УДК 004.312

В.В. Воронина, А.Д. Мухаметзянов, Ю.С. Балдина

РАЗРАБОТКА WEB-СЕРВИСА ДЛЯ РАБОТЫ С ПРОГРАММИРУЕМЫМИ ЛОГИЧЕСКИМИ ИНТЕГРАЛЬНЫМИ СХЕМАМИ

Воронина Валерия Вадимовна, *кандидат технических наук, доцент кафедры «Информационные системы» Ульяновского государственного технического университета. Окончила факультет информационных систем и технологий УлГТУ. Имеет статьи в области интеллектуального анализа временных рядов. [e-mail: vvsh85@mail.ru].*

Мухаметзянов Альберт Дамирович, *магистрант, окончил бакалавриат факультета информационных систем и технологий Ульяновского государственного технического университета. Имеет статьи в области разработки интеллектуальных информационных систем. [e-mail: desperado@x-cart.com].*

Балдина Юлия Сергеевна, *магистрант, окончила бакалавриат факультета информационных систем и технологий Ульяновского государственного технического университета. Имеет статьи в области разработки интеллектуальных информационных систем. [e-mail: uliya.baldina@gmail.com].*

Аннотация

Предлагаемая информационная система является интегральным ресурсом для полноценной работы с программируемыми логическими интегральными схемами (ПЛИС) и их удаленным программированием, хранилищем знаний в этой области и средой обмена опытом между разработчиками. Результатом работы системы является модель нечеткого контроллера, представляющая собой интеллектуальную систему автоматизированного управления, с возможностью адаптации к изменяющимся условиям функционирования (благодаря нечеткому контроллеру, алгоритм работы которого реализован на псевдонечетком языке). Цель данного проекта – создание единой удобной среды для программирования ПЛИС, а также обучения программированию. В данной работе представлены описание и способ реализации архитектуры проекта, предложены схемы и этапы процесса проектирования результирующего нечеткого контроллера, а также описание разработанной системы поддержки принятия решений для выбора ПЛИС.

Ключевые слова: облачный сервис, программируемые логические интегральные схемы (ПЛИС), интеллектуальная система автоматизированного управления.

DEVELOPMENT OF WEB-SERVICE FOR OPERATION WITH PROGRAMMABLE LOGIC DEVICES

Valeriia Vadimovna Voronina, *Candidate of Engineering, Associate Professor at the Department of Information Systems of Ulyanovsk State Technical University; graduated from the Faculty of Information Systems and Technologies of Ulyanovsk State Technical University; an author of articles in the field of intellectual analysis of time series. e-mail: vvsh85@mail.ru.*

Albert Damirovich Mukhametzianov, *Candidate for the Master's Degree; graduated from the Faculty of Information Systems and Technologies of Ulyanovsk State Technical University and got the Bachelor's Degree in 2015; an author of articles in the field of intelligent system development. e-mail: desperado@x-cart.com.*

Iuliia Sergeevna Baldina, *Candidate for the Master's Degree; graduated from the Faculty of Information Systems and Technologies of Ulyanovsk State Technical University and got the Bachelor's Degree in 2015; an author of articles in the field of intelligent system development. e-mail: uliya.baldina@gmail.com.*

Abstract

The current information system is an integral resource for full-fledged operation with programmable logic devices (PLD) and with the functions of remote programming, knowledge storage and experience exchange of the particular field between developers. The result of the system operation is the model of fuzzy controller that represents an intelligent automated control system (ISAU) in a small way with the ability of adaption to changing conditions of operation (due to the fuzzy controller implemented through a fuzzy algorithm on pseudo-fuzzy language). The goal of the project is to provide the convenient

integrated programming environment with the good condition for education. The article considers the description and the way of implementation of the project architecture; diagrams and stages of resulting fuzzy controller design process and the description of the developed decision support system for choosing PLD are shown.

Keywords: cloud service, programmable logic device (PLD), intelligent automated control system.

Введение

В настоящее время наиболее часто используемые в информационных технологиях микроконтроллеры (однокристальные микро-ЭВМ) не способны выполнять многочисленные задачи и алгоритмы цифровой обработки информации в связи с непрерывным развитием запросов IT-индустрии. Рынок микроэлектроники постепенно интегрирует в производство программируемые логические интегральные схемы по трем основным причинам. Во-первых, ПЛИС по производительности в разы превосходят микро-ЭВМ, во-вторых, они многофункциональны и мультизадачны, в-третьих, ПЛИС могут, в зависимости от полученных данных, менять свою архитектуру (перепрограммировать самих себя), т. е. «обучаться» [1 – 2, 4].

Постоянный рост и развитие рынка программируемой логики обуславливают появление следующих потребностей. Во-первых, существуют потребности в простых и доступных всем инструментальных средствах для работы с микросхемами программируемой логики. Во-вторых, появляются потребности в обучении и приобщении школьников и студентов технических направлений к научным новшествам в области IT-технологий, робототехники, конструирования и программирования цифровых устройств. В-третьих, необходимо разрабатывать алгоритмы, позволяющие ПЛИС эффективно работать в условиях неполноты информации. Описываемый в данной статье проект как раз и нацелен на решение этих задач.

