УДК 681.32

А.А. БАРКАЛОВ[1], Р.В. МАЛЬЧЕВА[2], А.А. ГРИЦЕНКО[2]

[1] ***Institute of Informatics and Electronic Zielenogorski University, Poland***
[2] ***Донецкий национальный технический университет, Украина***

РЕКОНФИГУРИРУЕМЫЙ СОПРОЦЕССОР

В статье предложена архитектура композитного модуля, основанного на применении динамических реконфигурируемых устройств. Рассмотрена архитектура реконфигурируемого сопроцессора, построенного на базе этого модуля, и их применение для построения систем высокой готовности.

композитная логика, реконфигурация, хеш-функция, синхронизация, сбой

Введение

Проектирование современных аппаратных систем происходит в условиях недостаточной формализации реализуемых ими алгоритмов, обоснованной как недостаточной формализацией требований в фазе проектирования, так и постоянным развитием самих алгоритмов. Для обеспечения высокого уровня качества создаваемых систем необходимо обеспечить их гибкость по отношению к алгоритмам, которые они реализуют. Эта проблема решается путем использования статически реконфигурируемых устройств, позволяющих изменять алгоритм работы после завершительной стадии проектирования. Однако использование статически реконфигурируемых устройств не решает проблему в полной мере, так как жестко ограничивает возможности изменения алгоритма физическими параметрами и отсутствием динамики изменения во время работы.

В данной работе предлагается архитектура композитного модуля, использующего динамически реконфигурируемые устройства [1], обладающего высоким уровнем гибкости и универсальности. Предлагаемая архитектура ориентирована на использование в системах, в которых формализация алгоритма отложена на фазы внедрения и эксплуатации. Предлагается архитектура реконфигурируемого сопроцессора, построенного на основе композитных модулей, предназначенного для решения круга задач широкого профиля и применение композитных модулей для проектирования систем высокой готовности. Проектные материалы представлены в соответствии стандарту UML 2.0 [2].

Архитектура композитного модуля

Композитный модуль (КМ) представляет собой систему, комбинирующую фиксированную и реконфигурируемую логику (рис. 1).

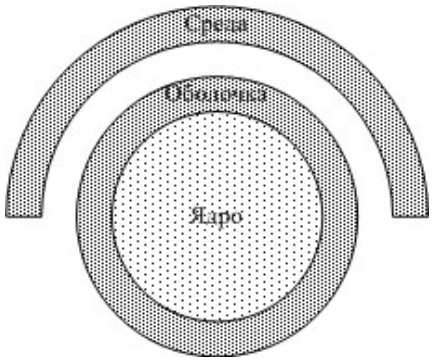

Рис. 1. Архитектура композитного модуля

Фиксированная логика (оболочка) КМ инкапсулирует его реконфигурируемое ядро от внешней среды, делая реконфигурацию прозрачной для нее. Оболочка обеспечивает получение, обработку и ретрансляцию запросов среды в ядро, играя роль драйвера, управляет динамической реконфигурацией ядра, играя роль конфигуратора (рис. 2). В каж-

дый момент времени ядро содержит одну секцию приложения, реализуемого им.

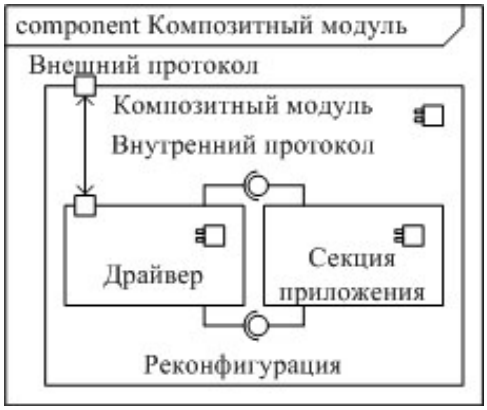

Рис. 2. Компоненты композитного модуля

Ядро развертывается на динамически реконфигурируемых устройствах (например, FPGA устройства, которые обеспечивают интерфейс физической реконфигурации (рис. 3)).

