Продолжение. Начало в № 2 `2007

Разработка базовых компонентов цифровых устройств, реализуемых на базе ПЛИС FPGA фирмы Xilinx®, с помощью генератора параметризированных модулей CORE Generator

Валерий ЗОТОВ
walerry@km.ru

Формирование описаний элементов, выполняющих операции умножения с накоплением, на основе параметризированного модуля Multiply Accumulator с помощью средств CORE Generator

Генерация описаний элементов, выполняющих операции умножения с накоплением, для последующей реализации в составе устройств, проектируемых на основе ПЛИС FPGA фирмы Xilinx, осуществляется с помощью параметризированного модуля *Multiply Accumulator*. В общем случае алгоритм функционирования этих элементов поясняет выражение (1).

$$Q = \sum_{n=0}^{count-1} A_n \times B_n \,. \qquad (1)$$

Умножители-накопители широко применяются в составе цифровых устройств различного назначения. Наиболее часто данные элементы используются в процессе проектирования устройств цифровой обработки сигналов и, в первую очередь, при разработке цифровых фильтров. Поэтому параметризированный модуль *Multiply Accumulator* представлен одновременно в составе группы ядер *Math Functions*, предназначенных для реализации математических функций, и группы ядер *Digital Signal Processing*, используемых в устройствах цифровой обработки сигналов. В рассматриваемой версии генератора параметризированных модулей CORE Generator актуальным вариантом данного ядра является модификация *Multiply Accumulator v4.0*, которая позволяет формировать описания умножителей-накопителей, предназначенные для реализации на базе кристаллов следующих семейств: Spartan-II; Spartan-IIE; Spartan-3; Spartan-3E; Virtex; QPRO Virtex Rad-Hard; QPRO Virtex Hi-Rel; Virtex-E; QPRO Virtex-E Military; Virtex-II; Virtex-II PRO и Virtex-4. Особенностями этой версии рассматриваемого параметризированного модуля являются:

- возможность выбора разрядности используемых операндов (входных шин данных) в создаваемых элементах в диапазоне от 2 до 32 двоичных разрядов (за исключением элементов, предназначенных для реализации на основе ПЛИС семейства Virtex-4, для которых верхней границей диапазона является 18 двоичных разрядов);
- поддержка различной формы представления входных данных и полученного результата (в форме значения со знаком или без знака);
- возможность выбора одного из трех методов округления (усечения) результата выполнения операций умножения с накоплением;
- автоматический подбор производительности формируемых элементов, соответствующей системному уровню;
- возможность генерации описаний умножителей-накопителей, реализуемых на базе различных ресурсов ПЛИС (таблиц преобразования LUT, аппаратных блоков умножения Multiplier Blocks или секций XtremeDSP);
- поддержка использования сигналов подтверждения (квитирования) при осуществлении операций умножения с накоплением;
- применение дополнительных регистров в составе формируемых элементов для организации конвейерного метода выполнения операций (по выбору пользователя);
- возможность создания описаний умножителей-накопителей с входными и выходными регистрами;
- возможность выборочного использования в создаваемых умножителях-накопителях входов сигнала синхронного сброса и сигнала разрешения синхронизации.

Обобщенная структура умножителей-накопителей, создаваемых на основе параметризированного модуля *Multiply Accumulator*, показана на рис. 68.

Основными компонентами представленной структуры являются умножитель, аккумулятор и блок управления. Кроме того, в составе архитектуры умножителей-накопителей могут применяться выходные регистры, присутствие которых определяется по выбору разработчика. Эти регистры изображены на рис. 68 пунктирными линиями. Умножитель параллельного типа, используемый в формируемых элементах, осуществляет вычисление произведения значений данных, представленных одновременно на входных шинах. Результат выполнения операции поступает непосредственно на вход аккумулятора. Блок управления обеспечивает согласование работы всех компонентов умножителя-накопителя, а также поддержку входных и выходных сигналов подтверждения.

Наличие совокупности входов и выходов сигналов подтверждения в формируемом умножителе-накопителе значительно облегчает согласование работы этого элемента с функционированием других блоков проектируемого устройства. В параметризированном модуле *Multiply Accumulator* предусмотрена поддержка следующих трех сигналов подтверждения: First Data (FD), New Data (ND) и Ready (RDY). Активный уровень входного сигнала First Data (уровень логической единицы) инициирует начало нового цикла работы умножителя-накопителя (нового цикла накопления произведений значений входных данных). При этом по фронту тактового сигнала (при наличии высокого логического уровня сигнала на входе ND) в регистр аккумулятора записывается результат операции перемножения значений данных, представленных на входных

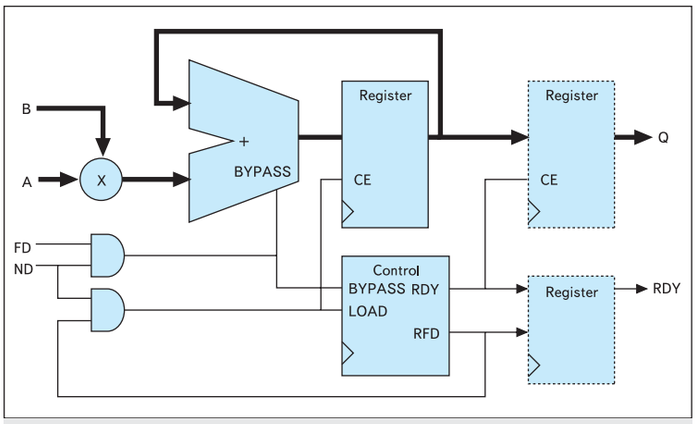

Рис. 68. Обобщенная структура элементов, выполняющих операции умножения с накоплением и формируемых на основе параметризированного модуля *Multiply Accumulator*

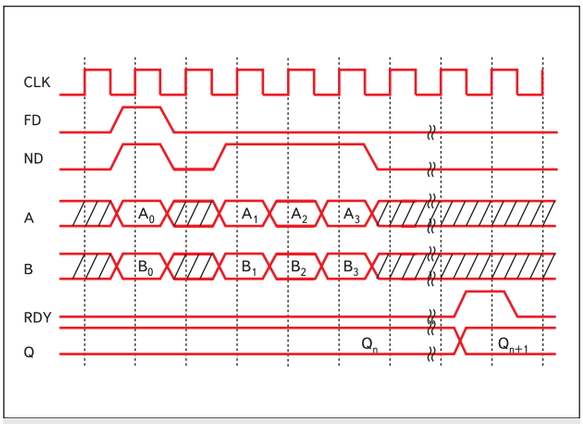

Рис. 69. Временные диаграммы, поясняющие функционирование умножителей-накопителей, формируемых на основе параметризированного модуля *Multiply Accumulator*

шинах A и B. Активный уровень сигнала New Data информирует о присутствии новых значений данных на входных шинах элемента, выполняющего операции умножения с накоплением. Высокий логический уровень сигнала на выходе Ready указывает на готовность (достоверность) выходных данных (полученного результата вычислений). На рис. 69 приведены временные диаграммы, которые наглядно поясняют функционирование умножителей-накопителей, формируемых на основе параметризированного модуля *Multiply Accumulator*.

Рассматриваемое ядро позволяет для каждого входного порта формируемого элемента индивидуально выбрать формат представления значений данных (со знаком или без знака). При этом следует учитывать, что если данные на одной из входных шин умножителя-накопителя воспринимаются как значения со знаком, то выходные данные (результат выполнения операций умножения с накоплением) также будут интерпретироваться как значения со знаком. Таким образом, чтобы данные на выходе создаваемого элемента соответствовали типу значений без знака, следует для каждой входной шины выбрать тот же формат представления данных (без знака).

В описаниях умножителей-накопителей, создаваемых с помощью параметризированного модуля *Multiply Accumulator*, используется следующая система условных обозначений входных и выходных портов:

- A[M:0] — входная шина данных A с разрядностью M+1;
- B[M:0] — входная шина данных B с разрядностью M+1;
- CLK — вход тактового сигнала;
- CE — вход сигнала разрешения синхронизации;
- RST — вход сигнала синхронного сброса;
- FD — вход сигнала First Data, инициирующего начало нового цикла накопления произведений значений входных данных;
- ND — вход сигнала New Data (ND), сообщающего о присутствии новых значений входных данных;
- Q[N:0] — выходная шина данных с разрядностью N+1;
- RDY — выход сигнала Ready, информирующего о готовности (достоверности) выходных данных генерируемого элемента.

Наиболее наглядным способом указания значений параметров генерируемого элемента, предназначенного для выполнения операций умножения с накоплением, является применение «мастера» настройки ядра *Multiply Accumulator*, в составе которого используются три диалоговые панели. Стартовая диалоговая панель данного «мастера» предназначена для определения количества операций умножения с накоплением, выполняемых формируемым элементом за один цикл работы, значения частоты тактового сигнала, разрядности и формы представления данных входных портов. Вид страницы *Parameters* данной диалоговой панели приведен на рис. 70.