Архитектура проекта

Основная сложность процесса программирования логических интегральных схем заключается в том, что существует много алгоритмов программирования и форматов данных, зависящих от производителя и типа ПЛИС, и время выполнения программирования ПЛИС может достигать 48 часов в зависимости от аппаратного обеспечения, компьютера, на котором проводятся вычисления [3].

Поэтому была спроектирована многоуровневая архитектура клиент-сервер, обеспечивающая главное преимущество разрабатываемой системы, а именно удаленное программирование ПЛИС, которое дает независимость от аппаратного обеспечения машины, на которой будут производиться вычисления (т. е. все вычислительные задачи выполняются на отдельном сервере), а также снижение времени программирования самой ПЛИС. Данный тип архитектуры позволяет вынести функцию обработки данных на отдельный сервер, что позволяет разделить функции хранения, обработки и представления данных для более эффективного использования возможностей серверов и клиентов.

На рисунке 1 представлена архитектура Web-сервиса, состоящая из трех частей, самая большая из них, в свою очередь, представлена четырьмя блоками. Рассмотрим эти части подробнее.

Web-сервис для хостинга IT-проектов представляет собой Web-ресурс для получения нужной информации или прохождения обучения по работе с программируемыми устройствами. Также на нем развернуты OnlineIDE для создания, редактирования и ведения проектов, написанных как на языках описания аппаратуры (HDL-языки), так и на высокоуровневых языках программирования. Этот блок реализован с использованием PHP, JavaScriptMVC, CoffeScript, AJAX, MySQL + Doctrine2 (ORM, работа с базами данных посредством объектов PHP, DQL).

Клиент-серверное приложение представляет собой небольшое установочное приложение, необходимое для удаленного подключения программируемого устройства к вычислительному серверу. Используемые средства при реализации: .NET Framework, C#, WPF.

Сервер-контроллер – это вычислительный сервер, состоящий из четырех программных блоков:

- Контроллер с подсистемой поддержки принятия решений для выбора архитектуры ПЛИС. Основная задача блока – контроль над всеми выполняемыми действиями на сервере.
- База данных, осуществляющая необходимые действия (редактирование / создание) для работы вычислительного сервера, а также Web-сервиса для хостинга IT-проектов.
- Транслятор языков или «программный блок правил», позволяющий транслировать языки описания аппаратуры в высокоуровневые языки программирования.

