УДК 004.272.43

ДВА ПОДХОДА К СОЗДАНИЮ МАКРООБЪЕКТОВ ДЛЯ РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ НА ОСНОВЕ ПЛИС

В.Б. Коваленко [1] М.С. Кочерга [1] Е.А. Семерников [2]

[1]Россия, г. Таганрог, НИИ многопроцессорных вычислительных систем ЮФУ
[2] Россия, г. Ростов-на-Дону, Южный научный центр РАН

В статье рассматриваются два подхода к созданию макрообъектов для реконфигурируемых вычислительных систем (РВС) на основе ПЛИС. РВС с макрообъектной архитектурой, с одной стороны, обеспечивают преимущества систем, структура которых программируется на схемотехническом уровне, а с другой, существенно упрощают программисту процесс адаптации архитектуры системы к структуре решаемых задач.

Введение

Эффективное использование РВС на основе ПЛИС является нетривиальной задачей, поскольку программировать нужно не только организацию вычислительного процесса, как это происходит в обычных многопроцессорных вычислительных системах, но и структуру РВС, адаптируя ее к структуре решаемой задачи. Эффективность вычислительного процесса при реконфигурации архитектуры РВС на низком (схемотехническом) уровне может быть повышена от 10 до 100 раз по сравнению с вычислительными системами, архитектура которых не может быть изменена. Это делает, с одной стороны, чрезвычайно привлекательными реконфигурируемые на низком уровне системы, а с другой стороны, их программирование становится по сложности сопоставимым с созданием новой вычислительной системы. Совершенствование методов адаптации архитектуры РВС к структуре решаемых задач возможно на пути создания РВС с макрообъектной архитектурой [1,2]. РВС с макрообъектной архитектурой, с одной стороны, обеспечивают эффективность РВС, реконфигурируемых на схемотехническом уровне, а с другой, позволяют существенно облегчить процесс адаптации архитектуры системы к структуре решаемых задач.

Аппаратный подход к созданию макрообъектов

Реконфигурируемые вычислительные системы могут содержать в своих решающих полях десятки и сотни ПЛИС сверхвысокой интеграции при этом количество логических блоков, потенциально доступных пользователю РВС, достигает нескольких десятков миллионов. Имея в своем распоряжении огромный схемотехнический ресурс, пользователь может реализовать практически любые вычислительные узлы с любыми техническими параметрами. Это позволяет не сдерживать разработчика в его стремлении к различным формам реализации, но в то же время усложняет формализацию процесса разработки прикладных программ.

В упрощенном виде программирование прикладной задачи заключается в следующем: необходимо, прежде всего, создать в аппаратном ресурсе РВС вычислительную структуру адекватную структуре алгоритма решения задачи и, подав на нее входные данные задачи, дождаться получения результата.

Однако при программировании РВС пользователь сталкивается с рядом проблем. Во-первых, сам процесс программирования вычислительной структуры в большом решающем поле ПЛИС – очень сложный и трудоемкий процесс, требующий высокой квалификации пользователя и специальных средств программирования. Во-вторых, перезагрузка конфигурационных файлов большого количества ПЛИС занимает длительное время. В-третьих, вычислительная структура получается более оптимальной, если она располагается в поле ячеек нескольких (а не одной) ПЛИС, в то время как существующие САПР не позволяют выйти за рамки одной микросхемы. В-четвертых, ничем не ограниченное многообразие технических решений, которые предоставляют пользователю технологии ПЛИС, не позволяют перейти к формальным методам создания вычислительных структур при программировании РВС. Список подобных проблем можно продолжить.

Для реализации принципов программирования РВС и преодоления перечисленных выше проблем было введено понятие макрообъекта [1, 2].

Макрообъектом называется архитектурно неделимая совокупность функциональных узлов (объектов), объединенных пространственной коммутационной системой и размещенных в одной или нескольких ПЛИС. При этом для макрообъекта допустимо изменение количества функциональных узлов того или иного типа, параметров узлов (разрядности операндов, числа информационных каналов, системы команд и т.п.), но не их назначения [2].