Число операций умножения-накопления, которые должны выполняться генерируемым элементом за один цикл работы, иными словами, значение параметра count в выражении (1), указывается в поле редактирования *Number MAC Cycles*, расположенном во встроенной панели *System Parameters*. Значение частоты тактового сигнала задается в поле редактирования *Clock Rate*, которое находится в этой же встроенной панели. Указываемые значения параметров *Number MAC Cycles* и *Clock Rate* не должны выходить за границы предельно допустимых диапазонов, информация о которых отображается справа от соответствующего поля редактирования (рис. 70).

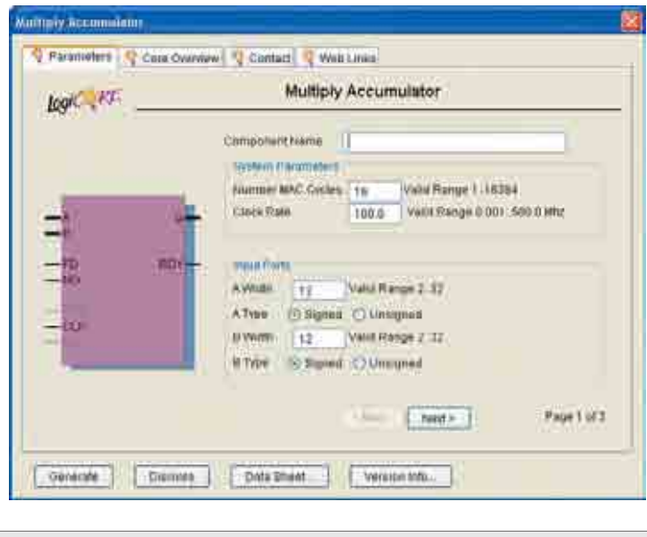

Рис. 70. Страница *Parameters* стартовой диалоговой панели «мастера» настройки параметров ядра умножителя-накопителя *Multiply Accumulator*

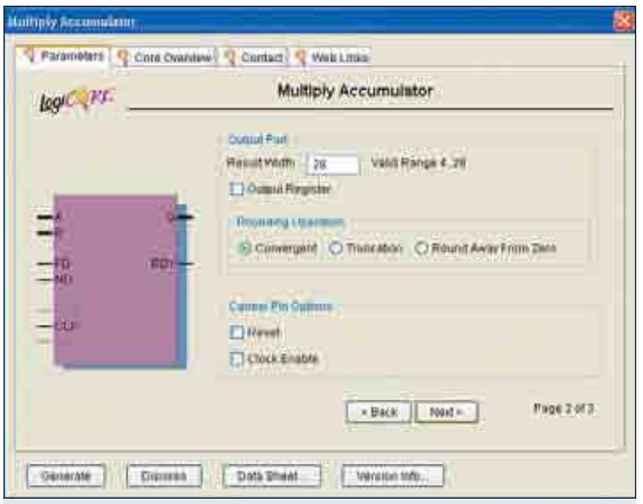

Рис. 71. Вид второй диалоговой панели «мастера» настройки ядра умножителя-накопителя *Multiply Accumulator*, предназначенной для определения параметров выходного порта и выбора сигналов управления (страница *Parameters*)

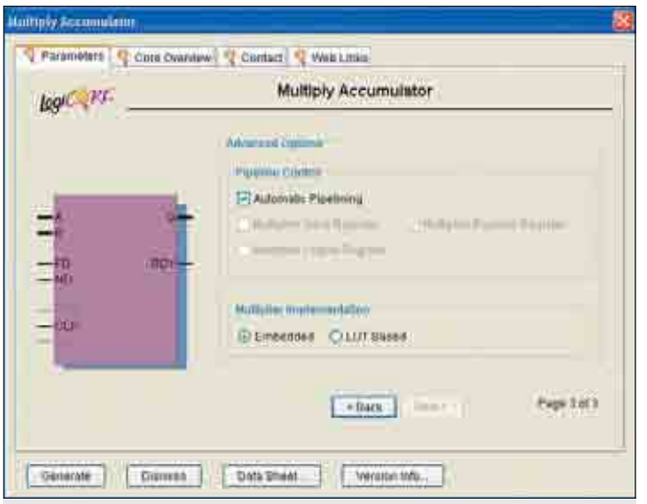

Рис. 72. Страница *Parameters* заключительной диалоговой панели «мастера» настройки ядра умножителя-накопителя *Multiply Accumulator*, предназначенная для выбора параметров конвейерной организации обработки данных и определения типа аппаратных ресурсов ПЛИС, используемых для реализации формируемого элемента

Разрядность входных портов (входных шин данных) A и B формируемого умножителя-накопителя определяется с помощью полей редактирования *A Width* и *B Width* соответственно. Эти поля редактирования расположены во встроенной панели *Input Ports* (рис. 70). Выбор формы представления значений для каждой входной шины данных осуществляется с помощью группы кнопок с зависимой фиксацией *A Type* и *B Type* соответственно. Чтобы данные, поступающие на какую-либо входную шину, интерпретировались как значения со знаком, следует зафиксировать в нажатом состоянии кнопку *Signed* в соответствующей группе (*A Type* или *B Type*). Если входные данные должны восприниматься как операнды без знака, то в нажатое состояние нужно переключить кнопку *Unsigned*. По умолчанию предлагается форма представления входных данных в виде значений со знаком.

Вторая диалоговая панель «мастера» настройки ядра умножителя-накопителя *Multiply Accumulator*, вид которой показан на рис. 71, позволяет определить параметры выходного порта и выбрать требуемые входы сигналов управления в формируемом элементе.

Значение разрядности выходного порта (выходной шины данных) создаваемого умножителя-накопителя указывается в поле редактирования *Result Width*, расположенном во встроенной панели *Output Port*. Справа от данного поля, в строке *Valid Range*, отображается информация о диапазоне допустимых значений разрядности выходной шины данных. Верхняя граница этого диапазона соответствует количеству разрядов выходного порта, которое необходимо для точного представления результата выполнения операций умножения и накопления заданной последовательности поступающих данных. Это значение автоматически вычисляется на основании количества разрядов входных шин данных и числа операций умножения и сложения, выполняемых за один цикл работы формируемого элемента, которые были указаны в соответствующих полях редактирования стартовой диалоговой панели.

Если разработчик выбирает меньшую разрядность выходного порта по сравнению с верхней границей допустимого диапазона значений этого параметра, то результат выполнения операций в формируемом умножителе-накопителе будет представлен в округленном или усеченном виде. При этом пользователь может выбрать один из трех возможных методов округления (усечения), используя группу кнопок с зависимой фиксацией *Rounding Operation*. Когда в нажатом состоянии зафиксирована кнопка *Truncation*, выполняется усечение полученного результата до указанного количества двоичных разрядов, при котором младшие неиспользуемые биты значения отбрасываются, а оставшиеся разряды поступают на выход формируемого элемента в неизменном виде. При нажатой кнопке *Convergent* округление результата вычислений до заданного количества двоичных разрядов производится в соответствии со следующими правилами. Если десятичное значение округляемой дробной части (неиспользуемых младших разрядов) полученного результата меньше 0,5, то осуществляется операция усечения, рассмотренная выше. При равенстве десятичного значения округляемой дробной части 0,5 точный результат вычислений округляется до ближайшего четного значения. Когда десятичное значение округляемой дробной части больше 0,5, целая часть результата (представленная оставшимися двоичными разрядами) увеличивается на единицу. Такой метод округления довольно часто применяется в устройствах цифровой обработки сигналов. При переключении в нажатое состояние кнопки *Round Away From Zero* в формируемом элементе будет использоваться стандартный способ округления дробного числа до ближайшего целого значения. Основное отличие данного способа от метода *Convergent* проявляется при округлении результатов вычислений, десятичное значение дробной части которых равно 0,5. В этом случае при использовании способа округления *Round Away From Zero* значение целой части результата, представленной оставшимися двоичными разрядами, увеличивается на единицу.

Чтобы добавить выходной регистр в состав структуры формируемого элемента, предназначенного для выполнения операций умножения с накоплением, нужно перевести в состояние «Включено» индикатор *Output Register*, который также находится во встроенной панели *Output Port*.

Выбор необходимых входов управляющих сигналов в генерируемом умножителе-накопителе осуществляется с помощью индикаторов состояния, которые представлены во встроенной панели *Control Pin Options* (рис. 71). Чтобы задействовать вход синхронного сброса, нужно установить в состояние «Включено» индикатор *Reset*. При наличии активного уровня сигнала (высокого логического уровня) на этом входе содержимое аккумулятора обнуляется по фронту тактового сигнала. Для использования в составе формируемого умножителя-накопителя входа разрешения сигнала синхронизации следует перевести во включенное состояние индикатор *Clock Enable*.