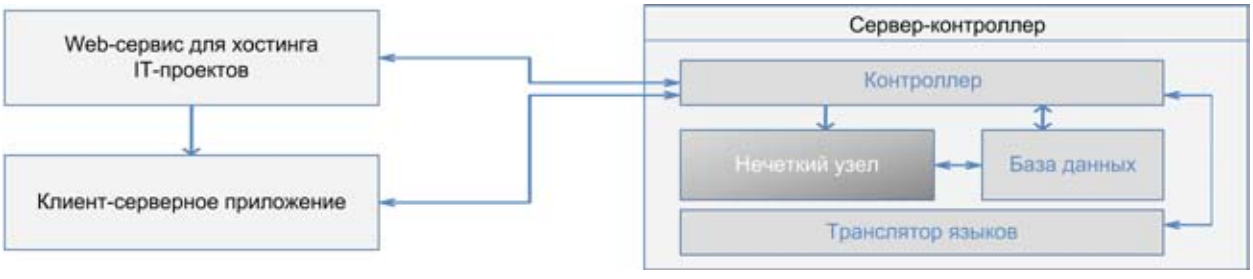

Рис. 1. Архитектура работы Web-сервиса

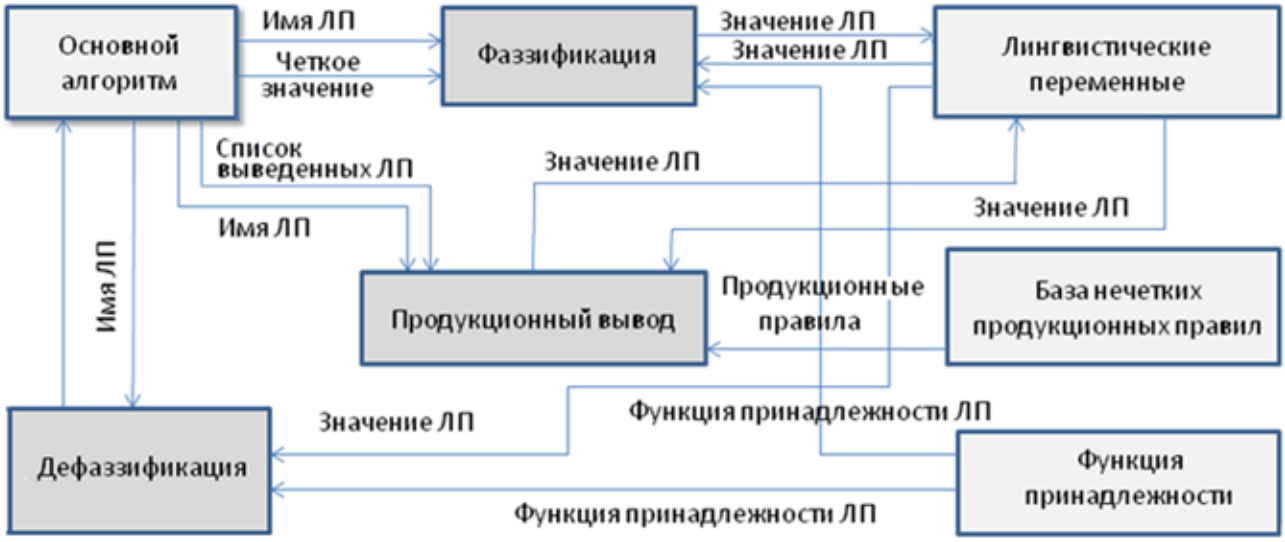

Рис. 2. Диаграмма потоков данных в реализации нечеткого контроллера

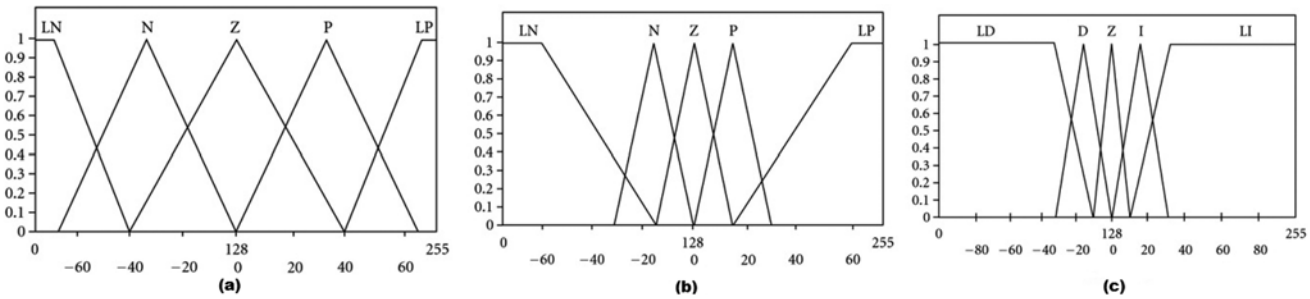

Рис. 3. Функции принадлежности

• Нечеткий узел. Один из самых важных программных компонентов Web-сервиса. Представляет собой отдельную архитектуру и программно-аппаратную реализацию результирующего нечеткого контроллера. Используемые средства при реализации: объектно-ориентированный язык программирования Java, модули на C++.

Рассмотрим более подробно нечеткий контроллер и подсистему поддержки принятия решений.

Нечеткий контроллер

При проектировании структурной схемы основное внимание было уделено созданию универсального нечеткого контроллера. Основными факторами универсальности и гибкости контроллера являются возможность изменения количества входных и выходных переменных, а также возможность масштабирования входного и выходного сигнала. Первый фактор обеспечивается наличием отдельного блока для вычисления функций каждой входной переменной и возможностью наращивания при необходимости входов и выходов блока активизации правил и блока фаззификации. Последний фактор обеспечивается внесением в схему блока масштабирования.

Процесс получения решения состоит из четырех основных стадий: фаззификации, логического вывода решения из множества нечетких правил вывода, композиции решения и дефаззификации [5–6].

Данное решение в контексте теории автоматического управления называется нечетким управлением. Оно применяется при синтезе нечетких регуляторов, гибридных регуляторов, нечетких поисковых систем автоматической оптимизации, нечетких устройств оценивания и фильтрации.

Чтобы задать требуемые нечеткие операции, необходимо реализовать следующие блоки. Как уже говорилось, нечеткий вывод включает в себя четыре основных этапа, однако второй и третий можно объединить в один.

Таким образом, программная реализация нечеткого контроллера будет содержать три модуля, реализующих данные этапы и задающих требуемые операции. Взаимодействие этих модулей с основной программой продемонстрировано на диаграмме (рис. 2).

На этапе фаззификации для каждого входа нечеткого контроллера (входа ошибки и входа погрешности ошибки) было определено пять функций принадлежности, задающих конкретную лингвистическую переменную (ЛП): LN (largenegative), N (negative), Z (zero), P (positive) и LP (largepositive). Диапазон «горизонта» для функций принадлежностей равен [–80, 80].

ce \ e	LN	N	Z	P	LP
LN	LI	LI	I	D	LD
N	LI	I	Z	Z	LD
Z	LI	I	Z	D	LD
P	LI	Z	Z	D	LD
LP	LI	I	D	LD	LD

Рис. 4. Матрица правил

На этапе дефаззификации для выхода нечеткого контроллера было определено также пять функций принадлежности, которые также задают конкретную лингвистическую переменную: LD (largedecrement), D (decrement), Z (zero), I (increment), и LI (largeincrement), с диапазоном «горизонта», равным [–100, 100].

Первая шкала «горизонта» [0, 255] обозначает реальные значения лингвистических переменных в нечетком контроллере. Вторая шкала «горизонта» [–80, 80] и [–100, 100] – значения операций при работе нечеткого контроллера. Шкала преобразований достигается за счет процессора схем программируемой логики [8–10].

На рисунке 3 представлены графики функций принадлежности для входа ошибки (a) и входа погрешности ошибки (b), а также выхода контроллера (c).

Результирующая матрица правил нечеткого контроллера сформулирована из 25 правил, приведенных на рисунке 4.

