УДК 004.315.5

Опанасенко В.Н., д-р. техн. наук
Лисовый А.Н.
Сорока Е.В.

РЕАЛИЗАЦИЯ УСКОРЕННЫХ АЛГОРИТМОВ ЦЕЛОЧИСЛЕННОГО ДЕЛЕНИЯ НА ПЛИС

Институт кибернетики им. В.М. Глушкова НАН Украины

Предложены структурные реализации модулей деления в элементном базисе ПЛИС типа FPGA, выполненные путем поведенческого описания алгоритмов на языке VHDL. Реализована проверка функционирования модулей деления методом моделирования в системе ModelSim Xilinx Edition - MXE III с помощью проверочного стенда

Введение

Проектирование и реализация цифровых устройств на современной элементной базе может быть выполнено на базе ПЛИС типа *FPGA*. Использование *HDL*-технологии, которая представляет собой комплекс инструментальных средств САПР и методологию проектирования, ориентированных на описание проекта с помощью языков *VHDL* и *Verilog* [1], позволяет в требуемые сроки разработать и реализовать устройство, используя минимум оборудования.

Наиболее важным свойством готового технического решения (*IP-core*) является его гарантированное воспроизведение в новом проекте в соответствии со спецификацией, определенной разработчиком этого решения и уточненной разработчиком проекта. Следует отметить, что описание модели с помощью *HDL*-технологии (*Hardware Description Language* – язык описания аппаратных средств) позволяет не только сделать ее перенастраиваемой и независимой от технологии, но и выполнять ее моделирование и синтез с использованием инструментальных средств различных фирм.

Постановка задачи

Операция деления, являясь наиболее сложной из списка базовых арифметических операций, может быть реализована множеством алгоритмов [2]. Актуальными остаются вопросы оптимизации алгоритмов, использования специальных алгоритмов ускоренного целочисленного деления, а также их аппаратной реализации.

Операция деления имеет следующие пути оптимизации по быстродействию:

- за счет замены делителя обратной величиной с последующем умножением ее на делимое;
- за счет сокращение времени выполнения операции сложения/вычитания;
- за счет уменьшения количества операций сложения/вычитания, при расчете частного;
- вычисление частного в избыточной системе исчисления.

Одним из способов организации устройства может быть использование конструкции ***Case*** (машины состояний). Принцип конструкции состоит в том, что в каждом состоянии выполняются заданные операции и определяется переход к следующему состоянию. Операции в каждом состоянии выполняются по новому синхросигналу. Данный подход используется в том случае, когда требуется уменьшить длительность синхросигнала, что приводит к увеличению аппаратных затрат. В конструкции ***Case*** легко организовать любые циклы, в том числе с переменным количеством итераций, что не всегда возможно при использовании конструкции ***For***, которая позволяет использовать ограниченное количество итераций [3].

На основе указанных свойств рассмотрим реализации модулей деления, поведенческое описание которых основано на конструкции ***Case***.

Реализация модулей деления

Рассмотрим реализацию модулей деления в элементном базисе ПЛИС типа *FPGA* фирмы *Xilinx*.

Модуль деления содержит четыре входа: ***a(32:0)*** – делимое; ***б(16:0)*** – делитель; ***Load*** – сигнал загрузки операндов (состояние «1» соответствует операции загрузки, «0» – простой устройства); ***Clk*** – последовательность синхроимпульсов, и два выхода: ***c(16:0)*** – частное; ***RG_FLAG(7:0)*** – регистр флагов.

Для входных и выходных операндов старший бит – знак операнда, операнды представлены в дополнительном коде. Регистр флагов представлен на рис 1.

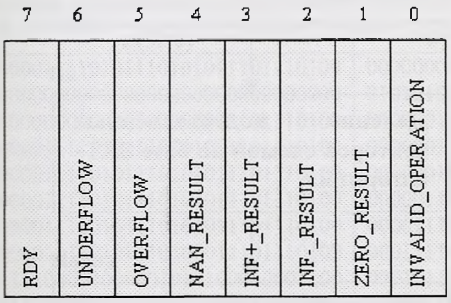

Рис. 1. Регистр флагов (8 бит)

Описание битов регистра флагов:

– ***(7)*** устанавливается в единицу, когда определен результат,

– ***(6)*** устанавливается в единицу, когда в ходе выполнения операции происходит потеря значимости,

– ***(5)*** устанавливается в единицу, когда в ходе выполнения операции происходит переполнение разрядной сетки,

– ***(4)*** устанавливается в единицу, когда в ходе выполнения операции результат *NaN число,* (не используется),

– ***(3)*** устанавливается в единицу, когда в ходе выполнения операции или во время анализа входных операндов определен результат *Infinity+,* (не используется),

– ***(2)*** устанавливается в единицу, когда в ходе выполнения операции или во время анализа входных операндов определен результат *Infinity–,* (не используется),

– ***(1)*** устанавливается в единицу, когда в ходе выполнения операции или во время анализа входных операндов определен результат *машинный нуль,*

– ***(0)*** устанавливается в единицу, когда операция над исходными операндами не может быть выполнена, например, a/(±0).