Третья, заключительная, диалоговая панель «мастера» настройки параметров ядра умножителя-накопителя *Multiply Accumulator*

используется для выбора параметров конвейерной организации обработки данных и определения типа аппаратных ресурсов ПЛИС, применяемых для реализации формируемого элемента. Вид страницы *Parameters* данной диалоговой панели показан на рис. 72.

Параметры конвейерной организации обработки данных в разрабатываемом умножителе-накопителе разработчик может определять автоматически или задавать «вручную». Выбор одного из этих способов осуществляется с помощью индикатора состояния *Automatic Pipelining*, который представлен во встроенной панели *Pipeline Control* (рис. 72). Если этот индикатор установлен в состояние «включено», то на основании значения тактовой частоты, указанного пользователем в стартовой диалоговой панели, производится автоматическое вычисление оптимального количества ступеней (дополнительных регистров) конвейерной организации обработки данных. При переводе индикатора *Automatic Pipelining* в выключенное состояние пользователю предоставляется возможность выбора дополнительных регистров, необходимых для организации конвейерной обработки данных в формируемом элементе. Для этой цели предназначены индикаторы состояния *Multiplier Input Register*, *Multiplier Output Register* и *Multiplier Pipelined Register*, которые также находятся во встроенной панели *Pipeline Control*. Эти индикаторы становятся доступными только после установки индикатора *Automatic Pipelining* в состояние «Выключено».

Для использования в составе структуры разрабатываемого умножителя-накопителя входных регистров нужно перевести во включенное состояние индикатор *Multiplier Input Register*. Если необходимо задействовать дополнительный регистр на выходе умножителя, применяемого в составе формируемого элемента, то следует установить индикатор *Multiplier Output Register* в состояние «Включено». Чтобы в умножителе, входящем в структуру генерируемого элемента, присутствовали дополнительные регистры, используемые для организации конвейерной обработки данных, индикатор *Multiplier Pipelined Register* должен находиться во включенном состоянии.

Определение типа аппаратных ресурсов ПЛИС, на основе которых будет выполняться реализация формируемого описания умножителя-накопителя, осуществляется с помощью двух кнопок с зависимой фиксацией, расположенных во встроенной панели *Multiplier Implementation*. Чтобы сгенерировать описание элемента, реализуемого на основе таблиц преобразования LUT, следует зафиксировать в нажатом состоянии кнопку *LUT Based*. Для реализации формируемого описания умножителя-накопителя на базе встроенных аппаратных 18-разрядных блоков умножения *Multiplier Blocks* или секций XtremeDSP нужно переключить в нажатое положение кнопку *Embedded*. Если кристаллы семейства, выбранного для реализации генерируемого элемента, не содержат указанных встроенных аппаратных блоков, то в нажатое состояние автоматически устанавливается кнопка *LUT Based*. В этом случае пользователь не может изменить предлагаемый вариант реализации разрабатываемого элемента, предназначенного для выполнения операций умножения с накоплением.

Пример описания элемента, предназначенного для выполнения операций умножения с накоплением, сформированного с помощью параметризированного модуля Multiply Accumulator

В качестве примера описания элемента, выполняющего операции умножения с накоплением, подготовленного с помощью параметризированного модуля *Multiply Accumulator*, в настоящем разделе приведен текст VHDL-описания умножителя-накопителя mac_16_24. Данный элемент выполняет операцию суммирования двадцати четырех произведений 16-разрядных значений входных данных. Данные, поступающие на входные шины умножителя-накопителя mac_16_24, интерпретируются в форме значений со знаком. Результат выполнения операций умножения с накоплением округляется до 36 разрядов по методу *Convergent*. В структуру сгенерированного элемента включены входные и выходной регистры, а также дополнительные регистры умножителя, используемые для организации конвейерной обработки данных. Реализация сформированного описания умножителя-накопителя осуществляется на базе встроенных аппаратных блоков ПЛИС.

Описание архитектуры элемента mac_16_24 соответствует обобщенной структуре умножителей-накопителей, формируемых на основе параметризированного модуля *Multiply Accumulator*, которая представлена на рис. 68. Текст сгенерированного VHDL-описания умножителя-аккумулятора для наглядности можно условно разбить на девять частей. Эти части рассматриваются далее в том же порядке, в котором они расположены в тексте описания сформированного элемента.

В первой части приведены ссылки на используемые библиотеки и пакеты этих библиотек. В отличие от рассмотренных ранее элементов, созданных с помощью генератора параметризированных модулей CORE Generator, в VHDL-описании элемента mac_16_24 кроме ссылок на стандартные логические библиотеки и унифицированные библиотеки фирмы Xilinx присутствуют ссылки на пакеты библиотеки XilinxCoreLib. Использование пакетов этой библиотеки связано с тем, что в состав сформированного описания включены компоненты, создаваемые на основе других параметризированных модулей. Текст первой части VHDL-описания рассматриваемого умножителя-накопителя имеет следующий вид:

```
LIBRARY std, ieee;
USE std.standard.ALL;
USE ieee.std_logic_1164.ALL;
--
LIBRARY unisim;
USE unisim.vcomponents.ALL;
--
LIBRARY XilinxCoreLib;
USE XilinxCoreLib.mult_gen_v7_0_comp.ALL;
USE XilinxCoreLib.c_shift_ram_v7_0_comp.ALL;
USE XilinxCoreLib.c_reg_fd_v7_0_comp.ALL;
USE XilinxCoreLib.c_accum_v7_0_comp.ALL;
USE XilinxCoreLib.c_addsub_v7_0_comp.ALL;
USE XilinxCoreLib.c_counter_binary_v7_0_comp.ALL;
```

Вторая часть содержит объявление объекта mac_16_24, представляющего разрабатываемый умножитель-накопитель, и описание его внешнего интерфейса. Эта часть выглядит следующим образом:

```
ENTITY mac_16_24 IS
   PORT (
            A : IN STD_LOGIC_VECTOR(15 DOWNTO 0);
            B : IN STD_LOGIC_VECTOR(15 DOWNTO 0);
            CLK : IN STD_LOGIC;
            CE : IN STD_LOGIC;
            RST : IN STD_LOGIC;
            FD : IN STD_LOGIC;
            ND : IN STD_LOGIC;
            Q : OUT STD_LOGIC_VECTOR(35 DOWNTO 0);
            RDY : OUT STD_LOGIC
            );
END mac_16_24;
```

