Продолжение. Начало в № 2`2010

Разработка VHDL-описаний цифровых устройств, проектируемых на основе ПЛИС фирмы Xilinx, с использованием шаблонов САПР ISE Design Suite

Валерий ЗОТОВ
walerry@km.ru

В двадцать третьей части статьи завершается изучение шаблонов VHDL-описаний элементов, выполненных на основе экземпляров библиотечных примитивов, которые реализуются на базе соответствующих аппаратных ресурсов ПЛИС серии Virtex-4. В этой части рассмотрены образцы описаний запоминающих устройств FIFO, конфигурируемых на основе модулей блочной памяти Block RAM кристаллов программируемой логики семейств Virtex-4 LX, Virtex-4 SX и Virtex-4 FX [34–43]. Здесь же приведена информация о шаблонах описаний компонентов, предназначенных для применения в составе проектов, реализуемых на базе ПЛИС серии Virtex-5.

Каждый модуль блочной памяти ПЛИС серии Virtex-4 может конфигурироваться не только как двухпортовое или однопортовое ОЗУ, но и в виде запоминающего устройства, работающего по принципу «первым вошел — первым вышел» (first-in first-out, FIFO) с различной организацией. Для подготовки описаний элементов FIFO-памяти предусмотрен библиотечный примитив ***FIFO16***. На основе экземпляра этого библиотечного примитива выполнены четыре шаблона описаний элементов запоминающих устройств FIFO с различными вариантами организации входных и выходных портов. В библиотечном примитиве ***FIFO16*** используются следующие параметры настройки:

- ALMOST_FULL_OFFSET — указывает число слов данных, на которое отличаются условия формирования активного уровня сигнала на выходах EMPTY и ALMOSTEMPTY (по умолчанию предлагается значение X“080”).
- ALMOST_EMPTY_OFFSET — задает количество слов данных, на которое должны отличаться условия формирования активного уровня сигнала на выходах FULL и ALMOSTFULL (по умолчанию присваивается значение X“080”).
- DATA_WIDTH — определяет разрядность входного и выходного портов формируемого экземпляра FIFO-памяти (по умолчанию установлено значение, соответствующее выбранному варианту организации портов записи и чтения информации).
- FIRST_WORD_FALL_THROUGH — предоставляет возможность использования в создаваемом элементе режима FWFT (по умолчанию для этого параметра предлагается значение, запрещающее применение указанной функции).

В состав системы условных обозначений входов и выходов, используемых в описании интерфейса элементов FIFO-памяти, формируемых на основе библиотечного примитива ***FIFO16***, входят следующие идентификаторы:

- ALMOSTEMPTY — выход сигнала, сообщающего о том, что не более заданного количества слов данных может быть считано из сформированного запоминающего устройства FIFO.
- ALMOSTFULL — выход сигнала, предупреждающего о том, что в элемент FIFO-памяти может быть записано не более указанного количества слов данных.
- DO — выходная шина данных с разрядностью, определяемой значением параметра DATA_WIDTH, на которую выводится информация, считываемая из запоминающего устройства FIFO.
- DOP — выходная шина, используемая для контроля четности считываемых информационных данных.
- EMPTY — выход сигнала, активный уровень которого информирует об опустошении запоминающего устройства (всех ячеек FIFO-памяти) и невозможности осуществления операции чтения новых данных.
- FULL — выход сигнала, активный уровень которого предупреждает о заполнении всего объема запоминающего устройства FIFO (всех ячеек FIFO-памяти) и невозможности осуществления операции записи новых данных.
- RDCOUNT — выходная шина, на которую выводится текущее значение счетчика количества считанных слов данных.
- RDERR — выход сигнала, активный уровень которого информирует об ошибке при выполнении операции чтения слова данных из элемента FIFO-памяти.
- WRCOUNT — выходная шина, на которой отображается текущее значение счетчика числа записанных слов данных.
- WRERR — выход сигнала, информирующего об ошибке при выполнении операции записи слова данных в запоминающее устройство FIFO.
- DI — входная шина с разрядностью, определяемой значением параметра DATA_WIDTH, на которую поступают информационные данные, записываемые в запоминающее устройство FIFO.
- DIP — входная шина, используемая для контроля четности записываемых информационных данных.
- RDCLK — вход сигнала синхронизации для порта чтения данных из элемента FIFO-памяти.
- RDEN — вход сигнала, разрешающего выполнение операции чтения (извлечения) данных из запоминающего устройства FIFO.
- RST — вход сигнала сброса.