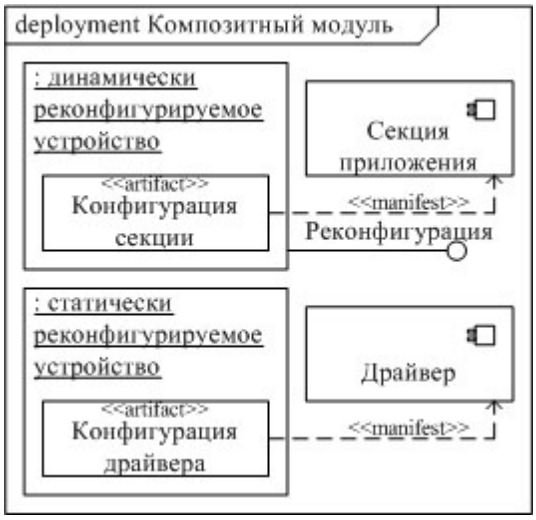

Рис. 3. Развертывание композитного модуля

Драйвер может развертываться на следующих типах устройств:

- КЛС [3] – протоколы взаимодействия со средой фиксированы, модуль является встроенным и не изменяет своей роли;
- статически реконфигурируемое устройство (СРУ) (рис. 3) – протоколы взаимодействия со средой могут изменяться, модуль может изменять свою роль, но потребует приостановки системы на относительно продолжительный период;
- динамически реконфигурируемое устройство (ДРУ) – протоколы взаимодействия со средой динамичны, модуль может изменять свою роль динамически.

Использование различных типов устройств отражается на цене и производительности системы (рис. 4, иллюстративный график).

Системы, спроектированные на базе КМ, характеризуются большой степенью гибкости, как следствие, имеют низкую производительность и высокую стоимость.

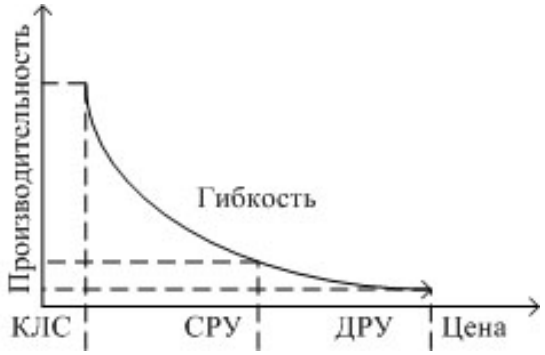

Рис. 4. Зависимость цены и производительности устройства от гибкости его логики

В зависимости от реализации логики драйвера, в процессе обработки одного внешнего запроса может быть произведено несколько реконфигураций ядра (рис. 5). Реконфигурация ядра инициируется драйвером, в результате анализа информации о запросе и состоянии его обработки. Для реконфигурации используется физический интерфейс ДРУ, поэтому с логической точки зрения это представляется уничтожением одной секции приложения и созданием другой.

Взаимодействие со средой осуществляется по асинхронно-синхронному [3, стр. 245-248] или асинхронному протоколам (рис. 6).

Асинхронный протокол является более приемлемым, из-за возможно больших потерь времени при использовании синхронизации.