Третью часть VHDL-описания умножителя-накопителя образуют приведенные далее выражения декларации внутренних сигналов, используемых для соединения компонентов, входящих в состав формируемого элемента:

```
ARCHITECTURE xilinx OF mac_16_24 IS
-- Signals for connecting to instantiations
   SIGNAL BU5_I0 : STD_LOGIC;
   SIGNAL BU5_I1 : STD_LOGIC;
   SIGNAL BU5_I2 : STD_LOGIC;
   SIGNAL BU5_I3 : STD_LOGIC;
   SIGNAL BU5_O : STD_LOGIC;
   SIGNAL BU8_I0 : STD_LOGIC;
   SIGNAL BU8_I1 : STD_LOGIC;
   SIGNAL BU8_I2 : STD_LOGIC;
   SIGNAL BU8_I3 : STD_LOGIC;
   SIGNAL BU8_O : STD_LOGIC;
   SIGNAL BU10_Q : STD_LOGIC_VECTOR(4 DOWNTO 0);
   SIGNAL BU10_CLK : STD_LOGIC;
   SIGNAL BU10_Q_THRESH0 : STD_LOGIC;
   SIGNAL BU10_CE : STD_LOGIC;
   SIGNAL BU10_SCLR : STD_LOGIC;
   SIGNAL BU239_I0 : STD_LOGIC;
   SIGNAL BU239_I1 : STD_LOGIC;
   SIGNAL BU239_I2 : STD_LOGIC;
   SIGNAL BU239_I3 : STD_LOGIC;
   SIGNAL BU239_O : STD_LOGIC;
   SIGNAL BU241_CLK : STD_LOGIC;
   SIGNAL BU241_D : STD_LOGIC_VECTOR(0 DOWNTO 0);
   SIGNAL BU241_Q : STD_LOGIC_VECTOR(0 DOWNTO 0);
   SIGNAL BU241_CE : STD_LOGIC;
   SIGNAL BU249_CLK : STD_LOGIC;
   SIGNAL BU249_D : STD_LOGIC_VECTOR(0 DOWNTO 0);
   SIGNAL BU249_Q : STD_LOGIC_VECTOR(0 DOWNTO 0);
   SIGNAL BU249_CE : STD_LOGIC;
   SIGNAL BU764_CLK : STD_LOGIC;
   SIGNAL BU764_D : STD_LOGIC_VECTOR(0 DOWNTO 0);
   SIGNAL BU764_Q : STD_LOGIC_VECTOR(0 DOWNTO 0);
   SIGNAL BU764_CE : STD_LOGIC;
   SIGNAL BU772_D : STD_LOGIC_VECTOR(0 DOWNTO 0);
   SIGNAL BU772_Q : STD_LOGIC_VECTOR(0 DOWNTO 0);
   SIGNAL BU772_CLK : STD_LOGIC;
   SIGNAL BU772_CE : STD_LOGIC;
   SIGNAL BU776_I0 : STD_LOGIC;
   SIGNAL BU776_I1 : STD_LOGIC;
   SIGNAL BU776_I2 : STD_LOGIC;
   SIGNAL BU776_I3 : STD_LOGIC;
   SIGNAL BU776_O : STD_LOGIC;
```

```
SIGNAL BU53_clk : STD_LOGIC;
SIGNAL BU53_a : STD_LOGIC_VECTOR(15 DOWNTO 0);
SIGNAL BU53_b : STD_LOGIC_VECTOR(15 DOWNTO 0);
SIGNAL BU53_o : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL BU53_q : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL BU53_a_signed : STD_LOGIC;
SIGNAL BU53_loadb : STD_LOGIC;
SIGNAL BU53_load_done : STD_LOGIC;
SIGNAL BU53_swapb : STD_LOGIC;
SIGNAL BU53_ce : STD_LOGIC;
SIGNAL BU53_aclr : STD_LOGIC;
SIGNAL BU53_sclr : STD_LOGIC;
SIGNAL BU53_rfd : STD_LOGIC;
SIGNAL BU53_nd : STD_LOGIC;
SIGNAL BU53_rdy : STD_LOGIC;
SIGNAL BU256_B : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL BU256_Q : STD_LOGIC_VECTOR(36 DOWNTO 0);
SIGNAL BU256_CLK : STD_LOGIC;
SIGNAL BU256_BYPASS : STD_LOGIC;
SIGNAL BU256_CE : STD_LOGIC;
SIGNAL BU256_ACLR : STD_LOGIC;
SIGNAL BU523_CLK : STD_LOGIC;
SIGNAL BU523_D : STD_LOGIC_VECTOR(0 DOWNTO 0);
SIGNAL BU523_Q : STD_LOGIC_VECTOR(0 DOWNTO 0);
SIGNAL BU523_CE : STD_LOGIC;
SIGNAL BU531_I0 : STD_LOGIC;
SIGNAL BU531_I1 : STD_LOGIC;
SIGNAL BU531_I2 : STD_LOGIC;
SIGNAL BU531_I3 : STD_LOGIC;
SIGNAL BU531_O : STD_LOGIC;
SIGNAL BU538_I0 : STD_LOGIC;
SIGNAL BU538_I1 : STD_LOGIC;
SIGNAL BU538_I2 : STD_LOGIC;
SIGNAL BU538_I3 : STD_LOGIC;
SIGNAL BU538_O : STD_LOGIC;
SIGNAL BU540_A : STD_LOGIC_VECTOR(35 DOWNTO 0);
SIGNAL BU540_B : STD_LOGIC_VECTOR(0 DOWNTO 0);
SIGNAL BU540_S : STD_LOGIC_VECTOR(36 DOWNTO 0);
SIGNAL BU690_D : STD_LOGIC_VECTOR(35 DOWNTO 0);
SIGNAL BU690_Q : STD_LOGIC_VECTOR(35 DOWNTO 0);
SIGNAL BU690_CLK : STD_LOGIC;
SIGNAL BU690_CE : STD_LOGIC;
SIGNAL n0 : STD_LOGIC := '0';
SIGNAL n1 : STD_LOGIC := '1';
SIGNAL n2 : STD_LOGIC;
SIGNAL n3 : STD_LOGIC;
SIGNAL n4 : STD_LOGIC;
SIGNAL n5 : STD_LOGIC;
SIGNAL n6 : STD_LOGIC;
SIGNAL n7 : STD_LOGIC;
SIGNAL n8 : STD_LOGIC;
SIGNAL n9 : STD_LOGIC;
SIGNAL n10 : STD_LOGIC;
SIGNAL n11 : STD_LOGIC;
SIGNAL n12 : STD_LOGIC;
SIGNAL n13 : STD_LOGIC;
SIGNAL n14 : STD_LOGIC;
SIGNAL n15 : STD_LOGIC;
SIGNAL n16 : STD_LOGIC;
SIGNAL n17 : STD_LOGIC;
SIGNAL n18 : STD_LOGIC;
SIGNAL n19 : STD_LOGIC;
SIGNAL n20 : STD_LOGIC;
SIGNAL n21 : STD_LOGIC;
SIGNAL n22 : STD_LOGIC;
SIGNAL n23 : STD_LOGIC;
SIGNAL n24 : STD_LOGIC;
SIGNAL n25 : STD_LOGIC;
SIGNAL n26 : STD_LOGIC;
SIGNAL n27 : STD_LOGIC;
SIGNAL n28 : STD_LOGIC;
SIGNAL n29 : STD_LOGIC;
SIGNAL n30 : STD_LOGIC;
SIGNAL n31 : STD_LOGIC;
SIGNAL n32 : STD_LOGIC;
SIGNAL n33 : STD_LOGIC;
SIGNAL n34 : STD_LOGIC;
SIGNAL n35 : STD_LOGIC;
SIGNAL n36 : STD_LOGIC;
SIGNAL n37 : STD_LOGIC;
SIGNAL n38 : STD_LOGIC;
SIGNAL n40 : STD_LOGIC;
SIGNAL n41 : STD_LOGIC;
SIGNAL n42 : STD_LOGIC;
SIGNAL n43 : STD_LOGIC;
SIGNAL n44 : STD_LOGIC;
SIGNAL n45 : STD_LOGIC;
SIGNAL n46 : STD_LOGIC;
SIGNAL n47 : STD_LOGIC;
SIGNAL n48 : STD_LOGIC;
SIGNAL n49 : STD_LOGIC;
SIGNAL n50 : STD_LOGIC;
SIGNAL n51 : STD_LOGIC;
SIGNAL n52 : STD_LOGIC;
SIGNAL n53 : STD_LOGIC;
SIGNAL n54 : STD_LOGIC;
SIGNAL n55 : STD_LOGIC;
SIGNAL n56 : STD_LOGIC;
SIGNAL n57 : STD_LOGIC;
SIGNAL n58 : STD_LOGIC;
SIGNAL n59 : STD_LOGIC;
SIGNAL n60 : STD_LOGIC;
SIGNAL n61 : STD_LOGIC;
SIGNAL n62 : STD_LOGIC;
SIGNAL n63 : STD_LOGIC;
SIGNAL n64 : STD_LOGIC;
SIGNAL n65 : STD_LOGIC;
SIGNAL n66 : STD_LOGIC;
SIGNAL n67 : STD_LOGIC;
SIGNAL n68 : STD_LOGIC;
SIGNAL n69 : STD_LOGIC;
SIGNAL n70 : STD_LOGIC;
SIGNAL n71 : STD_LOGIC;
SIGNAL n72 : STD_LOGIC;
SIGNAL n73 : STD_LOGIC;
SIGNAL n74 : STD_LOGIC;
SIGNAL n75 : STD_LOGIC;
SIGNAL n76 : STD_LOGIC;
SIGNAL n77 : STD_LOGIC;
SIGNAL n78 : STD_LOGIC;
SIGNAL n79 : STD_LOGIC;
SIGNAL n80 : STD_LOGIC;
SIGNAL n82 : STD_LOGIC;
SIGNAL n83 : STD_LOGIC;
SIGNAL n84 : STD_LOGIC;
SIGNAL n85 : STD_LOGIC;
SIGNAL n86 : STD_LOGIC;
SIGNAL n87 : STD_LOGIC;
SIGNAL n88 : STD_LOGIC;
SIGNAL n89 : STD_LOGIC;
SIGNAL n90 : STD_LOGIC;
SIGNAL n91 : STD_LOGIC;
SIGNAL n92 : STD_LOGIC;
SIGNAL n93 : STD_LOGIC;
SIGNAL n94 : STD_LOGIC;
SIGNAL n95 : STD_LOGIC;
SIGNAL n96 : STD_LOGIC;
SIGNAL n97 : STD_LOGIC;
SIGNAL n98 : STD_LOGIC;
SIGNAL n99 : STD_LOGIC;
SIGNAL n100 : STD_LOGIC;
SIGNAL n101 : STD_LOGIC;
SIGNAL n102 : STD_LOGIC;
SIGNAL n103 : STD_LOGIC;
SIGNAL n104 : STD_LOGIC;
SIGNAL n105 : STD_LOGIC;
SIGNAL n106 : STD_LOGIC;
SIGNAL n107 : STD_LOGIC;
SIGNAL n108 : STD_LOGIC;
SIGNAL n109 : STD_LOGIC;
SIGNAL n110 : STD_LOGIC;
SIGNAL n111 : STD_LOGIC;
SIGNAL n112 : STD_LOGIC;
SIGNAL n113 : STD_LOGIC;
SIGNAL n114 : STD_LOGIC;
SIGNAL n115 : STD_LOGIC;
SIGNAL n116 : STD_LOGIC;
SIGNAL n117 : STD_LOGIC;
SIGNAL n118 : STD_LOGIC;
SIGNAL n119 : STD_LOGIC;
SIGNAL n120 : STD_LOGIC;
SIGNAL n121 : STD_LOGIC;
SIGNAL n122 : STD_LOGIC;
SIGNAL n123 : STD_LOGIC;
SIGNAL n124 : STD_LOGIC;
SIGNAL n125 : STD_LOGIC;
SIGNAL n126 : STD_LOGIC;
SIGNAL n127 : STD_LOGIC;
SIGNAL n128 : STD_LOGIC;
SIGNAL n129 : STD_LOGIC;
SIGNAL n130 : STD_LOGIC;
SIGNAL n131 : STD_LOGIC;
SIGNAL n132 : STD_LOGIC;
SIGNAL n133 : STD_LOGIC;
SIGNAL n134 : STD_LOGIC;
SIGNAL n135 : STD_LOGIC;
SIGNAL n136 : STD_LOGIC;
SIGNAL n137 : STD_LOGIC;
SIGNAL n138 : STD_LOGIC;
SIGNAL n139 : STD_LOGIC;
SIGNAL n140 : STD_LOGIC;
SIGNAL n141 : STD_LOGIC;
SIGNAL n142 : STD_LOGIC;
SIGNAL n143 : STD_LOGIC;
SIGNAL n144 : STD_LOGIC;
SIGNAL n145 : STD_LOGIC;
SIGNAL n146 : STD_LOGIC;
SIGNAL n147 : STD_LOGIC;
SIGNAL n148 : STD_LOGIC;
SIGNAL n149 : STD_LOGIC;
SIGNAL n150 : STD_LOGIC;
SIGNAL n151 : STD_LOGIC;
SIGNAL n152 : STD_LOGIC;
SIGNAL n153 : STD_LOGIC;
SIGNAL n154 : STD_LOGIC;
SIGNAL n155 : STD_LOGIC;
SIGNAL n1018 : STD_LOGIC;
SIGNAL n1019 : STD_LOGIC;
SIGNAL n1020 : STD_LOGIC;
SIGNAL n1021 : STD_LOGIC;
SIGNAL n1022 : STD_LOGIC;
SIGNAL n1023 : STD_LOGIC;
SIGNAL n1024 : STD_LOGIC;
SIGNAL n1025 : STD_LOGIC;
SIGNAL n1026 : STD_LOGIC;
SIGNAL n1027 : STD_LOGIC;
SIGNAL n1028 : STD_LOGIC;
SIGNAL n1029 : STD_LOGIC;
SIGNAL n1030 : STD_LOGIC;
SIGNAL n1031 : STD_LOGIC;
SIGNAL n1032 : STD_LOGIC;
SIGNAL n1033 : STD_LOGIC;
SIGNAL n1034 : STD_LOGIC;
SIGNAL n1035 : STD_LOGIC;
SIGNAL n1036 : STD_LOGIC;
SIGNAL n1037 : STD_LOGIC;
SIGNAL n1038 : STD_LOGIC;
SIGNAL n1039 : STD_LOGIC;
SIGNAL n1040 : STD_LOGIC;
SIGNAL n1041 : STD_LOGIC;
SIGNAL n1042 : STD_LOGIC;
SIGNAL n1043 : STD_LOGIC;
SIGNAL n1044 : STD_LOGIC;
SIGNAL n1045 : STD_LOGIC;
SIGNAL n1046 : STD_LOGIC;
SIGNAL n1047 : STD_LOGIC;
SIGNAL n1048 : STD_LOGIC;
SIGNAL n1049 : STD_LOGIC;
SIGNAL n1050 : STD_LOGIC;
SIGNAL n1051 : STD_LOGIC;
SIGNAL n1052 : STD_LOGIC;
SIGNAL n1053 : STD_LOGIC;
SIGNAL n1055 : STD_LOGIC;
```

