УДК 004.3

О. И. Буренева, А. А. Пустовойтова, Д. М. Соболев
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

Аппаратная реализация арифметических операций в автомобильном радаре

Представлены возможные варианты реализации арифметических операций в базисе аппаратных ресурсов FPGA, ориентированные на применение в составе цифрового вычислителя автомобильного радара в диапазоне 76–77 ГГц. Рассмотрены операции: умножения и деления вещественных чисел, умножения комплексных чисел, извлечения корня квадратного. Выполнен сравнительный анализ результатов синтеза вычислителей, построенных в базисе библиотечных элементов и IP-ядер, предоставляемых производителем ПЛИС – компанией «Intel FPGA», а также предложены альтернативные реализации в виде RTL-описаний, синтезированных на основе разработанных программ на языке Verilog_HDL. Синтез устройств был ориентирован на применение микросхем семейства Cyclone 10 LP и выполнялся в системе проектирования Quartus Prime. С использованием рассмотренных решений в области арифметических вычислителей предложены варианты реализации амплитудного детектора – блока первичной обработки сигналов автомобильного радара, реализующего объединение данных двух квадратурных каналов.

Первичная обработка данных, аппаратная реализация, программируемые логические интегральные схемы, арифметические операции, встроенный умножитель, комплексный умножитель, извлечение квадратного корня

Автомобильный радар является одним из основных элементов комплексной системы обеспечения безопасности автомобильного транспорта и должен надежно обнаруживать, сопровождать и точно распознавать определенные цели с известными радиолокационными характеристиками.

На рынке радаров представлены изделия разных производителей: радары немецкой компании «Continental», узкополосные радары немецкой компании «Hella», радары дальнего действия компании «Bosch» [1]. Ведутся и отечественные разработки, например, компаниями «Cognitive Technologies» (Москва); российским отделением «Keysight Technologies» совместно с НПП «ИТЭЛМА». Среди отечественных разработчиков следует отметить АО «ПКК Миландр», выполняющее разработку «автомобильного радара», интегрируемого в интеллектуальную систему помощи водителю ADAS (Advanced Driver Assistance System) [2].

Принцип действия радара основан на считывании отраженных радиоволн с последующим выявлением объекта в зоне ответственности и определением расстояния до него, оценкой его параметров, скорости и направления движения.

В автомобильной радиолокации в основном применяются радары диапазона 24 ГГц, ограниченные возможности которых по идентификации объектов сдерживают их применение. Значительно улучшить качество распознавания позволяет переход систем обнаружения на работу на более высоких частотах. В частности, на частоте 76 ГГц увеличивается дальность обнаружения объектов, а в диапазоне 77–81 ГГц удается значительно повысить разрешение при детектировании объектов на малой дальности [3].

В состав автомобильного радара входят следующие модули:

– модуль фазированной антенной решетки с цифровым управлением, реализующий обзор пространства в заданной зоне обнаружения;

– модуль приемопередатчиков, обеспечивающий формирование модулированных высокочастотных сигналов, поступающих на модуль антенной решетки, прием отраженных сигналов и их предварительную обработку;

– модуль цифрового вычислителя, предназначенный для реализации алгоритмов обработки информации, обеспечивающих оценку дальности и скорости обнаруженных объектов, а также траекторное сопровождение целей.

К задачам цифрового вычислителя относятся первичная обработка, предполагающая предварительную фильтрацию и подготовку данных, полученных от приемника, и вторичная обработка, реализующая алгоритмы принятия решений на основе логического анализа результатов первичной обработки.

Анализ вариантов реализации цифрового вычислителя в радарах с диапазоном 76–77 ГГц показал, что оптимальной элементной базой для первичной обработки служат программируемые логические интегральные схемы класса FPGA (Field-Programmable Gate Array), а для вторичной – процессоры цифровой обработки сигналов (ЦОС). Применение FPGA позволяет достичь максимального уровня параллелизма и высокой производительности, а использование процессора ЦОС обеспечивает эффективную реализацию алгоритмов обработки информации.