Синтезированный нечеткий контроллер для решения различных задач является интеллектуальной системой, но в нем отсутствует возможность адаптации к изменяющимся условиям функционирования. Благодаря тому, что нечеткий контроль осуществлен посредством нечеткого алгоритма на псевдонечетком языке нечетких операций, данный недостаток является исправимым. Повышение интеллектуальности может производиться путем усложнения алгоритма. В этом случае необходимо расширить базу правил и включить в нее блок правил корректирующего или организационного уровня, производящих адаптацию существующих правил исправительного уровня. Помимо правил, корректировке могут подвергаться функции принадлежности термов с помощью специальных правил в зависимости от нечеткой функции ошибки системы.

Система поддержки принятия решений для выбора ПЛИС

В настоящее время из-за огромного количества ПЛИС, отличающихся между собой по целому ряду параметров, возникает потребность в разработке средства их объективного выбора на основании исходных данных по проектированию системы цифровой обработки информации.

Информационная система поддержки принятия решений является модулем Web-сервиса для программирования ПЛИС и предназначена для упрощения принятия стратегических решений поставленной задачи и выбора наиболее верного пути дальнейшего развития. Это дает возможность быстро и точно получать необходимые решения и данные.

Целью создания информационной системы является упрощение процесса принятия решений и получения знаний при проектировании и программировании устройств цифровой обработки сигналов на базе ПЛИС. В результате использования системы ожидается достижение таких технологических показателей, как сокращение времени и трудоемкости проектирования устройств цифровой обработки сигналов на базе ПЛИС, а также ряда производственно-экономических показателей, таких как сокращение затрат на заработную плату сотрудникам, минимизация издержек при неправильном выборе микросхемы на этапе проектирования.

За основу разработки была взята математическая модель, описанная в работе [11].

В ходе проведенного исследования были решены следующие задачи:

1. Доработана известная математическая модель принятия решений, позволяющая объединить критериальные функции в целевой функционал при выборе ПЛИС из множества альтернатив.

2. Разработан алгоритм принятия решений по объективному выбору ПЛИС при проектировании устройств цифровой обработки информации.

Предлагаемый метод выбора ПЛИС на основе целевого функционала включает следующую последовательность действий.

Шаг 1. Определение исходных данных: параметров ПЛИС и сигналов. Множество исходных параметров D предлагается разделить на две группы: параметры ПЛИС (аппаратные) $-(w_1, w_2, ..., w_n)$, $(y_1, y_2, ..., y_n)$ и параметры сигналов $-(s_1, s_2, ..., s_n)$, затем определить максимальные значения множества параметров $(y_1, y_2, ..., y_n)$. Такие параметры, как количество эквивалентных логических элементов, количество встроенных умножителей, количество эквивалентных макроячеек, количество встроенных микропроцессорных ядер и т.д., относятся к аппаратной группе. К параметрам сигналов принято относить те, которые влияют на выбор ПЛИС наиболее значительно: тип сигнала, амплитуда и частота.

Шаг 2. Нормирование параметров w_i по максимальным значениям, описываемое формулой (1).

$$K_i = \frac{w_i}{y_t}. \quad (1)$$

В результате такого нормирования будут получены безразмерные коэффициенты параметров. Причем здесь рассматривается только группа аппаратных параметров: K_{PCMAX} – коэффициент цены микросхемы; K_{CLE} – коэффициент количества эквивалентных логических элементов; K_{CMC} – коэффициент количества эквивалентных макроячеек; K_{CM18} – коэффициент количества встроенных умножителей 18x18 бит; K_{CM19} – коэффициент количества встроенных умножителей 18x19 бит; K_{CDSP27} – ко-

эффициент количества встроенных сигнальных процессоров 27x27 бит; K_{CDSPVP} – коэффициент количества встроенных сигнальных процессоров с регулируемой точностью; K_{TPD} – коэффициент времени задержки сигнала; K_{CGCN} – коэффициент количества глобальных цепей тактирования; K_{FMAX} – коэффициент тактовой частоты; K_{CPC} – коэффициент количества встроенных микропроцессорных ядер; K_{CUIO} – коэффициент количества программируемых пользователем выводов; K_{CHPSP} – коэффициент количества выводов встроенной микропроцессорной системы.

Шаг 3. Формирование частных критериальных функций модели $f_i^{(M)}(K_i)$. Для микросхем разного типа обычно вычисляют разные коэффициенты, чей приоритет указать сложно, поэтому необходимо выделить существенные коэффициенты параметров K_i^S. Применительно к группе аппаратных параметров существенный коэффициент – это коэффициент стоимости микросхемы, для которого частными критериальными функциями $f_i^{(M)}(K_i)$ могут быть:

- Объединение коэффициента количества встроенных умножителей 18x18 бит и коэффициента количества встроенных умножителей 18x19 бит в виде выражения (2).

$$f_1^{(M)} = K_{CM18} + K_{CM19}. \quad (2)$$

- Объединение коэффициента количества встроенных сигнальных процессоров 27x27 бит и коэффициента количества встроенных сигнальных процессоров с регулируемой точностью в виде выражения (3).

$$f_2^{(M)} = K_{CDSP27} + K_{CDSPVP}. \quad (3)$$

Некоторые из коэффициентов можно объединить в выражения, которые позволят оценить вычислительную мощность микросхемы количественно. Для этого в выражениях такого типа удобно использовать последовательность Фибоначчи: наиболее значимые коэффициенты умножаются на числа с наибольшим индексом в последовательности.

