

УДК 004.242

ОПАНАСЕНКО В.Н.,
ЩЕРБАКОВА И.А.,
ЖЕЛЄЗНЯК А.Л.,
ГОНЧАРОВА Л.Л. (ДЕТУТ)

Синтез реконфигурируемых вычислительных структур контроллеров на ПЛИС

Современные интегральные технологии изготовления заказных интегральных схем на ПЛИС предоставляют возможность реализовывать параллельные вычислительные структуры устройств с динамической реконфигурацией и тем самым решать задачи эффективной настройки на заданный алгоритм. Благодаря этому открывается возможность создания реконфигурируемых вычислительных устройств для современных систем, связанных с выполнением больших объемов вычислений, например, управления технологическими процессами в энергетике, контрольно-измерительных комплексах, при создании средств управления. Современные средства микропроцессорной техники – микроконтроллеры – реализуются, как правило, либо в виде наборов модулей, встраиваемых в компьютер, либо в виде автономных систем управления. Алгоритм управления при этом представляется программой, которая хранится в памяти. Для записи алгоритмов управления может использоваться как постоянная, так и оперативная память, которая позволяет перезаписывать программу большое число раз. Быстрое развитие современных технологий в микроэлектронике и создание программируемых логических интегральных схем высокой степени интеграции привели к появлению нового класса микроконтроллеров с возможностью реконфигурации вычислительной архитектуры, которая изменяется программным способом. Отличие реконфигурируемых микроконтроллеров состоит в том, что их структура не является фиксированной и меняется в зависимости от реализуемого алгоритма управления [1]. Обобщенный подход реализации такого класса реконфигурируемых микроконтроллеров может быть представлен в следующем виде. Архитектура реконфигурируемого микропроцессорного контроллера может быть описана как

$$S = \langle P, A_i, F \rangle,$$

где $P = \{P_i\}$ – множество объектов управления ($i = 1 \div n$); c – множество алгоритмов управления, реализующих функцию отображения множества входных сигналов $\{X_{ij}\}$, ($j = 1 \div m$) на множество выходных сигналов $\{Y_{ij}\}$ для i -го объекта; s – множество файлов конфигурации ($\gamma = 1 \div k, k = n \times m$), определяющих структуры реализации алгоритмов A_{ij} объектов управления P_i . В тех случаях, когда алгоритм A_{ij} не удается разместить в один кристалл, то он разбивается на фрагменты, которые выполняются последовательно. Сложность фрагментов алгоритма при этом определяется лишь логической емкостью кристалла. Файлы конфигурации с $l = 1 \div t$ (t – число фрагментов алгоритма A_{ij}), соответствующие этим фрагментам загружаются в кристалл последовательно. Очевидно, что при $t = 1$ алгоритм A_{ij} реализуется в одном кристалле. Конфигурация заказной интегральной схемы, изготавливаемой на ПЛИС, осуществляется посредством записи файла s . Структуры реконфигурируемых устройств, как правило, содержат один или несколько кристаллов ПЛИС, память для хранения файлов конфигурации, порты тестирования/отладки, а также разъемы для подключения внешних устройств.

Реконфигурируемые вычислительные модули могут быть двух типов, заказными и полужаказными. Заказные модули представляют собой вычислитель с жесткой логикой и содержат только те элементы, которые необходимы для выполнения заданных функций. Полузаказ-

© В.Н. Опанасенко, И.А. Щербакова, А.Л. Желєзняк, Л.Л. Гончарова, 2010

ные реконфигурируемые модули предназначены, в основном, для исследования, макетирования, и отработки разрабатываемых проектов. Можно указать несколько типов реконфигурируемых вычислительных устройств. Это автономный вычислитель, который не подключается к какой-либо шине и содержит, как правило, кэш-память внутри модуля (рис. 1). Обращение к памяти может осуществляться как со стороны внешнего устройства, так и со стороны устройства, размещаемого в ПЛИС. В состав модуля должен быть включен контроллер, подключаемый к шине.