- WRCLK — вход сигнала синхронизации для порта записи данных в элемент запоминающего устройства FIFO.
- WREN — вход сигнала разрешения операции записи данных в элемент FIFO-памяти.

1k x 18 (FIFO16) представляет собой шаблон VHDL-описания элемента запоминающего устройства, функционирующего по принципу «первым вошел — первым вышел», с организацией входного и выходного портов 1024×18 разрядов, реализуемого на основе модуля блочной памяти Block RAM в кристаллах программируемой логики семейств Virtex-4 LX, Virtex-4 SX и Virtex-4 FX:

```
-- FIFO16 : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the architecture body of the design code. The
-- declaration : instance name (FIFO16_inst) and/or the port declarations
-- code : after the "=>" assignment maybe changed to properly
-- : connect this function to the design. All inputs
-- : and outputs must be connected.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
   -- FIFO16: BlockRAM Asynchronous FIFO configured for 1k deep
x 18 wide
   -- Virtex-4
   -- Xilinx HDL Language Template, version 12.4
--
   FIFO16_inst : FIFO16
   generic map (
         ALMOST_FULL_OFFSET => X"080", -- Sets almost full
threshold
         ALMOST_EMPTY_OFFSET => X"080", -- Sets the almost
empty threshold
         DATA_WIDTH => 18, -- Sets data width to 4, 9, 18, or 36
         FIRST_WORD_FALL_THROUGH => FALSE) -- Sets the FIFO
FWFT to TRUE or FALSE
   port map (
         ALMOSTEMPTY => ALMOSTEMPTY, -- 1-bit almost empty
output flag
         ALMOSTFULL => ALMOSTFULL, -- 1-bit almost full output flag
         DO (31 DOWNTO 16) => unconnected (15 downto 0), --
Unused data output. Unconnected is a signal of 18 bits
         DO (15 DOWNTO 0) => DO, -- 16-bit data output
         DOP (3 DOWNTO 2) => unconnected (17 downto 16), --
Unused parity data output. Unconnected is a signal of 18 bits
         DOP (1 DOWNTO 0) => DOP, -- 2-bit parity data output
         EMPTY => EMPTY, -- 1-bit empty output flag
         FULL => FULL, -- 1-bit full output flag
         RDCOUNT => RDCOUNT, -- 12-bit read count output
         RDERR => RDERR, -- 1-bit read error output
         WRCOUNT => WRCOUNT, -- 12-bit write count output
         WRERR => WRERR, -- 1-bit write error
         DI (31 DOWNTO 16) => X"0000", -- Unused data inputs tied
to ground
         DI (15 downto 0) => DI, -- 16-bit data input
         DIP (3 DOWNTO 2) => "00", -- Unused parity inputs tied to
ground
         DIP (1 downto 0) => DIP, -- 2-bit parity input
         RDCLK => RDCLK, -- 1-bit read clock input
         RDEN => RDEN, -- 1-bit read enable input
         RST => RST, -- 1-bit reset input
         WRCLK => WRCLK, -- 1-bit write clock input
         WREN => WREN -- 1-bit write enable input
      );
   -- End of FIFO16_inst instantiation
```

На рис. 363 представлен условный графический образ элемента FIFO-памяти, описание которого формируется с помощью шаблона ***1k x 18 (FIFO16)*** для последующей реализации на базе модуля блочной памяти Block RAM ПЛИС серии Virtex-4.

