем этапе данным.

Применение оптимизации, описанное в работе [4], не отличается по принципу работы от описанного в работе [2].

Оптимизация Structure Splitting в варианте, описанном в работе [4], применена в компиляторе Open64.

В работе [9] описан полностью иной подход к Structure Splitting, реализованный с помощью отдельной программы, названной *On-the-fly Structure Splitting* (OSS). Подход состоит в профилировании исполняемого файла с перехватом обращений программы для выделения и освобождения памяти, в анализе и внесении изменений в исполняемый файл.

Заключение

Разработана и отлаживается новая версия оптимизации Structure Splitting, обобщенная на случай изменения размера массива структур с применением алгоритма сохранения корректности указателей на элементы измененного массива.

Достигнуто ускорение работы теста 181.mcf на 19 %. Задача 429.mcf ускорилась на 12 %, хотя при ручном применении оптимизации удалось достигнуть ускорения данной задачи на 20 %.

Дальнейшие направления работы заключаются в подборе констант для определения горячих и холодных полей, чтобы сделать ускорение кода максимальным и исключить случаи замедления кода вследствие применения оптимизации.

Список литературы

1. Шампаров В. Е., Маркин А. Л. Преобразование компилятором массива структур в несколько массивов // Труды 61-й Всероссийской научной конференции МФТИ. Радиотехника и компьютерные технологии. МФТИ. — 2018. — С. 9—10.

2. SPEC CPU. URL: <https://www.spec.org/benchmarks.html#cpu>

3. Hagos M., Tice C. Cache Aware Data Layout Reorganization Optimization in GCC // Proceedings of the GCC Developers' Summit. Ottawa, Ontario. — 2005. — P. 69—92.

4. Chakrabarti G., Chow F. Structure layout optimizations in the Open64 compiler: Design, implementation and measurements // Open64 workshop held in conjunction with the international symposium on code generation and optimization, 2008. URL: <https://www.capsl.udel.edu/conferences/open64/2008/Papers/111.pdf>

5. Prashantha N. R., Vikram T. V., Vaivaswatha N. Implementing Data Layout Optimizations in the LLVM Framework. 2014. URL: <http://llvm.org/devmtg/2014-10/Slides/Prashanth-DLO.pdf>

6. Chilimbi T., Davidson B., Larus J. R. Cache-Conscious Structure Definition // Proceedings of the ACM SIGPLAN'99 Con-

ference on Programming Language Design and Implementation. ACM. — 1999. — P. 13–24.

7. **Kim A. K., Perekatov V. I., Ermakov S. G.** Микропроцессоры и вычислительные комплексы семейства "Эльбрус". — СПб.: Питер, 2013. — 272 с.

8. **Loebel A.** Optimal Vehicle Scheduling in Public Transit: Ph.D. thesis. — Berlin, 1997. — 185 p.

9. **Golovanevsky O., Zaks A.** Struct-reorg: current status and future perspectives // Proceedings of the GCC Developers' Summit. Ottawa. — Ontario, 2007. — P. 47–56.

10. **Wang Zh., Wu Ch., Yew P.-Ch., Li J., Xu D.** On-the-Fly Structure Splitting for Heap Objects ACM Trans. Archit. Code Optim. — 2012. — Vol. 8, No. 4. — Article 26. URL: <https://doi.org/10.1145/2086696.2086705>

Structure Splitting for Elbrus Processor Compiler

V. E. Shamparov, Victor.E.Shamparov@mcst.ru, **A. L. Markin**, Alex.L. Markin@mcst.ru, MCST, Moscow, 117105, Russian Federation, Moscow Institute of Physics and Technology, Moscow Region, 141701, Russian Federation

Corresponding author:

Shamparov Viktor E., Software Engineer, MCST, Moscow, 117105, Russian Federation
E-mail: Victor.E. Shamparov@mcst.ru

*Received on October 13, 2020
Accepted on November 21, 2020*

This report presents a new version of Structure Splitting optimization, implemented for the compiler for Elbrus and SPARC processors. Structure Splitting tries to improve data locality by splitting arrays of structures into arrays of smaller structures. This solution helps to decrease probability of cache misses, which leads to execution time decrease. The optimization was generalized for the case of array of structures nested in another structure and possibility of its reallocation. Execution speed of two tests from SPEC CPU2000 and SPEC CPU2006 increased by 19 and 12 %.

Keywords: compiler, optimization, Structure Splitting, Elbrus, SPARC

For citation:

Shamparov V. E., Markin A. L. Structure Splitting for Elbrus Processor Compiler, *Programmnyaya Ingeneria*, 2021, vol. 12, no. 2, pp. 82–88

DOI: 10.17587/prin.12.82-88

References

1. **Shamparov V. E., Markin A. L.** Compiler transformation of array of structures into arrays, *Trudy 61-j Vserossijskoj nauchnoj konferencii MFTI. Radiotekhnika i komp'yuternye tekhnologii*, 2018, pp. 9–10 (in Russian).
2. **SPEC CPU**, available at: <https://www.spec.org/benchmarks.html#cpu>
3. **Hagog M., Tice C.** Cache Aware Data Layout Reorganization Optimization in GCC, *Proceedings of the GCC Developers' Summit*, Ottawa, Ontario, 2005, pp. 69–92.
4. **Chakrabarti G., Chow F.** Structure layout optimizations in the Open64 compiler: Design, implementation and measurements, *Open64 workshop held in conjunction with the international symposium on code generation and optimization*, 2008, available at: <https://www.capsl.udel.edu/conferences/open64/2008/Papers/111.pdf>
5. **Prashantha N. R., Vikram T. V., Vaivaswatha N.** Implementing Data Layout Optimizations in the LLVM Framework in

the LLVM Framework, 2014, available at: <http://llvm.org/devmtg/2014-10/Slides/Prashanth-DLO.pdf>

6. **Chilimbi T., Davidson B., Larus J. R.** Cache-Conscious Structure Definition, *Proceedings of the ACM SIGPLAN'99 Conference on Programming Language Design and Implementation*, ACM, 1999, pp. 13–24.

7. **Kim A. K., Perekatov V. I., Ermakov S. G.** *Mikroprocessory i vychislitel'nye kompleksy semejstva "El'brus"*, Saint-Petersburg, Piter, 2013, 272 p. (in Russian).

8. **Loebel A.** Optimal Vehicle Scheduling in Public Transit: Ph.D. thesis. Berlin, 1997, 185 p.

9. **Golovanevsky O., Zaks A.** Struct-reorg: current status and future perspectives, *Proceedings of the GCC Developers' Summit*, 2007, pp. 47–56.

10. **Wang Zh., Wu Ch., Yew P.-Ch., Li J., Xu D.** On-the-Fly Structure Splitting for Heap Objects, *ACM Trans. Archit. Code Optim.*, 2012, vol. 8, no. 4, article 26.

В. Г. Луцик, д-р техн. наук, вед. науч. сотр., vgl_41@mail.ru,
М. С. Макарова, канд. техн. наук, науч. сотр., april27_86@mail.ru,
А. И. Решмин, канд. техн. наук, вед. науч. сотр., alexreshmin@rambler.ru,
Научно-исследовательский институт механики МГУ имени М. В. Ломоносова

Применение трехпараметрической дифференциальной модели турбулентности для решения задач течения и теплообмена в каналах переменного сечения. Часть 2

Представлено описание методики численного исследования в приближении узкого канала задач течения и теплообмена в плоских и круглых каналах переменного сечения с использованием дифференциальной трехпараметрической модели сдвиговой турбулентности. Описаны основные результаты многочисленных исследований с использованием предложенной методики, одной из целей проведения которых стало обоснование возможности использования приближения узкого канала. Настоящее обзорное исследование выполнено в двух частях. В настоящей части обзора приведены результаты исследования ламинаризации потока при течении в конфузоре и трубе, интенсификации теплообмена при течении в диффузорах и в пластинчатом теплообменнике с диффузорными каналами.

Ключевые слова: дифференциальная модель турбулентности, диффузор, конфузор, градиент давления, вдув, отсос, ламинаризация течения, интенсификация теплообмена

Введение

Данная работа является продолжением обзорной статьи [1], в которой была подробно описана методика расчета течения и теплообмена в каналах постоянного и переменного сечения с использованием дифференциальной трехпараметрической модели сдвиговой турбулентности [2]. Представлено подробное описание вычислительной методики, приведены основные уравнения, характеризующие течение в каналах, а также уравнения и константы используемой авторами модели турбулентности [3–5]. Приведены основные результаты по исследованию смешанной конвекции в вертикальных трубах в условиях устойчивой и неустойчивой стратификации [6, 7], а также течений в каналах с проницаемыми стенками при наличии вдува или отсоса на стенке [8–10].

Часть 2 данной работы представляет собой подробный обзор полученных на сегодняшний день с использованием предложенной методики результатов численного моделирования задач о ламинаризации потока при течении в конфузоре и трубе, интенсификации теплообмена при течении в диффузорах и в пластинчатом теплообменнике с диффузорными каналами.

1. Методика расчета

В части 1 исследования [1] была подробно описана методика численного исследования задач течения и теплообмена в плоских и круглых каналах переменного сечения в приближении узкого канала.

Система уравнений неразрывности, движения и энергии описывает существенно дозвуковое (число Маха $M \ll 1$) течение, как в [1]. Для вычисления величин τ и q_T , входящих в уравнения движения и энергии, использована трехпараметрическая модель турбулентности [3], обобщенная на течение с теплообменом [5], в которой уравнения переноса записываются для энергии турбулентности $E = 0,5 \sum_{i=1}^2 \langle u_i'^2 \rangle$, напря-

жения сдвига $\tau = -\langle u'v' \rangle$ и предложенного А. Н. Колмогоровым параметра $\omega = E/L^2$ (L — поперечный интегральный масштаб турбулентности), а также уравнение переноса для $q_T = -c_p \langle v'T' \rangle$ [5].

Основные уравнения модели и константы приведены в части 1 [1].

Граничные условия на стенках, на оси симметрии и на входе в круглый (плоский) канал будут формулироваться для каждой из рассмотренных ниже задач.

2. Результаты численного моделирования

Ниже приведены основные результаты исследования ламинаризации потока при течении в конфузоре и трубе, интенсификации теплообмена при течении в диффузорах и в пластинчатом теплообменнике с диффузорными каналами.

2.1. Ламинаризация потока при течении в конфузоре и трубе

Продольный градиент давления является параметром, который оказывает существенное влияние

на турбулентное течение, приводя в пределе в случае большого отрицательного градиента давления к ламинаризации пограничного слоя.

Эффект ламинаризации (или реламинаризации) исходного турбулентного течения в пограничных слоях в случае внешних течений с отрицательным градиентом давления был обнаружен Штернбергом в 1954 г. [11]. Впоследствии исследованию процессов ламинаризации уделялось значительное внимание и на сегодняшний день имеется обширная литература (см., например, обзоры [4, 12–14]). Известно несколько локальных критериев параметра ускорения, которые были предложены для определения необходимого градиента скорости (или давления), обеспечивающего реламинаризацию [15–17]:

$$K = \frac{\nu}{u_e^2} \frac{du_e}{dx} > 3 \cdot 10^{-6}, \quad \Delta p = \frac{\nu}{\rho u_*^2} \frac{dp}{dx} > 0,02, \quad (1)$$

где u_e — скорость набегающего потока; u_* — динамическая скорость; ν — кинематическая вязкость; p — давление; ρ — плотность.

В работе [18] показано, что указанных локальных критериев параметра ускорения недостаточно, чтобы гарантировать реламинаризацию пограничного слоя. Для полной реламинаризации требуется [18], чтобы локальное число Рейнольдса $Re_\theta = u_e \theta / \nu$, определенное по толщине потери импульса θ , стало меньше критического $Re_\theta = 200 \dots 300$. Значение Re_θ , как следует из работы [12], станет меньше критического, если протяженность участка, на котором воздействует градиент давления, превышает длину релаксации характеристик турбулентности, зависящую от предыстории течения. Поэтому критериями типа (1) или их комбинациями нельзя однозначно предсказать реламинаризацию турбулентного пограничного слоя [12]. Так, в работе [4] путем численного моделирования получено, что для значения величины $K = 4 \cdot 10^{-6}$ число Re_θ заметно убывает по длине, приближаясь к критическому значению $Re_\theta = 200$, что свидетельствует о тенденции к ламинаризации пограничного слоя при сильном ускорении потока, однако полная ламинаризация на расчетной длине получена не была.

Тем не менее во многих работах (см., например, [19, 20]) критерии типа (1) или их комбинации используются, и значения их уточняются при определении условий реламинаризации турбулентных пристеночных пограничных слоев. Так, в работе [20], где представлен обзор работ по исследованию влияния отрицательного градиента давления на течение в пограничном слое, перечислены параметры реламинаризации, экспериментально получено значение параметра ускорения $K = 4,4 \cdot 10^{-6}$, обеспечивающего реламинаризацию турбулентного пограничного слоя.

Исследованию ламинаризации внутренних турбулентных течений посвящено значительно меньше работ по сравнению с исследованием для случая внешних течений. Так, достаточно подробно исследовано течение в трубах с сильным подогревом теплоносителя, создающим большой отрицательный градиент давления [21]. В работе [21] предполагается, что

начало ламинаризации в обогреваемой трубе можно определять также с помощью параметра ускорения K (1), в котором вместо скорости внешнего течения u_e подставляется средняя скорость течения в трубе U .

Что касается течения в конфузорах, то оно исследовано еще меньше. Так, в работе [22] проведено экспериментальное исследование течения и теплообмена в круглом конфузоре с углом сужения 8° . В работах [23, 24] исследованы течение и теплообмен в двумерном канале с плоским конфузуром. Детальный обзор опубликованных экспериментальных работ по реламинаризации течения в плоских конфузорах показывает, что в итоге большая их часть сводится к исследованию задачи внешнего обтекания плоской пластины при наличии отрицательного градиента давления, который, в свою очередь, задается с помощью сужения конфузурного канала [25–27]. При этом основной особенностью течения в канале с плоским конфузуром является то обстоятельство, что число Рейнольдса $Re = d_h U / \nu$, определенное по средней скорости течения U и гидравлическому диаметру d_h , равному двойной высоте конфузора h , в случае двумерного течения остается постоянным по длине канала. Это позволяет записать параметр ускорения потока K (1) в виде

$$K = \frac{\nu}{U^2} \frac{dU}{dx} = \frac{2\beta}{Re}.$$

где $\beta = -dh/dx = \operatorname{tg} \alpha$; α — угол наклона верхней стенки конфузора.

В работе [24] при исследовании влияния ламинаризации потока на теплообмен при течении в плоском конфузуре экспериментально установлено, что на выходе из конфузора число Нуссельта значительно меньше числа Нуссельта для развитого турбулентного течения при том же числе Рейнольдса, что свидетельствует о ламинаризации течения. Однако полной ламинаризации течения для чисел Рейнольдса $Re = 7300$ и $10\ 500$ на длине конфузора 200 мм даже при $K = 10^{-5}$ достигнуто не было [24], поскольку число Нуссельта на выходе из конфузора было заметно больше ламинарного значения.

В работе [25] экспериментально исследовано течение в плоском конфузуре с углом сужения 9° для числа Рейнольдса, определенного по гидравлическому диаметру конфузора, $Re = 10^5$. При этом параметр ускорения потока составлял $K = 3,1 \cdot 10^{-6}$. Были детально измерены осредненные профили скоростей, проведено их сравнение с известным автомодельным решением для ламинарного пограничного слоя [28]. Показано, что повышенный уровень турбулентности на входе в конфузурный участок не оказывает существенного влияния на процессы реламинаризации.

Численному исследованию ускоренных пограничных слоев уделяется существенно больше внимания, нежели экспериментальному. При этом исследования в первую очередь сводятся к проблеме выбора модели турбулентности, которая может адекватно описать процессы реламинаризации. Так, в работе [29] выполнено моделирование течений с отрицательным градиентом давления с использованием линейных и нелинейных RANS-моделей. Диапазон

изменения параметра ускорения при этом составлял $K = (1,5...3,0) \cdot 10^{-6}$. Показано, что лучшие из однопараметрических и двухпараметрических моделей турбулентности, рассмотренных в работе [29], в их стандартных формах нечувствительны к ускорению потока. Модификации этих моделей, предложенные их авторами, позволили улучшить результаты.

2.1.1. Ламинаризация потока при течении с теплообменом в плоском канале с конфузуром

Ниже приведены результаты численного моделирования течения с отрицательным градиентом давления в условиях внутренней задачи — в плоском канале с конфузуром при постоянном числе Рейнольдса [30].

Граничные условия на стенках канала ($y = 0, h$) и на входе ($x = x_{in}$) для рассматриваемой задачи имеют следующий вид:

$$\begin{aligned} u = v = 0, \quad -\left(\lambda \frac{\partial T}{\partial n}\right) &= 0, \\ E = \frac{\partial \omega}{\partial n} = \tau = q_T = 0 \quad (y = 0), \\ u = v = 0, \quad -\left(\lambda \frac{\partial T}{\partial n}\right) &= q_w, \\ E = \frac{\partial \omega}{\partial n} = \tau = q_T = 0 \quad (y = h), \\ p = p_1, \quad u = U_1, \quad T = T_1, \quad E = E_1, \\ \tau = 0, \quad \omega = \omega_1, \quad q_T = 0 \quad (x = x_{in}). \end{aligned} \quad (2)$$

В качестве граничных условий на входе используются постоянные по сечению профили скорости, температуры и характеристик турбулентности.

Задача решалась в следующей постановке (рис. 1). Параметры канала соответствовали принятым в работе [24]. Входу в конфузор предшествовал участок стабилизации длиной $L_1 = 1200$ мм, на котором устанавливалось развитое турбулентное течение при заданном числе Рейнольдса ($L_1/2h_1 = 40$, где $h_1 = 15$ мм — входная высота канала). Далее следовал конфузор длиной $L_k = 200$ мм с горизонтальной нижней стенкой и плоской наклонной верхней стенкой с линейно изменяющейся по длине высотой $h(x)$, на выходе равной $h_2 = h_1 + L_k \tan \alpha$. Отметим, что число Рейнольдса $Re = Ud_h/\nu$, определенное по гидравлическому диаметру $d_h = 2h$ и среднерасходной скорости U , остается постоянным по всей длине конфузора. За конфузуром следовал участок постоянного сечения $L_2 = 700$ мм и высотой, равной высоте конфузора на выходе h_2 , на котором восстанавливалось тече-

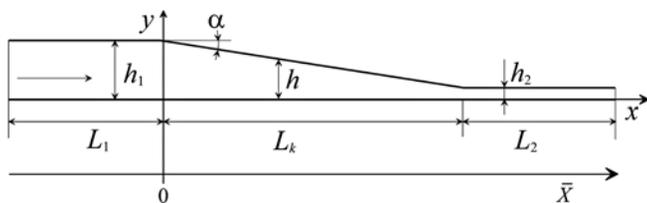


Рис. 1. Расчетная схема

ние, соответствующее заданному числу Рейнольдса. По всей длине рабочего участка осуществлялся подогрев воздуха нагревателем с постоянным тепловым потоком q_w в стенку. Значение q_w задавалось достаточно малым, чтобы не учитывать влияние тепловой конвекции.