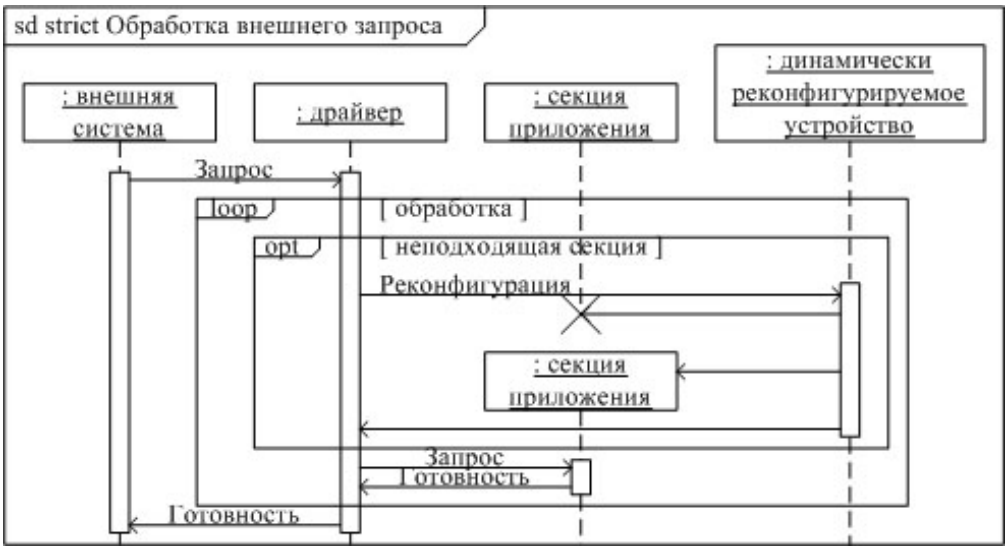

Рис. 5. Реконфигурация ядра композитного модуля

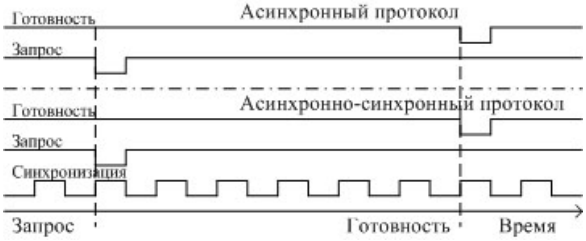

Рис. 6. Протоколы взаимодействия

Время обработки запроса составляет:

$$T_{обработки} = T_{драйвера} +$$

$$+\sum_{i=1}^{n}(T^{i}_{драйвера} + T^{i}_{реконфигурации} + T^{i}_{ядра}).$$

Статическая составляющая времени работы определяется временем получения и декодирования запроса драйвером, динамическая – временем конфигурации ядра и обработки запроса. Диапазон изменения динамической составляющей трудно прогнозировать из-за поздней фазы формализации алгоритма и его инкапсуляции, что не дает использовать синхронный протокол.

КМ взаимодействует с памятью конфигураций (ПК), в которой хранятся аппаратные конфигурации секций [4]. Доступ к ПК осуществляется по чтению. При реконфигурации драйвер осуществляет выборку необходимой конфигурации. ПК может рассматриваться как словарь, где ключом является значение некоторой хэш-функции, а значением – поток данных конфигурации (определяемый начальным адресом в ПК и размером). Хэш-функция определяется в зависимости от типа приложения, развернутого на КМ (для управляющего автомата зависит от предыдущей конфигурации и состояния системы в точке реконфигурации, для операционного устройства – от микрокоманды).

КМ является гибкой аппаратной платформой, которая позволяет формализовать алгоритм работы на фазе внедрения или эксплуатации. Недостатками являются проблемы синхронизации; пониженная производительность и повышенная стоимость, относительно традиционных систем с жесткой логикой.

Архитектура реконфигурируемого сопроцессора

Реконфигурируемый сопроцессор (РС) – это адаптивная аппаратная система обработки запросов [5].

РС базируется на архитектуре СЦВМ (рис. 7), где роль управляющего автомата [4] выполняет контроллер, а операционного устройства [6] – исполнитель. Контроллер и исполнитель реализуются на КМ, обеспечивая универсальность управляющего алгоритма и операционной части.

Предлагаемая архитектура является масштабируемой (рис. 8).