Числовая последовательность Фибоначчи строится по формуле (4).

$$U_1 = i;\ U_2 = i;\ U_n = U_{n-1} + U_{n-2}, \quad (4)$$

где U_n – элементы последовательности.

Одно из свойств этой последовательности можно записать выражением (5).

$$\sum_{i=1}^{n} f_{FIB}(i) = f_{FIB}(n+2) - 1. \quad (5)$$

Тогда для количественной оценки вычислительной мощности микросхемы общий вид выражения, который основан на свойстве последовательности Фибоначчи, будет следующим:

$$f_i^{(M)} = \frac{\sum_{i=1}^{n} f_{FIB}(i) * K_i}{f_{FIB}(n+2) - 1}. \quad (6)$$

Шаг 4. Выделение наиболее часто применяемых архитектур ПЛИС. Для этого необходимо выделить коэффициенты, определяющие быстродействие каждой из архитектур, среди них:

- Коэффициенты, определяющие быстродействие ПЛИС архитектуры CPLD [14]:

$$f_3^{(M)} = \frac{3 * K_{CMC} + 2 * K_{FMAX} + K_{CUIO} + K_{CGCN}}{7}. \quad (7)$$

- Коэффициенты, определяющие быстродействие ПЛИС архитектуры SoC [15]:

$$f_4^{(M)} = \frac{34 * K_{CLE} + 21 * K_{CPC} + 13 * K_{FMAX} + 8 * (1 - K_{TPD}) + 5 * K_{CHPSP} + 2 * K_{CGCN} + f_1 + f_2}{88}. \quad (8)$$

- Коэффициенты, определяющие быстродействие ПЛИС архитектуры FPGA [16]:

$$f_5^{(M)} = \frac{8 * K_{CLE} + 5 * K_{FMAX} + 3 * K_{CUIO} + 2 * K_{CGCN}}{20}. \quad (9)$$

- Коэффициенты, определяющие быстродействие ПЛИС комбинированной архитектуры PLD [17]:

$$f_6^{(M)} = \frac{13 * K_{CMC} + 8 * K_{CLE} + 5 * K_{FMAX} + 3 * K_{CUIO} + 2 * K_{CGCN}}{33}. \quad (10)$$

Рассмотренные коэффициенты (2, 3, 8, 9, 10) для упрощения можно объединить одним коэффициентом вычислительной мощности. Для этого необходимо применить обобщенную критериальную функцию, описанную выражением (11).

$$F^{(M)} \begin{cases} \text{если } K_{CPC} > 0, \text{ то } \exp\left(f_4^{(M)}\right); \\ \text{если } K_{CPC} = 0 \text{ и } K_{CMC} = 0, \\ \qquad \text{то } \exp\left(f_5^{(M)} + 3\right); \\ \text{если } K_{CPC} = 0 \text{ и } K_{CLE} = 0, \\ \qquad \text{то } \exp\left(f_3^{(M)} + 1\right); \\ \text{если } K_{CPC} = 0 \text{ и } K_{CMC} > 0 \text{ и } K_{CLE} > 0, \\ \qquad \text{то } \exp\left(f_6^{(M)} + 2\right). \end{cases} \quad (11)$$

Относительно группы аппаратных параметров обобщенная критериальная функция не позволяет выполнить анализ всего комплекса выделенных параметров ПЛИС: она не учитывает цену ПЛИС, а также соотношение цены и вычислительной мощности микросхемы.

Шаг 5. Формирование окончательного целевого функционала. Для соотношения цены и вычислительной мощности ПЛИС необходимо сформировать целевой функционал модели выбора $J^{(M)}(X^{(M)})$ варианта ПЛИС, который определяет поведение модели в зависимости от обобщенной критериальной функции $F^{(M)}(f_i^{(M)}, K_i)$ и существенных коэффициентов K_i^S. Он может быть описан выражением:

$$J^{(M)} = \left(F^{(M)}, K_i^S\right). \quad (12)$$

Опишем выражением (13) множество альтернативных вариантов:

$$X^{(M)} = \{X_1, X_2, ..., X_n\}. \quad (13)$$

Опишем выражением (14) зональное разделение области значений целевого функционала $\Delta J_{X_i}^{(M)}$, соответствующее альтернативным вариантам.

$$\Delta J_X^{(M)} \in \left\{\Delta J_{x_1}^{(M)}, \Delta J_{x_2}^{(M)}, ..., \Delta J_{x_n}^{(M)}\right\}. \quad (14)$$

Тогда целевой функционал примет следующий вид:

$$J^{(M)} = a_i * F^{(M)} + a_2 * \left(1 - K_{PCMAX}\right), \quad (15)$$

где a_i - весовые коэффициенты, которые должны удовлетворять условию $\sum_{i=1}^{2} a_i = 1$.

Коэффициент a_1 определяет вес требования вычислительной мощности микросхемы, а коэффициент a_2 определяет вес требования выгодной стоимости микросхемы.