Вторым вариантом является реконфигурируемый модуль с возможностью подключения к стандартной шине Host компьютера и содержащий кэш-память внутри модуля (рис. 2). Обращение к памяти должно осуществляться как со стороны шины Host компьютера, так и со стороны устройства, размещаемого в ПЛИС.

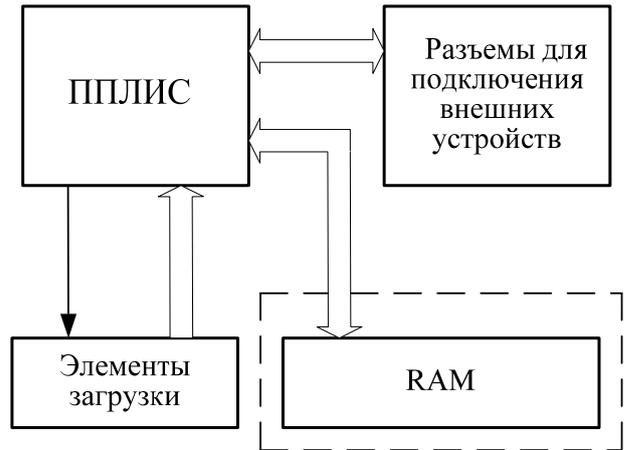


Рисунок 1 - Структурная схема реконфигурируемого модуля

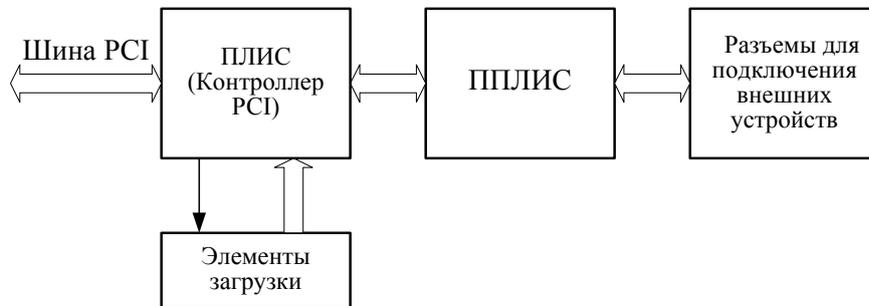


Рисунок 2 - Структурная схема реконфигурируемый модуль с возможностью подключения к шине host-компьютера

В подобных вычислительных структурах с программируемой архитектурой [1] фиксируется обрабатываемое поле заданной размерности, сконфигурированное специально для выполнения определенного заданного алгоритма или его части, обеспечивая таким образом реализацию этого алгоритма оптимальным способом, имея в виду, как время его выполнения, так и затраты аппаратных ресурсов. Третий вариант реализации вычислителей с программируемой архитектурой организован на основе принципа конвейерной реализации алгоритмов в структуре реконфигурируемого модуля. Структурная схема такого модуля представлена на рис. 3 и содержит s матриц, представляющих собой функциональное обрабатывающее поле (ФОП), канал ввода-вывода (КВВ), запоминающее устройство (ЗУ) файлов конфигурации, ОЗУ данных, устройство управления (УУ) шины данных и управления.

Конвейерный механизм предполагает загрузку файла конфигурации в очередную матрицу параллельно с обработкой данных в текущей матрице. Матрица ФОП представляет собой матрицу универсальных элементов, которым под управлением файла конфигурации F_γ назначается непосредственная функция и формируется структура связей между ними.