Рассчитывались параметры течений при числах Рейнольдса $Re = 7300$ и $10\,500$. При определении числа Рейнольдса в качестве характерного размера принимался гидравлический диаметр, который для плоского конфузора равен двойной высоте канала. Результаты численного исследования, проведенного для условий, соответствующих эксперименту [24], согласуются с экспериментальными данными по числу Нуссельта на длине конфузора ($x = 0...200$ мм). В канале постоянного сечения, следующим за конфузуром, происходит релаксация характеристик течения и теплообмена. Поведение их носит сложный характер и по числу Нуссельта соответствует эксперименту [24] имеет качественный характер.

Признаком ламинаризации исходного турбулентного течения будем считать выход на режим течения, соответствующий ламинарному, при котором число

Нуссельта (при $q_w = \text{const}$) $Nu = \frac{d_h q_w}{\lambda \Delta T} = 5,38$; коэффициент

трения $\lambda = 8\tau_w/\rho U^2 = 96/Re$ ($\tau_w = -(\eta \partial u / \partial n)_w$, η — динамическая вязкость); скорость на средней линии канала ($y = h/2$) $u_0/U = 1,5$; энергия турбулентности E во всем сечении потока близка к нулю.

Проведенное численное исследование показало, что для рассмотренного конфузора длиной $L_k = 200$ мм при числах Рейнольдса $Re = 7300$ и $10\,500$ ламинаризация наступает при значении параметра $K > 1 \cdot 10^{-5}$. Дальнейшее уменьшение параметра K , при котором наступает ламинаризация, связано с увеличением длины конфузора, т. е. возрастанием времени воздействия отрицательного градиента давления на течение. В подтверждение сказанного, на рис. 2 для $Re = 7300$ представлено изменение числа Нуссельта Nu (а) и максимальной интенсивности турбулентности $e_m = \sqrt{E_{\max}}/U$ (б) для конфузоров длиной $L_k = 200, 300$ и 400 мм при одном и том же наклоне стенки конфузора ($\alpha = 1,7^\circ$), т. е. при одинаковом значении параметра $K = 8,2 \cdot 10^{-6}$.

Как видно из рис. 2, при значении $K = 8,2 \cdot 10^{-6}$ ламинаризация наступает в конфузуре длиной $L_k = 400$ мм. Можно предположить, что дальнейшего

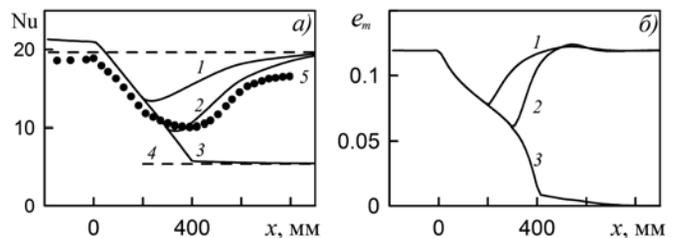


Рис. 2. Изменение по длине числа Нуссельта Nu (а) и максимальной интенсивности турбулентности $e_m = \sqrt{E_{\max}}/U$ (б) при значениях числа Рейнольдса $Re = 7300$ и параметра $K = 8,2 \cdot 10^{-6}$ для конфузоров длиной $L_k = 200$ (линия 1), 300 (линия 2) и 400 мм (линия 3); линия 4 — $Nu = 5,38$; точки 5 — экспериментальные данные [24] для $L_k = 200$ мм

уменьшения параметра ламинаризации можно достичь, увеличивая длину конфузора.

Полная ламинаризация течения, т. е. вырождение турбулентности, происходит, если на выходе из конфузора энергия турбулентности опускается ниже некоторого критического уровня. При этом ее дальнейшее уменьшение происходит уже в канале постоянной высоты. Число Рейнольдса, при котором наступает полная ламинаризация (Re^*), зависит от параметров конфузора.

Были проведены расчеты зависимости Re^* от относительной длины конфузора $L/L_{max} = 1 - h_2/h_1$, где L_{max} — максимально возможная длина конфузора, соответствующая предельному значению $h_2 \rightarrow 0$. Показано, что при $L/L_{max} > 0,95$ ($h_2/h_1 < 0,05$) значение Re^* практически не менялось. При заданном значении угла сужения конфузора эта величина определяет наибольшее значение числа Re , при котором будет иметь место полная ламинаризация потока, если длина конфузора будет близка к максимальной, что соответствует времени воздействия отрицательного градиента давления, близкого к максимальному.

Представляет интерес оценить значение параметра ускорения потока K для условий полной ламинаризации при течении в плоском конфузоре. На рис. 3 представлена зависимость параметра K от относительной длины конфузора L/L_{max} для ряда значений тангенса угла сужения конфузора β .

Как видно на рис. 3, при $L/L_{max} > 0,95$ для $\beta > 0,01$ значение параметра ламинаризации $K = 2\beta/Re^* \approx 6 \cdot 10^{-6}$ (линия 6). В остальных случаях параметр K не является константой и зависит от угла сужения конфузора α и его длины.

Таким образом, для внутреннего течения (в частном случае плоского конфузора), как и для внешнего течения, параметр ускорения потока вида $K = (v/U^2)(dU/dx)$ не позволяет в общем случае определить условие полной ламинаризации течения. Это можно сделать только в случае достаточного времени воздействия градиента давления на течение, и значение параметра ускорения при этом будет составлять $K = 6 \cdot 10^{-6}$.

Проведенное сравнение результатов расчета с экспериментальными данными [24] показало, что используемая модель турбулентности адекватно описывает течение с теплообменом в плоском конфузоре, включая процессы ламинаризации потока и последующего восстановления турбулентности. Однако полной ламинаризации течения как в эксперимен-

тах [24], так и в расчетах при параметрах, соответствующих эксперименту [24], получено не было.

Полная ламинаризация течения, т. е. вырождение турбулентности, происходит в том случае, если на выходе из конфузора энергия турбулентности опускается ниже некоторого критического уровня. При этом ее дальнейшее уменьшение происходит уже в канале постоянной высоты. Число Рейнольдса, при котором наступает полная ламинаризация (Re^*), зависит от параметров конфузора.

Показано, что для плоского конфузора только в случае достаточного времени воздействия градиента давления на течение значение параметра ускорения потока вида $K = (v/U^2)(dU/dx)$ может быть использовано для определения условия полной ламинаризации течения. Это значение будет составлять $K = 6 \cdot 10^{-6}$.

2.1.2. Реламинаризация потока при течении в трубе

Потери давления вследствие трения при турбулентном режиме течения, как отмечено в работе [31], связаны с ростом потребления энергии на прокачку газа или жидкости по трубам и, как следствие, с возрастанием эксплуатационных расходов. В случае реламинаризации (ламинаризации) течения, т. е. при переходе от турбулентного режима течения к ламинарному, потери из-за трения могут быть существенно снижены. Так, например, для течения с числом Рейнольдса 25 000 эти потери будут уменьшены на порядок.

В работе [31] представлен обзор ряда способов реламинаризации турбулентного течения в трубах. Не останавливаясь на всех способах реламинаризации, упомянутых в [31], где дан их достаточно подробный анализ, рассмотрим лишь те, которые имеют отношение к тематике настоящей статьи. В частности, в работах [32, 33] для управления турбулентным потоком использовались сетки и хонейкомбы. Полученные в этих работах результаты показали, что с помощью использованных сеток и хонейкомбов уровень турбулентности можно уменьшить, но реламинаризации течения достичь не удастся. Следует отметить, что в работах [34—36] при исследовании перехода к турбулентности в затопленных и спутных струях и в трубе [37] также использовались хонейкомбы для формирования течений с заданным входным профилем скорости и турбулентностью малой интенсивности и масштаба. В работах [38, 39] для формирования ламинарных затопленных струй использовалось устройство, позволяющее создавать в начальном сечении струи диаметром 0,12 м гладкий выпуклый профиль скорости с интенсивностью турбулентности 0,6 % при числах Рейнольдса выше 10 000. Низкий уровень турбулентности и требуемый профиль скорости формируются независимо. Для создания профиля скорости используется короткий диффузор с большой степенью расширения и пронизываемой перегородкой в выходном сечении [40].

В исследовании, описанном в работе [31], использовали два устройства формирования профиля скорости на входе в участок реламинаризации турбулентного потока. Как отмечено в [31], оба устройства

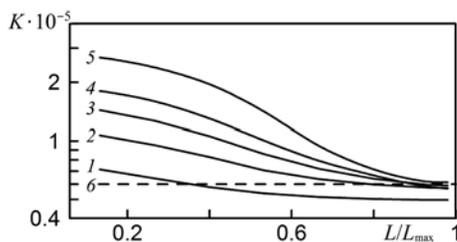


Рис. 3. Зависимость параметра K от L/L_{max} для ряда значений β : 1 — $\beta = 0,008$; 2 — $\beta = 0,012$; 3 — $\beta = 0,016$; 4 — $\beta = 0,020$; 5 — $\beta = 0,029$; 6 — $K = 6 \cdot 10^{-6}$

изменяют профиль скорости потока таким образом, что поток в центре трубы замедляется, а поток вблизи стенки ускоряется. Первым способом в работе [31] экспериментально получена реламинаризация турбулентного течения при $Re = 3800$, при втором способе — при $Re = 6000$. Увеличение числа Рейнольдса реламинаризации при втором способе управления потоком можно объяснить более низким уровнем турбулентности в центральном потоке по сравнению с уровнем турбулентности, генерируемым решеткой в первом способе. Дальнейшие исследования авторов [31] показали [41], что модификация профиля скорости в трубе несколькими способами может привести к подавлению турбулентности и поток может быть реламинирован и при более высоких, чем в [31], числах Рейнольдса. Таким образом, анализ полученных в работе [31] результатов экспериментального исследования позволяет сделать вывод, что для достижения более высокого числа Рейнольдса, при котором наступает реламинаризация, необходимо сформировать поток не только с заданным профилем скорости, но и с низким уровнем турбулентности. Этот вывод конечно же нельзя считать новым, поскольку в обзоре [42] приведены результаты экспериментальных работ, в которых режим течения оставался ламинарным в трубах на достаточно большой длине при низком уровне турбулентности на входе при числе Рейнольдса вплоть до $Re = 10^5$.

Численное исследование [43], проведенное с использованием дифференциальной модели турбулентности [3], также показало, что переход к турбулентному режиму течения в трубах с однородным профилем скорости при низком уровне турбулентности на входе может быть получен при числах Рейнольдса $Re > 10^5$. Эти результаты можно интерпретировать как возможность достижения реламинаризации при высоких числах Рейнольдса за счет низкого уровня турбулентности на входе.

В работах [4, 44] представлены результаты экспериментального и численного исследования нового класса течений с неоднородными (трапециевидным, U-образным и Λ-образным) профилями скорости и мелкомасштабной турбулентностью на входе, сформированными входными устройствами с переменным гидравлическим сопротивлением типа хонейкомбов, размер ячейки которых много меньше толщины слоя с градиентом скорости. Особенностью таких течений является то обстоятельство, что в них на значительной длине интенсивность процессов турбулентного переноса определяется энергией и масштабом турбулентности, генерируемыми входными устройствами, формирующими профиль скорости. Во всех типах этих течений было установлено, что на участке роста энергии турбулентности коэффициенты турбулентного переноса были существенно ниже, нежели на последующем участке развитой турбулентности. Для расчета неравновесных турбулентных течений, представленных в работах [4, 44], использовалась трехпараметрическая дифференциальная модель турбулентности [3].

Ниже представлены результаты расчета [45] течений, в которых получены экспериментальные дан-

ные [31] по реламинаризации турбулентного течения в трубе, а также численное исследование для трех вариантов входных условий, обеспечивающих реламинаризацию при больших числах Рейнольдса.

Граничные условия на стенке трубы диаметром D , на оси симметрии и на входе в трубу для рассматриваемой задачи имеют следующий вид:

$$\begin{aligned} u = 0, \quad v = 0, \quad E = \frac{\partial E}{\partial r} = \tau = 0 \quad (r = R = D/2), \\ \frac{\partial u}{\partial r} = v = 0, \quad \frac{\partial E}{\partial r} = \frac{\partial \omega}{\partial r} = \tau = 0 \quad (r = 0), \\ p = \text{const}, \quad u = u(r), \quad E = E(r), \quad L = L(r), \\ \tau = 0 \quad (x = 0). \end{aligned} \quad (3)$$

В качестве граничных условий на входе используются постоянные по сечению профили скорости, температуры и характеристик турбулентности.

Система уравнений, приведенная в работе [1] с граничными условиями (3), решалась численно методом прогонки с итерациями. Расчеты проводились на неравномерной сетке. Шаг по радиусу трубы вблизи стенки был малым. Шаг по продольной координате был также достаточно мал в сечениях, близких к входному сечению, и увеличивался по мере продвижения вниз по потоку, что позволило проводить расчеты даже при достаточно малом уровне турбулентности на входе в трубу.

Результаты численного исследования реламинаризации турбулентного течения, как будет показано ниже, существенным образом зависят от условий на входе в участок реламинаризации: профиля скорости и характеристик турбулентности.

Расчетная схема представлена на рис. 4. Профиль скорости на входе в участок реламинаризации был принят соответствующим профилю скорости при первом способе формирования потока работы [31]. В трубе с внутренним диаметром $D = 2R = 30$ мм было организовано два спутных потока: центральный поток (индекс 2), выходящий из кольцевой вставки с внутренним диаметром $d_2 = 2r_2 = 26$ мм со среднерасходной скоростью U_2 , и кольцевой пристеночный поток (индекс 1) в зазоре шириной 1 мм между кольцевой вставкой с наружным диаметром $d_1 = 2r_1 = 28$ мм и трубой со среднерасходной скоростью U_1 . Отношение скоростей спутных потоков было принято $U_1/U_2 = 1,67$. Что касается характеристик турбулентности на входе в участок реламинаризации, то они могут быть установлены лишь приближенно. На входе в участок реламинаризации

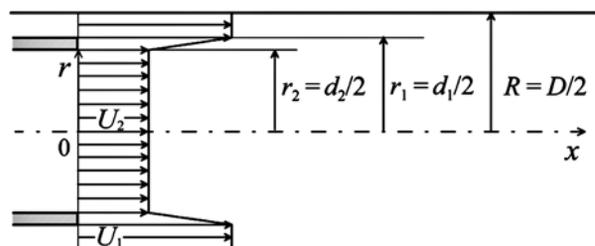


Рис. 4. Расчетная схема

(индекс 0) в центральном потоке в расчетах было принято значение интенсивности турбулентности $e_{02} = \sqrt{E_{02}}/U_2 = 0,05$ ($\sqrt{E_{02}}/U = 0,044$), которое на расстоянии $z = x/D = 2$ от входа на оси трубы (индекс cl) снижалось до $e_{cl} = \sqrt{E_{cl}}/U = 0,036$, что близко к значению $2u'/U$, полученному на том же расстоянии от входа в эксперименте [31]. Масштаб турбулентности на входе L_{02} в центральном потоке был принят примерно равным диаметру отверстий перфорации решетки, так что значение относительной величины масштаба составило $l_{02} = L_{02}/R = 0,2$.

Оценки интенсивности и масштаба турбулентности для спутного потока позволили установить следующее. Поскольку поток в зазоре шириной 1 мм между вставкой и трубой близок к ламинарному, то возмущения в нем будут минимальными. Однако наличие кромки кольцевой вставки толщиной 1 мм приводит к турбулизации спутного потока с масштабом турбулентности порядка толщины кромки. В расчетах для спутного потока были приняты следующие входные значения величины интенсивности $e_{01} = \sqrt{E_{01}}/U_1 = 0,007$ ($\sqrt{E_{01}}/U = 0,01$) и масштаба турбулентности $l_{01} = L_{01}/R = 0,07$. Напряжение сдвига в обоих потоках на входе было принято $\tau = -\langle u'v' \rangle = 0$. Профиль скорости в спутных потоках на входе был принят однородным с линейным изменением на кромке кольцевой вставки (см. рис. 4). Расчеты выполнены для чисел Рейнольдса $Re = DU/\nu$, приведенных в работе [31], которые определены по диаметру трубы D , среднерасходной скорости потока U и кинематической вязкости воздуха ν .

На рис. 5 для чисел Рейнольдса $Re = 3800$ и $Re = 4500$ при первом способе реламинаризации из работы [31] приведено изменение по длине участка реламинаризации $z = x/D$ относительной скорости потока на оси трубы u_{cl}/U . Как видно, реламинаризация, как и в эксперименте [31], достигается на длине $z = x/D = 100$ при числе Рейнольдса $Re = 3800$, что подтверждается значением $u_{cl}/U = 1,8$ (рис. 5, а), близким к значению $u_{cl}/U = 2$ для ламинарного режима течения. При этом интенсивность турбулентности на оси трубы $e_{cl} = \sqrt{E_{cl}}/U$ (рис. 5, б) стремится к нулю, и коэффициент сопротивления $\xi = (\rho\nu \partial u / \partial r)_w$ (рис. 5, в) приближается к значению $\xi = 64/Re$ для ламинарного режима течения, что также подтверждает достижение

реламинаризации при числе Рейнольдса $Re = 3800$. Для числа $Re = 4500$ при тех же входных условиях реламинаризации, как и в эксперименте [31], получено не было (рис. 5).

Расчетное изменение по длине профилей скорости u/U и интенсивности турбулентности $e = \sqrt{E}/U$ для тех же чисел Рейнольдса показало, что профиль скорости в случае реламинаризации течения ($Re = 3800$) при $z = 100$ близок к профилю Пуазейля. Для $Re = 4500$, когда реламинаризация не наступает, профиль скорости при $z = 100$ близок к профилю скорости для турбулентного течения. Показано, что в случае реламинаризации течения наиболее резкое снижение интенсивности турбулентности, как и в эксперименте [31], имеет место в области вблизи стенки трубы. Согласно [31] это означает подавление генерации турбулентности в пристеночной области, где она в отсутствие реламинаризации максимальна. Полученные результаты свидетельствуют о том, что в случае реламинаризации генерация турбулентности подавляется, и максимум ее смещается от стенки к оси трубы. В случае отсутствия реламинаризации ($Re = 4500$) максимум генерации турбулентности возрастает с выходом на режим развитого турбулентного течения при $z = 100$ и положение максимума генерации остается в пристеночной области. Эти результаты подтверждают сделанное в работе [31] предположение о том, что основополагающий физический механизм реламинаризации связан с ослаблением цикла производства пристеночной турбулентности.