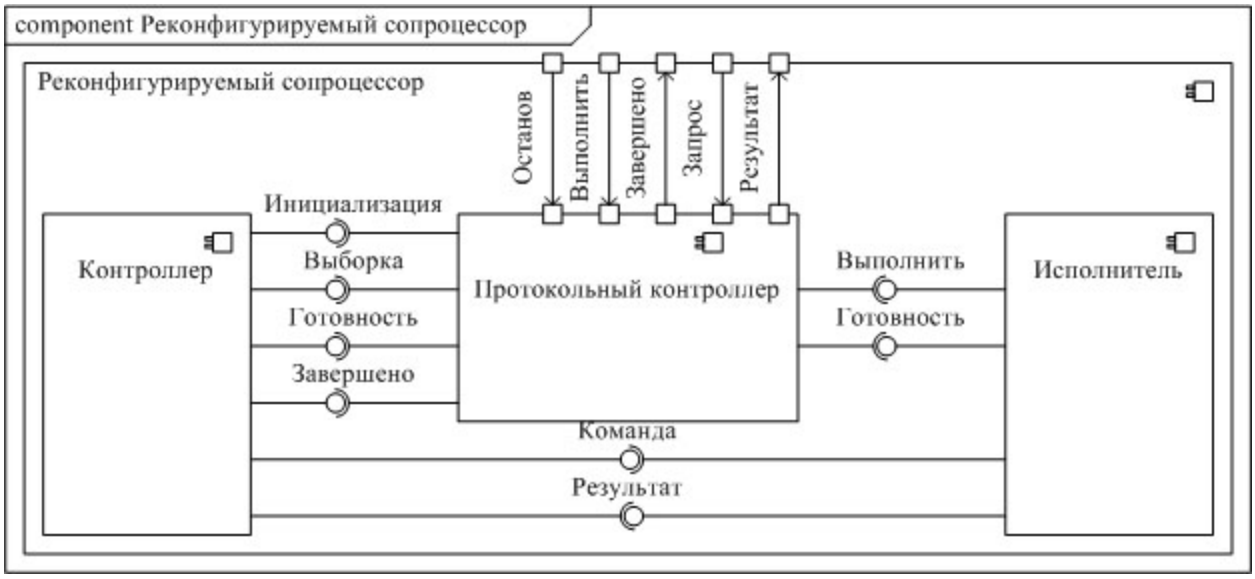

Рис. 7. Архитектура реконфигурируемого сопроцессора

Для масштабирования системы производится замена исполнителя сопроцессора высшего уровня (родительского) сопроцессором низшего уровня (дочерним), при этом протокольный контроллер, выступая делегатом интерфейсов реконфигурируемого сопроцессора, обеспечивает выполнение контрактов исполнителя.

Рис. 8. Масштабирование РС

Интерфейс инициализации, обеспечиваемый контроллером, необходим для загрузки начальной секции управляющего автомата после запуска РС, так как в момент запуска система еще не обладает состоянием, которое необходимо для правильной конфигурации контроллера, в частности, для формирования хэш-функции.

Кроме того, инициализация необходима в том случае, если поддерживается несколько точек входа в управляющую программу.

Инициализация исполнителя произойдет при первом обращении к нему в соответствии с переданным кодом команды.

Исполнитель может иметь дополнительный интерфейс с внешней памятью для систем обработки данных [4].

Интерфейсы, предоставляемые контроллером и исполнителем, обеспечиваются драйверами соответствующим им КМ.

Протокольный контроллер обладает только фиксированной логикой (рис. 9) и является фасадом РС [7, стр. 183-191].

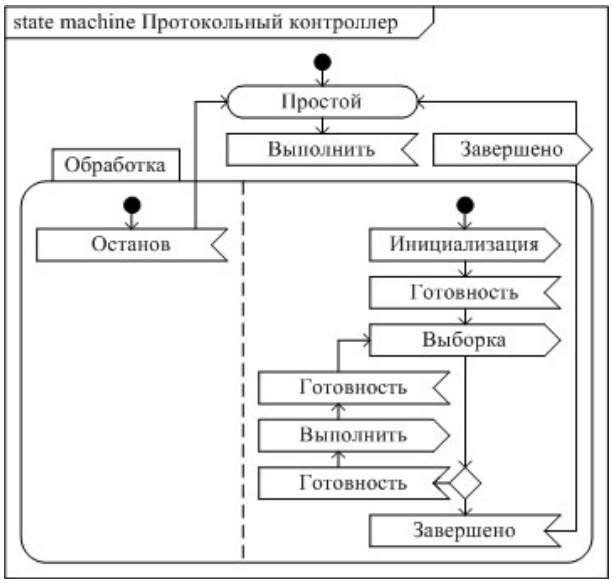

Рис. 9. Логика протокольного контроллера

РС представляет собой аппаратную систему, обладающую свойствами гибкости, универсальности и масштабируемости.

Пример применения предложенной архитектуры для реализации СВГ

КМ и построенный на его основе РС обладают устойчивостью к сбоям, что позволяет использовать их для построения систем высокой готовности (СВГ) (рис. 10):

Рис. 10. Обработка сбоев

1) при возникновении аппаратного сбоя КМ (РС) может быть реконфигурирован с использованием имеющихся ресурсов при полном развертывании подсистем;

2) РС может развертываться распределенно – контроллер и исполнитель развертываются попеременно;

3) при программном сбое в работе алгоритма система может использовать другой вариант алгоритма (при этом хэш-функция конфигураций включает информацию о наличии сбоя).