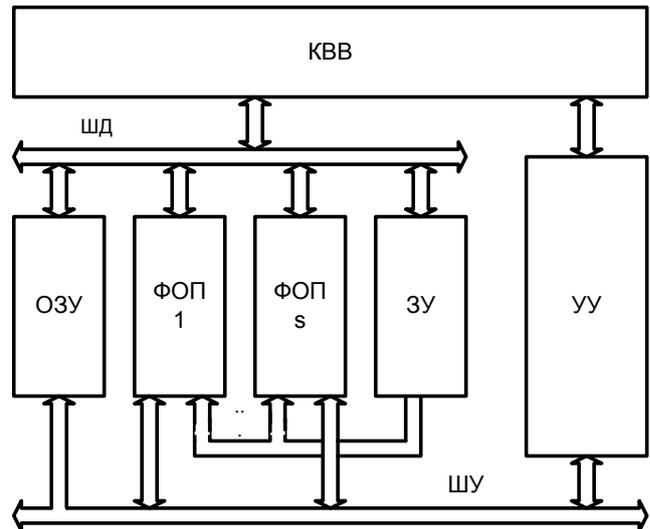


Рисунок 3 - Структурная схема РМК

Файлы конфигурации F_γ записываются в матрицу ФОП. Результаты обработки из матрицы ФОП могут поступать через канал как внешние выходные данные или в ОЗУ как промежуточные результаты. Множество

файлов конфигурации $F = \{F_\gamma\}$ записывается в ЗУ файлов конфигурации через канал КВВ. Инициализация системы состоит из нескольких этапов. На первом этапе реализуется запись множества файлов конфигурации F в ЗУ файлов конфигурации, далее реализуется загрузка файлов конфигурации F_γ в ФОП, после чего модуль запускается в работу. Формат команды содержит номер файла конфигурации и размер файла. Файлы конфигурации F_γ из ЗУ последовательно загружаются в соответствующую матрицу ФОП, в которой формируется сигнал, фиксирующий окончание загрузки файла конфигурации. После загрузки из ЗУ файла конфигурации в ФОП будет сформирована структура устройства для реализации соответствующего алгоритма, включающего в свой состав операционное и внутреннее управляющее устройство. Процедура обработки данных осуществляется следующим образом. В команде указывается номер матрицы ФОП, а УУ формирует сигнал, инициализирующий соответствующую матрицу ФОП. Затем осуществляется загрузка соответствующего файла конфигурации в очередную матрицу ФОП. В команде имеется также ссылка на соответствующую микропрограмму обработки данных. Микропрограмма реализуется в матрице ФОП под управлением внутреннего устройства управления. После завершения обработки данных α -ой матрицей результаты обработки данных записываются в ОЗУ и служат исходными данными для $(\alpha + 1)$ -ой матрицы. После выполнения программы в ФОП формируется прерывание, которое поступает на управляющий вход УУ, где выполняется его обработка. Подобный процессор может быть реализован в виде платы и ориентирован, например, для решения задач обработки сигналов. Внешние цифровые порты позволяют осуществить многоканальный ввод/вывод данных на плату при установке ее в компьютер, а внутренние разъемы расширения позволяют наращивать вычислительную мощь за счет стыка с процессорными платами.

Рассмотрим формализованный подход к проектированию вычислительных систем на основе принципа реконфигурации архитектуры. Модель проектируемой вычислительной системы можно представить следующей системой:

$$S = \langle M, A, B, P \rangle,$$

где M – множество математических методов для предметной области, лежащих в основе функционирования системы; s – множество алгоритмов реализации метода; $B = \{b\}$ – алфавит конструктивов, из которых синтезируется структура (алфавитом называется набор компонентов $\{b\}$, на основе которых строятся допустимые описание и реализация проекта); c – процедура описания объекта. Процесс проектирования состоит в

решении задачи синтеза вычислительной структуры на основе алфавита B для выполнения определенного алгоритма A , реализующего метод M , в соответствии с заданными требованиями. Результатом процедуры P является описание объекта на языке средств автоматизации проектирования. В отличие от традиционных подходов процесс разработки метода и алгоритма решения исходной задачи носит итеративный характер. Критериями эффективности искомого метода (алгоритма) являются обобщенные характеристики производительности, аппаратных затрат, точности решения задачи, сложности алгоритма, надежности проектируемой системы либо специальные критерии, такие как работа в реальном времени, трудоемкость разработки соответствующих метода и алгоритма и др. Создание общей методологии оптимального решения произвольной задачи в произвольной предметной области представляется весьма затруднительным. В этой связи более реалистичным является создание хорошо структурированной библиотеки методов и соответствующих архитектур проектируемой вычислительной системы, хранимых в виде программных файлов во внешней памяти базовой вычислительной подсистемы, и на их базе выбор подходящей пары (метод – архитектура) для конкретной проблемной ситуации [3]. В этом случае задача оптимального синтеза сводится к задаче рационального выбора из предварительно сформированных множеств решений библиотеки файлов конфигураций. Этот процесс можно достаточно строго формализовать. Такой подход позволяет не только получить эффективное решение задачи, но облегчить ее постановку и взаимодействие пользователя с компьютерной системой.