Представленные в работе [45] результаты подтверждают возможность реламинаризации течения с замедленным потоком в центральной области трубы и ускоренным в пристеночной области, а расчетное число Рейнольдса реламинаризации $Re = 3800$ соответствует экспериментальному значению, полученному в работе [31]. Такое невысокое значение числа Рейнольдса реламинаризации объясняется достаточно высокой интенсивностью турбулентности в центральной области, обусловленной постановкой турбулизирующей решетки для организации замедленного потока. При этом существенная зависимость возможности реламинаризации от входных условий была отмечена в работе [31].

Анализ представленных в работе [31] результатов экспериментального исследования позволяет сделать

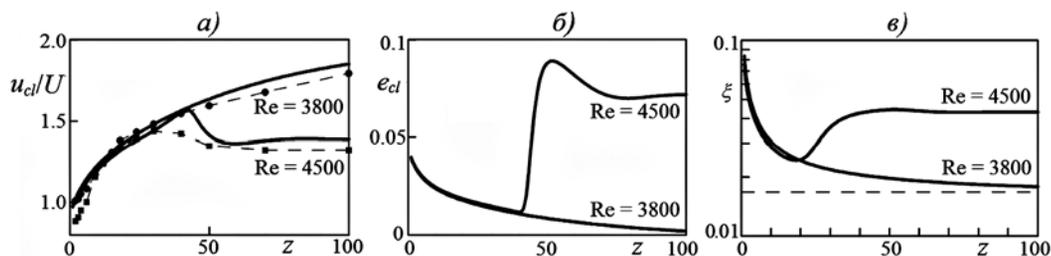


Рис. 5. Изменение по длине участка реламинаризации $z = x/D$ относительной скорости потока u_{cl}/U (а) интенсивности турбулентности $e_{cl} = \sqrt{E_{cl}}/U$ (б) на оси трубы и коэффициента сопротивления $\xi = (\rho\nu \partial u / \partial r)_w$ (в) для двух чисел Рейнольдса: $Re = 3800$ и $Re = 4500$. Точки — эксперимент [31], линии — расчет. Штриховая линия (в) $\xi = 64/Re$ для ламинарного режима течения при $Re = 3800$

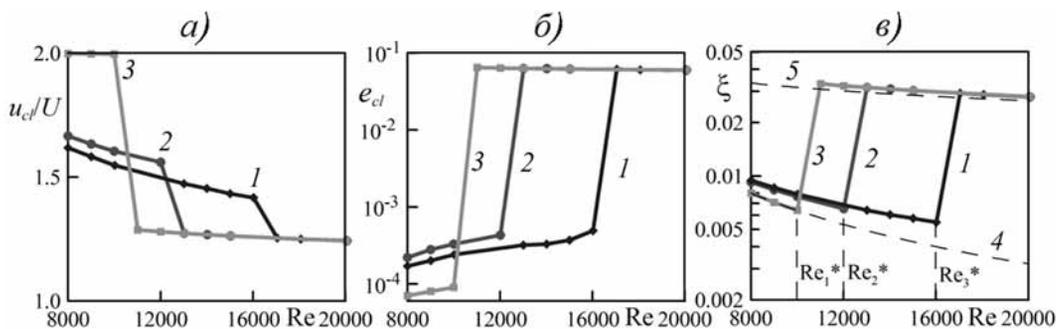


Рис. 6. Зависимости от числа Рейнольдса $Re = DU/\nu$ величин скорости u_{cl}/U (а), интенсивности турбулентности $e_{cl} = \sqrt{E_{cl}}/U$ (б) на оси трубы и коэффициента сопротивления ξ (в) для трех вариантов входных устройств:

1 — вариант 1; 2 — вариант 2; 3 — вариант 3; 4 — функция $\xi = 64/Re$; 5 — функция $\xi = 0,316/Re^{1/4}$

вывод, что для достижения более высокого числа Рейнольдса реламинаризации необходимо сформировать поток не только с заданным профилем скорости, но и с низким уровнем турбулентности. Поэтому представляет интерес исследовать влияние входного профиля скорости на возможность реламинаризации при низком уровне интенсивности турбулентности, генерируемой входным устройством.

При проведении численного исследования рассмотрено три варианта входных устройств:

- вариант 1 с U-образным профилем скорости (см. рис. 4), аналогичным рассмотренному в эксперименте [31], с параметрами $r_1/R = 0,9$, $r_2/R = 0,87$, $U_1/U_2 = 1,5$;
- вариант 2 с л-образным профилем скорости, с параметрами $r_1/R = 0,9$, $r_2/R = 0,87$, $U_1/U_2 = 0,6$;
- вариант 3 с параболическим профилем скорости.

Во всех трех вариантах характеристики турбулентности на входе были одинаковыми и приняты следующими: интенсивность турбулентности, определенная по местной скорости u , $e_0 = \sqrt{E_0}/u = 0,01$, масштаб турбулентности $L_0/R = 0,05$.

Отметим, что выбор варианта 3 входного устройства с параболическим профилем скорости на входе, реализованный с использованием хонейкомба [37], обусловлен тем обстоятельством, что профиль скорости при достижении реламинаризации будет стремиться к параболическому, т. е. остается близким к входному профилю скорости. При этом коэффициент сопротивления ξ на всей длине для варианта 3 наименьший из значений ξ для трех рассмотренных вариантов входных устройств и близок к ламинарному значению $\xi = 64/Re$.

На рис. 6 представлены зависимости от числа Рейнольдса $Re = DU/\nu$ величин u_{cl}/U , $e_{cl} = \sqrt{E_{cl}}/U$ и ξ для трех вариантов входных устройств.

Достижение режима реламинаризации в расчетах определялось, как и в [31], по изменению по длине трубы относительной скорости u_{cl}/U (рис. 6, а) и интенсивности турбулентности $e_{cl} = \sqrt{E_{cl}}/U$ (рис. 6, б) на оси трубы,

а также по изменению коэффициента сопротивления ξ (рис. 6, в).

В расчетах, проведенных с шагом по числу Рейнольдса $\Delta Re = 1000$, получено, что реламинаризация наступает для варианта входного устройства 1 в диапазоне чисел Рейнольдса $Re = (16...17) \cdot 10^3$, для варианта 2 — в диапазоне $Re = (12...13) \cdot 10^3$, для варианта 3 — в диапазоне $Re = (10...11) \cdot 10^3$.

Для вариантов расчета 1 и 2 с двухзонной организацией течения представляет интерес исследовать влияние различных входных условий в пристеночном потоке на число Рейнольдса реламинаризации течения. На рис. 7 представлены зависимости числа Рейнольдса реламинаризации Re^* от входных величин интенсивности турбулентности $e_{02} = \sqrt{E_{02}}/U_2$ ($l_{02} = L_{02}/R = 0,05$) и относительного масштаба турбулентности $l_{02} = L_{02}/R$ ($e_{02} = \sqrt{E_{02}}/U_2 = 0,01$) в пристеночном потоке для вариантов 1 и 2 входных устройств.

Как видно на рис. 7, а, снижение входной интенсивности турбулентности e_{02} в пристеночном потоке существенно повышает число Рейнольдса реламинаризации Re^* . Это повышение значения Re^* в большей степени относится к варианту 1 с увеличенной скоростью пристеночного потока. Что касается влияния входного масштаба турбулентности l_{02} (рис. 7, б), то влияние его на Re^* существенно для малых масштабов ($l_{02} < 0,2$). Для крупномасштабной входной турбулентности Re^* практически не зависит от l_{02} .

Анализ полученных и опубликованных в работе [31] результатов экспериментального исследования

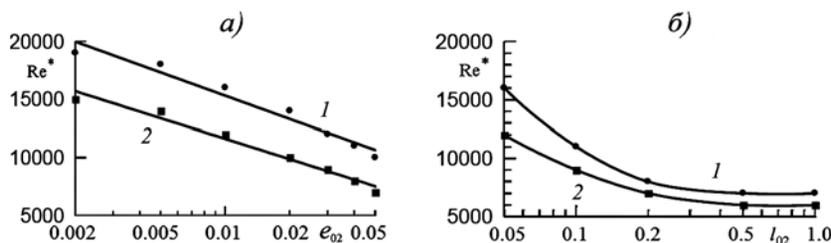


Рис. 7. Зависимость числа Рейнольдса реламинаризации Re^* от интенсивности турбулентности $e_{02} = \sqrt{E_{02}}/U_2$ (а) и относительного масштаба турбулентности $l_{02} = L_{02}/R$ (б) в пристеночном потоке для двух вариантов входных устройств: 1 — вариант 1; 2 — вариант 2

и результаты численного исследования позволяют сделать вывод, что для достижения более высокого числа Рейнольдса реламинаризации необходимо сформировать поток не только с заданным профилем скорости, но и с низким уровнем турбулентности.

2.2. Интенсификация теплообмена при течении в диффузорах

Повышение эффективности теплообменных аппаратов за счет интенсификации процессов теплообмена является актуальной задачей энергосбережения. К настоящему времени предложено большое число разнообразных способов интенсификации теплообмена, обзору и анализу которых посвящено множество работ [46—49].

В монографии [47] предпринята первая (к началу 1990-х годов) попытка обобщенного систематизированного изложения экспериментальных и аналитических результатов отечественных и зарубежных ученых по исследованию интенсификации теплообмена. Следует отметить, что число публикаций по исследованию интенсификации теплообмена непрерывно возрастает. Число их в виде статей, докладов и отчетов к 2011 г. составило более 1000 [49, 50].

Наиболее распространенным методом интенсификации теплообмена в каналах с однофазным теплоносителем является искусственная турбулизация потока, осуществляемая с помощью кольцевых или спиральных канавок, оребрения поверхности, спиральных или витых труб, шнеков, закрученных лент, лунок и т. п. [47]. Как отмечено в работе [47], наиболее доступным и эффективным методом управляемого воздействия на структуру турбулентного потока в целях интенсификации теплообмена является создание в нем отрывных зон или других организованных вихревых структур.

Одним из наиболее распространенных способов образования вихревых зон являются поперечные выступы или канавки, располагаемые на поверхности канала [47]. Структура вихревых зон зависит от очертаний поверхности выступа или канавки, являющихся причиной возникновения вихревой зоны.

Представляет интерес предложенный в монографии [46] способ интенсификации теплообмена с использованием трубчатой поверхности с волнистыми стенками, образующими систему диффузор—конфузор. Углы расширения диффузоров при этом выбираются из условия получения потока с нестационарными вихревыми зонами, которые интенсифицируют теплообмен.

Рассмотренный в работе [47] способ интенсификации теплообмена при использовании труб с кольцевыми турбулизаторами позволяет увеличить число Нуссельта Nu по сравнению с Nu_0 для гладких труб в 2,6—2,8 раза в диапазоне чисел Рейнольдса $10^4 \dots 10^5$ для оптимальной высоты и шага турбулизаторов (см. табл. 6,1—6,3 в работе [47]). Однако при этом относительная величина коэффициента гидравлического сопротивления ξ/ξ_0 возрастает до значений 6,3...9,2, существенно превышающих увеличение значения Nu/Nu_0 .

Таким образом, известные способы интенсификации теплообмена связаны с увеличением мощности на прокачку теплоносителя, что является недостатком этих способов.

Проведенное в работе [51] расчетное исследование течения в круглом коническом безотрывном диффузоре с малым углом раскрытия, выполненное с использованием трехпараметрической дифференциальной модели турбулентности [3], показало сильное влияние расширения на режим течения, а измеренные в [51] профили скорости и напряжений Рейнольдса хорошо согласуются с расчетами.

Расчет теплообмена в круглых конических расширяющихся каналах с переменным по длине числом Рейнольдса [52, 53] показал, что для течения в диффузоре при всех рассмотренных числах Рейнольдса основные характеристики теплообмена — число Нуссельта и коэффициент аналогии Рейнольдса — оказались значительно выше, чем в трубе постоянного сечения при том же числе Рейнольдса. Проведенные в работах [52, 53] расчеты продемонстрировали возможность повышения эффективности теплообменных аппаратов за счет интенсификации теплоотдачи в круглых диффузорах.

2.2.1. Интенсификация теплообмена в плоском безотрывном диффузоре

В работе [54] рассмотрена возможность интенсификации теплообмена в плоских конических расширяющихся каналах с постоянным по длине числом Рейнольдса и гидравлически гладкой поверхностью, обеспечивающей минимальные гидравлические потери.

При решении задач о течении и теплообмене в диффузоре для замыкания системы уравнений, приведенной в работе [1], использовались следующие граничные условия.

На стенках диффузора ($y = 0, h$) и на входе ($x = 0$) граничные условия имеют следующий вид:

$$\begin{aligned} u = v = 0, \quad -\left(\lambda \frac{\partial T}{\partial n}\right) &= 0, \\ E = \frac{\partial \omega}{\partial n} = \tau = q_T = 0 \quad (y = 0), \\ u = v = 0, \quad -\left(\lambda \frac{\partial T}{\partial n}\right) &= q_w, \\ E = \frac{\partial \omega}{\partial n} = \tau = q_T = 0 \quad (y = h), \\ p = p_1, \quad u = U_1, \quad T = T_1, \quad E = E_1, \quad \tau &= 0, \\ \omega = \omega_1, \quad q_T = 0 \quad (x = 0). \end{aligned} \quad (4)$$

В качестве граничных условий на входе используются постоянные по сечению профили скорости, температуры и характеристик турбулентности.

Задача решалась в следующей постановке (рис. 8). Входу в диффузор предшествует участок постоянного сечения (высотой h_1 и длиной $\bar{L}_1 = L_1/h_1 = 100$), на котором устанавливается развитое турбулентное течение. Далее следует участок с диффузором длиной $\bar{L}_d = L_d/h_1$. За диффузором расположен выход-

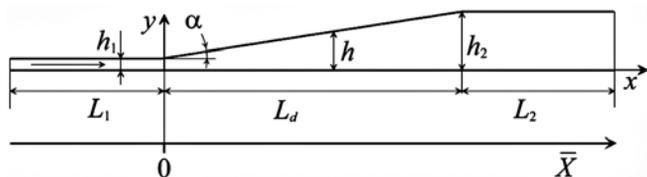


Рис. 8. Расчетная схема

ной участок постоянного сечения высотой $h_2 = h_1 + L_d \operatorname{tg} \alpha$. Отметим, что число Рейнольдса $Re = Ud_h/\nu$, определенное по гидравлическому диаметру $d_h = 2h$ и среднерасходной скорости U , остается постоянным по всей длине диффузора. В качестве продольной безразмерной координаты выбрана $\bar{X} = x/h_1$, где x — продольная координата с нулем в начале диффузора.

Сравнение результатов расчета течения и теплообмена в диффузоре проводилось с соответствующими результатами для течения в канале постоянного сечения при одинаковом числе Рейнольдса. Тепловой поток в стенку канала принимался постоянным по длине и небольшим по значению ($q_w = 10 \dots 100 \text{ Вт/м}^2$) с тем, чтобы теплофизические и переносные свойства теплоносителя можно было считать постоянными.

Результатами расчетов явились локальные значения безразмерного коэффициента теплоотдачи — числа Нуссельта Nu , коэффициента сопротивления ξ , относительного градиента давления ∇P° , определенные как

$$Nu = \frac{dq_w}{\lambda(T_w - \bar{T})}, \quad \xi = 8 \frac{\tau_w}{\rho U^2}, \quad \nabla P^\circ = \frac{dp/dx}{(dp/dx)_0},$$

где T_w — температура стенки; \bar{T} — средне-массовая температура; U — среднерасходная скорость; $\tau_w = -(\eta \partial u / \partial y)_w$; $q_w = -(\lambda \partial T / \partial y)_w$; $(dp/dx)_0 = \xi_0 \rho U^2 / 2d_h$ — абсолютная величина градиента давления для развитого течения в канале постоянного сечения при числе Рейнольдса, соответствующем течению в диффузоре.

В расчетах получены также безразмерные профили скорости $\bar{u} = u/U$, интенсивности турбулентности $e = \sqrt{E}/U$ и турбулентного трения $\rho \tau = -\rho \langle u'v' \rangle$.

Интегральные характеристики течения и теплообмена представлены в виде относительных величин числа Нуссельта Nu/Nu_0 , и коэффициента сопротивления ξ/ξ_0 , градиента давления ∇P° и фактора аналогии Рейнольдса (Reynolds Analogy Factor) $RAF = (Nu/Nu_0)/(\xi/\xi_0)$. Здесь число Нуссельта Nu_0 и коэффициент сопротивления ξ_0 вычислены для развитого течения в канале постоянного сечения при числе Рейнольдса, соответствующем течению в диффузоре.

Кроме того, определялись средние по длине диффузора L_d значения фактора аналогии Рейнольдса и относительных величин числа Нуссельта и коэффициента сопротивления:

$$\langle RAF \rangle = \frac{1}{L_d} \int_0^{L_d} RAF dx, \quad \langle Nu/Nu_0 \rangle = \frac{1}{L_d} \int_0^{L_d} \frac{Nu}{Nu_0} dx,$$

$$\langle \xi/\xi_0 \rangle = \frac{1}{L_d} \int_0^{L_d} \frac{\xi}{\xi_0} dx.$$

Параметрами задачи являются:

угол раскрытия диффузора — α ; число Рейнольдса — $Re = Ud_h/\nu$; длина диффузора — L_d ; число Прандтля — $Pr = \eta c_p/\lambda$; c_p — изобарная теплоемкость.

В работах [54, 55] приведены результаты численного исследования влияния на интенсификацию теплообмена указанных параметров, которые изменялись в диапазонах $\alpha = 1^\circ \dots 10^\circ$, $Re = (10 \dots 100) \cdot 10^3$, $L_d = 10 \dots 100$, $Pr = 0,026$ (ртуть) $\dots 55,7$ (трансформаторное масло). В качестве базового варианта был принят следующий: $\alpha = 3^\circ$, $Re = 40 \cdot 10^3$, $L_d = 50$, $Pr = 0,71$. Далее везде, где на рисунках какие-либо параметры не указаны, они равны базовым.

На рис. 9 представлено изменение по длине относительного числа Нуссельта Nu/Nu_0 и относительного коэффициента сопротивления ξ/ξ_0 в зависимости от угла раскрытия диффузора α .

Как видно на рис. 9, число Nu при течении в диффузоре превышает число Nu_0 при течении в канале постоянного сечения при том же числе Re и тем сильнее, чем больше угол раскрытия диффузора. Так, при $\alpha = 5^\circ$ (линия 5) число Nu примерно в 2 раза превышает соответствующее значение Nu_0 для канала постоянного сечения.

Коэффициент сопротивления при течении в диффузоре ξ (рис. 9, б) изменяется по длине не монотонно. Сначала он уменьшается, а затем возрастает и незначительно превышает соответствующее значение ξ_0 для течения в канале постоянного сечения при том же числе Re , но в меньшей степени, чем число Нуссельта.

В выходном участке после диффузора при большом угле раскрытия α течение не успевает выйти на развитое вследствие недостаточной относительной длины канала, определенной по выходной высоте, например, $(x - L_2)/h_2 \approx 9$ для $\alpha = 5^\circ$. Поэтому значения Nu/Nu_0 и ξ/ξ_0 (рис. 9) в выходном сечении заметно отличаются от единицы.