В первом случае после развертывания потребуются новые конфигурации секций приложения: при развертывании драйвер использует фиксированное количество ресурсов, поэтому неиспользуемые сбойные ресурсы должны компенсироваться ядром.

Для обеспечения возможности реконфигурации на меньшем количестве ресурсов память конфигураций должна содержать несколько наборов данных, определяющих конфигурации для различного количества доступных ресурсов (это могут быть различные варианты декомпозиции одного алгоритма, либо специальные модификации для работы после сбоя). Для различения модификаций алгоритма хэш-функция должна включать информацию о количестве доступных ресурсов.

Второй случай может возникнуть в случае, когда контроллер и исполнитель были развернуты на различных устройствах, одно из которых вышло из строя, при этом одновременное развертывание контроллера и исполнителя на доступных ресурсах невозможно. В этом случае модифицируется логика протокольного контроллера, который должен обеспечить попеременную конфигурацию контроллера и исполнителя (рис. 11).

Рис. 11. Протокольный контроллер СВГ

Диагностика включает оценку доступных аппаратных ресурсов (только после аппаратного сбоя) и анализ наличия необходимых данных (аппаратных конфигураций) в памяти конфигураций. Критическим является сбой, для которого нельзя найти подходящих данных для повторного конфигурирования системы после сбоя или нельзя перераспределить ресурсы для развертывания системы.

И в случае программных, и в случае аппаратных сбоев изменяется хэш-функция, отвечающая за идентификацию аппаратных конфигураций в процессе дальнейшей работы.

Предлагаемые архитектуры являются устойчивыми к сбоям, при этом не требует введения дублирующих ресурсов. Недостатком является то, что после обработки аппаратных сбоев, происходит снижение производительности. Преимущество в том, что архитектуры могут выдерживать серии сбоев.

Выводы

В статье предложена архитектура КМ – аппаратной системы, обладающей высокой степенью гибкости, универсальности и масштабируемости. Рассмотрено построение на основе КМ более сложных систем, в частности, РС – адаптивной системы обработки запросов. Рассмотрена возможность применения предложенных архитектур для построения СВГ, приведены специфические алгоритмы обработки сбоев, поддерживаемые архитектурами.

Эта статья описывает материал, используемый в следующих исследованиях:

- разработке методов декомпозиции конечных автоматов для их реализации с использованием КМ;
- программно-аппаратное моделирование предлагаемых архитектур динамических систем для определения взаимосвязей между производительностью систем и их реализацией (на физическом и логическом уровнях) и процессов обработки сбоев оборудования.

Литература

1. Палагин А.В. Реконфигурируемые вычислительные системы: Основы и приложения. – К.: Просвита, 2006. – 280 с.

2. Буч Г., Якобсон А., Рамбо Дж. UML. Классика CS. 2-е изд.: Пер. с англ. – СПб.: Питер, 2006. – 736 с.

3. Майоров С.А., Новиков Г.И. Принципы организации цифровых машин. – Л.: Машиностроение, 1974. – 432 с.

4. Bobda C., Majer M., Ahmadinia A., Haller T. The Erlangen Slot Machine: Increasing Flexibility in FPGA-based reconfigurable platforms // University of Erlangen-Nuremberg, Germany, 2004. – 6 p.

5. Neema S., Barty T., Scott J. Adaptive Computing and Runtime Reconfiguration // Vanderbilt University, Peabody, Nashville, USA, 1997. – 8 p.

6. Баркалов О.О., Ковальов С.О., Мальчева Р.В. Проектування операційних пристроїв. – Донецьк: РВА ДонНТУ, 2005. – 312 с.

7. Гамма Э. и др. Приемы объектно-ориентированного проектирования. – СПб.: Питер, 2006. – 366 с.

Поступила в редакцию 27.02.2007

Рецензент: д-р техн. наук, проф. В.А. Краснобаев, Харьковский национальный технический университет сельского хозяйства им. П. Василенко, Харьков.