2k x 9 (FIFO16) включает в себя образец VHDL-описания варианта конфигурирования модуля блочной памяти Block RAM кристаллов программируемой логики семейств Virtex-4 LX, Virtex-4 SX и Virtex-4 FX в виде запоминающего устройства FIFO с организацией портов записи и чтения данных 2048×9 разрядов:

```
-- FIFO16 : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the architecture body of the design code. The
-- declaration : instance name (FIFO16_inst) and/or the port declarations
-- code : after the "=>" assignment maybe changed to properly
-- : connect this function to the design. All inputs
-- : and outputs must be connected.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
   -- FIFO16: BlockRAM Asynchronous FIFO configuted for 2k deep
x 9 wide
   -- Virtex-4
   -- Xilinx HDL Language Template, version 12.4
--
   FIFO16_inst : FIFO16
   generic map (
         ALMOST_FULL_OFFSET => X"080", -- Sets almost full
threshold
         ALMOST_EMPTY_OFFSET => X"080", -- Sets the almost
empty threshold
         DATA_WIDTH => 9, -- Sets data width to 4, 9, 18, or 36
      T_WORD_FALL_THROUGH => FALSE) --Sets the FIFO FWFT
to TRUE or FALSE
   port map (
         ALMOSTEMPTY => ALMOSTEMPTY, -- 1-bit almost empty
output flag
         ALMOSTFULL => ALMOSTFULL, -- 1-bit almost full output flag
         DO (31 DOWNTO 8) => unconnected (23 downto 0), --
Unused data output. Unconnected is a signal of 27 bits
         DO (7 DOWNTO 0) => DO, -- 8-bit data output
         DOP (3 DOWNTO 1) => unconnected (26 downto 24), --
Unused parity data output. Unconnected is a signal of 27 bits
         DOP (0) => DOP, -- 1-bit parity data output
         EMPTY => EMPTY, -- 1-bit empty output flag
         FULL => FULL, -- 1-bit full output flag
         RDCOUNT => RDCOUNT, -- 12-bit read count output
         RDERR => RDERR, -- 1-bit read error output
         WRCOUNT => WRCOUNT, -- 12-bit write count output
         WRERR => WRERR, -- 1-bit write error
         DI (31 DOWNTO 8) => X"000000", -- Unused data inputs tied
to ground
         DI (7 downto 0) => DI, -- 8-bit data input
         DIP (3 DOWNTO 1) => "000", -- Unused parity inputs tied to
ground
         DIP (0) => DIP, -- 1-bit parity input
         RDCLK => RDCLK, -- 1-bit read clock input
         RDEN => RDEN, -- 1-bit read enable input
         RST => RST, -- 1-bit reset input
         WRCLK => WRCLK, -- 1-bit write clock input
         WREN => WREN -- 1-bit write enable input
      );
   -- End of FIFO16_inst instantiation
```

Условный графический образ элемента FIFO-памяти, формируемого с помощью шаблона ***2k x 9 (FIFO16)***, отличается от символа, представленного на рис. 363, только разрядностью входных и выходных шин данных.

4k x 4 (FIFO16) является шаблоном описания элемента FIFO-памяти с организацией входного и выходного портов 4096×4 разряда, реализуемого на основе модуля блочной памяти Block RAM ПЛИС серии Virtex-4:

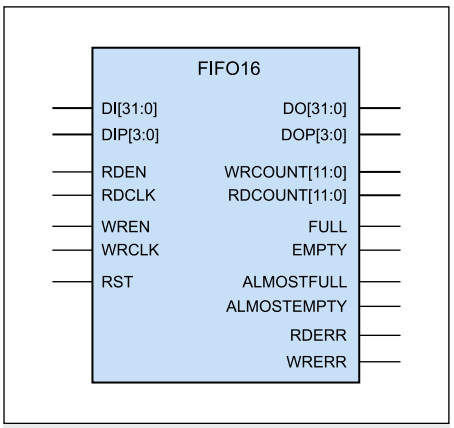

Рис. 363. Условный графический образ элемента FIFO-памяти, формируемого с помощью шаблона 1k x 18 (FIFO16)

```
-- FIFO16 : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the architecture body of the design code. The
-- declaration : instance name (FIFO16_inst) and/or the port declarations
-- code : after the "=>" assignment maybe changed to properly
-- : connect this function to the design. All inputs
-- : and outputs must be connected.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
   -- FIFO16: BlockRAM Asynchronous FIFO configured fro 4k deep x
4 wide
   -- Virtex-4
   -- Xilinx HDL Language Template, version 12.4
--
   FIFO16_inst : FIFO16
   generic map (
         ALMOST_FULL_OFFSET => X"080", -- Sets almost full
threshold
         ALMOST_EMPTY_OFFSET => X"080", -- Sets the almost
empty threshold
         DATA_WIDTH => 4, -- Sets data width to 4, 9, 18, or 36
         FIRST_WORD_FALL_THROUGH => FALSE) --Sets the FIFO
FWFT to TRUE or FALSE
   port map (
         ALMOSTEMPTY => ALMOSTEMPTY, -- 1-bit almost empty
output flag
         ALMOSTFULL => ALMOSTFULL, -- 1-bit almost full output flag
         DO (31 DOWNTO 4) => unconnected (27 downto 0), --
Unused data output. Unconnected is a signal of 32 bits
         DO (3 DOWNTO 0) => DO, -- 4-bit data output
         DOP => unconnected (31 downto 28), -- 4-bit Unused parity
data output. Unconnected is a signal of 32 bits
         EMPTY => EMPTY, -- 1-bit empty output flag
         FULL => FULL, -- 1-bit full output flag
         RDCOUNT => RDCOUNT, -- 12-bit read count output
         RDERR => RDERR, -- 1-bit read error output
         WRCOUNT => WRCOUNT, -- 12-bit write count output
         WRERR => WRERR, -- 1-bit write error
         DI (31 DOWNTO 4) => X"0000000", -- Unused data inputs
tied to ground
         DI (3 downto 0) => DI, -- 4-bit data input
         DIP => X"0", -- 4-bit Unused parity inputs tied to ground
         RDCLK => RDCLK, -- 1-bit read clock input
         RDEN => RDEN, -- 1-bit read enable input
         RST => RST, -- 1-bit reset input
         WRCLK => WRCLK, -- 1-bit write clock input
         WREN => WREN -- 1-bit write enable input
      );
   -- End of FIFO16_inst instantiation
```

512 x 36 (FIFO16) содержит образец VHDL-описания варианта конфигурирования модуля блочной памяти Block RAM кристаллов программируемой логики семейств Virtex-4 LX, Virtex-4 SX и Virtex-4 FX в виде запоминающего устройства, работающего по принципу «первым вошел — первым вышел», с организацией портов записи и чтения данных 512×36 разрядов:

```
-- FIFO16 : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the architecture body of the design code. The
-- declaration : instance name (FIFO16_inst) and/or the port declarations
-- code : after the "=>" assignment maybe changed to properly
-- : connect this function to the design. All inputs
-- : and outputs must be connected.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
   -- FIFO16: BlockRAM Asynchrnous FIFO configured for 512 deep
x 36 wide
   -- Virtex-4
   -- Xilinx HDL Language Template, version 12.4
--
   FIFO16_inst : FIFO16
   generic map (
         ALMOST_FULL_OFFSET => X"080", -- Sets almost full
threshold
         ALMOST_EMPTY_OFFSET => X"080", -- Sets the almost
empty threshold
         DATA_WIDTH => 36, -- Sets data width to 4, 9, 18, or 36
         FIRST_WORD_FALL_THROUGH => FALSE) -- Sets the FIFO
FWFT to TRUE or FALSE
   port map (
         ALMOSTEMPTY => ALMOSTEMPTY, -- 1-bit almost empty
output flag
         ALMOSTFULL => ALMOSTFULL, -- 1-bit almost full output flag
         DO => DO, -- 32-bit data output
         DOP => DOP, -- 4-bit parity data output
         EMPTY => EMPTY, -- 1-bit empty output flag
         FULL => FULL, -- 1-bit full output flag
         RDCOUNT => RDCOUNT, -- 12-bit read count output
         RDERR => RDERR, -- 1-bit read error output
         WRCOUNT => WRCOUNT, -- 12-bit write count output
         WRERR => WRERR, -- 1-bit write error
         DI => DI, -- 32-bit data input
         DIP => DIP, -- 4-bit parity input
         RDCLK => RDCLK, -- 1-bit read clock input
         RDEN => RDEN, -- 1-bit read enable input
         RST => RST, -- 1-bit reset input
         WRCLK => WRCLK, -- 1-bit write clock input
         WREN => WREN -- 1-bit write enable input
      );
   -- End of FIFO16_inst instantiation
```

Несколько элементов FIFO-памяти, формируемых с помощью рассмотренных шаблонов, могут соединяться соответствующим образом для повышения разрядности входного и выходного портов. В качестве примера на рис. 364 изображена структурная схема запоминающего устройства, функционирующего по принципу «первым вошел — первым вышел», с организацией портов записи и чтения данных 512×72 разряда, выполненная на основе двух элементов FIFO-памяти, для подготовки описаний которых используется шаблон ***512 x 36 (FIFO16)***.