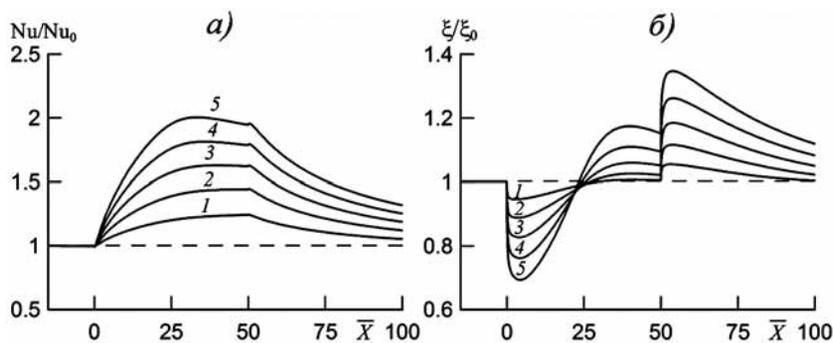


Рис. 9. Изменение по длине \bar{X} относительного числа Нуссельта Nu/Nu_0 (а) и относительного коэффициента сопротивления ξ/ξ_0 (б) для ряда значений угла α :

линии 1 — 5 — $\alpha = 1^\circ, 2^\circ, 3^\circ, 4^\circ, 5^\circ$ соответственно

Результаты расчета градиента давления показывают, что с ростом угла раскрытия диффузора появляющийся в диффузоре положительный градиент давления возрастает. Это приводит к турбулизации течения, что подтверждается изменением профилей скорости $\bar{u} = u/U$, температуры $\theta = (T - T_w)/(\bar{T} - T_w)$ и интенсивности турбулентности $e = \sqrt{\bar{E}}/U$.

Одним из критериев эффективности теплообмена является соотношение между теплоотдачей и гидравлическими потерями — так называемый коэффициент аналогии Рейнольдса. Известно [46, 47], что наиболее низкий теплообмен наблюдается в газовых ($Pr \approx 1$) теплообменниках при невысоких числах Рейнольдса. Так, для развитого турбулентного ($Re > 10^4$) течения воздуха в трубах коэффициент аналогии Рейнольдса не превосходит значения 1,05...1,15.

При течении в диффузоре (рис. 9) число Nu существенно превышает число Nu_0 при течении в канале постоянного сечения при том же числе Re , в то время как коэффициент сопротивления ξ изменяется незначительно. Поэтому RAF при течении в диффузоре существенно выше единицы. С увеличением угла раскрытия диффузора α значение RAF возрастает. Так, для $\alpha = 5^\circ$ его значение примерно в 2 раза превышает единицу.

Учитывая непостоянство значений RAF , Nu/Nu_0 , ξ/ξ_0 по длине диффузора, представляется целесообразным использовать для анализа процесса средние по длине диффузора значения величин числа Нуссельта, коэффициента сопротивления и фактора аналогии Рейнольдса. Как видно на рис. 10, средние по длине диффузора значения $\langle RAF \rangle$ и $\langle Nu/Nu_0 \rangle$ существенно зависят от угла раскрытия диффузора, в отличие от $\langle \xi/\xi_0 \rangle$, которое практически не зависит от угла α . Это обстоятельство подчеркивает принципиальное отличие рассмотренного способа интенсификации теплообмена от известных [46—50], где увеличение теплоотдачи (числа Нуссельта) достигается ценой значительного роста гидравлических потерь (коэффициента сопротивления).

Численное исследование влияния числа Рейнольдса на характеристики течения и теплообмена в диффузоре позволило установить следующее. С ростом числа Рейнольдса (с $10 \cdot 10^3$ до $100 \cdot 10^3$) фак-

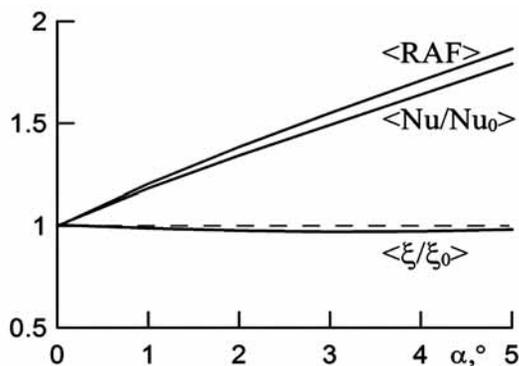


Рис. 10. Средние по длине диффузора значения фактора аналогии Рейнольдса $\langle RAF \rangle$, числа Нуссельта $\langle Nu/Nu_0 \rangle$ и коэффициента сопротивления $\langle \xi/\xi_0 \rangle$ в зависимости от угла α

тор аналогии Рейнольдса, как текущий — RAF , так и средний по длине диффузора — $\langle RAF \rangle$ немного уменьшаются. Уменьшается также среднее по длине число Нуссельта $\langle Nu/Nu_0 \rangle$, в то время как коэффициент сопротивления $\langle \xi/\xi_0 \rangle$ остается практически постоянным.

Полученное в расчетах влияние числа Рейнольдса на интенсификацию теплообмена качественно согласуется с приведенными в [47] опытными данными по уменьшению значения Nu/Nu_0 с ростом числа Рейнольдса. Полученная в расчетах зависимость интенсификации теплообмена от длины диффузора показала, что для сохранения высокого текущего значения фактора аналогии Рейнольдса RAF на большой длине следует использовать длинный диффузор.

Как отмечено в работе [47], попытка ответить на вопрос, на каких теплоносителях (жидкометаллических, газах или капельных жидкостях) можно получить лучшие эффекты интенсификации, была сделана на основе теоретического анализа, проведенного Нуннером [56], который пришел к выводу, что наилучшие результаты увеличения теплоотдачи достигаются для жидких металлов, т. е. при $Pr \ll 1$. Однако все последующие работы, как экспериментальные [49], так и теоретические, не подтвердили выводов Нуннера [56]. Приведенные в работе [49] опытные данные позволили авторам сделать вывод, что при развитом турбулентном течении Pr слабо влияет на эффект интенсификации в области значений $Pr = 0,7...80$.

Численное исследование влияния числа Прандтля на характеристики течения и теплообмена в диффузоре, проведенное в диапазоне $Pr = 0,026...55,7$, позволило установить следующее. С ростом числа Прандтля фактор аналогии Рейнольдса, как текущий RAF , так и средний по длине диффузора $\langle RAF \rangle$ (рис. 11) меняются немонотонно. Аналогичным образом меняется среднее по длине число Нуссельта $\langle Nu/Nu_0 \rangle$, в то время как коэффициент сопротивления $\langle \xi/\xi_0 \rangle$ (рис. 11) остается практически постоянным.

Максимальный эффект интенсификации теплообмена (рис. 11) достигается для газов с малым чис-

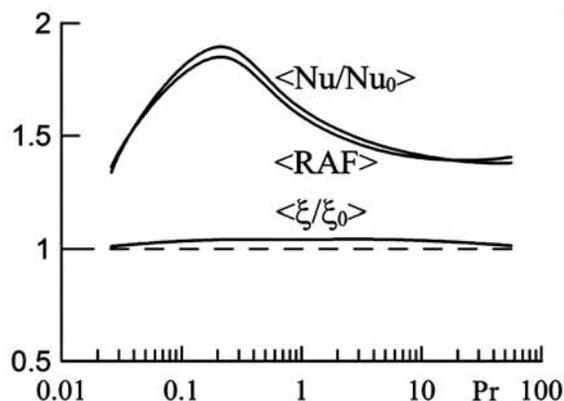


Рис. 11. Средние по длине диффузора значения числа Нуссельта $\langle Nu/Nu_0 \rangle$, фактора аналогии Рейнольдса $\langle RAF \rangle$ и коэффициента сопротивления $\langle \xi/\xi_0 \rangle$ в зависимости от числа Прандтля Pr теплоносителя

лом Прандтля (например, для смеси гелий—ксенон). С ростом числа Прандтля эффект интенсификации снижается, выходя на уровень, соответствующий жидким металлам, что качественно согласуется с приведенными в работе [47] опытными данными. Это позволяет рекомендовать использовать в теплообменниках газовые теплоносители с малым числом Прандтля.

В заключение отметим, что принципиальное отличие процессов течения и теплообмена в плоском диффузоре [54] и в круглом диффузоре [52, 53] состоит в следующем. В плоском диффузоре число Рейнольдса остается постоянным по всей длине диффузора, а в круглом диффузоре оно уменьшается по длине обратно пропорционально диаметру диффузора. Это сказывается на изменении относительных характеристик течения и теплообмена по длине диффузора, поскольку для канала постоянного сечения, с которым проводится сравнение, значения этих характеристик зависят от значения числа Рейнольдса.

2.2.2. Интенсификация теплообмена в пластинчатых теплообменниках с диффузорными каналами

Представляет интерес исследовать возможность интенсификации теплообмена в пластинчатых теплообменниках, которые нашли широкое применение, в частности, в химической, нефтяной, пищевой, молочной и пивоваренной промышленности [57].

В пластинчатых теплообменниках обычно используются пластины с различного типа гофрированной поверхностью, которые приводят к интенсификации теплообмена и увеличению поверхности теплообмена в пределах 15...25 % [57]. При этом коэффициенты трения в пластинчатых теплообменниках с гофрированными пластинами намного выше, чем в гладких каналах, при одинаковом числе Рейнольдса.

Предлагаемое в работе [58] использование в пластинчатых теплообменниках диффузорных каналов с гладкой поверхностью, обеспечивающей минимальные гидравлические потери, позволит обеспечить интенсификацию теплообмена в диффузоре практически без роста коэффициента трения, что характеризует принципиальное отличие рассмотренного способа интенсификации теплообмена от других известных способов, где увеличение теплоотдачи достигается ценой значительного роста гидравлических потерь.

Задача решалась в следующей постановке (рис. 12). Рассмотрен элемент пластинчатого противоточного теплообменника с двумя смежными плоскими диффузорами длиной L и высотой $h(x)$ (рис. 12, а) с линейным профилем общей теплопередающей стенки по длине. Две другие стенки диффузоров полагались теплоизолированными. Входу в диффузоры предшествовали участки постоянного сечения высотой h_1 и длиной L_1 с теплоизолированными стенками. За диффузорами следовали участки постоянного сечения высотой h_2 и длиной L_2 также с теплоизолированными стенками. Высота выходного сечения зависела от угла наклона образующей диффузоров α и определялась как $h_2 = h_1 + Ltg\alpha$. Отметим, что число Рейнольдса $Re = Ud_h/\nu$ ($d_h = 2h$ — гидравлический диаметр; U — среднерасходная скорость) в плоском диффузоре при любом угле раскрытия для несжимаемой жидкости с постоянными теплофизическими свойствами остается постоянным по длине. Для теплоносителей с переменными свойствами при не очень существенном нагреве (охлаждении) число Рейнольдса будет меняться по длине диффузора также незначительно.

Для сравнения рассмотрен также элемент пластинчатого противоточного теплообменника с двумя плоскими каналами постоянного сечения (рис. 12, б). Длины каналов соответствуют приведенным в рабо-

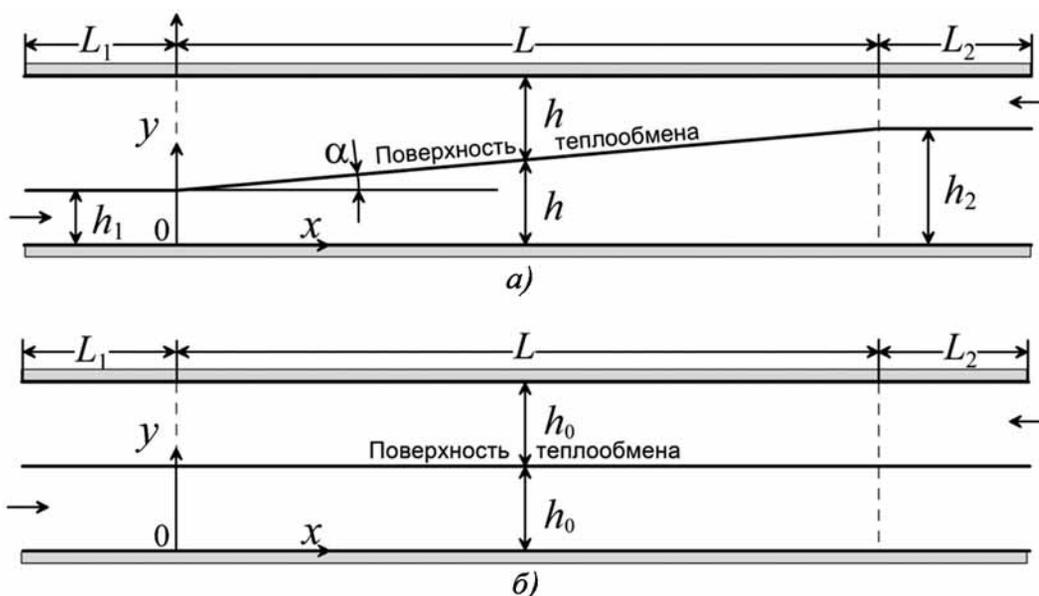


Рис. 12. Расчетная схема элемента пластинчатого противоточного теплообменника с двумя плоскими диффузорами (а) и с двумя каналами постоянного сечения (б)

те [58], а высота каналов определялась из условий равенства средних по длине площадей сечений и составляла $h_0 = (h_1 + h_2)/2$.

Граничные условия на входе:

$$p = p_1, \quad u = U_1, \quad T = T_1, \quad E = E_1, \\ \tau = 0, \quad \omega = \omega_1, \quad q_T = 0 \quad (x = -L_1).$$

Граничные условия на стенках:

$$u = v = 0, \quad E = \frac{\partial \omega}{\partial y} = \tau = q_T = 0; \quad (5)$$

на теплопередающей стенке, разделяющей потоки в каналах:

$$T_{wh} = T_{wc} = T_w, \quad -\left(\lambda \frac{\partial T}{\partial y}\right)_{wh} = -\left(\lambda \frac{\partial T}{\partial y}\right)_{wc} = q_w;$$

на теплоизолированных стенках:

$$\left(\frac{\partial T}{\partial y}\right) = 0.$$

Отметим, что температура и тепловой поток на теплопередающей стенке должны быть непрерывны. Индексы c и h в граничных условиях и далее относятся соответственно к условиям на стенке и в "холодном", и "горячем" потоках, а 1 и 2 соответственно к входным и выходным каналам.

В качестве граничных условий на входе использовались постоянные по сечению профили скорости, температуры и характеристик турбулентности. Интенсивность турбулентности в обоих потоках на входе принималась равной $e_1 = \sqrt{E_1}/U = 0,04$. Масштаб турбулентности в потоке принимался достаточно большим, чтобы вырождение турбулентности на длине входного участка было незначительным.

Таким образом, система уравнений, представленная в работе [1], с граничными условиями (5) позволяет решить задачу и найти распределения как средних, так и турбулентных характеристик течения и теплообмена.

Теплообменник противоточный, поэтому одновременное совместное решение уравнений в приближении узкого канала невозможно. Решение находилось итерационным методом. Последовательно решались приведенные выше уравнения гидродинамики и теплообмена для "горячего" и "холодного" каналов. При этом в "горячем" канале для уравнения теплопроводности в качестве граничного условия на теплообменной стенке задается значение температуры, полученное из расчета для "холодного" канала. При решении уравнений в "холодном" канале на теплообменной стенке задается тепловой поток, полученный из расчета очередной итерации в "горячем" канале. Теплообменная стенка принята бесконечно тонкой, что позволяет не учитывать термическое сопротивление стенки, зависящее от материала и толщины стенки, и при этом температура и тепловой поток на обеих сторонах стенки считаются одинаковыми. По этой же причине тепловой поток вдоль стенки внутри нее считался нулевым.

Пластинчатые теплообменники, согласно [57], наиболее широко применяют с теплоносителями жидкость—жидкость при рабочем давлении 2 МПа и температуре ниже 150 °С.

В расчетах для диффузорных каналов были приняты следующие исходные данные:

- длина диффузоров $L = 500$ мм;
- угол наклона образующей диффузоров $\alpha = 1^\circ$;
- длина входных участков $L_1 = 50$ мм, высота $h_1 = 4$ мм;
- длина выходных участков $L_2 = 50$ мм, высота $h_2 = h_1 + L \operatorname{tg} \alpha = 12,7$ мм.

Для теплообменников с каналами постоянного сечения были приняты следующие исходные данные:

- длина каналов постоянного сечения $L = 500$ мм;
- длина входных и выходных участков $L_1 = L_2 = 50$ мм, высота $h_0 = (h_1 + h_2)/2 = 8,4$ мм.

Для обоих теплообменников были приняты следующие исходные данные:

- теплоноситель — вода при давлении на входе 1 МПа;
- входная температура "холодного" потока $T_c = 300$ К;
- входная температура "горячего" потока $T_h = 400$ К;
- число Рейнольдса "холодного" потока принято постоянным $Re = 10\,000$;
- число Рейнольдса "горячего" потока варьировалось в диапазоне $Re = 3000 \dots 6000$.

Представленные в работе [58] результаты расчетов получены для "холодного" потока с числом Рейнольдса $Re_c = 10000$, а для "горячего" — с меньшим числом Рейнольдса $Re_h = 3000$, что обеспечивает (см. ниже) максимальное охлаждение "горячего" потока.

Приведенное на рис. 13, а изменение по длине температуры теплопередающей стенки $T_w(x)$ свидетельствует о том, что в теплообменнике с диффузорными каналами эта температура более круто меняется по длине (линии 1) по сравнению с теплообменником с каналами постоянного сечения, где $T_w(x)$ изменяется слабо (линии 2). При этом тепловой поток в стенку $q_w(x) = -(\lambda \partial T / \partial y)_w$ (рис. 13, б) ведет себя аналогичным образом и в теплообменнике с диффузорными каналами (линии 1) существенно превосходит тепловой поток в теплообменнике с каналами постоянного сечения (линии 2).

Как установлено в работах [53, 54], перестройка течения и появляющийся в диффузоре положительный градиент давления приводят к турбулизации течения. Подтверждением этого являются изменение интенсивности турбулентности $e = \sqrt{E}/U$ и напряжения сдвига τ/U^2 , которые существенно возрастают в диффузоре (рис. 14), что приводит к интенсификации теплообмена в теплообменнике с диффузорными каналами.