Четвертая часть включает в себя следующую совокупность операторов, описывающих интерфейс между внутренними сигналами и внешними (входными и выходными) портами сгенерированного умножителя-накопителя:

```
BEGIN
--
   n82 <= A(0);
   n83 <= A(1);
   n84 <= A(2);
   n85 <= A(3);
   n86 <= A(4);
   n87 <= A(5);
   n88 <= A(6);
   n89 <= A(7);
   n90 <= A(8);
   n91 <= A(9);
   n92 <= A(10);
   n93 <= A(11);
   n94 <= A(12);
   n95 <= A(13);
   n96 <= A(14);
   n97 <= A(15);
   n98 <= B(0);
   n99 <= B(1);
   n100 <= B(2);
   n101 <= B(3);
   n102 <= B(4);
   n103 <= B(5);
   n104 <= B(6);
   n105 <= B(7);
   n106 <= B(8);
   n107 <= B(9);
   n108 <= B(10);
   n109 <= B(11);
   n110 <= B(12);
   n111 <= B(13);
   n112 <= B(14);
   n113 <= B(15);
   n150 <= CLK;
   n151 <= CE;
   n152 <= RST;
   n153 <= FD;
   n154 <= ND;
   Q(0) <= n114;
   Q(1) <= n115;
   Q(2) <= n116;
   Q(3) <= n117;
   Q(4) <= n118;
   Q(5) <= n119;
   Q(6) <= n120;
   Q(7) <= n121;
```

```
Q(8) <= n122;
Q(9) <= n123;
Q(10) <= n124;
Q(11) <= n125;
Q(12) <= n126;
Q(13) <= n127;
Q(14) <= n128;
Q(15) <= n129;
Q(16) <= n130;
Q(17) <= n131;
Q(18) <= n132;
Q(19) <= n133;
Q(20) <= n134;
Q(21) <= n135;
Q(22) <= n136;
Q(23) <= n137;
Q(24) <= n138;
Q(25) <= n139;
Q(26) <= n140;
Q(27) <= n141;
Q(28) <= n142;
Q(29) <= n143;
Q(30) <= n144;
Q(31) <= n145;
Q(32) <= n146;
Q(33) <= n147;
Q(34) <= n148;
Q(35) <= n149;
RDY <= n155;
```