Используемые алгоритмы

Деление с неподвижным делимым и сдвигаемым вправо делителем [4]. Операция вычитания применяется для 32 бит делимого (для модуля делимого) и только тогда, когда делимое больше делителя, в этом же случае в бит частного записывается «1», в противном случае «0». Первоначально делитель формируется как ***'0'&б&"000000000000000"*** (где ***б*** модуль делителя). Сдвиг делителя происходит на каждой итерации.

Деление с неподвижным делителем и восстановлением остатка. Делимое сдвигается влево на каждой итерации. Вычитание делителя из делимого происходит в том случае, когда старшая часть делимого ***a(31 downto 16)*** больше делителя и результат записывается в старшую часть делителя. Главное отличие от первого алгоритма заключается в том, что операция вычитания производится только над 16 битами, и сдвиг происходит над делимым влево.

Деление с неподвижным делителем без восстановления остатка. Идея алгоритма заключается в том, что над старшей частью делимого с знаком (17 бит) и делителем с знаком может выполнятся как сложение так и вычитание. Операция определяется в соответствии с знаком результата получившимся в предыдущей операции. В случае сложения в бит частного записывается «1», вычитания – «0».

SRT–алгоритм представляет собой модификацию алгоритма деления без восстановления остатка. Отличие состоит в отсутствии необходимости суммирования или вычитания на каждой итерации в зависимости от получившегося значения в регистре делимого. Частное представляется двумя регистрами и формируется от старшего бита к младшему поразрядно: «0» – соответствует комбинации «00», когда три ставших бита в регистре делимого совпадают; «1» – соответствует комбинации «10», когда три старших бита в регистре делимого не совпадают и делимое положительное; «–1» - соответствует «01», когда три старших бита в реги-

стре делимого не совпадают и делимое отрицательное. Частное формируется путем вычитания из регистра 1 значения регистра 2, и коррекции результата.

Предлагаемая реализация модуля деления чисел с фиксированной точкой, выполнена средствами поведенческого описания *SRT*–алгоритма на языке *VHDL*. Архитектура модуля представлена машиной состояний, имеющей 4 состояния. В состоянии ***"idle"*** выполняется анализ входного операнда, и формирование дополнительных векторов, в ***"norm"*** – поиск старшей единицы делителя и "нормализация" делимого и делителя, в ***"srt"*** – непосредственно операция деления, в ***"corekt"*** – коррекция частного, остатка и формирование выходного результата. Алгоритм работы представлен на рис. 2.

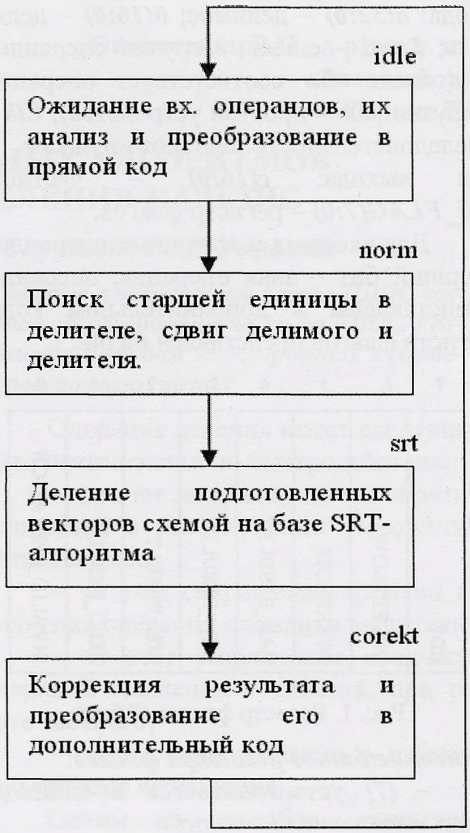

Рис. 2. Алгоритм деления на базе *SRT*–алгоритма

Таблица 1. Оценки временных и аппаратных параметров модулей деления

Оценка аппаратных ресурсов и быстродействия	Слайсы (*Slices*)	Синхросигналы *Clk*, период (нс)
Деления с неподвижным делимым и сдвигаемым вправо делителем	251 (7%)	11,5
Деление с неподвижным делителем и восстановлением остатка	222 (6%)	9,8
Деление с неподвижным делителем без восстановления остатка	221 (6%)	10,8
Деление на базе *SRT*-алгоритма	410 (11%)	12,6