В результате проведенного численного исследования установлена (рис. 15) зависимость количества переданной теплоты (мощности) от "горячего" потока к "холодному" от числа Рейнольдса "горячего" потока Re_h (при постоянном значении числа Рейнольдса "холодного" потока $Re_c = 10\,000$). Как видно

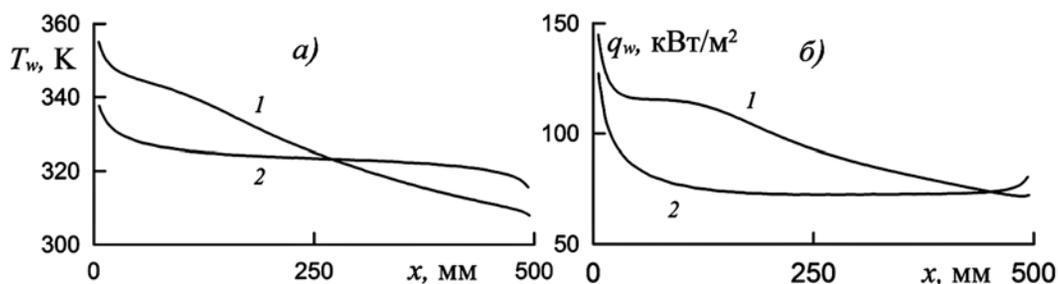


Рис. 13. Изменение по длине температуры теплопередающей стенки (а) и теплового потока в стенку (б): линии 1 — диффузоры с углом расширения $\alpha = 1^\circ$; линии 2 — каналы постоянного сечения ($\alpha = 0^\circ$)

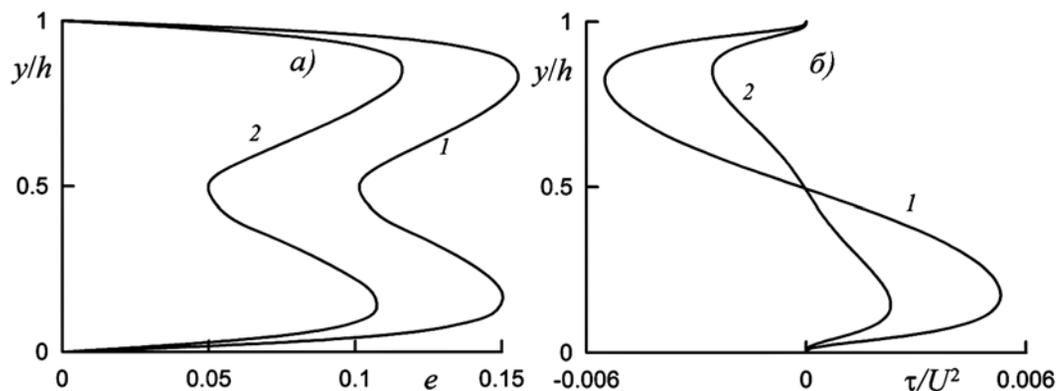


Рис. 14. Профили интенсивности турбулентности (а) и напряжения сдвига (б) в середине каналов ($x = 250$ мм) с "горячими" потоками: линии 1 — диффузор с углом расширения $\alpha = 1^\circ$; линии 2 — канал постоянного сечения

на рис. 15, для теплообменника с диффузорными каналами (линия 1) количество переданной теплоты Q примерно на 21...26 % (в зависимости от числа Рейнольдса Re_h) больше, чем в теплообменнике с каналами постоянного сечения.

Полученная в расчетах зависимость подогрева "холодного" потока и охлаждения "горячего" потока от числа Рейнольдса "горячего" потока (при постоянном значении числа Рейнольдса "холодного" потока $Re_c = 10\ 000$) продемонстрировала эффективность теплообменников с диффузорными каналами. Для них получена разность температур в "холодном" и "горячем" потоках на 20...25 % больше по сравнению с теплообменником с каналами постоянного сечения.

Эффективность повышения тепловых характеристик теплообменника определяется разными факторами. Одним из важных факторов является то, к каким гидравлическим потерям приведет это повышение при одинаковых расходах и входных температурах теплоносителей. В работе [58] проведено сравнение гидравлических потерь в противоточных пластинчатых теплообменниках с разными углами раскрытия каналов от 0° (каналы постоянной высоты) до $0,8^\circ$.

На рис. 16 показаны зависимости изменения температуры "горячего" потока и потеря статического давления в нем от угла раскрытия канала для чисел Рейнольдса $Re = 10\ 000$ как в "холодном", так и в "горячем" каналах.

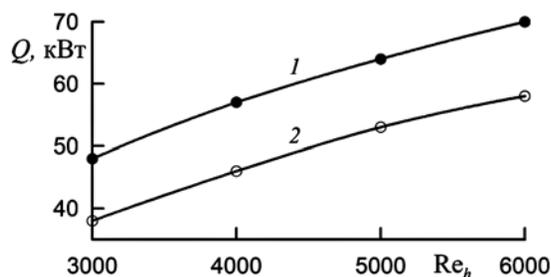


Рис. 15. Зависимость количества переданной теплоты от "горячего" потока к "холодному" от числа Рейнольдса "горячего" потока Re_h : линия, точки 1 — диффузоры с углом расширения $\alpha = 1^\circ$; линия, точки 2 — каналы постоянного сечения

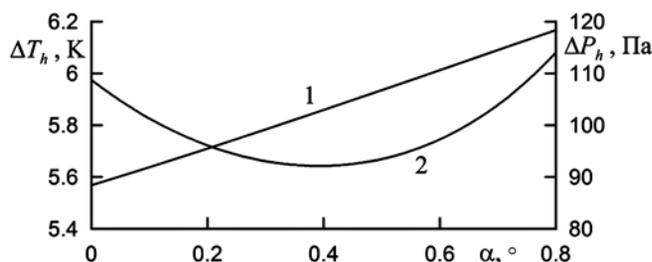


Рис. 16. Зависимость значения величины охлаждения "горячего" потока (линия 1) и потери статического давления в нем (линия 2) от угла раскрытия канала

Видно, что охлаждение горячего потока почти линейно растет с увеличением угла раскрытия. При этом потери статического давления сначала уменьшаются, а после угла $0,4^\circ$ начинают возрастать и при угле $0,75$ градуса сравниваются с потерями статического давления в канале постоянного сечения. В этом диапазоне углов и тепловые, и гидравлические характеристики теплообменника с расширяющимися каналами превышают такие же характеристики теплообменника с каналами постоянного сечения.

Проведенный расчет показал, что в теплообменнике с коническими каналами с углом раскрытия $0,8^\circ$ охлаждение "горячего" потока составляет $6,2^\circ$. Такое же охлаждение достигается в прямом теплообменнике с каналами, длина которых на 12 % больше. При этом потери статического давления в нем немного превышают потери давления в диффузорном теплообменнике, поэтому последний можно считать более эффективным.

Отметим, что эффективность теплообменника в части гидравлических потерь в большей степени зависит от конструктивных особенностей входа и выхода из проточных частей конкретного теплообменника, в основном определяющих суммарные потери давления, что требует специального рассмотрения.

Заключение

С использованием трехпараметрической дифференциальной модели турбулентности разработана методика численного исследования в приближении узкого канала задач течения и теплообмена в плоских и круглых каналах переменного сечения. Описаны основные результаты численного моделирования с использованием предложенной методики для решения задач о ламинаризации потока при течении в конфузоре и трубе; интенсификации теплообмена при течении в диффузорах и в пластинчатом теплообменнике с диффузорными каналами.

Проведено численное моделирование течения с отрицательным градиентом давления в плоском канале с конфузоре при постоянном числе Рейнольдса. Сравнение результатов расчета с известными экспериментальными данными показало, что используемая модель турбулентности адекватно описывает течение с теплообменом в плоском конфузоре, включая процессы ламинаризации потока и последующего восстановления турбулентности. Однако полной ламинаризации течения как в экспериментах, так и в расчетах при параметрах, соответствующих эксперименту, получено не было. Показано, что для плоского конфузора только в случае достаточного времени воздействия градиента давления на течение значение параметра ускорения потока вида $K = (\nu/U^2)(dU/dx)$ может быть использовано для определения условия полной ламинаризации течения. При этом значение будет составлять $K = 6 \cdot 10^{-6}$.

Проведен расчет течений, для которых имеются экспериментальные данные по реламинаризации турбулентного течения в трубе. Получено удовлетворительное согласие результатов расчета с извест-

ными экспериментальными данными по профилям скорости, интенсивности турбулентности и числам Рейнольдса реламинаризации. Полученные результаты расчета подтвердили возможность реламинаризации течения с замедленным потоком в центральной области трубы и ускоренным в пристеночной области. Для трех вариантов входных устройств с различными профилями скорости и одинаковой мелкомасштабной турбулентностью на входе показана возможность достижения реламинаризации течения в трубах при числах Рейнольдса $Re > 10\ 000$. Из трех рассмотренных вариантов входных устройств наиболее эффективным оказался вариант с организацией двухзонного течения с замедленным потоком в центральной области трубы и ускоренным в пристеночной области. В этом варианте получено число Рейнольдса реламинаризации $Re^* = 16\ 000$. Показано, что дальнейшее повышение числа Рейнольдса реламинаризации может быть получено при уменьшении интенсивности и масштаба турбулентности на входе. Полученные результаты численного исследования показали, что независимое формирование в начальном сечении профиля скорости и характеристик турбулентности открывает большие возможности для решения как практических, так и исследовательских задач.

Проведено численное исследование течения и теплообмена в плоских безотрывных диффузорах с различной степенью расширения для ряда чисел Рейнольдса и Прандтля. Сравнение характеристик течения и теплообмена проводилось с соответствующими характеристиками для течения в канале постоянного сечения при одинаковом числе Рейнольдса. Показано, что для течения в диффузоре при всех рассмотренных изменениях угла раскрытия диффузора и чисел Рейнольдса и Прандтля основные характеристики теплообмена — число Нуссельта и фактор аналогии Рейнольдса — оказываются значительно выше, чем в канале постоянного сечения при том же числе Рейнольдса. Это превышение слабо зависит от числа Рейнольдса, но возрастает с увеличением угла раскрытия диффузора и при использовании газового теплоносителя с малым числом Прандтля. Установлено, что перестройка течения и появившийся в диффузоре положительный градиент давления приводят к росту турбулизации течения. Показано, что интенсификация теплообмена в диффузоре достигается практически без роста коэффициента трения, что характеризует принципиальное отличие рассмотренного способа интенсификации теплообмена от других известных способов, где увеличение теплоотдачи (числа Нуссельта) достигается ценой значительного роста гидравлических потерь.

Проведено численное моделирование теплообмена в пластинчатых противоточных теплообменниках с диффузорными каналами с малыми углами раскрытия. Полученные в расчетах значения величин интенсивности турбулентности и напряжения сдвига существенно возрастают в диффузоре, что приводит к интенсификации теплообмена в теплообменнике с диффузорными каналами. Показано, что в теплообменниках с диффузорными каналами за счет ин-

тенсификации теплообмена количество переданной теплоты от "горячего" теплоносителя к "холодному" возрастает по сравнению с теплообменниками с каналами постоянного сечения. Следствием этого является большая разность температур в "холодном" и "горячем" потоках в теплообменнике с диффузорными каналами по сравнению с теплообменником с каналами постоянного сечения.

Работа выполнена в рамках исследований, частично финансируемых гос. бюджетом (АААА-А16-116021110203-6, АААА-А19-119012990110-0; АААА-А16-116021110200-5; АААА-А19-119012990115-5) и Советом по грантам Президента РФ (№ СП-3993.2018.1).

Список литературы

1. Лущик В. Г., Макарова М. С., Решмин А. И. Применение трехпараметрической дифференциальной модели турбулентности для решения задач течения и теплообмена в каналах переменного сечения. Часть 1 // Программная инженерия. — 2021. — Т. 12, № 1. — С. 21—30.
2. Лущик В. Г., Макарова М. С., Якубенко А. Е. Применение трехпараметрической модели сдвиговой турбулентности для решения задач внешнего обтекания проницаемых поверхностей потоком сжимаемого газа // Программная инженерия. — 2017. — Т. 8, № 12. — С. 563—574.
3. Лущик В. Г., Павельев А. А., Якубенко А. Е. Трехпараметрическая модель сдвиговой турбулентности // Изв. АН СССР. МЖГ. — 1978. — № 3. — С. 13—25.
4. Лущик В. Г., Павельев А. А., Якубенко А. Е. Уравнения переноса для характеристик турбулентности: модели и результаты расчетов // В сб.: Итоги науки и техники. Сер. Механика жидкости и газа. — М.: ВИНТИ, 1988. — Т. 22. — С. 3—61.
5. Лущик В. Г., Павельев А. А., Якубенко А. Е. Трехпараметрическая модель турбулентности: расчет теплообмена // Изв. АН СССР. МЖГ. — 1986. — № 2. — С. 40—52.
6. Лущик В. Г., Якубенко А. Е. Дифференциальная модель турбулентности: численное исследование смешанной конвекции в вертикальных трубах // Изв. РАН. МЖГ. — 1996. — № 2. — С. 73—86.
7. Лущик В. Г., Якубенко А. Е. Численное исследование смешанной конвекции в вертикальных трубах в условиях неустойчивой стратификации // Изв. РАН. МЖГ. — 2002. — № 3. — С. 50—57.
8. Леонтьев А. И., Лущик В. Г., Макарова М. С. Численное исследование течения в трубе с отсосом газа через проницаемые стенки // Изв. РАН. МЖГ. — 2014. — № 3. — С. 74—81.
9. Makarova M. S., Lushchik V. G. Numerical simulation of turbulent flow and heat transfer in tube under injection of gas through permeable walls // Journal of Physics: Conference Series. — 2017. — Vol. 891, No. 1. — Article 012066.
10. Leontev A. I., Lushchik V. G., Makarova M. S. Heat and mass transfer in a tube with permeable walls: influence of suction and the Prandtl number // Proceedings of Eight International Symposium on Turbulence, Heat and Mass Transfer. — 2015. — P. 145—147.
11. Sternberg J. The transition from a turbulent to a laminar boundary layer. US Army Bal. Res. Lab. Aberdeen, USA, 1954. Rep. 906.
12. Гиневский А. С., Иоселевич В. А., Колесников А. В. и др. Методы расчета турбулентного пограничного слоя // Итоги науки и техники. Сер. Механика жидкости и газа. — М.: ВИНТИ. — 1978. — Т. 11. — С. 155—304.
13. Narasimha R., Sreenivasan K. R. Relaminarization fluid flows // Advances in Applied Mechanics. — 1979. — Vol. 19. — P. 221—309.
14. Кадер Б. А., Яглом А. М. Влияние шероховатости и продольного градиента давления на турбулентные пограничные слои // Итоги науки и техники. Сер. Механика жидкости и газа. — М.: ВИНТИ. — 1984. — Т. 18. — С. 3—111.
15. Moretti P. H., Kays V. M. Heat transfer in turbulent boundary layer with varying free-stream velocity and varying surface temperature — an experimental study // Int. J. Heat Mass Transfer. — 1965. — Vol. 8. — P. 1187—1202.
16. Sreenivasan K. R. Laminar-turbulent, relaminarizing and transitional flows // Acta Mech. — 1982. — Vol. 44. — P. 1—48.
17. Badry Narayanan M. A., Ramjee V. On the criteria for reverse transition in a two-dimensional boundary-layer flow // J. Fluid Mech. — 1969. — Vol. 35, No. 2. — P. 225—241.
18. Секундов А. Н. Применение дифференциального уравнения для турбулентной вязкости к анализу плоских неавтономных течений // Изв. АН СССР. МЖГ. — 1971. — № 5. — С. 114—127.
19. Volchkov E. P., Makarov M. S., Sakhnov A. Yu. Boundary layer with asymptotic favourable pressure gradient // Int. J. Heat Mass Transfer. — 2010. — Vol. 53. — P. 2837—2843.
20. Bourassa C., Thomas F. O. An experimental investigation of a highly accelerated turbulent boundary layer // J. Fluid Mech. — 2009. — No. 634. — P. 359—404.
21. Bankston C. A. The Transition From Turbulent to Laminar Gas Flow in a Heated Pipe // J. Heat Transfer. — 1970. — Vol. 92, No. 4. — P. 569—579.
22. Леонтьев А. И., Обливин А. Н., Романенко П. Н. Исследование сопротивления и теплообмена при турбулентном течении воздуха в осесимметричных каналах с продольным градиентом давления // ПМТФ. — 1961. — № 5. — С. 17—25.
23. Tanaka H., Shimizu Jun-ichi. Laminarization in Low Reynolds Number Turbulent Duct Flows // J. Heat Transfer. — 1977. — Vol. 99, No. 4. — P. 682—684.
24. Tanaka H., Kawamura H., Tateno A., Hatamiya S. Effect of Laminarization and Retransition on Heat Transfer for Low Reynolds Number Flow Through a Converging to Constant Area Duct // J. Heat Transfer. — 1982. — Vol. 104, No. 2. — P. 363—371.
25. Talamelli A., Fornaciari N., Johan K., Westin A., Alfredsson P. H. Experimental investigation of streaky structures in a relaminarizing boundary layer // J. Turbulence. — 2002. — No. 3. — P. 018.
26. Ichimiya M., Nakamura I., Yamashita S. Properties of a relaminarizing turbulent boundary layer under a favorable pressure gradient // Exp. Thermal and Fluid Science. — 1998. — Vol. 17, No. 1-2. — P. 37—48.
27. Ichimiya M., Nakase Y., Nakamura I., Yamashita S., Fukutomi J., Yoshikawa M. Properties of a relaminarizing turbulent boundary layer under a favorable pressure gradient (Analysis of bursting structure with VITA technique) // Transactions of the Japan Society of Mechanical Engineers, Part B. — 1998. — Vol. 62, N. 594. — P. 483—490.
28. Schlichting H. Boundary-Layer Theory. 7th edn. New York: McGraw-Hill, 1979.
29. Orijji U. R., Karimisani S., Tucker P. G. RANS modeling of accelerating boundary layers // Journal of Fluids Engineering, Transactions of the ASME. — 2015. — Vol. 137, No. 1. — Paper № A12.
30. Лущик В. Г., Макарова М. С., Решмин А. И. Ламинаризация потока при течении с теплообменом в плоском канале с конфузуром // Изв. РАН. МЖГ. — 2019. — № 1. — С. 68—77.
31. Kühnen J., Scarselli D., Schaner M., Hof B. Relaminarization by steady modification of the streamwise velocity profile in a pipe // Flow Turbulence Combust. — 2018. — No. 100. — P. 919—943.
32. Laws E. M., Livesey J. L. Flow through screens // Annu. Rev. Fluid Mech. — 1978. — Vol. 10, No. 1. — P. 247—266.
33. Lumley J. L., McMahon J. F. Reducing water tunnel turbulence by means of a honeycomb // J. Fluids Eng. — 1967. — Vol. 89, No. 4. — P. 764—770.
34. Навознов О. И., Павельев А. А. О переходе к турбулентности в спутных струях // Изв. АН СССР. МЖГ. — 1969. — № 6. — С. 131—136.
35. Навознов О. И., Павельев А. А., Яценко А. В. О переходе к турбулентности в затопленных и спутных струях // Изв. АН СССР. МЖГ. — 1972. — № 4. — С. 148—154.
36. Навознов О. И., Павельев А. А. Влияние начальных условий на течение осесимметричных спутных струй // Изв. АН СССР. МЖГ. — 1980. — № 4. — С. 18—24.
37. Коляда В. В., Павельев А. А. Влияние профиля скорости на входе в круглую трубу на переход к турбулентности // Изв. АН СССР. МЖГ. — 1986. — № 4. — С. 166—169.
38. Зайко Ю. С., Решмин А. И., Тепловодский С. Х., Чичерина А. Д. Исследование затопленных струй с увеличенной длиной начального ламинарного участка // Изв. РАН. МЖГ. — 2018. — № 1. — С. 97—106.
39. Zayko Ju., Teplovodskii S., Chicherina A., Vedenev V., Reshmin A. Formation of free round jets with long laminar regions at large Reynolds numbers // Phys. Fluids. — 2018. — Vol. 30, No. 4. — Article 043603.