В состав пятой части входят операторы, используемые для описания сигналов и компонентов, которые относятся к блоку управления разрабатываемого умножителя-накопителя. В этой части описания, текст которой приведен далее, применяются компоненты, созданные на основе параметризированных модулей двоичного счетчика Binary Counter, сдвигового регистра RAM-based Shift Register и параллельного регистра FD-based Parallel Register:

```
BU5_I0 <= n154;
BU5_I1 <= '1';
BU5_I2 <= n151;
BU5_I3 <= n152;
n34 <= BU5_O;
BU5 : LUT4
    GENERIC MAP (
                INIT => X»ff80»
                )
    PORT MAP (
                I0 => BU5_I0,
                I1 => BU5_I1,
                I2 => BU5_I2,
                I3 => BU5_I3,
                O => BU5_O
                );
--
BU8_I0 <= n35;
BU8_I1 <= n153;
BU8_I2 <= n152;
BU8_I3 <= n151;
n36 <= BU8_O;
BU8 : LUT4
    GENERIC MAP (
                INIT => X»fef0»
                )
    PORT MAP (
                I0 => BU8_I0,
                I1 => BU8_I1,
                I2 => BU8_I2,
                I3 => BU8_I3,
                O => BU8_O
                );
--
BU10_CLK <= n150;
n35 <= BU10_Q_THRESH0;
BU10_CE <= n34;
BU10_SCLR <= n36;
BU10 : C_COUNTER_BINARY_V7_0
    GENERIC MAP (
                c_count_mode => 0,
                c_has_aset => 0,
                c_load_enable => 0,
                c_load_low => 0,
                c_count_to => «00000»,
                c_sync_priority => 0,
                c_has_iv => 0,
                c_has_sclr => 1,
                c_restrict_count => 0,
                c_width => 5,
                c_has_q_thresh1 => 0,
                c_enable_rlocs => 0,
                c_has_q_thresh0 => 1,
                c_thresh1_value => «00000»,
                c_has_load => 0,
                c_has_up => 0,
                c_thresh_early => 1,
                c_has_thresh1 => 0,
                c_has_thresh0 => 0,
                c_ainit_val => «00000»,
                c_has_ce => 1,
                c_pipe_stages => 0,
                c_has_aclr => 0,
                c_sync_enable => 0,
                c_has_ainit => 0,
                c_sinit_val => «00000»,
                c_has_sset => 0,
                c_has_sinit => 0,
                c_count_by => «00001»,
                c_has_l => 0,
                c_thresh0_value => «10111»
                )
    PORT MAP (
                Q => BU10_Q,
                CLK => BU10_CLK,
                Q_THRESH0 => BU10_Q_THRESH0,
                CE => BU10_CE,
                SCLR => BU10_SCLR
                );
--
BU239_I0 <= n153;
BU239_I1 <= '0';
BU239_I2 <= '0';
BU239_I3 <= '0';
n37 <= BU239_O;
BU239 : LUT4
    GENERIC MAP (
                INIT => X»5555»
                )
    PORT MAP (
                I0 => BU239_I0,
                I1 => BU239_I1,
                I2 => BU239_I2,
                I3 => BU239_I3,
                O => BU239_O
                );
--
BU241_CLK <= n150;
BU241_D(0) <= n37;
n38 <= BU241_Q(0);
BU241_CE <= n151;
BU241 : C_SHIFT_RAM_V7_0
    GENERIC MAP (
                c_has_aset => 0,
                c_read_mif => 0,
                c_has_a => 0,
                c_sync_priority => 0,
                c_has_sclr => 0,
                c_width => 1,
                c_enable_rlocs => 0,
                c_default_data_radix => 2,
                c_generate_mif => 0,
                c_ainit_val => «0»,
                c_has_ce => 1,
                c_has_aclr => 0,
                c_mem_init_radix => 2,
                c_sync_enable => 0,
                c_depth => 3,
                c_has_ainit => 0,
                c_sinit_val => «0»,
                c_has_sset => 0,
                c_has_sinit => 0,
                c_shift_type => 0,
                c_mem_init_file => «null»,
                c_default_data => «0»,
                c_reg_last_bit => 1,
                c_addr_width => 1
                )
    PORT MAP (
                CLK => BU241_CLK,
                D => BU241_D,
                Q => BU241_Q,
                CE => BU241_CE
                );
--
BU249_CLK <= n150;
BU249_D(0) <= n154;
BU249_CE <= n151;
BU249 : C_SHIFT_RAM_V7_0
    GENERIC MAP (
                c_has_aset => 0,
                c_read_mif => 0,
                c_has_a => 0,
                c_sync_priority => 0,
                c_has_sclr => 0,
                c_width => 1,
                c_enable_rlocs => 0,
                c_default_data_radix => 2,
                c_generate_mif => 0,
                c_ainit_val => «0»,
                c_has_ce => 1,
                c_has_aclr => 0,
                c_mem_init_radix => 2,
                c_sync_enable => 0,
                c_depth => 3,
                c_has_ainit => 0,
                c_sinit_val => «0»,
                c_has_sset => 0,
                c_has_sinit => 0,
                c_shift_type => 0,
                c_mem_init_file => «null»,
                c_default_data => «0»,
                c_reg_last_bit => 1,
                c_addr_width => 1
                )
    PORT MAP (
                CLK => BU249_CLK,
                D => BU249_D,
                Q => BU249_Q,
                CE => BU249_CE
                );
--
BU764_CLK <= n150;
BU764_D(0) <= n35;
n79 <= BU764_Q(0);
BU764_CE <= n151;
BU764 : C_SHIFT_RAM_V7_0
    GENERIC MAP (
                c_has_aset => 0,
                c_read_mif => 0,
                c_has_a => 0,
                c_sync_priority => 0,
                c_has_sclr => 0,
                c_width => 1,
                c_enable_rlocs => 0,
                c_default_data_radix => 2,
                c_generate_mif => 0,
                c_ainit_val => «0»,
                c_has_ce => 1,
                c_has_aclr => 0,
                c_mem_init_radix => 2,
                c_sync_enable => 0,
                c_depth => 4,
                c_has_ainit => 0,
                c_sinit_val => «0»,
                c_has_sset => 0,
                c_has_sinit => 0,
                c_shift_type => 0,
                c_mem_init_file => «null»,
                c_default_data => «0»,
                c_reg_last_bit => 1,
                c_addr_width => 1
                )
    PORT MAP (
                CLK => BU764_CLK,
                D => BU764_D,
                Q => BU764_Q,
                CE => BU764_CE
                );
--
BU772_D(0) <= n79;
n80 <= BU772_Q(0);
BU772_CLK <= n150;
BU772_CE <= n151;
BU772 : C_REG_FD_V7_0
    GENERIC MAP (
                c_width => 1,
                c_has_ce => 1,
                c_sinit_val => «0»,
                c_has_sinit => 0,
                c_ainit_val => «0»,
                c_has_aset => 0,
                c_sync_enable => 0,
                c_enable_rlocs => 0,
                c_has_aclr => 0,
                c_has_sset => 0,
                c_sync_priority => 0,
                c_has_ainit => 0,
                c_has_sclr => 0
                )
    PORT MAP (
                D => BU772_D,
                Q => BU772_Q,
                CLK => BU772_CLK,
                CE => BU772_CE
                );
--
BU776_I0 <= n79;
BU776_I1 <= n80;
BU776_I2 <= '0';
BU776_I3 <= '0';
n155 <= BU776_O;
BU776 : LUT4
```

```
    GENERIC MAP (
                INIT => X»2222»
                )
    PORT MAP (
            I0 => BU776_I0,
            I1 => BU776_I1,
            I2 => BU776_I2,
            I3 => BU776_I3,
            O => BU776_O
            );
```

Шестая часть содержит следующую совокупность операторов, которые описывают умножитель, выполненный на базе параметризированного модуля *Multiplier Generator* версии v7.0, и внутренние сигналы, назначаемые соответствующим интерфейсным портам этого компонента:

```
BU53_clk <= n150;
BU53_a(15) <= n97;
BU53_a(14) <= n96;
BU53_a(13) <= n95;
BU53_a(12) <= n94;
BU53_a(11) <= n93;
BU53_a(10) <= n92;
BU53_a(9) <= n91;
BU53_a(8) <= n90;
BU53_a(7) <= n89;
BU53_a(6) <= n88;
BU53_a(5) <= n87;
BU53_a(4) <= n86;
BU53_a(3) <= n85;
BU53_a(2) <= n84;
BU53_a(1) <= n83;
BU53_a(0) <= n82;
BU53_b(15) <= n113;
BU53_b(14) <= n112;
BU53_b(13) <= n111;
BU53_b(12) <= n110;
BU53_b(11) <= n109;
BU53_b(10) <= n108;
BU53_b(9) <= n107;
BU53_b(8) <= n106;
BU53_b(7) <= n105;
BU53_b(6) <= n104;
BU53_b(5) <= n103;
BU53_b(4) <= n102;
BU53_b(3) <= n101;
BU53_b(2) <= n100;
BU53_b(1) <= n99;
BU53_b(0) <= n98;
n33 <= BU53_q(31);
n32 <= BU53_q(30);
n31 <= BU53_q(29);
n30 <= BU53_q(28);
n29 <= BU53_q(27);
n28 <= BU53_q(26);
n27 <= BU53_q(25);
n26 <= BU53_q(24);
n25 <= BU53_q(23);
n24 <= BU53_q(22);
n23 <= BU53_q(21);
n22 <= BU53_q(20);
n21 <= BU53_q(19);
n20 <= BU53_q(18);
n19 <= BU53_q(17);
n18 <= BU53_q(16);
n17 <= BU53_q(15);
n16 <= BU53_q(14);
n15 <= BU53_q(13);
n14 <= BU53_q(12);
n13 <= BU53_q(11);
n12 <= BU53_q(10);
n11 <= BU53_q(9);
n10 <= BU53_q(8);
n9 <= BU53_q(7);
n8 <= BU53_q(6);
n7 <= BU53_q(5);
n6 <= BU53_q(4);
n5 <= BU53_q(3);
n4 <= BU53_q(2);
n3 <= BU53_q(1);
n2 <= BU53_q(0);
BU53_a_signed <= '0';
BU53_loadb <= '0';
BU53_swapb <= '0';
BU53_ce <= n151;
BU53_aclr <= '0';
BU53_sclr <= '0';
BU53_nd <= '0';
BU53 : mult_gen_v7_0
    GENERIC MAP (
                c_a_type => 0,
                c_mem_type => 0,
                c_has_sclr => 0,
                c_has_q => 1,
                c_reg_a_b_inputs => 1,
                c_has_o => 0,
                bram_addr_width => 8,
                c_v2_speed => 0,
                c_baat => 16,
                c_output_hold => 0,
                c_b_constant => 0,
                c_has_loadb => 0,
                c_has_b => 1,
                c_use_luts => 1,
                c_has_rdy => 0,
                c_has_nd => 0,
                c_pipeline => 1,
                c_has_a_signed => 0,
                c_b_type => 0,
                c_sqm_type => 0,
                c_standalone => 0,
                c_b_value => «0000000000000001»,
                c_enable_rlocs => 0,
                c_mult_type => 1,
                c_has_aclr => 0,
                c_mem_init_prefix => «mem»,
                c_has_load_done => 0,
                c_has_swapb => 0,
                c_out_width => 32,
                c_b_width => 16,
                c_a_width => 16,
                c_has_rfd => 0,
                c_sync_enable => 1,
                c_has_ce => 1,
                c_stack_adders => 0
                )
    PORT MAP (
                clk => BU53_clk,
                a => BU53_a,
                b => BU53_b,
                o => BU53_o,
                q => BU53_q,
                a_signed => BU53_a_signed,
                loadb => BU53_loadb,
                load_done => BU53_load_done,
                swapb => BU53_swapb,
                ce => BU53_ce,
                aclr => BU53_aclr,
                sclr => BU53_sclr,
                rfd => BU53_rfd,
                nd => BU53_nd,
                rdy => BU53_rdy
                );
```