Постановка задачи. Тракт первичной обработки данных радара Обзор-77 (АО «ПКК Миландр») состоит из следующих модулей:

– режекторного фильтра (РФ), выполняющего фильтрацию цифровых отсчетов, поступающих на вход тракта первичной обработки сигнала;

– блока подавления несинхронной импульсной помехи (НИП), который сглаживает импульсную помеху в потоке отсчетов;

– весового окна (ВО), реализующего наложение оконных функций на поток отсчетов с целью управления эффектами растекания спектра;

– блока быстрого преобразования Фурье (БПФ), выполняющего преобразование сигнала из временной последовательности импульсов в спектральное (частотное) представление сигнала и реализованного в двух вариантах для двумерного преобразования;

– блока фазовой коррекции (ФК), который выполняет выравнивание фаз между приемными каналами;

– блока амплитудно-фазовой коррекции (АФК), обеспечивающего компенсацию амплитудных и фазовых погрешностей в каналах антенной решетки, вызванных особенностями аналоговых приемных трактов;

– блока формирования лучей (ФЛ), формирующего произвольное число лучей в различных направлениях;

– амплитудного детектора (АД), который объединяет данные двух квадратурных каналов;

– адаптивного порогового устройства (АПУ), реализующего алгоритм рангового вычисления порога и выполняющего сравнение с полученным порогом текущих отсчетов.

Временные характеристики и аппаратные затраты тракта первичной обработки во многом определяются эффективностью выполнения следующих арифметических операций:

– умножение вещественных чисел (в блоках РФ, НИП, АД, ВО),

– деление вещественных чисел (в блоке НИП),

– умножение комплексных чисел (в блоках ФК, АФК, ВО, ФЛ, БПФ),

– извлечение квадратного корня (в блоках НИП, АД).

Актуальной становится задача проведения анализа возможных реализаций перечисленных операций с последующим выбором оптимального решения с точки зрения сочетания частоты работы устройства и аппаратных затрат.

Для реализации канала первичной обработки цифрового вычислителя выбрана микросхема FPGA компании «Intel FPGA» семейства Cyclone 10 LP. Микросхемы Cyclone 10 LP имеют классическую архитектуру устройств семейства MAX 10, а к их отличительным особенностям относятся логические элементы с четырьмя входами, встроенные умножители 18 × 18 и модули блочной памяти M9K [4].

Варианты модулей тракта первичной обработки разработаны с использованием языков Verilog_HDL и VHDL. В качестве среды отладки применялась система ModelSim Altera, а для проверки синтезируемости подготовленных описаний и анализа характеристик – среда проектирования Quartus Prime 18.0 компании «Intel FPGA».

Реализация умножения и деления вещественных чисел. Операция умножения в аппаратном базисе FPGA может быть реализована различными способами [5], [6]. Это может быть имплементация умножителя в варианте soft multiplier, когда для формирования частичных результатов умножения используются LUT (Look-Up Table) логических ячеек программируемых логических интегральных микросхем (ПЛИС), также могут применяться встроенные умножители или элементы блоков ЦОС.

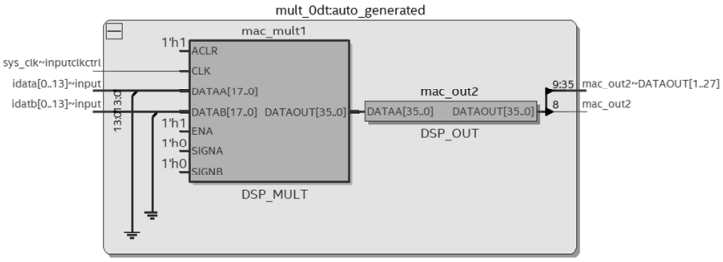
При наличии в архитектуре ПЛИС встроенных умножителей естественна реализация операции умножения на их базе. При этом результат синтеза не зависит от оформления операции: в программе может использоваться как оператор умножения (*) (рис. 1, *а*), так и оператор включения соответствующего аппаратного вычислителя

```
always @( posedge sys_clk )
begin
  idata_r <= idata;
  idatb_r <= idatb;
end

assign odat1=idata_r * idatb_r;
```