Шаг 6. Принятие решения. Решение для математической модели в условиях определенности формулируется так: попадание реального целевого функционала $J(R)$ в интервал значений $\Delta J_{X_i}^{(M)}$ будет определять альтернативный вариант $X_i^{(R)}$, при этом чем выше показатель $J^{(R)}$, тем предпочтительней модель. Тогда процесс принятия решения можно записать в виде:

$$X_i^{(R)} = X_i^{(M)} \in \{X_1, X_2, ..., X_n\}$$
$$при\ J^{(R)}\left(F^{(R)}, K_i^{S_R}\right) \in \Delta J_{X_i}^{(M)},\ J^{(R)} \to \infty. \quad (16)$$

Шаг 7. Оптимизация решений. На этом шаге необходимо сформировать ограничения и учесть сигнальные параметры. При разработке метода выбора ПЛИС с использованием базы данных алгоритм принятия решения приобретает иерархический вид. Таким образом, первостепенная задача – выбрать из представленных моделей ПЛИС те, которые будут удовлетворять формальному описанию при ограничении выбора. После этого полученные решения оптимизируются.

При выборе ПЛИС можно говорить о следующих параметрах обрабатываемого сигнала, являющихся ограничивающим фактором при выборе микросхемы. Во-первых, FS (МГц) – частота обрабатываемого сигнала, во-вторых, NPS – количество обрабатываемых сигналов.

Учитывая, что при обработке сигнала с частотой FS должно выполняться условие $FS << FMAX$, можно сформировать условное описание зависимостей параметров микросхем и параметров обрабатываемого сигнала.

Разработаем правила для формирования ограничения. Согласно условию однозначной дискретизации ПЛИС сигнала, вытекающему из теоремы Котельникова, получаем ограничение:

$$F_{S,\,MAX} \le \frac{F_{MAX}}{2}. \quad (17)$$

Согласно условию совместимости количества обрабатываемых сигналов с количеством линий ввода-вывода получаем ограничение:

$$N_{PS} \le CUIO. \quad (18)$$

Алгоритм метода выбора ПЛИС, реализованный в рамках модуля описываемого Web-сервиса, состоит из следующих шагов:

1. Постановка задачи.
2. Определение исходных данных.
3. Заполнение ключевых полей поиска.
4. Формирование первичного ограничения выбора.
5. Формирование вторичного ограничения выбора.
6. Формирование основного списка микросхем.
7. Нормирование параметров ПЛИС.
8. Вычисление значения целевого функционала для каждой микросхемы.
9. Построение списка вариантов решений.
10. Разделение на интервалы области значения целевого функционала.
11. Если решение найдено – вывод решения, иначе переход к шагу 3.

Для анализа точности разработанной математической модели было проведено статистическое исследование. Его задачей было получение выборки векторов приоритетов критериев выбора модели ПЛИС. Нами были составлены опросные письма, ответ на которые позволил заполнить матрицу парных сравнений для критериев. Опрашивались люди, которые занимаются разработкой цифровых устройств на базе ПЛИС, пользователи специализированных форумов. В качестве моделируемой задачи решалась задача лучшей модели ПЛИС для учебных целей.

В таблице 1 приведены результаты итоговых расчетов для четырех микросхем: Cyclone IV – EP4CGX30, Cyclone V – 5CEA2, Spartan 3 – XC3S1500 и Spartan 6 – LX – XC6SLX25.

Таблица 1

Результаты расчетов

Модель ПЛИС	$J^{(M)}$
Cyclone IV – EP4CGX30	0,2457
Cyclone V – 5CEA2	0,5317
Spartan 3 – XC3S1500	0,36725
Spartan 6 – LX – XC6SLX25	0,4446

На основании разработанной математической модели наилучшим вариантом признается модель с наибольшим значением целевого функционала $J^{(M)}$. Таким образом, модель Cyclone V – 5CEA2 является наиболее подходящей для заданной области.

Далее в порядке релевантности: Spartan 6 – LX – XC6SLX25, Spartan 3 – XC3S1500, Cyclone IV – EP4CGX30.

Следует учитывать тот факт, что при принятии решения не были задействованы частные критериальные функции параметров сигнала.

Результаты выбора ПЛИС, полученные от подсистемы поддержки принятия решений, сравнивались с результатами, полученными от экспертов. Коэффициент сходства результатов 93%. Таким образом, разработанную систему можно считать эффективной.

Заключение

В данной статье описана спроектированная среда для программирования логических устройств (ПЛИС), основными плюсами которой являются:

- независимость разработчика от технических характеристик компьютера;
- «облачность» создаваемой системы: нет необходимости устанавливать программное обеспечение;
- возможность удаленного программирования микросхем;
- комфортная среда для работы, учебы и обмена опытом между разработчиками;
- наличие сервиса и базы для обучения программированию ПЛИС;
- возможность моделирования нечеткого контроллера и поддержка принятия решения по выбору ПЛИС.

СПИСОК ЛИТЕРАТУРЫ

1. Flynn M.J. and W. Luk. Computer System Design: System-on-Chip, John Wiley & Sons, Hoboken, NJ, USA, 2011.

2. Idkhajine L., E. Monmasson, M. W. Naouar, A. Prata, and K. Bouallaga. Fully integrated FPGA-based controller for synchronous motor drive. // IEEE Transactions on Industrial Electronics. 2009. Vol. 56, no. 10, pp. 4006–4017.

3. Sun F., H. Wang, F. Fu, and X. Li. Survey of FPGA low power design. // Proc. of the International Conference on Intelligent Control and Information Processing (ICICIP '10), August 2010, pp. 547–550.