Для увеличения объема запоминающего устройства применяется каскадное соединение нескольких элементов FIFO-памяти.

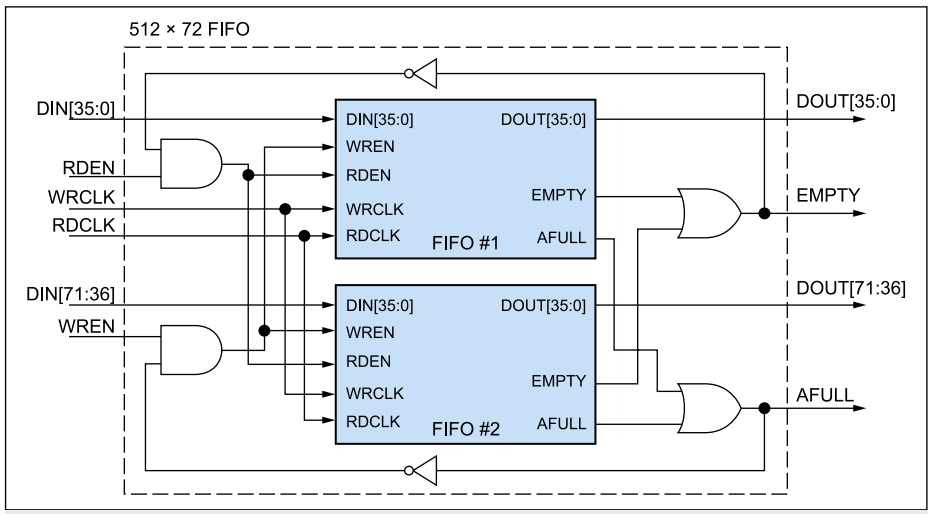

Рис. 364. Структура элемента FIFO-памяти с организацией 512×72 разряда

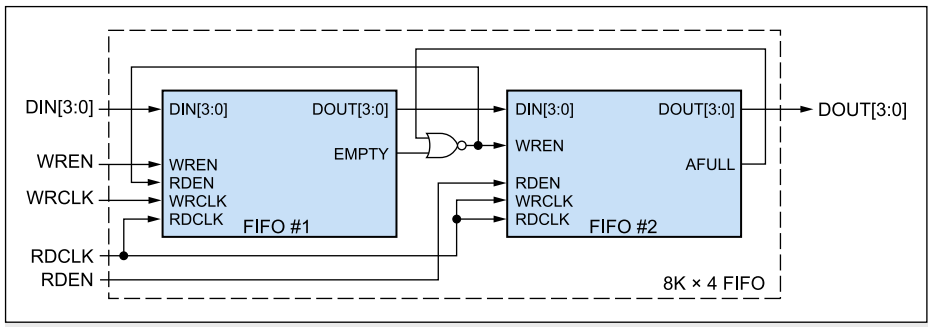

Рис. 365. Схема каскадного соединения двух элементов FIFO-памяти с организацией 4096×4 разряда

Пример такого соединения демонстрирует структурная схема запоминающего устройства FIFO с организацией входного и выходного портов 8192×4 разряда, выполненная на базе двух элементов, формируемых с помощью шаблона ***4k x 4 (FIFO16)*** (рис. 365).

Раздел ***Virtex-5*** каталога ***Device Primitive Instantiation*** шаблонов языка VHDL объединяет в себе образцы описаний компонентов, выполненных на основе экземпляров библиотечных примитивов, которые реализуются на базе соответствующих аппаратных ресурсов ПЛИС одноименной серии [44–54]. Структура этого раздела в основном повторяет структуру каталога ***Spartan-3*** (рис. 51, КиТ № 8 `2010, стр. 101). Большинство шаблонов, сосредоточенных в подразделах каталога ***Virtex-5***, включают в себя те же конструкции, что и одноименные образцы описаний, расположенные в папках ***Spartan-3***, ***Spartan-3A***, ***Spartan-3A DSP***, ***Spartan-3E***, ***Spartan-6*** и ***Virtex-4***, которые были рассмотрены ранее. Поэтому далее приводится информация о тех шаблонах, которые не были представлены в предыдущих частях статьи.