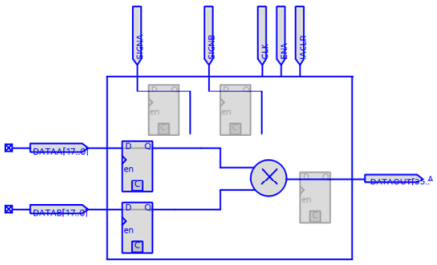
а

```
always @( posedge sys_clk )
begin
  idata_r <= idata;
  idatb_r <= idatb;
end

M_1 Mult_inst
(
   .dataa (idata_r),
   .datab (idatb_r),
   .result (odat2)
   );
```

б



в



г

Рис. 1

(рис. 1, *б*). В результате синтезируется схема, ориентированная на подключение умножителя, реализованного во встроенном ЦОС-блоке FPGA (рис. 1, *в*). Если используются регистры-защелки для фиксации аргументов операции, как реализовано в рассматриваемом примере с помощью оператора always, то в качестве таких элементов задействуются регистры блока ЦОС (рис. 1, *г*).

Согласно информации, представленной в файле отчета, для имплементации представленного 14-разрядного вычислителя задействованы только 2 встроенных 9-битных умножителя, остальные ресурсы FPGA не используются.

Операция деления также может быть реализована различными способами [7], [8]. При этом результат синтеза определяется оформлением операции деления в программе Verilog_HDL. Анализ результатов синтеза показал, что использование оператора деления (/) приводит к синтезу схемы, которая соответствует алгоритму, построенному на операциях вычитания (сложения) и сдвига. Алгоритм основан на представлении деления как последовательности вычитаний делителя на первом шаге из делимого, а на последующих шагах – из образующихся в процессе вычисления частичных остатков. В итоге определяется частное y и остаток z от деления x на d такие, что

$$y = \text{int}\frac{x}{d};\ z = x - yd,$$

при условии $z < d$. Подключение библиотечного элемента lpm_divide, представляющего собой параметризируемую мегафункцию, оптимизированную для имплементации в ПЛИС «Intel FPGA», позволяет получить более эффективное с точки

Таблица 1

Аппаратный ресурс	Варианты реализации	
	Мегафункция lpm_divide	Оператор деления
Общее количество логических элементов	155/119.088 (<1 %)	181/119.088 (<1 %)
– комбинационные элементы без регистров	141	160
– регистры	0	7
– комбинационные элементы регистрами	14	14
Использование LUT логических ячеек – 4-входовые – 3-входовые – <=2-входовые – только регистры	 12 113 30 0	 19 135 20 7
Логические элементы – в нормальном режиме – в арифметическом режиме	77 78	93 81
Общее количество регистров	14/120.403 (<1 %)	21 / 120.403 (<1 %)
– выделенные регистры	14/119.088 (<1 %)	21 / 119.088 (<1 %)
Общее количество логических блоков (частично или полностью использованных)	10/7.443 (<1 %)	16 / 7.443 (< 1 %)
Среднее использование связей	0.1 %/0.1 %/0.1 %	0.1 %/0.1%/0.1%
Пиковое использование связей	1.5 %/1.4 %/1.7 %	1.9 %/1.9%/2.0%

зрения аппаратных затрат устройство. Сравнение требуемых аппаратных ресурсов для обоих вариантов реализации приведено в табл. 1.

Очевидно, что при необходимости минимизации аппаратных затрат использование мегафункции предпочтительно, поскольку по большинству параметров превосходит реализацию с использованием оператора деления.

Реализация умножения комплексных чисел. Традиционно реализация комплексного умножения выполняется в соответствии со следующими формулами:

$$\left(a_0 + jb_0\right)\left(a_1 + jb_1\right) = a_0a_1 + jb_0a_1 + \\ + jb_1a_0 - b_0b_1 = \left(a_0a_1 - b_0b_1\right) + j\left(b_0a_1 + b_1a_0\right). \quad (1)$$

Для вычисления результата комплексного умножения можно воспользоваться соответствующим ядром ALTMULT_COMPLEX компании «Intel FPGA». Ядро может быть настроено в двух режимах. Режим классических преобразований (Conventional Representation) обеспечивает синтез схемы, реализующей (1) без модификаций, что предполагает минимизацию затрат в комбинационных ячейках FPGA.