Иными словами, макрообъект является некоторой заготовкой, которая может доопределяться пользователем в процессе создания конкретного технического решения, а затем тиражироваться в схемотехническом ресурсе ПЛИС базовых модулей в необходимом количестве и соединяться с подобными или другими макрообъектами в вычислительные структуры, которые оптимально соответствуют структуре решаемой задачи.

Архитектура РВС, основанная на использовании макрообъектов, получила название *макрообъектной архитектуры* [1, 2].

Описанный выше вариант построения РВС с макрообъектной архитектурой отражает аппаратный подход, который предполагает создание аппаратных проблемно-ориентированных решений, разработка которых предполагает участие специалиста, владеющего схемотехникой ПЛИС. Схемотехник участвует как на этапе создания библиотечных элементов, входящих в состав макрообъекта, так и на этапе создания самих макрообъектов. В процессе создания проблемно-ориентированных макрообъектов участвуют также алгоритмисты, которые представляют задачу в виде, пригодном для решения с помощью соответствующих макрообъектов. Примеры аппаратного подхода при создании макрообъектов подробно описаны в [1, 2, 3].

Программный подход к созданию макрообъектов

Рассмотрим несколько иной подход к построению проблемно-ориентированных макрообъектов, который предполагает участие пользователя только с квалификацией программиста.

Известно [1, 2], что программирование РВС отличается от программирования МВС традиционной архитектуры, и его можно условно разделить на две составляющие: программирование структурное, которое создает необходимые вычислительные структуры в поле логических ячеек ПЛИС, и программирование процедурное – программирование в традиционном смысле, заключающееся в организации вычислительного процесса в РВС. При этом программирование

вычислительных структур вызывает у пользователей наибольшие трудности [1, 2, 4]. Это связано с тем, что традиционно, пользователи привыкли программировать только организацию вычислительного процесса, опираясь на неизменяемую аппаратную поддержку средств вычислительной техники, в то время как для программирования вычислительных структур РВС требуется совершенно другая квалификация, а именно - квалификация схемотехника. При программировании пользовательской задачи структура РВС приобретает черты специализированной многопроцессорной ЭВМ, которая оптимально соответствует структуре решаемой задачи из предметной области.

В [1, 2, 4] приведено описание языка параллельного программирования высокого уровня COLAMO, позволяющего преодолеть семантический разрыв между параллельным алгоритмом и его реализацией в РВС. Язык COLAMO позволяет эффективно реализовывать различные способы организации параллельных вычислений, имеет средства для описания алгоритмов без изменения их изначальной параллельной структуры и, самое главное, легко реализуется на РВС различных конфигураций и различных аппаратных решений.

Используя средства языка COLAMO, можно с той или иной эффективностью запрограммировать алгоритм решения практически любой задачи. Транслятор в процессе трансляции COLAMO-программы создает структурную составляющую параллельного алгоритма, которая соответствует его информационному графу [5]. В процессе реализации структурной составляющей в аппаратуре РВС каждой вершине информационного графа будет поставлен в соответствие библиотечный элемент, реализующий арифметическую операцию, а каждой дуге – библиотечный элемент соответствующего интерфейса. Входным и выходным вершинам информационного графа будут поставлены в соответствие каналы распределенной памяти РВС [1, 2]. Таким образом, в РВС будет создана вычислительная структура, соответствующая информационному графу. Наборы библиотечных элементов соответствуют списку допустимых операций языка COLAMO и создаются схемотехниками на этапе проектирования РВС.

В базовую версию языка COLAMO включен функционально полный, но все-таки ограниченный набор общеупотребимых арифметических и логических операций, таких как сложение, вычитание, умножение, деление, сравнение и т.п. Очевидно, что запрограммировать объект, реализующий сложные информационные зависимости, содержащие множество условных переходов из допустимых операций базовой версии языка COLAMO весьма проблематично. А, самое главное, эффективность такого объекта при реализации в аппаратном ресурсе РВС будет невысокой.