Процесс проектирования вычислительных устройств на ПЛИС полностью поддерживается инструментальными средствами, которые позволяют выполнять: описание проекта, логическое моделирование, размещение и трассировку проекта в заданном кристалле, временную верификацию и формирование стандартного файла для конфигурации кристалла. Процесс проектирования основан на некотором алфавите, в качестве которого могут использоваться: набор сложных проектных решений – $B^4 = \bigcup_{\gamma} b_{\gamma}^4, (\gamma = 1 \div p)$; набор примитивов и макроэлементов стандартной библиотеки – $B^3 = \bigcup_z b_z^3, (z = 1 \div k)$; набор макроэлементов, формируемых разработчиком – $B^2 = \bigcup_j b_j^2, (j = 1 \div m)$; набор параметрических макромоделей – $B^1 = \bigcup_i b_i^1, (i = 1 \div l)$. В конечном итоге конкретный проект будет использовать алфавит конструктивов B :

$B = \bigcup_s B^s, (\forall s = \overline{1 \div 4})$, где часть подмножеств B^s

могут являться пустыми множествами:

$$B^1 = \emptyset \quad (\forall i = 0); \quad B^2 = \emptyset \quad (\forall j = 0);$$

$$B^3 = \emptyset (\forall z = 0); \quad B^4 = \emptyset (\forall \gamma = 0).$$

Рассмотрим синтез структурной реализации последовательности алгоритмов, когда библиотека файлов конфигураций содержит алгоритм в виде одного библиотечного элемента. Пусть метод/задача (М) представляется последовательностью алгоритмов $(A_i, \forall i = \overline{1 \div n}) - M = \bigcup_i A_i$. Для эффективной ра-

боты необходимо сформировать хорошую библиотеку структурных реализаций методов (алгоритмов), выполняющих отображение алгоритма в структурную реализацию $(F : A_i \Rightarrow B_i)$. Каждый алгоритм имеет отображение $F : A_i \Rightarrow B_i$ в структурную реализацию (B_i) , которая представляет собой файл конфигурации для кристалла заказной интегральной схемы. Возможна реализация алгоритмов в параллельной, последовательной, конвейерной и последовательно-параллельной формах:

$$B_i = \bigcup_z B_{iz}, (z = \overline{1 \div k}).$$

Каждый вариант реализации характеризуется параметрами быстродействия (время выполнения – t_{iz}) и аппаратными затратами (q_{iz}). Будем считать, что мощность множества B достаточная для реализации широкого набора алгоритмов. В случае, если требуемая реализация i -го алгоритма в библиотеке отсутствует ($B_i = \emptyset$), то необходимо с помощью инструментальных средств создать этот проект и включить его в качестве стандартного элемента в библиотеку.

При таком походе задача оптимизации сводится к упорядоченному назначению каждой i -й вершине графа реализуемого алгоритма (B_{iz})-го элемента библиотеки для реализации задачи с целью получения экстремального значения некоторого критерия качества. В результате выполнении указанных процедур определяется структура, реализующая заданный граф. Тогда решение задачи может быть получено методами целочисленного математического программирования [4]. В зависимости от требуемого критерия качества можно предложить следующие варианты постановки задачи оптимизации, которая состоит в определении минимума целевой функции.

1. Критерием качества является суммарные аппаратные затраты на реализацию всех алгоритмов:

$$Q = \sum_i \sum_z q_{iz} x_{iz} = \min,$$

$$(\forall i = \overline{1 \div n}, \forall z = \overline{1 \div k}),$$

$$\sum_{z=1}^k x_{iz} = 1, \quad i = \overline{1 \div n}, z = \overline{1 \div k},$$

$$it_r + \sum_i \sum_z (t_{iz} x_{iz}) \leq T_0,$$

где t_r – время реконфигурирования кристалла ПЛИС с помощью файла конфигурации ($t_r = const$), зависит от типа кристалла,

T_0 – допустимое время выполнения всех алгоритмов.