Анализ формулы показывает, что для ее аппаратной реализации необходимо использовать 4 умножителя для вычисления промежуточных произведений и 2 сумматора для формирования вещественной и мнимой частей результата.

Канонический режим работы комплексного умножителя (Canonical Representation) обеспечивает синтез устройства с минимизацией количества умножителей, что достигается выполнением предварительных вычислений. Вывод формулы для вещественной части результата основан на следующих преобразованиях:

$$a_0a_1 - b_0b_1 = \left(a_0a_1 - b_0b_1\right) + \\ + \left(a_0b_1 - b_0a_1\right) - \left(a_0b_1 - b_0a_1\right) = \\ = \left(a_0a_1 - b_0b_1 - a_0b_1 + b_0a_1\right) + \left(a_0b_1 - b_0a_1\right) = \\ = \left(a_0 + b_0\right)\left(a_1 - b_1\right) + \left(a_0b_1 - b_0a_1\right). \quad (2)$$

Мнимая часть результата рассчитывается в соответствии с (1):

$$b_0a_1 + b_1a_0 .$$

Такое представление позволяет использовать два умножителя для формирования мнимой части результата и для формирования второго слагаемого в выражении для вещественной части одновременно. Третий умножитель потребуется для вычисления первого слагаемого выражения (2). Дополнительно потребуется применить 5 сумматоров, для синтеза которых будут задействованы ресурсы логических ячеек, настроенных на работу в арифметическом режиме.

Возможны и иные варианты, основанные на использовании предварительного разложения, как предложено, например, в [9]. Дополнительное преобразование выполняется по следующим вспомогательным переменным:

$$k_1 = a_0\left(a_1 + b_1\right) = a_0a_1 + a_0b_1; \\ k_2 = b_1\left(a_0 + b_0\right) = a_0b_1 + b_1b_0; \\ k_3 = a_1\left(b_0 - a_0\right) = a_1b_0 - a_0a_1.$$

Вещественная (Re) и мнимая (Im) части результата рассчитываются следующим образом:

$$\mathrm{Re} = k_1 - k_2; \\ \mathrm{Im} = k_1 + k_3.$$

Для имплементации этой функции, как и в предыдущем варианте, необходимо использовать 3 умножителя и 5 сумматоров, что с точки зрения задействования аппаратных умножителей экономично. RTL-представление, полученное в результате синтеза VHDL-описания комплексного умножителя, приведено на рис. 2. В разработанной программе использовано поведенческое описание устройства, необходимые аппаратные элементы FPGA в явном виде не указывались.

Все рассмотренные варианты были синтезированы, при этом назначались параметры, которые требуются при реализации блоков ФК и АФК, выполняющих умножение знаковых отсчетов на знаковые весовые коэффициенты: разрядность обрабатываемых отсчетов – 14 бит, разрядность коэффициентов – 17 бит, разрядность результата – 20 бит. Выбрана реализация без организации конвейера, латентность блоков – 0 тактов. Конвейер может быть организован при необходимости поднятия частоты работы устройства. Функциональность блока при этом останется без изменений, однако результат будет формироваться с латентностью, обусловленной количеством ступеней конвейера.

Характеристики синтезированных устройств приведены в табл. 2.

Анализ приведенных результатов показывает, что по некоторым параметрам описанный на языке VHDL модуль превосходит библиотечные элементы, что, вероятно, объясняется универсальностью библиотечных модулей, допускающих широкую вариативность настроек.