Поэтому предлагается ввести в язык COLAMO специальные функции, которые, с одной стороны, позволяют оперировать с ними как с элементами языка, а с другой стороны, они, будучи реализованными в виде библиотечных элементов, имеют эффективную аппаратную реализацию. Аппаратная реализация алгоритма задачи, запрограммированного средствами языка COLAMO с использованием таких функций, будет иметь все основные признаки макрообъекта. В то же время этот макрообъект создан программистом путем написания COLAMO-программы. Специалисты схемотехник и алгоритмист принимают участие только на этапе создания библиотечных элементов.

Наборы специальных функций создаются для различных предметных областей и для реализации специальных алгоритмов. Подключение библиотек специальных функций к базовой версии языка COLAMO возможно по мере необходимости посредством операторов типа include.

Рассмотрим пример реализации проблемно-ориентированного макрообъекта цифровой обработки сигналов из элементов библиотеки специальных функций «LibDSP_1».

Библиотека «LibDSP_1» включает три специальные функции:

- *FFT_IT(A,B,At,Bt,Param_FFT_IT)* – функция выполнения одной ступени конвейерного БПФ, включая формирование коэффициентов Wt и устройство конвейерной задержки. A, B – имена входных массивов. At, Bt – имена выходных массивов. Param_FFT_IT – набор параметров настройки ступени конвейера БПФ;

- *Kash(A,B,C,D,Param_Kash)* – функция, определяющая двухпортовую КЭШ-память объемом 8К 64-разрядных слов с адресным процессором. КЭШ-память используется для хранения входных данных, промежуточных и конечных результатов при выполнении алгоритмов БПФ размерности до 8192 комплексных отсчетов. A, B – имена входных массивов. C, D – имена выходных массивов. Param_Kash – набор параметров настройки, необходимых для работы КЭШ-памяти и адресного процессора при выполнении алгоритмов БПФ необходимой размерности;

- *Mashtab(A,B,C,D,Param_Mashtab)* – функция, реализующая устройство масштабирования результатов алгоритма БПФ. A, B – имена входных массивов. C, D – имена выходных массивов. Param_Mashtab – параметр масштабирования результатов.

На рис. 1 приведен фрагмент программы FFT_Lib на языке COLAMO для алгоритма БПФ над массивом длиной 4096 комплексных отсчетов с использованием специальных функций.

```
for j := 0 to 1 do
begin
  if j = 0 then         //Условный оператор, порождающий мультиплексор M
  begin
    Re_t[*] := Re[*];
    im_t[*] := Im[*];
    if last_kadr = 1 then paramKt := param3K
    else paramKt := param1K;
  end
  else if j = 1 then
  begin
    Re_t[*] := re_c[k,*,7];
    Im_t[*] := im_c[k,*,7];
    paramKt := param2K
  end;

  Kash(re_t[*], im_t[*], re_c[k,*,0], im_c[k,*,0], paramKt);

  if j=0 then
  begin
    FFT_IT(re_c[k,*,0], im_c[k,*,0], re_c[k,*,1], im_c[k,*,1], param_FFT_1[k,*, 0]);
    FFT_IT(re_c[k,*,1], im_c[k,*,1], re_c[k,*,2], im_c[k,*,2], param_FFT_1[k,*, 1]);
    FFT_IT(re_c[k,*,2], im_c[k,*,2], re_c[k,*,3], im_c[k,*,3], param_FFT_1[k,*, 2]);
    FFT_IT(re_c[k,*,3], im_c[k,*,3], re_c[k,*,4], im_c[k,*,4], param_FFT_1[k,*, 3]);
    FFT_IT(re_c[k,*,4], im_c[k,*,4], re_c[k,*,5], im_c[k,*,5], param_FFT_1[k,*, 4]);
    FFT_IT(re_c[k,*,5], im_c[k,*,5], re_c[k,*,6], im_c[k,*,6], param_FFT_1[k,*, 5]);
    Mashtab(re_c[*,6], im_c[*,6], re_c[*,7], re_c[*,7], N2, param_mash[k])
  end;
end;
if last_cadr = 1 then
begin
  re_out_c[*] := re_c[k, *, 0];
  im_out_c[*] := im_c[k, *, 0];
end;
```

Рис. 1. Фрагмент COLAMO-программы FFT_Lib

На рис. 2 показана структура макрообъекта, созданного по приведенному фрагменту программы FFT_Lib. Созданный макрообъект является проблемно-ориентированным вычислительным устройством, способным выполнять алгоритмы БПФ над массивами длиной от 64 до 8192 комплексных отсчетов.

