Моделювання та ін формаційні технології **Збірник наукових праць Інституту проблем моделювання в енергетиці ім. Г.Є.Пухова НАН України. Вип. 21.** - К., 2003. - с.

СПОСОБ МИНИМИЗАЦИИ АППАРАТУРНЫХ ЗАТРАТ МИКРОКОНТРОЛЛЕРОВ

В.Л.Лепеха, А.М.Сергиенко, Ю.Шевченко

Аннотация.

В работе рассматриваются вопросы разработки настраиваемых вычислительных модулей микроконтроллеров для их реализации в системах на кристалле. Предлагается способ минимизации аппаратурных затрат такого микроконтроллера, основанный на настройке его структуры под исполняемую программу. Начальная структура микроконтроллера имеет набор функциональных блоков, ассоциируемых с отдельными подмножествами команд и адресов памяти. Эта структура описывается на языке VHDL с возможностями настраиваемого отключения функциональных блоков. Автоматический анализ программы микроконтроллера, проверяя, какие команды и адреса памяти не используются, дает информацию для настройки VHDL-модели. Компилятор-синтезатор при трансляции VHDL-модели исключает из нее неиспользуемые функциональные блоки и дает результирующую минимизированную логическую схему микроконтроллера. Предложенный способ при разработке микроконтроллера с архитектурой i8051, который был реализован в ПЛИС Xilinx Virtex, позволил уменьшить аппаратурные затраты до 1,6 раза.

Ключевые слова: система на кристалле, совместное аппаратно-программное проектирование, микроконтроллер, VHDL , ПЛИС.

1.Введение

Технология разработки систем на кристалле (СНК) становится все более распространенной, так как она обеспечивает удешевление новых изделий электроники, повышение их надежности, уменьшение энергопотребления. СНК, реализованные в ПЛИС, имеют преимущества в быстроте разработки и изготовления, возможности производства малых партий. Одним из основных принципов разработки СНК является применение крупных библиотечных вычислительных модулей (intellectual property cores), которые должны быть надежно повторяемыми и настраиваемыми под решаемые задачи в ряде проектов СНК. Применение таких модулей, которые можно назвать вычислительными заготовками за их функциональную и технологическую адаптируемость, позволяет уменьшить трудозатраты проектирования СНК и значительно уменьшить сроки проектирования [1].

Вычислительные заготовки различаются по степени гибкости своей настройки под условия потребителя как гибкие (описанные языком описания аппаратуры, таком как VHDL, на уровне регистровых передач), жесткие (логическая схема) и твердые (маски под определенную технологию). Гибкие заготовки обычно подстраиваются к условиям нового проекта в широких пределах и независимы от его технологии. Минимизация аппаратурных затрат вычислительных заготовок обеспечивает не только уменьшение стоимости СНК, но и уменьшение его энергопотребления, что является важным фактором для портативных и энергонезависимых приложений.

В современных встраиваемых цифровых системах микроконтроллеры играют роль "рабочей лошадки". Среди микроконтроллеров с различной архитектурой наиболее распространенными являются 8-разрядные микроконтроллеры, наиболее популярным из которых считается микроконтроллер с архитектурой i8051. Поэтому вычислительные заготовки с архитектурой i8051 очень часто применяются в СНК [2].

Микроконтроллеры предназначены для выполнения заранее заданных вычислительных и управляющих функций. Так как эти функции, как правило, неизменны в

течение срока службы микроконтроллера, то желательно, чтобы микроконтроллер был узкоспециализированным. Но применяемая ранее технология микросхем требовала, чтобы изделия микроэлектроники были унифицированными и поэтому микроконтроллер обычно проектировался универсальным. Из-за этой универсальности нередко больше половины оборудования микроконтроллеров во встраиваемых приложениях постоянно простаивает.

Современная технология СНК дает разработчикам широкие возможности по разработке крупных проектов в краткие сроки с экономической целесообразностью изготовления малых партий изделий. Это делает выгодной разработку специализированных микроконтроллеров. Но для этого нужны новые методы их проектирования. В данной работе рассматриваются вопросы минимимизации аппаратурных затрат гибких заготовок микроконтроллеров для СНК и приводится пример заготовки микроконтроллера i8051 с минимизацией аппаратурных затрат путем адаптации к алгоритму приложения.