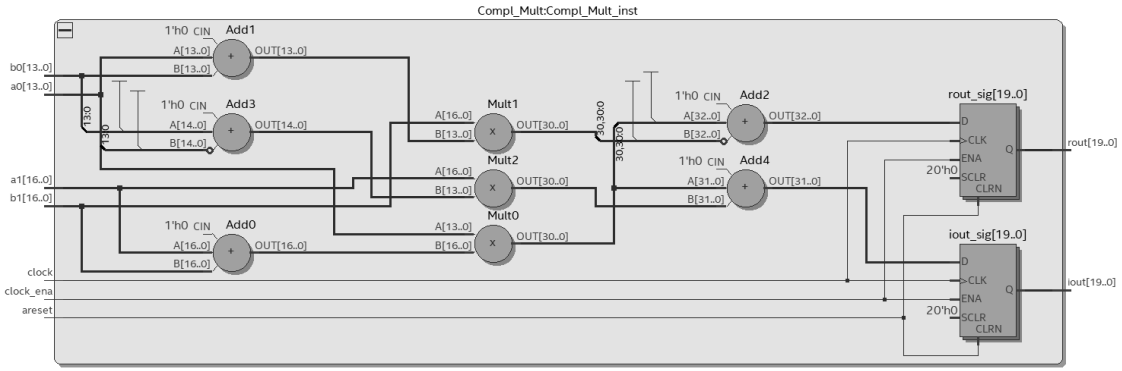


Рис. 2

Таблица 2

Аппаратный ресурс	Варианты реализации		
	Ядро в режиме Canonical	Ядро в режиме Conventional	VHDL-программа
Количество логических элементов	155	102	51
Количество комбинационных функций	93	40	51
Использование LUT логических ячеек			
– 4-входовые	0	0	0
– 3-входовые	88	38	48
– <=2-входовые	5	2	3
Логические элементы			
– в нормальном режиме	5	2	3
– в арифметическом режиме	88	38	48
Общее количество регистров	62	62	20
– выделенные регистры	62	62	20
Контакты ввода/вывода	125	125	125
Встроенные 9-битные умножители	6	8	4

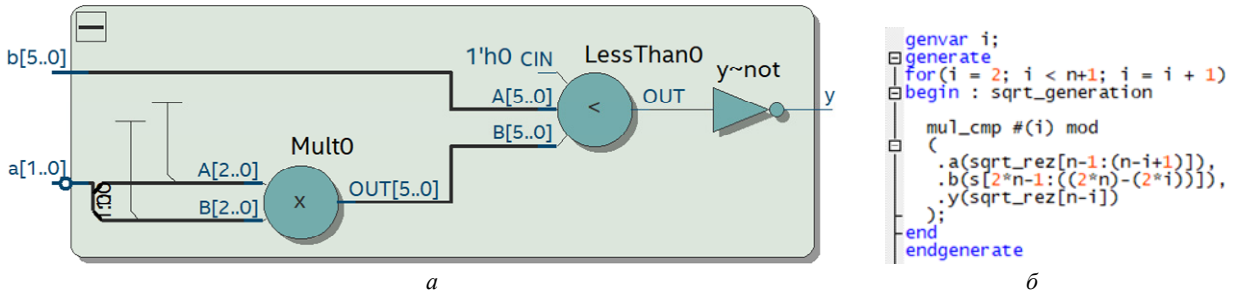
Реализация извлечения корня квадратного. Для вычисления корня квадратного существуют специальные алгоритмы, также возможно использование соответствующих IP- (intellectual property) ядер.

В качестве примера применения IP-ядра рассмотрим вариант включения в проект IP-ядра ALTSQRT, которое реализуется с помощью логических элементов и модулей адаптивной логики. Вычисление может быть конвейеризовано: обработка каждой пары битов входного кода будет выполняться на отдельном оборудовании. Это позволит поднять частоту работы устройства, однако увеличит латентность, что приведет к увеличению времени вычисления функции.

Альтернативный вариант построен на базе алгоритма извлечения корня квадратного из двоичного числа «в столбик», реализованного на языке Verilog_HDL. В соответствии с этим алгоритмом выполняется попарное сравнение квадрата промежуточного результата и аргумента функции. В зависимости от результата сравнения определяется текущий бит выходного кода. В основе такого сравнения лежит модуль mul_cmp, RTL-вид которого приведен на рис. 3, *а*. Для получения нужного количества таких элементов можно воспользоваться оператором генерации языка Verilog_HDL, формат которого приведен в листинге на рис. 3, *б*. В рассмотренном программном фрагменте параметр *n* определяет разрядность результата sqrt_rez, разрядность аргумента функции s, соответственно, равна 2*n*.