Седьмая часть представляет собой описание аккумулятора, сгенерированного на базе параметризированного модуля *Accumulator*, и локальных сигналов, которые используются для сопряжения входных портов данного компонента с другими блоками умножителя-накопителя. В состав этой части описания элемента mac_16_24 входит следующая совокупность операторов:

```
BU256_B(0) <= n2;
BU256_B(1) <= n3;
BU256_B(2) <= n4;
BU256_B(3) <= n5;
BU256_B(4) <= n6;
BU256_B(5) <= n7;
BU256_B(6) <= n8;
BU256_B(7) <= n9;
BU256_B(8) <= n10;
BU256_B(9) <= n11;
BU256_B(10) <= n12;
BU256_B(11) <= n13;
BU256_B(12) <= n14;
BU256_B(13) <= n15;
BU256_B(14) <= n16;
BU256_B(15) <= n17;
BU256_B(16) <= n18;
BU256_B(17) <= n19;
BU256_B(18) <= n20;
BU256_B(19) <= n21;
BU256_B(20) <= n22;
BU256_B(21) <= n23;
BU256_B(22) <= n24;
BU256_B(23) <= n25;
BU256_B(24) <= n26;
BU256_B(25) <= n27;
BU256_B(26) <= n28;
BU256_B(27) <= n29;
BU256_B(28) <= n30;
BU256_B(29) <= n31;
BU256_B(30) <= n32;
BU256_B(31) <= n33;
n40 <= BU256_Q(0);
n41 <= BU256_Q(1);
n42 <= BU256_Q(2);
n43 <= BU256_Q(3);
n44 <= BU256_Q(4);
n45 <= BU256_Q(5);
n46 <= BU256_Q(6);
n47 <= BU256_Q(7);
n48 <= BU256_Q(8);
n49 <= BU256_Q(9);
n50 <= BU256_Q(10);
n51 <= BU256_Q(11);
n52 <= BU256_Q(12);
n53 <= BU256_Q(13);
n54 <= BU256_Q(14);
n55 <= BU256_Q(15);
n56 <= BU256_Q(16);
n57 <= BU256_Q(17);
n58 <= BU256_Q(18);
n59 <= BU256_Q(19);
n60 <= BU256_Q(20);
n61 <= BU256_Q(21);
n62 <= BU256_Q(22);
n63 <= BU256_Q(23);
n64 <= BU256_Q(24);
n65 <= BU256_Q(25);
n66 <= BU256_Q(26);
n67 <= BU256_Q(27);
n68 <= BU256_Q(28);
n69 <= BU256_Q(29);
n70 <= BU256_Q(30);
n71 <= BU256_Q(31);
n72 <= BU256_Q(32);
n73 <= BU256_Q(33);
n74 <= BU256_Q(34);
n75 <= BU256_Q(35);
n76 <= BU256_Q(36);
BU256_CLK <= n150;
BU256_BYPASS <= n38;
BU256_CE <= n151;
BU256_ACLR <= n152;
BU256 : C_ACCUM_V7_0
    GENERIC MAP (
                c_has_bypass_with_cin => 0,
                c_has_sclr => 0,
                c_has_b_out => 0,
                c_sync_priority => 0,
                c_has_aset => 0,
                c_has_s => 0,
                c_bypass_enable => 0,
                c_b_constant => 0,
                c_has_ovfl => 0,
                c_high_bit => 36,
                c_sinit_val => «000000000000000000000000000»,
                c_has_bypass => 1,
                c_pipe_stages => 0,
                c_has_sset => 0,
                c_has_ainit => 0,
                c_has_q_c_out => 0,
                c_b_type => 0,
                c_has_add => 0,
                c_has_sinit => 0,
                c_has_b_in => 0,
                c_has_b_signed => 0,
                c_bypass_low => 1,
                c_enable_rlocs => 0,
                c_b_value => «000000000000000000000000000»,
                c_add_mode => 0,
                c_has_aclr => 1,
                c_out_width => 37,
                c_saturate => 0,
                c_low_bit => 0,
                c_ainit_val => «000000000000000000000000000»,
                c_has_q_ovfl => 0,
                c_has_q_b_out => 0,
                c_has_c_out => 0,
                c_b_width => 32,
                c_scale => 0,
                c_sync_enable => 0,
                c_has_ce => 1,
                c_has_c_in => 0
                )
    PORT MAP (
                B => BU256_B,
                Q => BU256_Q,
                CLK => BU256_CLK,
                BYPASS => BU256_BYPASS,
                CE => BU256_CE,
                ACLR => BU256_ACLR
                );
```