40. Решмин А. И., Тепловодский С. Х., Трифонов В. В. Круглый короткий диффузор с большой степенью расширения и проницаемой перегородкой // Изв. РАН. МЖГ. — 2012. — № 5. — С. 32–39.
41. Kühnen J., Song B., Scarselli D., Budanur N. B., Riedl M., Willis A. P., Avila M., Hof B. Destabilizing turbulence in pipe flow // Nat. Phys. — 2018. — Vol. 14. — P. 386–390.
42. Eckhardt B. Introduction. Turbulence transition in pipe flow: 125th anniversary of the publication of Reynolds' paper // Phil. Trans. R. Soc. A. — 2009. — Vol. 367, Issue 1888. — P. 449–455.
43. Павельев А. А., Решмин А. И. Переход к турбулентности на начальном участке круглой трубы // Изв. РАН. МЖГ. — 2001. — № 4. — С. 113–121.
44. Лущик В. Г., Павельев А. А., Якубенко А. Е. Управление турбулентными пограничными слоями: результаты экспериментов и расчетные модели // Механика и научно-технический прогресс. Т. 2. Механика жидкости и газа. — М.: Наука, 1987. — С. 67–89.
45. Решмин А. И., Трифонов В. В., Лущик В. Г., Макарова М. С. О возможности снижения трения за счет ламинаризации течения в трубе // Физико-химическая кинетика в газовой динамике. — 2019. — Т. 20, № 2. — С. 1–9.
46. Мигай В. К. Повышение эффективности современных теплообменников. — Л.: Энергия, 1980. — 144 с.
47. Калинин Э. К., Дрейцер Г. А., Копп И. З., Мякочин А. С. Эффективные поверхности теплообмена. — М.: Энергоатомиздат, 1998. — 408 с.
48. Дзюбенко Б. В., Кузма-Кичта Ю. А., Леонтьев А. И. и др. Интенсификация тепло- и массообмена в макро-, микро- и наномасштабах. — М.: ФГУП "ЦНИАТОМИНФОРМ", 2008. — 532 с.
49. Bergles A. E. Recent developments in enhanced heat transfer // Heat Mass Transfer. — 2011. — Vol. 47, No. 8. — Article 1001.
50. Leontiev A. I., Kiselev N. A., Burtsev S. A., Strongin M. M., Vinogradov Yu. A. Experimental investigation of heat transfer and drag on surfaces with spherical dimples // Experimental Thermal and Fluid Science. — 2016. — Vol. 79. — P. 74–84.
51. Решмин А. И., Тепловодский С. Х., Трифонов В. В. Турбулентное течение в круглом безотрывном диффузоре при числах Рейнольдса, меньших 2000 // Изв. РАН. МЖГ. — 2011. — № 2. — С. 121–130.
52. Лущик В. Г., Решмин А. И. Особенности турбулентного течения в круглых каналах с уменьшающимся по длине числом Рейнольдса // Изв. РАН. МЖГ. — 2015. — № 2. — С. 76–82.
53. Леонтьев А. И., Лущик В. Г., Решмин А. И. Теплообмен в конических расширяющихся каналах // ТВТ. — 2016. — Т. 54, № 2. — С. 287–293.
54. Лущик В. Г., Решмин А. И. Интенсификация теплообмена в плоском безотрывном диффузоре // ТВТ. — 2018. — Т. 56, № 4. — С. 586–593.
55. Лущик В. Г., Макарова М. С., Медвецкая Н. В., Решмин А. И. Численное исследование течения и теплообмена в плоских каналах переменного сечения // Тепловые процессы в технике. — 2019. — Т. 11, № 9. — С. 386–394.
56. Nunner W. Wärmeübergang und Druckabfall in rauen Röhren // VDI Forschungsheft. — 1956. — No. 455. — P. 5.
57. Справочник по теплообменникам в 2-х томах. Перевод с англ. М.: Энергоатомиздат, 1987.
58. Лущик В. Г., Макарова М. С., Решмин А. И. Пластинчатый теплообменник с диффузорными каналами // ТВТ. — 2020. — Т. 58, № 3. — С. 376–383.

Application of the Three-Parameter Differential Model of Turbulence for Solving Problems of Flow and Heat Transfer in Channels of Variable Cross-Section. Part 2

V. G. Lushchik, vgl_41@mail.ru, M. S. Makarova, mariia.makarova@gmail.com,
A. I. Reshmin, alexreshmin@rambler.ru, Lomonosov Moscow State University, Institute of mechanics,
Moscow, 119192, Russian Federation

Corresponding author:

Makarova Mariia S., Researcher, Lomonosov Moscow State University, Institute of mechanics, Moscow, 119192, Russian Federation
E-mail: mariia.makarova@gmail.com

Received on September 18, 2020

Accepted November 24, 2020

A description of the method of numerical study in the approximation of a narrow channel of the problems of flow and heat transfer in flat and circular channels of variable cross-section using a differential three-parameter model of shear turbulence is presented. The main results of numerous studies using the proposed method are described, one of the goals of which was to substantiate the possibility of using the narrow channel approximation. This review study is carried out in two parts. In the second part the results of the study of laminarization during flow in the confuser and the pipe, heat transfer intensification during flow in diffusers and in a plate heat exchanger with diffuser channels are presented.

Keywords: differential turbulence model, diffuser, confuser, pressure gradient, blowing, suction, laminarization of the flow, intensification of heat transfer

Acknowledgements:

This work was partially supported by the state budget (AAAA-A16-116021110203-6, AAAA-A19-119012990110-0; AAAA-A16-116021110200-5; AAAA-A19-119012990115-5) and the Grant Council of the President of the Russian Federation (СП-3993.2018.1).

For citation:

Lushchik V. G., Makarova M. S., Reshmin A. I. Application of the Three-Parameter Differential Model of Turbulence for Solving Problems of Flow and Heat Transfer in Channels of Variable Cross-Section. Part 2, *Programmnaya Ingeneria*, 2021, vol. 12, no. 2, pp. 89–106

DOI: 10.17587/prin.12.89-106

References

1. **Lushchik V. G., Makarova M. S., Reshmin A. I.** Application of the Three-Parameter Differential Model of Turbulence for Solving Problems of Flow and Heat Transfer in Channels of Variable Cross-Section: Part 1, *Programmnaya Ingeneria*, 2020, vol. 12, no. 1, pp. 21–30 (in Russian).
2. **Lushchik V. G., Makarova M. S., Yakubenko A. E.** Application of the Three-Parameter Model of Shear Turbulence for Solving Problems of External Flowing on Permeable Surfaces by a Compressible Gas Flow, *Programmnaya Ingeneria*, 2017, vol. 8, no. 12, pp. 563–574 (in Russian).
3. **Lushchik V. G., Pavel'ev A. A., Yakubenko A. E.** Three-parameter model of shear turbulence, *Fluid Dynamics*, 1978, vol. 13, no. 3, pp. 350–360.
4. **Lushchik V. G., Pavel'ev A. A., Yakubenko A. E.** The transport equations for turbulence characteristics: models and calculation results, *Itogi Nauki i Tekhniki. Ser. Mechanics of fluid and gas*, Moscow, VINITI, 1988, vol. 22, pp. 3–61 (in Russian).
5. **Lushchik V. G., Pavel'ev A. A., Yakubenko A. E.** Three-parameter model of turbulence: Heat transfer calculations, *Fluid Dynamics*, 1986, vol. 21, no. 2, pp. 200–211.
6. **Lushchik V. G., Yakubenko A. E.** Differential model of turbulence: A numerical study of mixed convection in vertical pipes, *Fluid Dynamics*, 1996, vol. 31, no. 2, pp. 73–86.
7. **Lushchik V. G., Yakubenko A. E.** Numerical Study of Mixed Convection in Vertical Tubes under Conditions of Unstable Stratification, *Fluid Dynamics*, 2002, vol. 37, no. 3, pp. 396–402.
8. **Leont'ev A. I., Lushchik V. G., Makarova M. S.** Numerical investigation of tube flow with suction through permeable walls, *Fluid Dynamics*, 2014, vol. 49, no. 3, pp. 362–368.
9. **Makarova M. S., Lushchik V. G.** Numerical simulation of turbulent flow and heat transfer in tube under injection of gas through permeable walls, *Journal of Physics: Conference Series*, 2017, vol. 891, no. 1, article 012066.
10. **Leontiev A. I., Lushchik V. G., Makarova M. S.** Heat and mass transfer in a tube with permeable walls: influence of suction and the Prandtl number, *Proceedings of Eight International Symposium on Turbulence, Heat and Mass Transfer*, 2015, pp. 145–147.
11. **Sternberg J.** The transition from a turbulent to a laminar boundary layer. US Army Bal. Res. Lab. Aberdeen, USA, 1954. Rep. 906.
12. **Ginevsky A. S., Ioselevich V. A., Kolesnikov A. V., Lapin Yu. V., Pilipenko V. N., Secundov A. N.** Methods for calculating the turbulent boundary layer, *Itogi Nauki i Tekhniki. Ser. Mechanics of fluid and gas*. Moscow, VINITI, 1978, vol. 11, pp. 155–304 (in Russian).
13. **Narasimha R., Sreenivasan K. R.** Relaminarization fluid flows, *Advances in Applied Mechanics*, 1979, vol. 19, pp. 221–309.
14. **Kader B. A., Yaglom A. M.** Influence of roughness and longitudinal pressure gradient on turbulent boundary layers, *Itogi Nauki i Tekhniki. Ser. Mechanics of fluid and gas*, Moscow, VINITI, 1984, vol. 18, pp. 3–111 (in Russian).
15. **Moretti P. H., Kays V. M.** Heat transfer in turbulent boundary layer with varying free-stream velocity and varying surface temperature — an experimental study, *Int. J. Heat Mass Transfer*, 1965, vol. 8, pp. 1187–1202.
16. **Sreenivasan K. R.** Laminarescent, relaminarizing and retransitional flows, *Acta Mech.*, 1982, vol. 44, pp. 1–48.
17. **Badry Narayanan M. A., Ramjee V.** On the criteria for reverse transition in a two-dimensional boundary-layer flow, *J. Fluid Mech.*, 1969, vol. 35, no. 2, pp. 225–241.
18. **Secundov A. N.** Application of a differential equation for turbulent viscosity to the analysis of plane non-self-similar flows, *Fluid Dynamics*, 1971, vol. 6, no. 5, pp. 828–840.
19. **Volchkov E. P., Makarov M. S., Sakhnov A. Yu.** Boundary layer with asymptotic favourable pressure gradient, *Int. J. Heat Mass Transfer*, 2010, vol. 53, pp. 2837–2843.
20. **Bourassa C., Thomas F. O.** An experimental investigation of a highly accelerated turbulent boundary layer, *J. Fluid Mech.*, 2009, no. 634, pp. 359–404.
21. **Bankston C. A.** The Transition from Turbulent to Laminar Gas Flow in a Heated Pipe, *J. Heat Transfer*, 1970, vol. 92, no. 4, pp. 569–579.
22. **Leontiev A. I., Oblivin A. N., Romanenko P. N.** Investigation of resistance and heat transfer in turbulent air flow in axisymmetric channels with a longitudinal pressure gradient, *J. Applied Mechanics and Technical Physics*, 1961, no. 5, pp. 17–25 (in Russian).
23. **Tanaka H., Shimizu Jun-ichi.** Laminarization in Low Reynolds Number Turbulent Duct Flows, *J. Heat Transfer*, 1977, vol. 99, no. 4, pp. 682–684.
24. **Tanaka H., Kawamura H., Tateno A., Hatamiya S.** Effect of Laminarization and Retransition on Heat Transfer for Low Reynolds Number Flow Through a Converging to Constant Area Duct, *J. Heat Transfer*, 1982, vol. 104, no. 2, pp. 363–371.
25. **Talamelli A., Fornaciari N., Johan K., Westin A., Alfredsson P. H.** Experimental investigation of streaky structures in a relaminarizing boundary layer, *J. Turbulence*, 2002, no. 3, article 018.
26. **Ichimiya M., Nakamura I., Yamashita S.** Properties of a relaminarizing turbulent boundary layer under a favorable pressure gradient, *Exp. Thermal and Fluid Science*, 1998, vol. 17, no. 1–2, pp. 37–48.
27. **Ichimiya M., Nakase Y., Nakamura I., Yamashita S., Fukutomi J., Yoshikawa M.** Properties of a relaminarizing turbulent boundary layer under a favorable pressure gradient (Analysis of bursting structure with VITA technique), *Transactions of the Japan Society of Mechanical Engineers, Part B*, 1998, vol. 62, no. 594, pp. 483–490.
28. **Schlichting H.** *Boundary-Layer Theory*. 7th edn, New York, McGraw-Hill, 1979.
29. **Orijji U. R., Karimisani S., Tucker P. G.** RANS modeling of accelerating boundary layers, *Journal of Fluids Engineering, Transactions of the ASME*, 2015, vol. 137, no. 1, paper № A12.
30. **Lushchik V. G., Makarova M. S., Reshmin A. I.** Laminarization of Flow with Heat Transfer in a Plane Channel with a Confuser, *Fluid Dynamics*, 2019, vol. 54, no. 1, pp. 67–76.
31. **Kühnen J., Scarselli D., Schaner M., Hof B.** Relaminarization by steady modification of the streamwise velocity profile in a pipe, *Flow Turbulence Combust.*, 2018, no. 100, pp. 919–943.
32. **Laws E. M., Livesey J. L.** Flow through screens, *Annu. Rev. Fluid Mech.*, 1978, vol. 10, no. 1, pp. 247–266.
33. **Lumley J. L., McMahan J. F.** Reducing water tunnel turbulence by means of a honeycomb, *J. Fluids Eng.*, 1967, vol. 89, no. 4, pp. 764–770.
34. **Navoznov O. I., Pavel'ev A. A.** Transition to turbulence in coflowing jets, *Fluid Dynamics*, 1969, vol. 4, no. 6, pp. 84–88.
35. **Navoznov O. I., Pavel'ev A. A., Yatsenko A. V.** The transition to turbulence in submerged jets and wakes, *Fluid Dynamics*, 1972, vol. 7, no. 4, pp. 672–678.
36. **Navoznov O. I., Pavel'ev A. A.** Influence of the initial conditions on axisymmetric jets in a parallel flow, *Fluid Dynamics*, 1980, vol. 15, no. 4, pp. 488–493.
37. **Kolyada V. V., Pavel'ev A. A.** Effect of the velocity profile at the inlet to a circular pipe on the transition to turbulence, *Fluid Dynamics*, 1986, vol. 21, no. 4, pp. 650–653.
38. **Zayko Yu. S., Reshmin A. I., Teplovodskii S. Kh., Chicherina A. D.** Investigation of Submerged Jets with an Extended Initial Laminar Region, *Fluid Dynamics*, 2018, vol. 53, no. 1, pp. 95–104.
39. **Zayko Ju., Teplovodskii S., Chicherina A., Vedenev V., Reshmin A.** Formation of free round jets with long laminar regions at large Reynolds numbers, *Phys. Fluids.*, 2018, vol. 30, article 043603.
40. **Reshmin A. I., Teplovodskii S. Kh., Trifonov V. V.** Short round diffuser with a high area ratio and a permeable partition, *Fluid Dynamics*, 2012, vol. 47, no. 5, pp. 583–589.
41. **Kühnen J., Song B., Scarselli D., Budanur N. B., Riedl M., Willis A. P., Avila M., Hof B.** Destabilizing turbulence in pipe flow, *Nat. Phys.*, 2018, vol. 14, pp. 386–390.

42. **Eckhardt B.** Introduction. Turbulence transition in pipe flow: 125th anniversary of the publication of Reynolds' paper, *Phil. Trans. R. Soc. A.*, 2009, vol. 367, issue 1888, pp. 449–455.

43. **Pavelyev A. A., Reshmin A. I.** Turbulent Transition in the Inlet Region of a Circular Pipe, *Fluid Dynamics*, 2001, vol. 36, no. 4, pp. 626–633.

44. **Lushchik V. G., Pavel'ev A. A., Yakubenko A. E.** Turbulent boundary layer control: experimental results and computational models, *Mekhanika i nauchno-tekhnicheskij progress. T. 2. Mekhanika zhidkosti i gaza*, Moscow, Nauka, 1987, pp. 67–89 (in Russian).

45. **Reshmin A. I., Trifonov V. V., Lushchik V. G., Makarova M. S.** On the Possibility of Reducing Friction due to Relaminarization of the Flow in the Pipe, *Fiziko-himicheskaya kinetika v gazovoj dinamike*, 2019, vol. 20, no. 2, pp. 1–9 (in Russian).

46. **Migai V. K.** *Improving the efficiency of modern heat exchangers*, Leningrad, Energy, 1980. 144 p. (in Russian).

47. **Kalinin E. K., Dreitser G. A., Kopp I. Z., Myakochin A. S.** *Effective surfaces of heat transfer*, Moscow, Energoatomizdat, 1998, 408 p. (in Russian).

48. **Dzubenko B. V., Kuzma-Kichta Yu. A., Leontiev A. I.** *Intensification of heat and mass transfer in macro-, micro- and nanoscales*, Moscow, FGUP Tsniiatominform, 2008, 532 p. (in Russian).

49. **Bergles A. E.** Recent developments in enhanced heat transfer, *Heat Mass Transfer*, 2011, vol. 47, no. 8, article 1001.

50. **Leontiev A. I., Kiselev N. A., Burtsev S. A., Strongin M. M., Vinogradov Yu. A.** Experimental investigation of heat

transfer and drag on surfaces with spherical dimples, *Experimental Thermal and Fluid Science*, 2016, vol. 79, pp. 74–84.

51. **Reshmin A. I., Teplovodskii S. Kh., Trifonov V. V.** Turbulent flow in a circular separationless diffuser at Reynolds numbers smaller than 2000, *Fluid Dynamics*, 2011, vol. 46, no. 2, pp. 278–285.

52. **Lushchik V. G., Reshmin A. I.** Distinctive features of turbulent flows in round channels with the Reynolds number decreasing along the length, *Fluid Dynamics*, 2015, vol. 50, no. 2, pp. 223–228.

53. **Leontiev A. I., Lushchik V. G., Reshmin A. I.** Heat transfer in conical expanding channels, *High Temperature*, 2016, vol. 54, no. 2, pp. 270–276.