4. Saha D. and S. Sur-Kolay. SoC: a real platform for IP reuse, IP infringement, and IP protection // VLSI Design 2011, vol. 2011, Article ID 731957, 10 p.

5. Tian L., H. Pan, and D. Li. Efficient Memory Processors Design of Multiple Applications for Multiprocessors Architecture. // Software Engineering and Knowledge Engineering: Theory and Practice in Advances in Intelligent and Soft Computing, Y. Wu, Ed. Springer, Berlin, Germany. 2012. pp. 693–697.

6. Precup R.E. and H. Hellendoorn. A survey on industrial applications of fuzzy control. // Computers in Industry. 2011. vol. 62, no. 3, pp. 213–226.

7. Kackprzyk J. Multistage fuzzy control: a model-based approach to fuzzy control and decision making. // Journal of Multi-Criteria Decision Analysis. 1998. vol. 7, no. 4, pp. 239–240, .

8. Ross T. J. Fuzzy Logic With Engineering Applications. John Wiley & Sons, Singapore, 3rd edition, 2010.

9. Jim L. Embedded Control Systems in C/C++, CMP Books, Berkley, Calif, USA, 2004.

10. Thareja V., M. Bolic, and V. Groza. Design of a fuzzy logic coprocessor using handel-C // Proceedings of the 2nd IEEE International Workshop on Soft Computing Applications (SOFA '07). Oradea, Romania. August, 2007. pp. 83–88.

11. Турыгин И.Г., Литвинская О.С. Выбор на основе целевого функционала программируемых логических интегральных схем при проектировании специализированных устройств // Успехи современного естествознания. – 2012. – № 6. – С. 100–112.

12. Стешенко В.Б. EDA. Практика автоматизированного проектирования радиоэлектронных устройств. – М. : Нолидж, 2002. – 768 с.

13. Котельников В.А. О пропускной способности «эфира» и проволоки в электросвязи // Успехи физических наук. – 2006. – Т. 176, № 7. – С. 762–770.

14. CPLD [Электронный ресурс]: материалы свободной энциклопедии Википедиа. – Режим доступа: https://ru.wikipedia.org/wiki/CPLD (дата обращения 01.08.2015).

15. Система на кристалле [Электронный ресурс]: материалы свободной энциклопедии Википедиа. – Режим доступа: https://ru.wikipedia.org/wiki/Система_на_кристалле (дата обращения 01.08.2015).

16. Программируемая пользователем вентильная матрица [Электронный ресурс]: материалы свободной энциклопедии Википедиа. – Режим доступа: https://ru.wikipedia.org/wiki/Программируемая_пользователем_вентильная_матрица (дата обращения 01.08.2015).

REFERENCES

1. Flynn M. J. and Luk W. *Computer System Design: System-on-Chip*. John Wiley & Sons, Hoboken, NJ, USA, 2011.

2. Idkhajine E., M. Monmasson, W. Naouar, A. Prata, and K. Bouallaga. Fully integrated FPGA L.-based controller for synchronous motor drive. *IEEE Transactions on Industrial Electronics,* 2009, vol. 56, no. 10, pp. 4006–4017.

3. Sun F., H. Wang, F. Fu, and X. Li. Survey of FPGA low power design. *Proceedings of the International Conference on Intelligent Control and Information Processing (ICICIP '10),* August 2010, pp. 547–550.

4. Saha D. and S. Sur-Kolay. SoC: a real platform for IP reuse, IP infringement, and IP protection. *VLSI Design,* 2011, vol. 2011, Article ID 731957, 10 p.

5. Tian L., H. Pan, and D. Li. Efficient Memory Processors Design of Multiple Applications for Multiprocessors Architecture. *Software Engineering and Knowledge Engineering: Theory and Practice in Advances in Intelligent and Soft Computing,* Y. Wu, Ed., Germany, Berlin, Springer Verlag, 2012, pp. 693–697.

6. Precup R. E. and H. Hellendoorn. A survey on industrial applications of fuzzy control. *Computers in Industry,* 2011, vol. 62, no. 3, pp. 213–226.

7. Kackprzyk J. Multistage fuzzy control: a model-based approach to fuzzy control and decision making. *Journal of Multi-Criteria Decision Analysis,* 1998, vol. 7, no. 4, pp. 239–240.

8. Ross T. J. *Fuzzy Logic With Engineering Applications. 3rd edition*. John Wiley & Sons, Singapore, 2010.

9. Jim L. *Embedded Control Systems in C/C++*. CMP Books, Berkley, Calif, USA, 2004.

10. Thareja V., M. Bolic, and V. Groza. Design of a fuzzy logic coprocessor using handel-C. *Proceedings of the 2nd IEEE International Workshop on Soft Computing Applications (SOFA '07),* Oradea, Romania, 2007, August, pp. 83–88.

11. Turygin I.G., Litvinskaia O.S. Vybor na osnove tselevogo funktsionala programmiruemykh logicheskikh integralnykh skhem pri proektirovanii spetsializirovannykh ustroistv [Choice on the Base of Criterion Functional of Programmable Logic Device in Special-Purpose Hardware Designing]. *Uspekhi Sovremennogo Estestvoznaniia* [Advances in Current Natural Sciences], 2012, no. 6, pp. 100–112.