При выполнении синтеза устанавливались следующие значения параметров вычислителя: разрядность обрабатываемых отсчетов – 28 бит, разрядность результата – 14 бит, конвейеризация не применялась. Характеристики синтезированных устройств приведены в табл. 3.

При синтезе языкового описания были использованы встроенные умножители, количество которых определяется разрядностью результата. Также следует отметить преобладание логических ячеек, настроенных на работу в арифметическом режиме. Эта особенность обусловлена тем, что арифметический режим обеспечивает синтез сумматоров и цепей сравнения (компараторов). Специальные цепи переноса логических ячеек в этом режиме задействованы при арифметических преобразованиях многоразрядных аргументов.



а *б*

Рис. 3

Таблица 3

Аппаратный ресурс	Варианты реализации	
	Ядро ALTSQRT	Verilog_HDL-программа
Количество логических элементов	247	238
Количество комбинационных функций	247	238
Использование LUT логических ячеек		
– 4-входовые	12	24
– 3-входовые	170	179
– <=2-входовые	65	35
Логические элементы		
– в нормальном режиме	117	60
– в арифметическом режиме	130	178
Общее количество регистров	0	0
– выделенные регистры	0	0
Контакты ввода/вывода	42	42
Встроенные 9-битные умножители	0	15

Синтез библиотечного ядра оказался более экономичным с точки зрения встроенных умножителей. Для обеспечения необходимой вычислительной мощности задействованы логические ячейки, которые, будучи настроены в нормальном режиме, реализуют комбинационные функции.

Реализация амплитудного детектора. Модуль квадратурного сигнала $S(n)$ вычисляется модулем амплитудного детектора посредством извлечения корня квадратного из суммы квадратов вещественной $x_{\text{re}}(n)$ и мнимой $x_{\text{im}}(n)$ частей отсчетов в соответствии с формулой

$$S(n) = \sqrt{x_{\text{re}}(n)^2 + x_{\text{im}}(n)^2}.$$

С точки затрат на разработку простой путь создания амплитудного детектора связан с использованием алгоритма цифрового вычисления поворота системы координат (CORDIC) [10] и соответствующего типового IP-ядра [11], поставляемого в составе системы автоматизированного проектирования Quartus Prime.

Это ядро может настраиваться на работу в четырех режимах, для работы блока АД актуален режим преобразования координат из декартовых в полярные (режим Vector Translate). В результате работы ядра в данном режиме может быть рассчитан модуль и угол вектора на комплексной плоскости. Для реализации амплитудного детектора интерес представляет модуль вектора.

Для корректной работы IP-ядра возникла необходимость введения в схему сигнала rst, выполняющего асинхронный сброс схемы. При настройке параметров установлена разрядность входных сигналов 14 бит, для представления результата предусмотрены 15 бит. При этом в ядре формируется не представляющее интереса в разрабатываемом амплитудном детекторе значение угла вектора разрядностью 15 бит. Затраты на реализацию проекта амплитудного детектора приведены в табл. 4.

Частота работы схемы для худших случаев составляет:

– модель Slow 85 °C (скорость, обусловленная кремнием, минимальна; низкое напряжение; температура 85 °C): 182.68 МГц;

– модель Slow 0 °C (скорость, обусловленная кремнием, минимальна; низкое напряжение; температура 0 °C): 199.8 МГц.