54. **Lushchik V. G., Reshmin A. I.** Heat Transfer Enhancement in a Plane Separation-free Diffuser, *High Temperature*, 2018, vol. 56, no. 4, pp. 569–575.

55. **Lushchik V. G., Makarova M. S., Medvetskaya N. V., Reshmin A. I.** Numerical Investigation of Flow and Heat Transfer in Flat Channels of Variable Cross-Section, *Teplovye processy v tekhnike*, 2019, vol. 11, no. 9, pp. 386–394 (in Russian).

56. **Nunner W.** *Wärmeübergang und Druckabfall in rauhen Röhren*, VDI Forschungsheft, 1956, no. 455, 5 p.

57. **Handbook** on heat exchangers in 2 vol., Moscow, Energoatomizdat, 1987 (in Russian).

58. **Lushchik V. G., Makarova M. S., Reshmin A. I.** Plate Heat Exchanger with Diffuser Channels, *High Temperature*, 2020, vol. 58, no. 3, pp. 352–359.

ИНФОРМАЦИЯ

22—23 апреля 2021 г. состоится

Отраслевая научно-техническая конференция

приборостроительных организаций ГК "РОСКОСМОС"

"ИНФОРМАЦИОННО-УПРАВЛЯЮЩИЕ И ИЗМЕРИТЕЛЬНЫЕ СИСТЕМЫ",

посвященная 55-й годовщине образования АО "НПО ИТ"

Конференция проводится АО "Научно-производственное объединение измерительной техники" (АО "НПО ИТ") при поддержке государственной корпорации по космической деятельности "Роскосмос" и АО "Российские космические системы".

Заседания конференции будут проходить в АО "НПО ИТ" по адресу: Московская область, г. Королёв, ул. Пионерская, дом 2.

Тематика научно-технической конференции

1. Состояние и основные направления развития телеметрических и информационно-управляющих систем
2. Новое поколение интеллектуальной датчиково-преобразующей и управляющей аппаратуры. Теория и принципы построения
3. Информационно-измерительное обеспечение летных испытаний
4. Волоконно-оптические гироскопы и приборы на их основе
5. Перспективы развития средств выведения и наземной космической инфраструктуры
6. Новые конструкции, технологии и материалы. Качество производства продукции приборостроительной отрасли
7. Двойные технологии. Техническое перевооружение приборостроительных организаций "Роскосмоса"

Адрес оргкомитета:

Российская Федерация, 141074, Московская область, г. Королёв, ул. Пионерская, дом 2, Акционерное общество "Научно-производственное объединение измерительной техники" (АО "НПО ИТ")

Контактные телефоны:

+7(499)750-40-50, доб. 11-66 Ачкасов Виталий Анатольевич
+7(499)750-40-50, доб. 10-01 Сабко Владимир Леонидович
+7(499)750-40-50, доб. 10-29 Вербицкая Карина Викторовна

И. Е. Харлампенков, науч. сотр., harlampenkov@ict.sbras.ru,
А. Ю. Ощепков, аспирант, aosivt@gmail.com, Федеральное государственное бюджетное научное учреждение "Федеральный исследовательский центр информационных и вычислительных технологий", г. Новосибирск

Преобразование и отображение расчетных данных оптико-электронных данных дистанционного зондирования Земли на основе расширений GeoServer и распределенных технологий хранения

Рассмотрен способ визуализации данных спутниковых снимков с помощью служб программного обеспечения GeoServer и распределенных технологий хранения. Особенностью результатов, представленных в статье, является подход, основанный на визуализации данных посредством преобразования колоночного хранилища Apache Parquet в растр формата BIL и сохранение его в виде временного файла. Такой подход позволяет обеспечить взаимодействие с распределенной файловой системой через библиотеку Kite SDK и включить дополнительные обработчики данных, основанные на технологии Apache Hadoop в качестве внешних сервисов.

Ключевые слова: GeoServer, WMS Extensions, Apache Hadoop, HDFS, Apache Parquet, Kite SDK, Java, GDAL, ENVI BIL, распределенные информационные системы

Введение

На основе проведенных ранее исследований, касающихся хранения данных современных космических аппаратов (КА) [1], выявлена проблема представления результатов в виде набора карт в связи с подходом хранения спектральных данных дистанционного зондирования (ДДЗ), для которого не реализован, на текущий момент, механизм отображения. Соответственно принято решение разработать компонент информационной системы, который позволит считывать данные из колоночного Parquet-файла в распределенной файловой системе и отображать их для внешних ресурсов.

В настоящее время существует программное обеспечение, предоставляющее возможности отображения геопространственных данных в виде встроенных в настольное приложение карт или опубликованное в виде внешних сервисов. В его состав входят:

- QGIS Server — модуль, позволяющий организовать доступ к растровым геоданным на основе стандарта Web Map Service (WMS) [2];
- MapVox — внешний сервис, предоставляющий возможности создания пользовательских карт и интегрирования их на веб-сайты и мобильные приложения с использованием данных Open Street Map [3];
- MapTiler — набор внешних сервисов для формирования отображения векторных данных [4];
- uDig — комплекс решений [5];

- Mapnik — открытая библиотека для рендеринга растровых карт из векторных данных [6];
- MapServer [7];
- GeoServer [8];
- ESRI Geoportal Server [9];
- Geoclip [10];
- MapLarge, работающий в качестве внешнего сервиса [11];
- Map Kit Framework — библиотека формирования отображения карт для платформ семейства Apple [12];
- GeniusGis [13].

Следует также отметить проекты, которые непосредственно реализуют инфраструктуру для решения задач, подобных поставленной в данной статье, но ограниченных коммерческой лицензией или по технологическому стеку. Однако эти ограничения не позволяют оперативно отображать результаты обработки в силу необходимости дополнительных преобразований на этапе конвертации в один из широко применяемых форматов (например, BIL [14]). К числу таких проектов, в частности, можно отнести перечисленные далее.

- GeoMesa — набор инструментов с открытым исходным кодом, который позволяет проводить крупномасштабную геопространственную аналитику в облачных и распределенных вычислительных системах. Обеспечивает пространственно-временную сохраняемость данных поверх распределенных столбцово-ориентированных баз данных Accumulo, HBase и Cassandra [15].

Сравнительный анализ программного обеспечения

Наименование программного обеспечения	Критерий				
	1	2	3	4	5
QGis Server	+	-	+	+	-
MapBox	-	-	+	+	+
MapTiler	-	-	-	+	+
uDig	+/-	+/-	+	+	+
Mapnik	-	-	-	+	-
GeoServer	+	+	+	+	+
ESRI Geoportal Server	+	-	+	+	+
Geoclip	-	-	-	+	-
MapLarge	-	+/-	+	+	+
Map Kit Framework	-	-	-	+	+
GeniusGis	-	-	-	+	-
MapServer	+	+/-	+	+	+

• GeoWave, который построен на основе модели расширяемости проекта GeoTools, что означает возможность интегрировать его непосредственно с любым GeoTools-совместимым проектом. GeoWave обеспечивает стандартную поддержку распределенных хранилищ ключ—значение [16].

• GeoTrellis — каркас, построенный на основе Apache Spark для работы с растровыми данными [17].

• GIS Tools for Hadoop — набор инструментов с открытым исходным кодом, который позволяет проводить пространственный анализ больших данных, но в основе своей направлен на использование в проектах Esri [18].

Таким образом, рассмотренные проекты нацелены на построение информационной системы, позволяющей проводить анализ спектральных ДДЗ или их представление. Однако они не предоставляют функциональные возможности отображения растровых данных из файлов формата Parquet в виде внешних сервисов. В связи с изложенным выше задача по реализации компонента отображения спектральных ДДЗ из колоночного хранилища, содержащего данные и метаданные спектральных снимков, является актуальной и ее решение востребовано.

Постановка задачи

Для внедрения возможности отображения полученных результатов обработки растровых данных спектральных спутниковых снимков, которые хранятся в Apache Parquet, были сформулированы следующие задачи:

- 1) провести сравнительный анализ программных компонентов, реализующих отображение геоданных;
- 2) определить требования к разрабатываемому модулю;
- 3) реализовать сервис отображения растровых данных, сохраненных в колоночном файле хранилища Apache Parquet, и провести его апробацию.

Анализ программного обеспечения

В рамках решения отмеченных выше задач проведено сравнение существующего программного обеспечения (ПО) по следующим критериям:

- 1) работа с растровыми данными;
- 2) работа с распределенными хранилищами;
- 3) модульность системы в части расширения;
- 4) предоставление возможности работы в качестве внешнего сервиса;
- 5) возможность удаленного управления по стандартным протоколам (например, HTTP, SOAP).

В табл. 1 приведены результаты сравнения 12 наименований ПО, где "+" — ПО удовлетворяет критерию, "-" — ПО не удовлетворяет критерию и "+/-" — ПО предоставляет возможности удовлетворить критерий, но не собственными инструментами.

На основе табл. 1 и анализа приведенных выше проектов [15—18] авторами принято решение использовать для разработки модуля отображения растровых данных спутниковых снимков

и результатов их обработки, хранящихся в файле Apache Parquet, возможности GeoServer 2.16. Сформулированы следующие требования к разрабатываемому расширению для GeoServer:

- 1) выгрузка растровых данных из файла Parquet заданного формата в распределенной файловой системе HDFS;
- 2) возможность удаленного управления отображением посредством REST-запросов;
- 3) работа модуля в качестве внешнего сервиса для предоставления бесшовной интеграции с другими облачными решениями.

Программная реализация

Для начала рассмотрим бинарный формат Apache Parquet. Он состоит из набора бинарных файлов и объединяющего их метаописания, сохраненного в текстовый файл. Данные в описываемом формате представлены в виде колоночного хранилища.

В программной реализации, предлагаемой в статье, используется формат хранения данных, на основе которого будет реализовано расширение растрового хранилища GeoServer. Ранее [1] предлагалась схема колоночного хранилища растровых данных мультиспектральных снимков. Для возможности позиционирования отображаемых растровых данных необходимо расширить ее следующими параметрами:

- width — число колонок в снимке;
- height — число строк в снимке;
- projection — определения системы координат;
- geoTransform — вращение относительно системы координат.

```

{ "type": "record",
  "name": "SatelliteImage",
  "namespace": "aos.model",
  "fields": [
    { "name": "rowId", "type": "int" },
    { "name": "width", "type": "int" },
    { "name": "height", "type": "int" },
    { "name": "projection", "type": "string" },
    { "name": "geoTransform", "type": { "type": "array", "items": "double" }, "default": [0] },
    { "name": "dataDeepBlue", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataBlue", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataGreen", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataRed", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataNIR", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataSWIR2", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataSWIR3", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataSWIR1", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataCirrus", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataTex", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataTIRS1", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataTIRS2", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataVCID1", "type": { "type": "array", "items": "float" }, "default": [0] },
    { "name": "dataVCID2", "type": { "type": "array", "items": "float" }, "default": [0] }
  ]
}

```

Рис. 1. Метаописание схемы хранения данных в Parquet-файле

Модифицированная схема представлена на рис. 1. Следует также отметить работу по отображению растровых данных сервером GeoServer. Она включает в себя последовательность действий по формированию позиционирования отображения, созданию хранилища, которое будет содержать бинарный файл снимка, и заданию способа отображения посредством слоя или группы слоев. Таким образом, работа модулей GeoServer (с растровыми данными) сводится к выполнению описанной последовательности. На основе изложенного выше предложен следующий алгоритм разрабатываемого модуля для GeoServer 2.16 (рис. 2).

Шаг 1. Передача REST-запроса разработанному модулю и преобразование его в набор параметров, содержащих полный путь до Parquet-файла предлагаемой схемы, представленной на рис. 1.

Шаг 2. Выделение параметров запроса.

Шаг 3. Проверка существования преобразованного файла.

Шаг 4. Считывание Parquet-файла.

Шаг 5. Построчная итерация считанных данных.

Шаг 6. Создание объекта, содержащего метаданные и данные снимка: расположение в координатной

сетке, число столбцов, строк и значение свечения точек (пикселей) в итоговом растровом файле.

Шаг 7. Завершение итерации и сохранение данных.

Следует отметить, что объект итогового файла представляет собой экземпляр класса DataSet

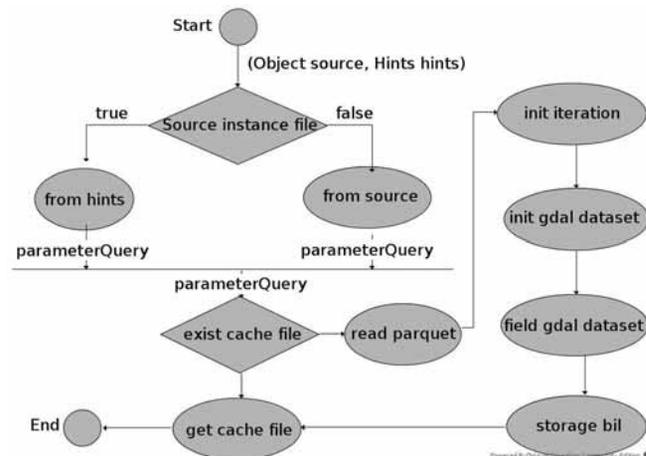


Рис. 2. Алгоритм работы расширения

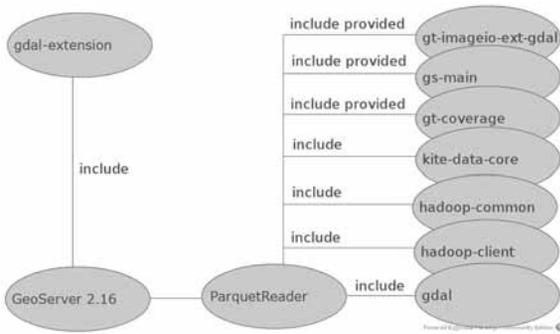


Рис. 3. Общая схема интеграции расширения

библиотеки GDAL [19], который дает возможность создавать различные бинарные файлы изображений в среде операционной системы. Считывание Parquet-файла из распределенной файловой системы проводится посредством Kite SDK [20].

Итоговый файл представляет собой изображение (растровые данные) для хранилища GeoServer, а также выполняет задачу "горячих данных", которые позволяют снизить расчетную нагрузку на сервис.

Программная реализация включает в себя зависимости и библиотеки, представленные на рис. 3.

Таким образом, для включения разработанного модуля (ParquetReader, GGRP) в работу GeoServer необходимо выполнить следующие действия:

1) собрать и установить в операционную систему библиотеки GDAL и расширения с используемой JRE для запуска GeoServer;

- 2) включить расширение в GeoServer gdal-extensions;
- 3) включить расширение в GeoServer ParquetReader.

По завершению установки разработанного расширения в оснастке GeoServer 2.16 для создания хранилища появится пункт GGRP — RasterParquet with GeoTrans (рис. 4).

Апробация

В качестве примера авторами был выбран набор растровых данных, составленный из снимка КА Landsat 7 и сохраненный в файл Parquet, расположенный в распределенной файловой системе HDFS.

Использование сервиса на основе разработанного модуля сводится к выполнению следующей серии запросов:

- создание рабочей области (рис. 5);
- создание хранилища (рис. 6);
- создание слоя (рис. 7).

Запросы, представленные на рис. 5–7, являются типовыми запросами GeoServer 2.16. Следует отметить запрос на создание хранилища данных геосервера (см. рис. 6) и, в частности, параметр "url", который содержит:

- адрес геосервера (<http://localhost:8080/geoserver>);
- тип файловой системы (HDFS);
- адрес расположения Parquet-файла (`cluster@localhost:8020/user/claster/temp/parquet/ndvi/1/LE71310222013244EDC00`);
- имя Parquet-файла (`LE71310222013244EDC00.parquet`).

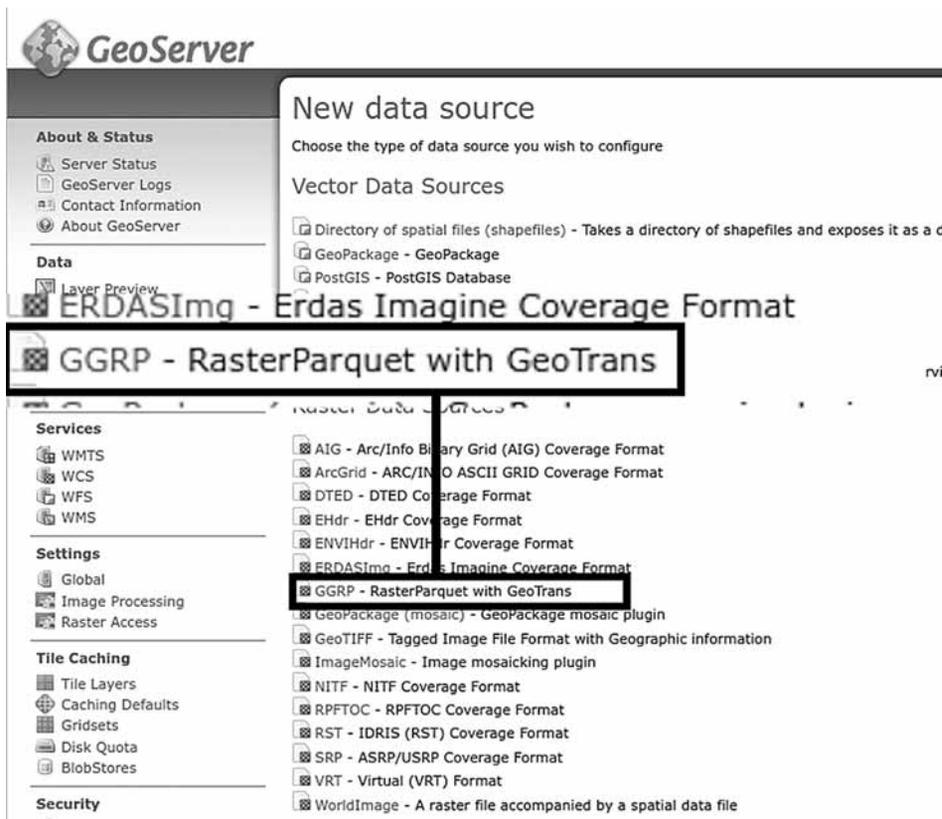


Рис. 4. Оснастка GeoServer для добавления нового хранилища данных

```
curl -d '{"workspace":{"name":"wsparquet"}}'
-H "Content-Type: application/json"
-u admin:geoserver
-X POST http://localhost:8080/geoserver/rest/workspaces
```

Рис. 5. Запрос на создание рабочей области

```
curl -d '{
  "coverageStore":{
    "workspace":"wsparquet",
    "name":"parquet",
    "enabled":true,
    "type":"GGRP",
    "url":"http://localhost:8080/geoserver?
      hdfs&cluster@localhost:8020/user/cluster/temp/parquet/ndvi/1
      /LC08_L1TP_142020_20190907_01_RT&LC08_L1TP_142020_20190907_01_RT.parquet"
  }
}'
-H "Content-Type: application/json"
-u admin:geoserver
-X POST http://localhost:8080/geoserver/rest/workspaces/wsparquet/coveragestores
```

Рис. 6. Запрос на создание хранилища данных в GeoServer

```
curl -d '{ "coverage":{
  "name":"parquet",
  "title":"parquet",
  "srs":"EPSG:4326"
}}'
-H "Content-Type: application/json"
-u admin:geoserver
-X POST
http://localhost:8080/geoserver/rest/workspaces/wsparquet/coveragestores/parquet/coverages
```

Рис. 7. Запрос на создание слоя

На рис. 8, см. третью сторону обложки, представлен результат работы WMS-службы GeoServer 2.16 на основе растровых данных, полученных в итоге серии запросов к разработанному модулю.