2. Пути минимизации аппаратурных затрат микроконтроллеров.

Так как микроконтроллеры состоят из множества логических схем и схем памяти, то традиционно их оборудование уменьшают путем логической минимизации и оптимизации числа регистров, объема ОЗУ, ПЗУ. За полувековую историю развития вычислительной техники разработано большое множество методов минимимизации оборудования ЭВМ, которые можно применить к разработке микроконтроллеров. Эти методы направлены на выбор системы команд, адресации, минимизацию оборудования АЛУ, управляющих автоматов, ПЗУ [3]. Но современная технология разработки СНК вносит свои коррективы в выбор методов и требует поиска новых методов оптимизации. Сейчас логическая оптимизация обычно выполняется автоматически на стадии отображения описания микроконтроллера на уровне регистровых передач в логические схемы с помощью компиляторов-синтезаторов с языков VHDL или Verilog. И поэтому, концентрируясь на оптимизации булевских функций, практически невозможно получить существенную дополнительную минимизацию оборудования.

Наиболее эффективным подходом является синтез специализированной системы команд микроконтроллера. Такой синтез может быть направленным не только на оптимальную реализацию некоторого класса прикладных задач, но и на минимизацию аппаратурных затрат результирующей схемы микроконтроллера [3], [4]. Однако, при этом возникают трудности с программированием в такой нестандартной системе команд. Хотя уже существуют компиляторы, которые настраиваются на произвольную архитектуру микропроцессора, но они все еще находятся на экспериментальной стадии разработки, а их эффективность далека от желаемой [5].

Большую долю аппаратурных затрат микроконтроллера представляет ПЗУ программ. Так, ПЗУ объемом 4-8 килобайт может занимать на кристалле такую же площадь, как и сам 8-разрядный микроконтроллер. Поэтому имеет смысл минимизировать объем этого ПЗУ, например, путем упаковки информации. В работе [6] показано, что таким способом можно уменьшить необходимый объем ПЗУ на 7-20%.

В настоящее время наиболее распространенным практическим подходом является выбор вычислительной заготовки микроконтроллера с наименьшими аппаратурными затратами из числа имеющихся на рынке. В [7] представлена заготовка микроконтроллера i8051, которую можно настраивать, изменяя в проекте вручную число необходимых таймеров, страниц ОЗУ, подключая порт последовательного ввода-вывода и другие периферийные устройства. Такой подход, когда в заготовке микроконтроллера регулируется множество подключаемых блоков, в настоящий период представляется как наиболее перспективный .

3. Минимизация аппаратурных затрат путем настройки структуры микроконтроллера.

Так как микроконтроллер предназначен для выполнения заранее заданных вычислительных и управляющих функций, то он может быть узкоспециализированным. В нем, в зависимости от реализованного алгоритма, могут не использоваться все порты ввода-вывода, периферийные устройства, предусмотренные его архитектурой. Также могут не использоваться многие команды из набора команд, который обеспечивает универсальность микроконтроллера. Так, в микроконтроллере i8051 такие команды, как десятичная коррекция, умножение, деление, битовые операции и другие используются далеко не во всех приложениях, но требуют для своей реализации существенных дополнительных аппаратурных затрат. Поэтому, в зависимости от решаемых задач, система команд микроконтроллера может быть сокращена.

Предлагаемый способ минимизации аппаратурных затрат микроконтроллеров, основан на настройке его структуры под исполняемую программу. Для этого структура микроконтроллера должна иметь как можно более широкое множество функциональных блоков, ассоциируемых с отдельными подмножествами команд и адресов памяти. Путем анализа исполняемой программы можно выявить, какие его команды и адреса памяти не используются в вычислениях. Тогда при трансляции описания микроконтроллера на уровне регистровых передач в логическую схему из него можно удалить все функциональные блоки, ненужные для исполнения данной программы.