Восьмую часть образует приведенная далее последовательность операторов, описывающих поведение сигналов и компоненты, применяемые для осуществления функции округления результатов выполнения операций умножения с накоплением до заданного количества двоичных разрядов. В составе этой части VHDL-описания используются компоненты, сформированные на базе параметризированных модулей сдвигового регистра *RAM-based Shift Register* и сумматора/вычитающего устройства *Adder/Subtracter*.

```
BU523_CLK <= n150;
BU523_D(0) <= n35;
n77 <= BU523_Q(0);
BU523_CE <= n151;
BU523 : C_SHIFT_RAM_V7_0
    GENERIC MAP (
                c_has_aset => 0,
                c_read_mif => 0,
                c_has_a => 0,
                c_sync_priority => 0,
                c_has_sclr => 0,
                c_width => 1,
                c_enable_rlocs => 0,
                c_default_data_radix => 2,
                c_generate_mif => 0,
                c_ainit_val => «0»,
                c_has_ce => 1,
                c_has_aclr => 0,
                c_mem_init_radix => 2,
                c_sync_enable => 0,
                c_depth => 3,
                c_has_ainit => 0,
                c_sinit_val => «0»,
                c_has_sset => 0,
                c_has_sinit => 0,
                c_shift_type => 0,
                c_mem_init_file => «null»,
                c_default_data => «0»,
                c_reg_last_bit => 1,
                c_addr_width => 1
                )
    ORT MAP (
            CLK => BU523_CLK,
            D => BU523_D,
            Q => BU523_Q,
            CE => BU523_CE
            );
--
BU531_I0 <= n151;
BU531_I1 <= n77;
BU531_I2 <= '0';
BU531_I3 <= '0';
n78 <= BU531_O;
BU531 : LUT4
    GENERIC MAP (
                INIT => X»8888»
                )
    PORT MAP (
            I0 => BU531_I0,
            I1 => BU531_I1,
            I2 => BU531_I2,
            I3 => BU531_I3,
            O => BU531_O
            );
--
BU538_I0 <= n41;
BU538_I1 <= n40;
BU538_I2 <= '0';
BU538_I3 <= '0';
n1055 <= BU538_O;
BU538 : LUT4
    GENERIC MAP (
                INIT => X»8888»
                )
    PORT MAP (
            I0 => BU538_I0,
            I1 => BU538_I1,
            I2 => BU538_I2,
            I3 => BU538_I3,
            O => BU538_O
            );
--
BU540_A(0) <= n41;
BU540_A(1) <= n42;
BU540_A(2) <= n43;
BU540_A(3) <= n44;
BU540_A(4) <= n45;
BU540_A(5) <= n46;
BU540_A(6) <= n47;
BU540_A(7) <= n48;
BU540_A(8) <= n49;
BU540_A(9) <= n50;
BU540_A(10) <= n51;
BU540_A(11) <= n52;
BU540_A(12) <= n53;
BU540_A(13) <= n54;
BU540_A(14) <= n55;
BU540_A(15) <= n56;
BU540_A(16) <= n57;
BU540_A(17) <= n58;
BU540_A(18) <= n59;
BU540_A(19) <= n60;
BU540_A(20) <= n61;
BU540_A(21) <= n62;
BU540_A(22) <= n63;
BU540_A(23) <= n64;
BU540_A(24) <= n65;
BU540_A(25) <= n66;
BU540_A(26) <= n67;
BU540_A(27) <= n68;
BU540_A(28) <= n69;
BU540_A(29) <= n70;
BU540_A(30) <= n71;
BU540_A(31) <= n72;
BU540_A(32) <= n73;
BU540_A(33) <= n74;
BU540_A(34) <= n75;
BU540_A(35) <= n76;
BU540_B(0) <= n1055;
n1018 <= BU540_S(0);
n1019 <= BU540_S(1);
n1020 <= BU540_S(2);
n1021 <= BU540_S(3);
n1022 <= BU540_S(4);
n1023 <= BU540_S(5);
n1024 <= BU540_S(6);
n1025 <= BU540_S(7);
n1026 <= BU540_S(8);
n1027 <= BU540_S(9);
n1028 <= BU540_S(10);
n1029 <= BU540_S(11);
n1030 <= BU540_S(12);
n1031 <= BU540_S(13);
n1032 <= BU540_S(14);
n1033 <= BU540_S(15);
n1034 <= BU540_S(16);
n1035 <= BU540_S(17);
n1036 <= BU540_S(18);
n1037 <= BU540_S(19);
n1038 <= BU540_S(20);
n1039 <= BU540_S(21);
n1040 <= BU540_S(22);
n1041 <= BU540_S(23);
n1042 <= BU540_S(24);
n1043 <= BU540_S(25);
n1044 <= BU540_S(26);
n1045 <= BU540_S(27);
n1046 <= BU540_S(28);
n1047 <= BU540_S(29);
n1048 <= BU540_S(30);
n1049 <= BU540_S(31);
n1050 <= BU540_S(32);
n1051 <= BU540_S(33);
n1052 <= BU540_S(34);
n1053 <= BU540_S(35);
BU540 : C_ADDSUB_V7_0
    GENERIC MAP (
                c_has_bypass_with_cin => 0,
                c_a_type => 0,
                c_has_sclr => 0,
                c_has_aset => 0,
                c_has_b_out => 0,
                c_sync_priority => 1,
                c_has_s => 1,
                c_has_q => 0,
                c_bypass_enable => 0,
                c_b_constant => 0,
                c_has_ovfl => 0,
                c_high_bit => 36,
                c_latency => 1,
                c_sinit_val => «000000000000000000000000000»,
                c_has_bypass => 0,
                c_pipe_stages => 0,
                c_has_sset => 0,
                c_has_ainit => 0,
                c_has_a_signed => 0,
                c_has_q_c_out => 0,
                c_b_type => 1,
                c_has_add => 0,
                c_has_sinit => 0,
                c_has_b_in => 0,
                c_has_b_signed => 0,
                c_bypass_low => 0,
                c_enable_rlocs => 0,
                c_b_value => «000000000000000000000000000»,
                c_add_mode => 0,
                c_has_aclr => 0,
                c_out_width => 37,
                c_low_bit => 0,
                c_ainit_val => «000000000000000000000000000»,
                c_has_q_ovfl => 0,
                c_has_q_b_out => 0,
                c_has_c_out => 0,
                c_b_width => 1,
                c_a_width => 36,
                c_sync_enable => 0,
                c_has_ce => 0,
                c_has_c_in => 0
                )
    PORT MAP (
                A => BU540_A,
                B => BU540_B,
                S => BU540_S
                );
```

Заключительная, девятая, часть сформированного VHDL-описания умножителя-накопителя содержит последовательность операторов, представляющих выходной регистр и сигналы, соответствующие его входным и выходным портам. В качестве выходного регистра применяется компонент, сгенерированный на основе параметризированного модуля параллельного регистра *FD-based Parallel Register*. Текст заключительной части описания элемента mac_16_24 выглядит следующим образом:

```
BU690_D(0) <= n1018;
BU690_D(1) <= n1019;
BU690_D(2) <= n1020;
BU690_D(3) <= n1021;
BU690_D(4) <= n1022;
BU690_D(5) <= n1023;
BU690_D(6) <= n1024;
BU690_D(7) <= n1025;
BU690_D(8) <= n1026;
BU690_D(9) <= n1027;
BU690_D(10) <= n1028;
BU690_D(11) <= n1029;
BU690_D(12) <= n1030;
BU690_D(13) <= n1031;
BU690_D(14) <= n1032;
BU690_D(15) <= n1033;
BU690_D(16) <= n1034;
BU690_D(17) <= n1035;
BU690_D(18) <= n1036;
BU690_D(19) <= n1037;
BU690_D(20) <= n1038;
BU690_D(21) <= n1039;
BU690_D(22) <= n1040;
BU690_D(23) <= n1041;
BU690_D(24) <= n1042;
BU690_D(25) <= n1043;
BU690_D(26) <= n1044;
BU690_D(27) <= n1045;
BU690_D(28) <= n1046;
BU690_D(29) <= n1047;
BU690_D(30) <= n1048;
BU690_D(31) <= n1049;
BU690_D(32) <= n1050;
BU690_D(33) <= n1051;
BU690_D(34) <= n1052;
BU690_D(35) <= n1053;
n114 <= BU690_Q(0);
n115 <= BU690_Q(1);
n116 <= BU690_Q(2);
n117 <= BU690_Q(3);
n118 <= BU690_Q(4);
n119 <= BU690_Q(5);
n120 <= BU690_Q(6);
n121 <= BU690_Q(7);
n122 <= BU690_Q(8);
n123 <= BU690_Q(9);
n124 <= BU690_Q(10);
n125 <= BU690_Q(11);
n126 <= BU690_Q(12);
n127 <= BU690_Q(13);
n128 <= BU690_Q(14);
n129 <= BU690_Q(15);
n130 <= BU690_Q(16);
n131 <= BU690_Q(17);
n132 <= BU690_Q(18);
n133 <= BU690_Q(19);
n134 <= BU690_Q(20);
```

```
n135 <= BU690_Q(21);
n136 <= BU690_Q(22);
n137 <= BU690_Q(23);
n138 <= BU690_Q(24);
n139 <= BU690_Q(25);
n140 <= BU690_Q(26);
n141 <= BU690_Q(27);
n142 <= BU690_Q(28);
n143 <= BU690_Q(29);
n144 <= BU690_Q(30);
n145 <= BU690_Q(31);
n146 <= BU690_Q(32);
n147 <= BU690_Q(33);
n148 <= BU690_Q(34);
n149 <= BU690_Q(35);
BU690_CLK <= n150;
BU690_CE <= n78;
BU690 : C_REG_FD_V7_0
    GENERIC MAP (
                c_width => 36,
                c_has_ce => 1,
                c_sinit_val => «000000000000000000000000»,
                c_has_sinit => 0,
                c_ainit_val => «000000000000000000000000»,
                c_has_aset => 0,
                c_sync_enable => 0,
                c_enable_rlocs => 0,
                c_has_aclr => 0,
                c_has_sset => 0,
                c_sync_priority => 0,
                c_has_ainit => 0,
                c_has_sclr => 0
                )
    PORT MAP (
                D => BU690_D,
                Q => BU690_Q,
                CLK => BU690_CLK,
                CE => BU690_CE
                );
END xilinx;
```

Для декларации сформированного умножителя-накопителя в составе VHDL-описания разрабатываемого устройства нужно использовать следующую последовательность выражений:

```
component mac_16_24
    port (
            A: IN std_logic_VECTOR(15 downto 0);
            B: IN std_logic_VECTOR(15 downto 0);
            Q: OUT std_logic_VECTOR(35 downto 0);
            CLK: IN std_logic;
            CE: IN std_logic;
            RST: IN std_logic;
            FD: IN std_logic;
            ND: IN std_logic;
            RDY: OUT std_logic
            );
end component;
--
-- FPGA Express Black Box declaration
attribute fpga_dont_touch: string;
attribute fpga_dont_touch of mac_16_24: component is «true»;
--

-- Synplicity black box declaration
attribute syn_black_box : boolean;
attribute syn_black_box of mac_16_24: component is true;
```

Описание каждого экземпляра умножителя-накопителя типа mac_16_24 в составе проектируемого устройства выполняется с помощью оператора, шаблон которого имеет следующий вид:

```
<идентификатор_экземпляра_умножителя-накопителя_mac_16_24> : mac_16_24
    port map (
                A => A,
                B => B,
                Q => Q,
                CLK => CLK,
                CE => CE,
                RST => RST,
                FD => FD,
                ND => ND,
                RDY => RDY
                );
```

Продолжение следует