Аналогично проведена работа разработанным модулем по выгрузке растровых данных Sentinel-2A (рис. 9, см. третью сторону обложки).

Также воссозданные бинарные файлы имеют привязку к системе координат, как показано на рис. 10, 11, см. третью сторону обложки.

Результаты качественных характеристик работы модуля GGRP представлены в табл. 2, где:

- 1) наименование Parquet-файла, содержащего растровые данные снимка;
- 2) объем занимаемого места Parquet-файла, содержащего растровые данные всего снимка, Мбайт;
- 3) объем занимаемого места временного бинарного представления в формате ENVI BIL конкретного канала спектрального снимка, Мбайт;
- 4) время создания временного бинарного представления, с.

Таким образом, данное расширение предоставляет возможность выгрузки растровых данных из распределенной файловой системы на сторону гео-

Таблица 2

Качественные характеристики использования модуля GGRP

1	2	3	4
LE71310222013244EDC00.parquet	259,8	482,2	8
S2A_OPER_MSI_LIC_TL_SGS__20160106T092224_A002819_T44UPF_N02_01.parquet	187,5	256,4	7

сервера и отображения полученных результатов в соответствии со стандартом WMS. Стоит отметить, что выгрузка происходит на первый запрос, затем будет использоваться сформированный бинарный файл.

Вывод

Результаты, представленные в статье, демонстрируют возможности использования библиотек KiteSDK и расширения геосервера с кешированием полученных результатов в виде ENVI BIL- формата. Такое расширение может эффективно применяться к отображению различных растровых ДДЗ, представленных в виде Parquet-файла, расположенного в распределенной файловой системе, когда необходимо использовать программные механизмы представления полученных результатов обработки больших объемов информации, сохраняя при этом высокий уровень доступности к данным.

Список литературы

1. Потапов В. П., Попов С. Е., Ощепков А. Ю. Хранение и обработка данных спутниковых мульти- и гиперспектральных снимков на основе формата Apache Parquet // Программная инженерия. — 2018. — Т. 9, № 3. — С. 123—131.
2. Документация модуля QGIS Server. URL: https://docs.qgis.org/3.4/ru/docs/training_manual/qgis_server/index.html/ (дата обращения: 11.09.2020).
3. Документация MapBox. URL: <https://docs.mapbox.com/help/tutorials/> (дата обращения: 11.09.2020).
4. Документация MapTiler Server. URL: <https://support.maptiler.com/s9-server> (дата обращения: 11.09.2020).
5. Документация uDig. URL: <http://www.refractive.net/products/udig/> (дата обращения: 11.09.2020).
6. Введение в SAGA GIS. URL: <http://www.saga-gis.org/en/index.html> (дата обращения: 11.09.2020).
7. Введение в MapServer. URL: <http://download.osgeo.org/mapserver/docs/MapServer.pdf> (дата обращения: 11.09.2020).
8. Документация GeoServer. URL: <http://geoserver.org/about/> (дата обращения: 11.09.2020).
9. Представление Geoportal server. URL: <https://www.esri.com/en-us/arcgis/products/geoportal-server/overview> (дата обращения: 11.09.2020).
10. Учебное пособие GeoClip. URL: <https://www.geoclip.fr/documentation/?lang=en#guides> (дата обращения: 11.09.2020).
11. Учебное пособие MapLarge. URL: <https://maplarge.com/tutorial> (дата обращения: 11.09.2020).
12. Учебное пособие Map Kit Framework. URL: <https://developer.apple.com/documentation/mapkit> (дата обращения: 11.09.2020).
13. GeniusGis. URL: <https://www.geniusgis.com/> (дата обращения: 11.09.2020).
14. Документация форматов ENVI. URL: <https://www.harrisgeospatial.com/docs/ENVIImageFiles.html> (дата обращения: 11.09.2020).
15. Документация GeoMesa. URL: <https://www.geomesa.org/documentation/stable/user/index.html> (дата обращения: 11.09.2020).
16. Документация GeoWave. URL: <https://s3.amazonaws.com/geowave/latest/docs/userguide.pdf> (дата обращения: 11.09.2020).
17. Документация GeoTrellis. URL: <https://readthedocs.org/projects/geotrellis/downloads/pdf/latest/> (дата обращения: 11.09.2020).
18. Документация Gis tools for hadoop. URL: <http://esri.github.io/gis-tools-for-hadoop/> (дата обращения: 11.09.2020).
19. Overview GDAL/OGR Java Bindings API. URL: <http://gdal.org/java/> (дата обращения: 11.09.2020).
20. Документация KiteSDK. URL: <http://kitesdk.org/docs/current/> (дата обращения: 11.09.2020).

Conversion and Display of the Calculated Data of Spectral Remote Sensing Data on the Basis of GeoServer Extensions and Distributed Storage Technologies

I. E. Kharlampenkov, harlampenkov@ict.sbras.ru, A. U. Oshchepkov, aosivt@gmail.com, Federal Research Center for Information and Computational Technologies SB RAS, Novosibirsk, 630090, Russian Federation

Corresponding author:

Kharlampenkov Ivan E., Scientist, Federal Research Center for Information and Computational Technologies SB RAS, Novosibirsk, 630090, Russian Federation
E-mail: harlampenkov@ict.sbras.ru

*Received on October 26, 2020
Accepted on December 01, 2020*

The article presents methods for caching and displaying data from spectral satellite images using libraries of distributed computing systems that are part of the Apache Hadoop ecosystem, and GeoServer extensions. The authors gave a brief overview of existing tools that provide the ability to present remote sensing data using distributed information technologies. A distinctive feature is the way to convert remote sensing data inside Apache Parquet files for further display. This approach allows you to interact with the distributed file system via the Kite SDK libraries and switch on additional data processors based on Apache Hadoop technology as external services. A comparative analysis of existing tools, such as: GeoMesa, GeoWawe, etc is performed. The following steps are described: extracting data from Apache Parquet via the Kite SDK, converting this data to GDAL Dataset, iterating the received data, and saving it inside the file system in BIL format. In this article, the BIL format is used for the GeoServer cache. The extension was implemented and published under the Apache License on the GitHub resource. In conclusion, you will find instructions for installing and using the created extension.

Keywords: GeoServer, WMS Extensions, Apache Hadoop, HDFS, Apache Parquet, Kite SDK, Java, GDAL, ENVI BIL, distributed information systems

For citation:

Kharlampenkov I. E., Oshchepkov A. U. Conversion and Display of the Calculated Data of Spectral Remote Sensing Data on the Basis of GeoServer Extensions and Distributed Storage Technologies, *Programmnaya Ingeneria*, 2021, vol. 12, no. 2, pp. 107–112

DOI: 10.17587/prin12.107-112

References

1. Potapov V. P., Popov S. E., Oshchepkov A. U. Storage and data handling satellite spectral images based on apache parquet, *Programmnaya ingeneria*, 2018, vol. 9, no. 3, pp. 123–131 (in Russian).
2. **Documentation** QGIS Server, available at: https://docs.qgis.org/3.4/ru/docs/training_manual/qgis_server/index.html/ (date of access 11.09.2020).
3. **Documentation** MapBox, available at: <https://docs.mapbox.com/help/tutorials/> (date of access 11.09.2020).
4. **Documentation** MapTiler Server, available at: <https://support.maptiler.com/s9-server> (date of access 11.09.2020).
5. **Documentation** uDig, available at: <http://www.refractions.net/products/udig/> (date of access 11.09.2020).
6. **Documentation** SAGA GIS, available at: <http://www.saga-gis.org/en/index.html> (date of access 11.09.2020).
7. **Documentation** MapServer, available at: <http://download.osgeo.org/mapserver/docs/MapServer.pdf> (date of access 11.09.2020).
8. **Documentation** GeoServer, available at: <http://geoserver.org/about/> (date of access 11.09.2020).
9. **Documentation** Geoportal server, available at: <https://www.esri.com/en-us/arcgis/products/geoportal-server/overview> (date of access 11.09.2020).
10. **Documentation** GeoClip, available at: <https://www.geoclip.fr/documentation/?lang=en#guides> (date of access 11.09.2020).
11. **Documentation** MapLarge, available at: <https://maplarge.com/tutorial> (date of access 11.09.2020).
12. **Documentation** Map Kit Framework, available at: <https://developer.apple.com/documentation/mapkit> (date of access 11.09.2020).
13. **GeniusGis**, available at: <https://www.geniusgis.com/> (date of access 11.09.2020).
14. **Documentation** formats' ENVI, available at: <https://www.harris-geospatial.com/docs/ENVIImageFiles.html> (date of access 11.09.2020).
15. **Documentation** GeoMesa, available at: <https://www.geomesa.org/documentation/1.2.0/user/introduction.html> (date of access 11.09.2020).
16. **Documentation** GeoWawe, available at: <https://s3.amazonaws.com/geowave/latest/docs/userguide.pdf> (date of access 11.09.2020).
17. **Documentation** GeoTrellis, available at: <https://readthedocs.org/projects/geotrellis/downloads/pdf/latest/> (date of access 11.09.2020).
18. **Documentation** Gis tools for hadoop, available at: <http://esri.github.io/gis-tools-for-hadoop/> (date of access 11.09.2020).
19. **Overview** GDAL/OGR Java Bindings API, available at: <http://gdal.org/java/> (date of access 11.09.2020).
20. **Documentation** KiteSDK, available at: <http://kitesdk.org/docs/current/> (date of access 11.09.2020).

ООО "Издательство "Новые технологии". 107076, Москва, Стромьинский пер., 4
Технический редактор Е. М. Патрушева. Корректор Е. В. Комиссарова

Сдано в набор 13.01.2021 г. Подписано в печать 25.02.2021 г. Формат 60×88 1/8. Заказ П1221
Цена свободная.

Оригинал-макет ООО "Авансед солюшнз". Отпечатано в ООО "Авансед солюшнз".
119071, г. Москва, Ленинский пр-т, д. 19, стр. 1. Сайт: www.aov.ru

Рисунки к статье И. Е. Харламченкова, А. Ю. Ощепкова
«ПРЕОБРАЗОВАНИЕ И ОТОБРАЖЕНИЕ РАСЧЕТНЫХ ДАННЫХ
ОПТИКО-ЭЛЕКТРОННЫХ ДАННЫХ ДИСТАНЦИОННОГО
ЗОНДИРОВАНИЯ ЗЕМЛИ НА ОСНОВЕ РАСШИРЕНИЙ GEOSERVER
И РАСПРЕДЕЛЕННЫХ ТЕХНОЛОГИЙ ХРАНЕНИЯ»

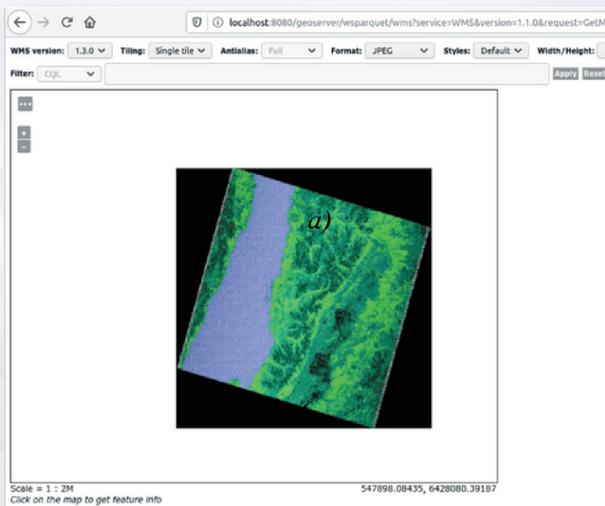


Рис. 8. Отображение полученных результатов
в виде слоя OpenLayers Landsat 7

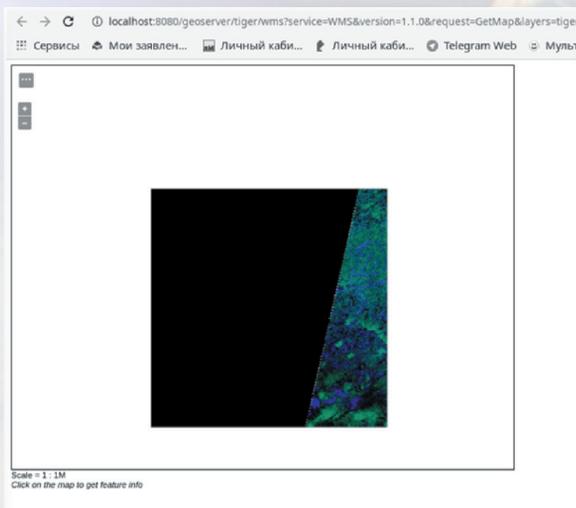


Рис. 9. Отображение полученных результатов
в виде слоя OpenLayers Sentinel-2A

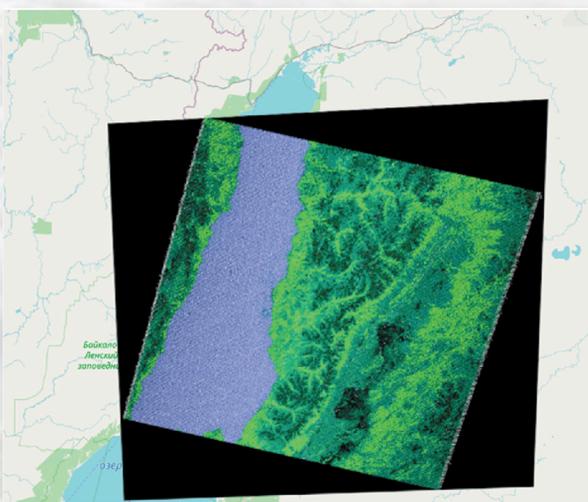


Рис. 10. Отображение на карте
восстановленного канала B2
LE71310222013244EDC00

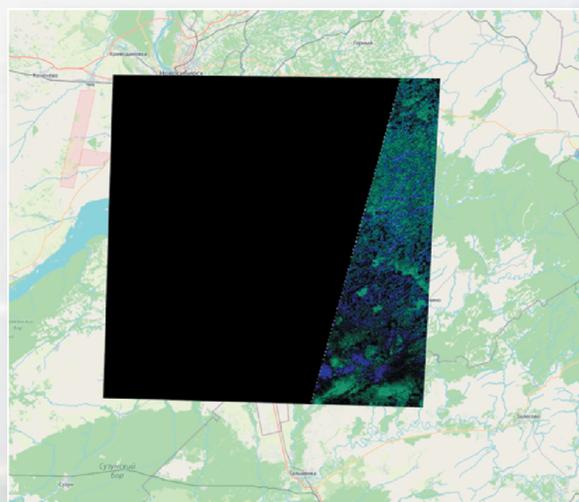
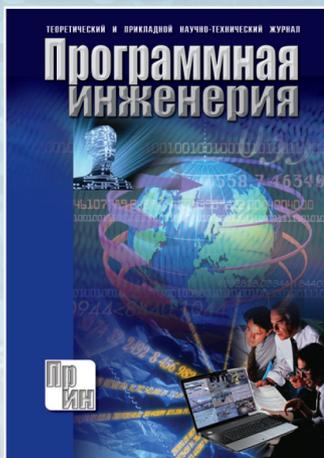


Рис. 11. Отображение на карте
восстановленного канала B2
снимка S2A_OPER_MSI_L1C_TL_
SGS_20160106T092224_A002819_T44UPF_N02_01

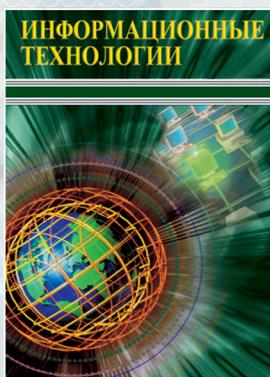
Издательство «НОВЫЕ ТЕХНОЛОГИИ» выпускает научно-технические журналы



Теоретический и прикладной научно-технический журнал **ПРОГРАММНАЯ ИНЖЕНЕРИЯ**

В журнале освещаются состояние и тенденции развития основных направлений индустрии программного обеспечения, связанных с проектированием, конструированием, архитектурой, обеспечением качества и сопровождением жизненного цикла программного обеспечения, а также рассматриваются достижения в области создания и эксплуатации прикладных программно-информационных систем во всех областях человеческой деятельности.

Подписной индекс по Объединенному каталогу
«Пресса России» – 22765



Ежемесячный теоретический
и прикладной научно-
технический журнал

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

В журнале освещаются современное состояние, тенденции и перспективы развития основных направлений в области разработки, производства и применения информационных технологий.

Подписной индекс по
Объединенному каталогу
«Пресса России» – 72656

Междисциплинарный
теоретический и прикладной
научно-технический журнал

НАНО- и МИКРОСИСТЕМНАЯ ТЕХНИКА

В журнале освещаются современное состояние, тенденции и перспективы развития нано- и микросистемной техники, рассматриваются вопросы разработки и внедрения нано микросистем в различные области науки, технологии и производства.



Подписной индекс по
Объединенному каталогу
«Пресса России» – 79493



Ежемесячный теоретический
и прикладной
научно-технический журнал

МЕХАТРОНИКА, АВТОМАТИЗАЦИЯ, УПРАВЛЕНИЕ

В журнале освещаются достижения в области мехатроники, интегрирующей механику, электронику, автоматику и информатику в целях совершенствования технологий производства и создания техники новых поколений. Рассматриваются актуальные проблемы теории и практики автоматического и автоматизированного управления техническими объектами и технологическими процессами в промышленности, энергетике и на транспорте.

Подписной индекс по
Объединенному каталогу
«Пресса России» – 79492

Научно-практический
и учебно-методический журнал

БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

В журнале освещаются достижения и перспективы в области исследований, обеспечения и совершенствования защиты человека от всех видов опасностей производственной и природной среды, их контроля, мониторинга, предотвращения, ликвидации последствий аварий и катастроф, образования в сфере безопасности жизнедеятельности.



Подписной индекс по
Объединенному каталогу
«Пресса России» – 79963

Адрес редакции журналов для авторов и подписчиков:

107076, Москва, Стромьинский пер., 4. Издательство "НОВЫЕ ТЕХНОЛОГИИ".
Тел.: (499) 269-55-10, 269-53-97. Факс: (499) 269-55-10. E-mail: antonov@novtex.ru