Обычно структура микроконтроллера имеет две четко выраженные части: устройство управления (УУ), которое представляет собой отдельный микропрограммный автомат и операционное устройство, включающее ПЗУ программ (ПЗУП), арифметико-логическое устройство (АЛУ), блок памяти данных (ОЗУД), вычислитель адресов (счетчик) команд и данных (СК), блок управления прерываниями (БП), блок специальных регистров и портов (БСР). То есть управление микроконтроллера сосредоточено в УУ. Стандартный микроконтроллер i8051 с выходами портов P0,…,P3 имеет такую же структуру, которая показана на рис.1.

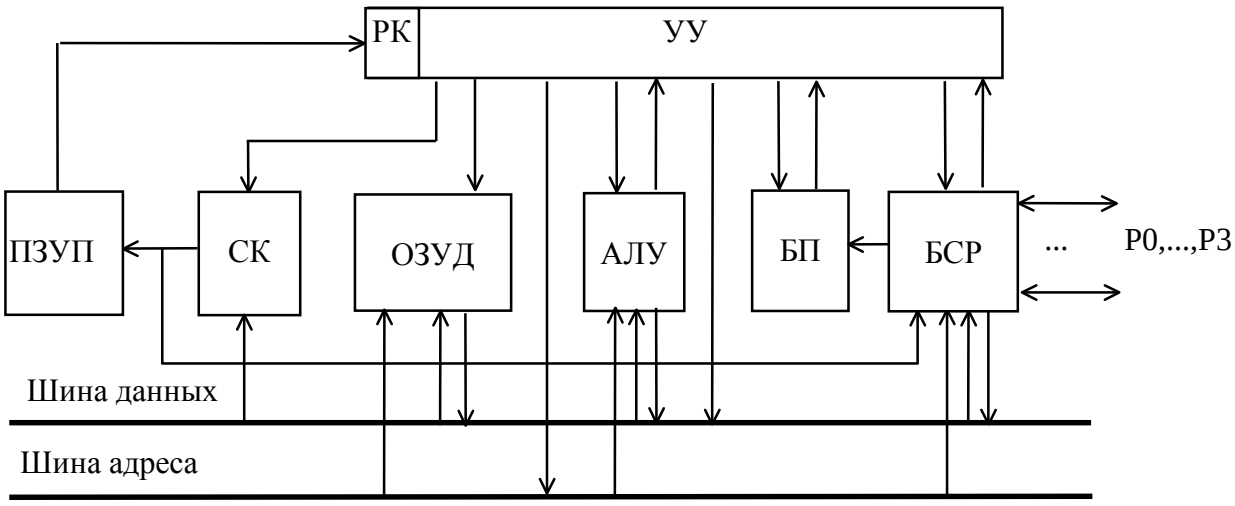

Рис.1. Структура микроконтроллера с сосредоточенным управлением

Управляющий автомат УУ реализован как автомат Мили, или микропрограммируемый автомат. Раньше при разработке микропроцессоров отдавалось предпочтение микропрограммируемому автомату, благодаря простоте его разработки и модификации. Однако при его реализации в СНК, как аппаратурные затраты, так и критическая задержка схем могут оказаться чрезмерно большими.

Современные средства логического синтеза, такие как Synopsys™, значительно упрощают разработку микропрограммного автомата на основе автомата Мили, минимизируя при этом аппаратные затраты. Однако, при локализации управления в одном микропро-

граммном автомате также возникают длинные задержки логических схем и большие аппаратурные затраты, что объясняется следующими особенностями СНК. Во-первых, сложное устройство управления предполагает использование логических уравнений с большим числом аргументов. Такие уравнения чаще всего плохо минимизируются и требуют для своей реализации схему с длинными логическими цепочками. Во-вторых, задержку в линиях связи в ПЛИС и современных СБИС можно оценить как пропорциональную их длине и она может быть большей, чем суммарная задержка в логических схемах. Если всё управление сосредоточить в одном блоке, то его многочисленные входы и выходы необходимо подключать к остальным блокам микропроцессора через длинные маршруты с большими задержками. Кроме того, количество нагрузок на один выход блока оказывается большим, что также увеличивает общую задержку.

Поэтому для минимизации как задержки в критических путях, так и аппаратурных затрат предлагается структура микроконтроллера с распределенным управлением показанная на рис.2.