2. Критерием качества является суммарное время выполнения всех алгоритмов:

$$T = \sum_i \sum_z t_{iz} x_{iz} = \min, \quad (\forall i = \overline{1 \div n}, \forall z = \overline{1 \div k}),$$

$$\sum_{z=1}^k x_{iz} = 1, \quad i = \overline{1 \div n}, z = \overline{1 \div k},$$

$$\sum_i \sum_z q_{iz} x_{iz} \leq Q_0,$$

где Q_0 – допустимые аппаратные затраты.

3. Комплексный критерий качества, учитывающий суммарное время выполнения всех алгоритмов и затраты оборудования:

$$\alpha \sum_i \sum_z t_{iz} x_{iz} + \beta \sum_i \sum_z q_{iz} x_{iz} = \min,$$

$$(\forall i = \overline{1 \div n}, \forall z = \overline{1 \div k}),$$

$$\sum_{z=1}^k x_{iz} = 1, \quad i = \overline{1 \div n}, z = \overline{1 \div k}.$$

$$\sum_i \sum_z q_{iz} x_{iz} \leq Q_0,$$

$$it_r + \sum_i \sum_z (t_{iz} x_{iz}) \leq T_0,$$

где α, β – весовые коэффициенты.

Для решения этих задач достаточно применить один из известных методов.

Выводы

1. Предложен подход к постановке задачи оптимального синтеза вычислительной структуры, который в отличие от известных позволяет реализовать синтез архитектуры путем выбора подходящей пары

(метод – архитектура) на основе множества решений из библиотеки файлов конфигураций.

2. Изложены пути применения устройств с реконфигурируемой структурой на базе кристаллов ПЛИС для эффективного решения сложных задач путем динамического изменения логической структуры процессора благодаря чему формируются необходимые проблемно-ориентированные конфигурации.

Литература

1. Палагин А.В., Опанасенко В.Н. Реконфигурируемые вычислительные системы. – К.: Просвіта, 2006. – 295 с.
2. Палагин А.В., Опанасенко В.Н., Лисовый А.Н. Проектирование реконфигурируемых систем на ПЛИС // Технология и конструирование электронной аппаратуры. – 2007. - №3 – С. 15–24.
3. Палагин А.В., Опанасенко В.Н. О проектировании реконфигурируемых устройств на основе программируемых логических интегральных схем // Электронное моделирование. – 2006. - № 4. – С. 65–74.
4. Сергиенко И.В., Шило В.П. Задачи дискретной оптимизации. Проблемы, методы решения, исследования. – Киев: Наук. Думка, 2003. – 261 с.

Резюме

Предложен подход к синтезу вычислительных структур контроллеров, построенных на основе ПЛИС, основанный на создании хорошо структурированной библиотеки алгоритмов управления и соответствующих вычислительных структур проектируемого контроллера. Выполнена формализация задачи оптимального синтеза вычислительной структуры контроллера, критерием качества которой являются суммарные аппаратные затраты на реализацию и время выполнения всех алгоритмов

Запропоновано підхід до синтезу обчислювальних структур контролерів, побудованих на основі ПЛИС, заснований на створенні добре структурованої бібліотеки алгоритмів керування та відповідних обчислювальних структур проектного контролера. Виконана формалізація задачі оптимального синтезу обчислювальної структури контролера, критерієм якості якої є сумарні апаратні витрати на реалізацію і час виконання всіх алгоритмів

An approach to the synthesis of computing structures of controllers that are built on the FPGA, based on a well-structured library of control algorithms and appropriate computing structures of the designed controller is proposed. It was accomplished formalization of the problem of optimal synthesis for the controller computing structure, quality criterion of which structure is the total hardware expense for implementation and execution time of all algorithms

Ключовые слова: контроллер, вычислительная структура, ПЛИС, алгоритм

Поступила 14.12.2009 г.