Также амплитудный детектор может быть построен с использованием рассмотренных ранее вариантов реализации операции умножения и извлечения квадратного корня, для чего потребуются два умножителя, сумматор и устройство для извлечения квадратного корня, соединенные, как

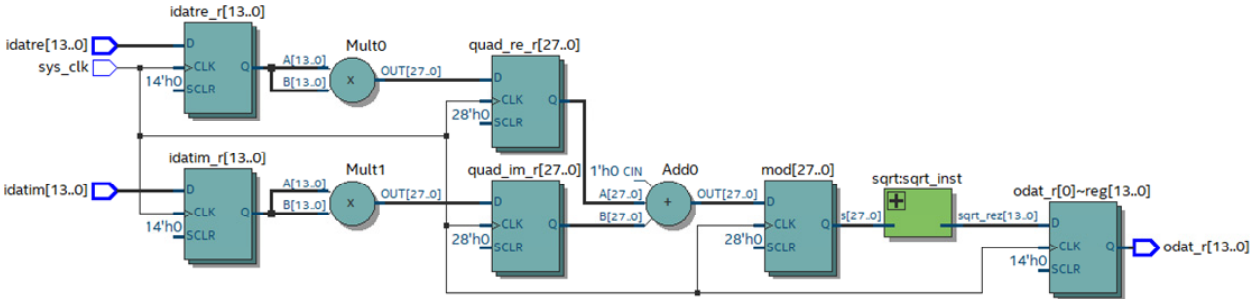


Рис. 4

Таблица 4

Аппаратный ресурс	Варианты реализации		
	Verilog_HDL реализация, поведенческое описание	Verilog_HDL реализация, с ядром ALTSQRT	IP-ядро CORDIC
Количество логических элементов	358	482	1191
Количество комбинационных функций	276	253	1190
Использование LUT логических ячеек – 4-входовые – 3-входовые – <=2-входовые	 24 205 47	 6 166 81	 0 636 554
Логические элементы – в нормальном режиме – в арифметическом режиме	 72 204	 97 156	 310 880
Общее количество регистров – выделенные регистры	 0	 327	 697
Контакты ввода/вывода	43	43	45
Встроенные 9-битные умножители	0	4	0

показано на синтезированной RTL-схеме на рис. 4. Перед вычислительными элементами использованы регистры для хранения промежуточных результатов: квадратов вещественной и мнимой частей сигнала, результатов суммирования и вычисления корня квадратного. Для управления процессом записи данных в регистры в схему введен глобальный тактовый сигнал sys_clk.

Частота работы схемы для худших случаев моделей составляет:

– Slow 85 °C: 11.8 МГц;

– Slow 0 °C: 12.4 МГц.

Такая низкая частота работы обусловлена длинными комбинационными цепочками, образующимися в модуле sqrt. Аналогичная ситуация возникает и при использовании библиотечного модуля ALTSQRT, который при реализации в комбинационном варианте также характеризуется задержками в комбинационных цепях.

Повысить частоту работы можно посредством конвейеризации указанных вычислителей. Для упрощения процесса проектирования была выполнена конвейеризация библиотечного элемента: в нем организован внутренний 7-ступенчатый конвейер. Такая коррекция позволила получить следующие значения максимально возможной частоты работы моделей:

– Slow 85 °C: 135.81 МГц;

– Slow 0 °C: 142.02 МГц.

Введение конвейера увеличило необходимое для проекта количество регистров, при этом несколько уменьшились требования к объему логических ресурсов и значительно повысилась тактовая частота. Сравнение аппаратных затрат по всем рассмотренным вариантам реализации амплитудного детектора приведено в табл. 4.

Сравнительный анализ трех вариантов реализации показал, что что использование IP неэффективно. Оптимальным решением является комбинация поведенческого описания блока с использованием специфически настроенных библиотечных элементов.

В статье проведен анализ вариантов аппаратной реализации основных арифметических операций, используемых при первичной обработке данных, снимаемых с антенной решетки автомобильного радара: умножения и деления вещественных чисел, умножения комплексных чисел,

устройства для извлечения квадратного корня. Рассмотренные варианты реализации основывались на применении как библиотечных элементов компании «Intel FPGA», так и дополнительных арифметических преобразований и специальных алгоритмов.