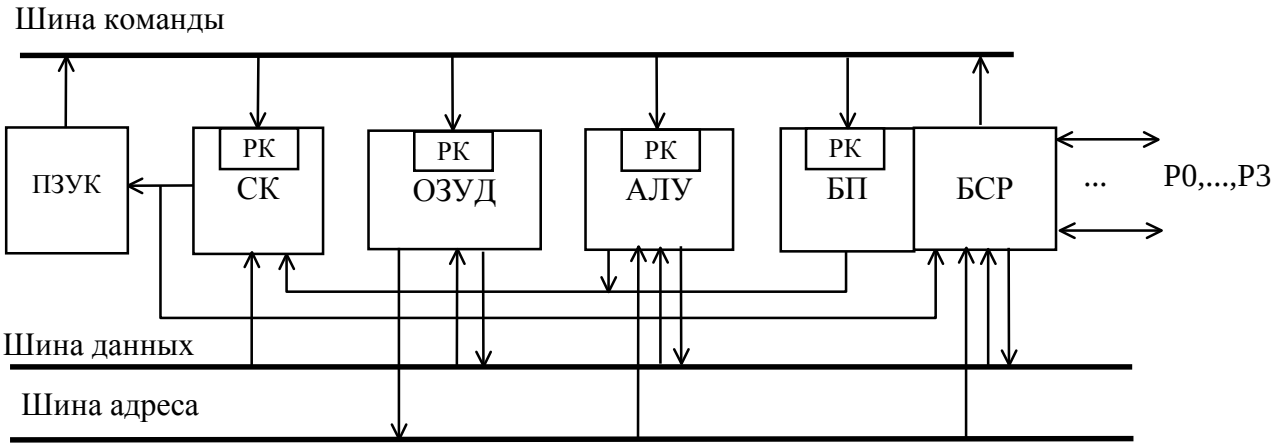

Рис.2. Структура микроконтроллера с распределенным управлением

Четыре операционных блока: СК, ОЗУД, АЛУ и БП имеют локальные регистры команды РК и блоки управления, которые дешифрируют только те команды, и выдают такие управляющие сигналы, которые имеют отношение только к данному блоку. Поэтому каждый из блоков управления имеет в несколько раз меньшие аппаратурные затраты и более высокое быстродействие, чем общее УУ. Благодаря такой структуре, все блоки микропроцессора представляют собой локализованные кластеры, которые связаны между собой малым количеством межсоединений. Такая кластеризованность проекта благоприятствует получению схемы с минимальными критическими задержками при автоматических размещении и разводке СНК.

Разделения ресурсов считается эффективным способом минимизации аппаратурных затрат. Для этого к входам разделяемых ресурсов, таких как АЛУ, счетчики, блоки памяти, подключаются многовходовые мультиплексоры, а к разделяемым общим шинам присоединяют большое число тристабильных буферов.

Если эти разделяемые ресурсы реализовать в ПЛИС, то и аппаратурные затраты, и временные задержки будут существенно высокими. Это объясняется тем, что большие мультиплексоры реализуются в ПЛИС как несколько уровней логических схем. Также многовходовые мультиплексоры предполагают наличие большого числа межсоединений.

Автоматическая разводка таких межсоединений в СНК, даже при сравнительно малом объеме оборудования, может привести к увеличенной площади, занимаемой схемой на кристалле и чрезмерным задержкам в межсоединениях.

С другой стороны, например, в ПЛИС сложность сумматора сравнительно невелика и сравнима со сложностью двухвходового мультиплексора, а его задержка значительно меньше, чем у сложного мультиплексора. Поэтому в заготовке микроконтроллера эффективнее увеличить количество таких ресурсов, как сумматор, счетчик, регистр, чем минимизировать их число за счет добавления большого числа многовходовых мультиплексоров. Например, сложение, умножение, деление, битовые операции и сравнение в АЛУ можно выполнять в отдельных узлах. Структура такого АЛУ показана на рис.3.