12. Steshenko V.B. EDA. *Praktika avtomatizirovannogo proektirovaniia radioelektronnykh ustroistv* [Computer-Aided Design Practice of Radioelectronic Devices]. Moscow, Nolidzh Publ., 2002. p. 768.

13. Kotelnikov V.A. O propusknoi sposobnosti «efira» i provoloki v elektrosviazi [On the Transmission Capacity of 'ether' and Wire in Electric Communications]. *Uspekhi Fizicheskikh Nauk* [Physics – Uspekhi. Advances in Physical Sciences], 2006, vol. 176, no.7, pp. 762–770.

14. Materialy svobodnoi entsiklopedii Vikipedia [Wikipedia. Free Encyclopedia]. Available at: https://ru.wikipedia.org/wiki/Sistema_na_kristalle. (accessed 01.08.2015).

15. Materialy svobodnoi entsiklopedii Vikipedia [Wikipedia. Free Encyclopedia]. Available at: https://ru.wikipedia.org/wiki/Programmiruemaya_polzovatelem_ventilnaia_matritsa. (accessed 01.08.2015).

16. Materialy svobodnoi entsiklopedii Vikipedia [Wikipedia. Free Encyclopedia]. Available at: URL: https://ru.wikipedia.org/wiki/PLIS. (accessed 01.08.2015).

УДК 621.391.037

Г.М. Тамразян, А.А. Гладких, Д.В. Ганин

АППАРАТНАЯ РЕАЛИЗАЦИЯ ОПТИМАЛЬНОГО ДЕКОДЕРА НИЗКОПЛОТНОСТНЫХ КОДОВ

Тамразян Георгий Михайлович, *аспирант кафедры «Телекоммуникации» Ульяновского государственного технического университета, окончил УлГТУ. Имеет публикации и патенты в области мягкого декодирования избыточных кодов. [e-mail: tamrazz@bk.ru].*

Гладких Анатолий Афанасьевич, *кандидат технических наук, окончил Военную академию связи им. С.М. Буденного, адъюнктуру ВАС. Профессор кафедры «Телекоммуникации» Ульяновского государственного технического университета. Имеет монографию, учебные пособия, статьи и патенты РФ в области помехоустойчивого кодирования и защиты информации. [e-mail: a.gladkikh@ulstu.ru].*

Ганин Дмитрий Владимирович, *кандидат экономических наук, доцент, окончил Нижегородскую государственную сельскохозяйственную академию. Заведующий кафедрой «Инфокоммуникационные технологии и системы связи» Нижегородского государственного инженерно-экономического университета. Имеет статьи в области инфокоммуникаций. [e-mail: ngiei135@mail.ru].*

Аннотация

В современных инфокоммуникационных системах все большее применение стали находить коды с низкой плотностью проверок (Low Density Parity Check – LDPC) за счет своей корректирующей способности. В настоящее время LDPC-коды максимально приблизились к порогу Шеннона. Кроме того, использование таких кодов, в отличие от турбо-кодов, не имеет ограничений, связанных с патентами. Эти факторы послужили причиной растущего интереса к низкоплотностным кодам. Несмотря на относительную простоту реализации такого кодека, мягкое декодирование LDPC-кода имеет значительную вычислительную сложность. В данной работе подробно рассмотрены основные проблемные области, связанные с аппаратной реализацией LDPC-кодека, а также пути их преодоления. В работе продемонстрированы результаты моделирования различных способов реализации декодера и их сравнение. Кроме того, представлен метод списочного декодирования LDPC-кода, который значительно снижает вычислительную нагрузку на декодер и ускоряет его работу. Использование тех или иных техник и алгоритмов, показанных в данной работе, позволит разработать оптимальный декодер низкоплотностного кода, спроектированного под определенную задачу.

Ключевые слова: LDPC-код, мягкий декодер, ПЛИС, кластер, граф Таннера, списочное декодирование.

HARDWARE IMPLEMENTATION OF THE OPTIMAL LDPC DECODER

Georgii Mikhailovich Tamrazian, *Post-Graduate Student at the Department of Telecommunications of Ulyanovsk State Technical University; graduated from Ulyanovsk State Technical University; an author of articles and patents in the field of redundant code soft decoding. e-mail: tamrazz@bk.ru.*

Anatolii Afanasievich Gladkikh, *Candidate of Engineering; graduated from S.M. Budyonny Military Communications Academy; finished his post-graduate studies at the same academy; Professor at the Department of Telecommunications at Ulyanovsk State Technical University; an author of a monograph, textbooks, articles, and patents in the field of noiseless coding and information security. e-mail: a.gladkikh@ulstu.ru.*

Dmitrii Vladimirovich Ganin, *Candidate of Economics, Associate Professor; graduated from Nizhny Novgorod State Agricultural Academy; Head of the Department of Infocommunication Technologies and Telecommunications at Nizhny Novgorod State University of Engineering and Economics; an author of articles in the field of infocommunications. e-mail: ngiei135@mail.ru.*

Abstract

Low Density Parity Check (LDPC) codes become more useful in modern infocommunication systems because of their error-correcting capability. At the present time, LDPC codes have reached Shannon's limit. Moreover, the application of such codes unlike the turbo codes don't have any license limitations. These factors have become the cause of growing interest to LDPC codes. Despite an easy way of the codec implementation, soft decoding of LDPC codes is a complex computational process.