Сравнение результатов синтеза показало, что при наличии в структуре ПЛИС встроенных аппаратных умножителей их использование позволяет повысить эффективность проекта с точки зрения аппаратных затрат. При этом всегда существует альтернатива, основанная на применении аппаратных ресурсов логических ячеек, настроенных на работу в арифметическом режиме. При превышении количества необходимых для реализации проекта умножителей есть альтернативные варианты организации соответствующих вычислений на логических ячейках.

Рассмотренные аппаратные вычислители разработаны с использованием языков Verilog_HDL и VHDL. Отладка проводилась в системе моделирования ModelSim Altera. Для проверки синтезируемости подготовленных программных описаний использована среда проектирования Quartus Prime 18.0 компании «Intel».

Работа выполнена в СПбГЭТУ «ЛЭТИ» при финансовой поддержке Министерства науки и высшего образования Российской Федерации в рамках Соглашения № 075-11-2019-053 от 20.11.2019 г. (на основании постановления Правительства Российской Федерации от 9 апреля 2010 г. № 218) по теме: «Создание отечественного высокотехнологичного производства систем безопасности автотранспорта на основе блока управления и интеллектуальных датчиков, включающих миллиметровые радары диапазона 76–77 ГГц».

СПИСОК ЛИТЕРАТУРЫ

1. Stevenson R. A driver's sixth sense // IEEE Spectrum. 2011. Vol. 48, № 10. С. 51–55.

2. Мякочин Ю. О., Бирюков М. Ю. Миллиметровые радары АО «ПКК Миландр» для применения на автотранспорте и в системах безопасности // Электроника: наука, технология, бизнес. 2019. № 8. С. 90–95.

3. Мякочин Ю. О., Бирюков М. Ю. Автомобильные радары частотных диапазонов 24 и 77 ГГц // Электроника: наука, технология, бизнес. 2018. № 8. С. 84–88.

4. Cyclone 10 LP Advance Information Brief. URL: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/cyclone-10aib-01029.pdf (дата обращения 10.09.20).

5. Implementing Multipliers in FPGA Devices URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/an/an306.pdf (дата обращения 10.09.20).

6. Мо Чжо Чо. Особенности реализации операции умножения на ПЛИС // Современные наукоемкие технологии. 2008. № 4. С. 114–116.

7. FPGA implementation of fixed-point divider using pre-computed values / Kasim M. F., Trio Adiono, Fahreza M., Fadhli Zakiy // Procedia Technology. 2013. № 11. P. 206–211.

8. Intel FPGA integer arithmetic IP cores user guide. URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_lpm_alt_mfug.pdf (дата обращения 10.09.20).

9. Лайонс Р. Цифровая обработка сигналов. М.: БИНОМ, 2015.

10. Lakshmi B., Dhar A. S. CORDIC Architectures: A Survey // VLSI Design. 2010 URL: http://www.hindawi.com/journals/vlsi/2010/ (дата обращения 10.09.20).

11. ALTERA_CORDIC IP Core user guide. URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_cordic.pdf (дата обращения 10.09.20).

O. I. Bureneva, A. A. Pustovoitova, D. M. Sobolev
Saint Petersburg Electrotechnical University

HARDWARE IMPLEMENTATION OF ARITHMETIC OPERATIONS IN CAR'S RADAR

Presents possible variants of realization of arithmetic operations in the basis of FPGA hardware resources, aimed at application in the digital computer of automobile radar in the range 76–77 GHz. The operations are considered: multiplication and division of real numbers, multiplication of complex numbers, extraction of the square root. The comparative analysis of the synthesis results of calculators built in the basis of bi-library elements and IP cores, provided by the manufacturer of FPGA - Intel FPGA company, is carried out, and also alternative implementations in the form of RTL descriptions synthesized on the basis of developed programs in Verilog_HDL language are offered. Synthesis of devices was oriented to Cyclone 10 LP chip application and was performed using Quartus Prime design system. With the use of the considered solutions in the field of arithmetic calculators the variants of implementation of the amplitude detector – the block of primary processing of signals of automobile radar, which implements the integration of data of two quadrature channels, have been offered.

Primary data processing, hardware implementation, programmable logic device, arithmetic operations, built-in multiplier, complex multiplier, square root calculation