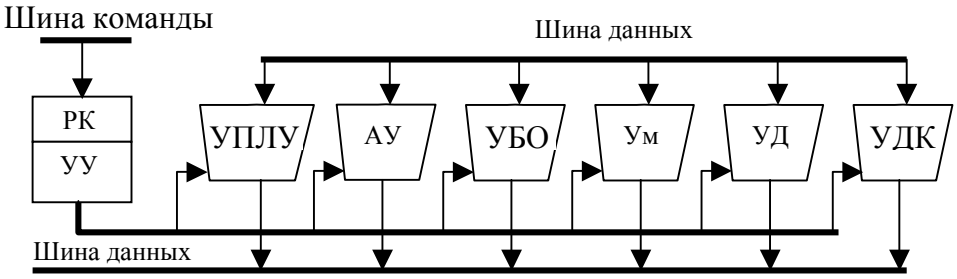

Рис.3. Структура АЛУ

Это АЛУ состоит из устройства проверки логических условий (УПЛУ), арифметического устройства (АУ), устройства битовых операций (УБО), умножителя (Ум), устройства деления (УД), устройства десятичной коррекции (УДК). Каждое из этих устройств может быть исключено из АЛУ, если соответствующие им команды не используются в программе. При этом соответственно упрощается устройство управления (УУ), которое не дешифрирует эти команды. В предельном случае в АЛУ остается только лишь УПЛУ, необходимое для выполнения команд условного перехода.

Таким образом, отказ от разделения ресурсов позволяет экономить аппаратурные затраты за счет исключения из микроконтроллера неиспользуемых узлов и минимизации коммутирующих цепей и межсоединений. Одновременно это позволяет минимизировать критический путь задержки как в логических схемах и буферах, так и в межсоединениях.

Существенный эффект дает также уменьшение количества используемых сигналов запроса прерывания. Это приводит к упрощению БП и существенной минимизации длительности тактового интервала, так как многоуровневая схема анализа приоритета имеет большие аппаратурные затраты и задержку. Часто в приложениях состояние многих периферийных регистров, например, регистр управления прерыванием, регистр таймера, устанавливается при инициализации и в последующей работе не изменяется. Такие регистры можно заменить генераторами констант.

Процесс получения структуры микроконтроллера с минимизированными аппаратурными затратами выглядит следующим образом. Приложение с микроконтроллером разрабатывается по общепринятым методикам. При этом при его программировании желательно минимизировать количество используемых ресурсов микроконтроллера (портов ввода-вывода, регистров спецфункций и т.п.) и типов команд. В последнем случае многие редко исполняемые команды, например, десятичной коррекции, могут быть заменены на эквивалентные цепочки других команд. После того, как программа полностью отлажена, она

анализируется с целью выявления множества команд и множества периферийных устройств, задействованных в вычислениях. Затем модель микроконтроллера, описанная на языке VHDL или Verilog, настраивается на исполнение заданного множества команд и из нее исключаются все неиспользуемые периферийные устройства. Такая модель, будучи скомпилированной программой-синтезатором, например, Synopsys, даст логическую схему микроконтроллера с минимальными аппаратурными затратами.

4.Экспериментальные результаты.

В соответствии с вышеописанным способом была разработана модель заготовки микроконтроллера с архитектурой i8051. Модель микроконтроллера описана на VHDL и предназначена для реализации в ПЛИС серии Xilinx Virtex™ и Spartan II™. Разработана также программа HEXANALYSER, которая анализирует исполнительный код программы микроконтроллера и в результате выдает файл настройки модели (заголовок модели с константами generic и портами).

В таблице приведены параметры аппаратурных затрат в зависимости от исполненной конфигурации микроконтроллера. Там же приведены параметры для реализации микроконотроллера в четырех реальных приложений. Аппаратурные затраты измерены в числе эквивалентных конфигурируемых логических блоков ПЛИС (ЭКЛБ, CLB slices).