DSP Block (DSP48E) предоставляет шаблон описания элементов, формируемых на базе аппаратных секций цифровой обработки сигналов DSP48E, которые представлены в составе архитектуры кристаллов программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT. Структура и функциональные возможности этих секций имеют существенные отличия по сравнению с аппаратными модулями цифровой обработки сигналов DSP48 A и DSP48 A1, входящими в состав ПЛИС семейств Spartan-3 A DSP, Spartan-6 LX и Spartan-6 LXT, образцы описаний которых были рассмотрены ранее. Аппаратные секции цифровой обработки сигналов DSP48E представляют собой результат дальнейшего развития архитектуры модулей ЦОС DSP48, используемых в кристаллах программируемой логики семейств Virtex-4 LX, Virtex-4 SX и Virtex-4 FX. Наиболее заметные особенности секций DSP48E:

- применение аппаратного умножителя 25×18, позволяющего вычислять произведение 25- и 18-разрядных значений входных данных;
- поддержка свыше 40 различных режимов (реализуемых функций), выбираемых разработчиком;
- оптимизированная архитектура, предоставляющая возможность эффективной реализации высокопроизводительных цифровых фильтров и выполнения операций комплексного умножения;
- возможность применения многоступенчатой конвейерной организации выполнения операций, повышающей производитель-

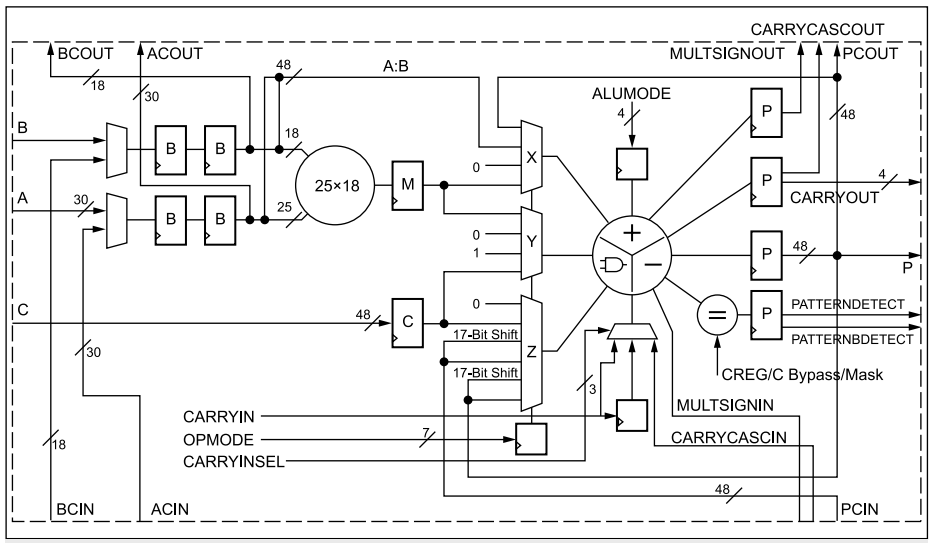

Рис. 366. Структурная схема аппаратной секции ЦОС DSP48E, применяемой в ПЛИС серии Virtex-5

ность разрабатываемых устройств цифровой обработки сигналов;
- поддержка поразрядных логических операций;
- возможность реализации трехвходовых сумматоров и вычитающих устройств;
- поддержка динамического управления режимами работы функциональных блоков модуля;
- возможность выполнения арифметических функций без использования умножителя;
- низкий уровень потребляемой мощности (для каждой секции DSP48E это значение составляет 1,38 мВт/100 МГц).

Подробная структурная схема аппаратной секции цифровой обработки сигналов DSP48E, применяемой в ПЛИС серии Virtex-5, показана на рис. 366.

В состав представленной структуры входят следующие основные элементы:
- аппаратный умножитель 25×18;
- арифметическо-логический блок (АЛБ);
- селекторы-мультиплексоры, осуществляющие коммутацию сигналов на входах аппаратного умножителя 25×18 и АЛБ;
- регистры, предназначенные для реализации конвейерной организации выполнения операций.