В табл. 1 приведены данные об аппаратных ресурсах, максимальном быстродействии разработанных модулей относительно используемого алгоритма.

Верификация модулей деления

Проверка функционирования модулей осуществлена методом моделирования в системе *ModelSim Xilinx Edition - MXE* III.

На рис. 3 представлена схема стенда, выполненная средствами схематического редактора *Engineering Capture System (ECS)*, входящего в состав системы *Xilinx ISE Foundation*. В табл. 2 представлены результаты верификации.

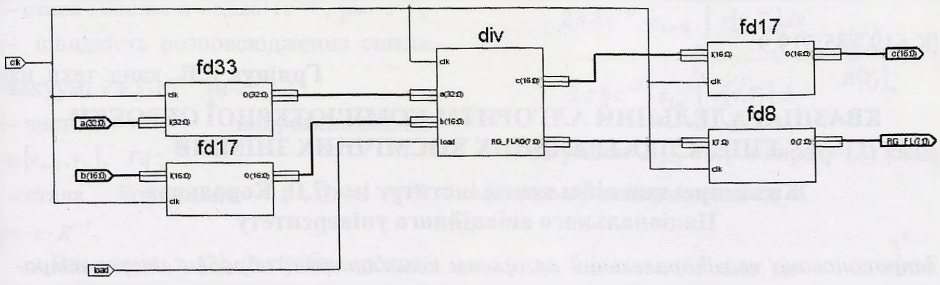

Рис. 3. Стенд для проверки модулей деления

Таблица 2. Исходные и результирующие данные системы моделирования

a (32:0)	б (16:0)	c (16:0)	RG FLAG(7:0)
00000011110101110101011101110100	00000000000000000	00000000000000000	10000001
00000000000000000000000000000000	01010101010010110	00000000000000000	10000010
00000000000000000000000001101110100	00000010011101000	00000000000000000	11000000
00000011110101110101011101110100	00000010011101000	00000000000000000	10100000
00000011110101110101011101110100	00000011110100000	01000000011110110	10000000
00000011110101110101011101110100	00000111101000000	00100000001111011	10000000
00000011110101110101011101110100	00001111010000000	00010000000111101	10000000
00000011110101110101011101110100	00011110100000000	00001000000011110	10000000
11111100001010001010100010001100	00011110100000000	11110111111100010	10000000
11111100001010001010100010001100	00001111010000000	11101111111000011	10000000
11111100001010001010100010001100	00000111101000000	11011111110000101	10000000
11111100001010001010100010001100	00000011110100000	10111111100001010	10000000

Выводы

Модули разработаны с помощью инструментальных средств *Xilinx Foundation ISE 9.2*, протестированы путем моделирования в системе *ModelSim Xilinx Edition–III* с помощью стенда для проверки, что подтверждает правильность функционирования. Проект реализован на ПЛИС *Spartan 3 XC3X400*.

Полученные результаты будут использованы при разработке сложных вычислительных устройств с использованием типовых технических решений (в виде библиотечных элементов *IP-Core*) в качестве составных компонентов для реализации таких устройств на одном кристалле. Такие типовые решения в совокупности представляют собой «открытую» библиотеку файлов конфигураций, входящих в состав реконфигурируемой вычислительной системы с виртуальной архитектурой.

Список литературы

1. *Семенец В.В., Хаханова И.В., Хаханов В.И.* Проектирование цифровых систем с использованием языка VHDL. – Харьков: ХНУРЕ, 2003. – 492 с.

2. *Карцев М.А.* Арифметика цифровых машин. – М: Наука, 1969. – 576 с.

3. VHDL'93. IEEE Standard VHDL Language Reference Manual // IEEE Std 1076 – 1993.

4. *Опанасенко В.Н., Сахарин В.Г., Лисовый А.Н.* Проектирование модулей с плавающей точкой на ПЛИС с использованием языка VHDL. – К: Математические машины и системы. – №3. 2005. – 195 с.