Таблица. Аппаратурные затраты микроконтроллера в зависимости от конфигурации

Конфигурация микроконтроллера	Аппаратурные затраты, ЭКЛБ	% от полной конфигурации
Микроконтроллер, исполняющий всю систему команд и имеющий все периферийные устройства	950	100
То же, но без команд MUL,DIV,DA	865	91
То же, но без команд MOVC, сдвига и битовых операций	825	86,8
То же, но без трех параллельных портов	799	84,1
То же, но без одного таймера	758	79,8
То же, но без таймеров и UART	619	65,2
Контроллер для управления автоматом продажи жетонов метро	799	84,1
Контроллер механизма выдачи торгового автомата	744	78,3
Контроллер для управления теплицей	730	76,8
Контроллер для управления блоком питания	588	61,8

Анализ таблицы показывает, что при отказе от использования команд умножения, деления, десятичной коррекции, что типично для многих приложений преимущественно логического контроля, аппаратурные затраты уменьшаются на 9%. Если не используются таймеры и порт последовательного обмена, то затраты уменьшаются на 19%. Каждый из параллельных портов имеет аппаратурные затраты около 1% от общих затрат. При глубокой минимизации аппаратурные затраты можно уменьшить до 1,6 раза. Кроме того, разработанная заготовка микроконтроллера имеет в 1,35 раза меньше аппаратурные затраты, чем аналогичная заготовка, описанная в [7].

Разработанная модель микроконтроллера также имеет минимизированный цикл выполнения команд. Каждая команда выполняется только за 2, 3 или 4 такта, что в среднем в

6,2 раза меньше, чем в оригинальном микроконтроллере. Также в ней задержка отработки прерывания в тактах приблизительно в 8 раз меньше. Тактовая частота микроконтроллера при его реализации в ПЛИС Xilinx XCV300-06 достигает 60 МГц, ее значение в 1,7 раза больше, чем у микроконтроллера, описанного в [7].

В реальных приложениях, в которых необходимо обслуживать десятки и сотни параллельных разрядов ввода-вывода нередко применяют несколько микроконтроллеров параллельно, только потому, что не достаточно выводов портов кристалла микроконтроллера. В этом случае целесообразно использовать ПЛИС с предлагаемой моделью микроконтроллера, которая обеспечит функциональную замену нескольких микроконтроллеров. Такая замена поддерживается повышенным быстродействием микроконтроллера, большим числом выводов ПЛИС и тем, что дополнительные аппаратурные затраты на параллельные порты микроконтроллера сравнительно невелики. При этом соответственно, получается многократное уменьшение аппаратурных затрат.

5. Заключение.

Микроконтроллеры представляют важный класс вычислительных заготовок. Предложеный способ минимизации аппаратурных затрат микроконтроллера, основанный на настройке его структуры под исполняемую программу позволяет более чем в 1,6 раза уменьшить аппаратурные затраты. Этот способ также обеспечивает уменьшение минимальной длительности тактового интервала за счет минимизации длины критического пути в логических схемах. Высокая эффективность предложенного способа проверена при разработке микроконтроллера с архитектурой i8051, который был реализован в ПЛИС Xilinx Virtex.

ЛИТЕРАТУРА

[1]. Keating M., Bricaud P. *Reuse methodology manual² for System-On-a-Chip designs,* Boston, Dordrecht, London, -Kluwer, –1999, -286p.

[2]. De Micheli G., Gupta R.K. *Hardware/ Software Co-Design,* Proc. of the IEEE, –1997, –V.85, - №3, –p.349-365.

[3]. Палагин А.В., Иванов В.А., Кургаев А.Ф., Денисенко В.П. Мини-ЭВМ. Принципы построения и проектирования. Под ред. Б.Н.Малиновского, –Киев, Наукова думка, -1975, -175с.

[4]. Huang I.-J., Despain A.M. *Synthesis of Application Specific Instruction Sets,* IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, –V14, -№6, –1995,- p.663-676.

[5]. Bhattacharya S.S., Leupers R. Marwedel P. *Software Synthesis and Code Generation for Signal Processing Systems,* IEEE Trans. on Circuits and Systems, Part II, Analog and Digital Signal Processing, –2000, –V47, -№9, –p.849-875.

[6]. Liao S.Y., Devadas S., Keuzer K. *Code Density Optimization for Embedded DSP Processors Using Data Compression Techniques,* IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, –1998, –V17, -№7, –p. 601-608.

[7] Dolphin Provies Industry Fastest 8051 Core for Xilinx Virtex FPGAs, *Available at* http//www.dolphin.fr/.