Программа для выполнения алгоритма БПФ, написанная на COLAMO без привлечения специальных функций, довольно громоздка, и на одну базовую операцию БПФ требует десять каналов распределенной памяти. Программа, приведенная на рис. 1, требует всего четыре канала распределенной памяти на шесть базовых операций

БПФ. Помимо этого количество операторов в программе FFT_Lib почти в четыре раза меньше, чем в COLAMO-программе без использования специальных функций.

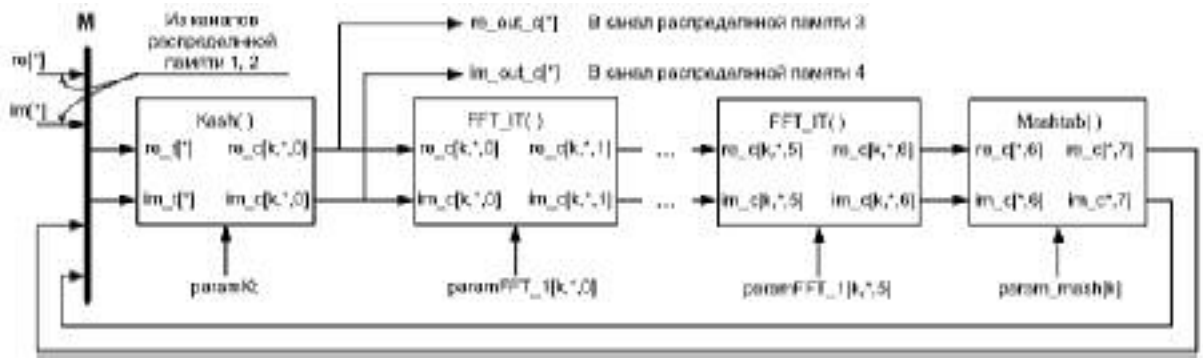

Рис. 2 Структура макрообъекта, созданного по COLAMO-программе

Заключение

В статье описаны два подхода к созданию макрообъектов для РВС с макрообъектной архитектурой. Конечно, аппаратный подход обеспечивает более эффективное использование аппаратного ресурса РВС. Однако создание таких макрообъектов является сложной, нетривиальной задачей, требующей участия разнородных специалистов: алгоритмиста, схемотехника и программиста.

Программный подход позволяет создавать проблемно-ориентированные макрообъекты с участием одного программиста. Алгоритмист и схемотехник участвуют только на этапе создания специальных функций для данной предметной области. Программист создает параллельную программу средствами языка высокого уровня COLAMO с использованием специальных функций. В результате транслятор при поддержке комплекса средств разработки программ для РВС создает в аппаратном ресурсе РВС необходимые проблемно-ориентированные макрообъекты для решения прикладной задачи.

Список литературы

1. Каляев А.В., Левин И.И. Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений. - М.: Янус-К, 2003. – 380 с.

2. Каляев И.А., Левин И.И., Семерников Е.А., Шмойлов В.И. Реконфигурируемые мультиконвейерные вычислительные структуры. - Ростов н/Д: Издательство ЮНЦ РАН, 2008. - 320 с.

3. Доронченко Ю.И., Семерников Е.А. Конвейерный макропроцессор цифровой обработки сигналов со структурно-процедурной организацией вычислений // Вестник компьютерных и информационных технологий. - М: Машиностроение, 2005. - №6. – С. 49-55.

4. Левин И.И. Язык параллельного программирования высокого уровня для структурно-процедурной организации вычислений // Труды Всероссийской научной конференции. - М.: Изд-во МГУ, 2000. – С. 108-112.

5. Воеводин В.В. Параллельные вычисления / В.В. Воеводин, Вл.В. Воеводин - С.-Петербург: «БХВ-Петербург», 2002. - 599 с.