Шаблон описания элементов, формируемых на базе аппаратных секций цифровой обработки сигналов, предназначенных для использования в составе проектов, реализуемых на основе кристаллов программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT, выполнен на основе экземпляра библиотечного примитива ***DSP48E***:

```
-- DSP48E : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the body of the design code. The instance name
-- declaration : (DSP48E_inst) and/or the port declarations after the
-- code : "=>" declaration maybe changed to properly reference and
-- : connect this function to the design. All inputs and outputs
-- : must be connected.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
   -- DSP48E: DSP Function Block
   -- Virtex-5
   -- Xilinx HDL Language Template, version 13.2
--
   DSP48E_inst : DSP48E
   generic map (
         ACASCREG => 1, -- Number of pipeline registers between A/
ACIN input and ACOUT output, 0, 1, or 2
         ALUMODEREG => 1, -- Number of pipeline registers on
ALUMODE input, 0 or 1
         AREG => 1, -- Number of pipeline registers on the A input, 0,
1 or 2
         AUTORESET_PATTERN_DETECT => FALSE, -- Auto-reset
upon pattern detect, TRUE or FALSE
         AUTORESET_PATTERN_DETECT_OPTINV => "MATCH",
-- Reset if "MATCH" or "NOMATCH"
         A_INPUT => "DIRECT", -- Selects A input used, "DIRECT"
(A port) or "CASCADE" (ACIN port)
         BCASCREG => 1, -- Number of pipeline registers between B/
BCIN input and BCOUT output, 0, 1, or 2
          BREG => 1, -- Number of pipeline registers on the B input, 0,
1 or 2
         B_INPUT => "DIRECT", -- Selects B input used, "DIRECT"
(B port) or "CASCADE" (BCIN port)
         CARRYINREG => 1, -- Number of pipeline registers for the
CARRYIN input, 0 or 1
         CARRYINSELREG => 1, -- Number of pipeline registers for
the CARRYINSEL input, 0 or 1
         CREG => 1, -- Number of pipeline registers on the C input,
0 or 1
         MASK => X"3FFFFFFFFFFF", -- 48-bit Mask value for pattern
detect
         MREG => 1, -- Number of multiplier pipeline registers, 0 or 1
         MULTCARRYINREG => 1, -- Number of pipeline registers for
multiplier carry in bit, 0 or 1
         OPMODEREG => 1, -- Number of pipeline registers on
OPMODE input, 0 or 1
         PATTERN => X"000000000000", -- 48-bit Pattern match for
pattern detect
         PREG => 1, -- Number of pipeline registers on the P output, 0
or 1
       SIM_MODE => "SAFE", -- Simulation: "SAFE" vs "FAST", see
"Synthesis and Simulation
                  -- Design Guide" for details
         SEL_MASK => "MASK", -- Select mask value between the
"MASK" value or the value on the "C" port
         SEL_PATTERN => "PATTERN", -- Select pattern value between
the "PATTERN" value or the value on the "C" port
         SEL_ROUNDING_MASK => "SEL_MASK", -- "SEL_MASK",
"MODE1", "MODE2"
         USE_MULT => "MULT_S", -- Select multiplier usage, "MULT"
(MREG => 0),
                  -- "MULT_S" (MREG => 1), "NONE" (not using
multiplier)
         USE_PATTERN_DETECT => "NO_PATDET", -- Enable
pattern detect, "PATDET", "NO_PATDET"
          USE_SIMD => "ONE48") -- SIMD selection, "ONE48",
"TWO24", "FOUR12"
     port map (
         ACOUT => ACOUT, -- 30-bit A port cascade output
         BCOUT => BCOUT, -- 18-bit B port cascade output
         CARRYCASCOUT => CARRYCASCOUT, -- 1-bit cascade
carry output
         CARRYOUT => CARRYOUT, -- 4-bit carry output
         MULTSIGNOUT => MULTSIGNOUT, -- 1-bit multiplier sign
cascade output
         OVERFLOW => OVERFLOW, -- 1-bit overflow in add/acc
output
         P => P, -- 48-bit output
         PATTERNBDETECT => PATTERNBDETECT, -- 1-bit active
high pattern bar detect output
         PATTERNDETECT => PATTERNDETECT, -- 1-bit active high
pattern detect output
         PCOUT => PCOUT, -- 48-bit cascade output
         UNDERFLOW => UNDERFLOW, -- 1-bit active high
underflow in add/acc output
         A => A, -- 30-bit A data input
         ACIN => ACIN, -- 30-bit A cascade data input
         ALUMODE => ALUMODE, -- 4-bit ALU control input
         B => B, -- 18-bit B data input
         BCIN => BCIN, -- 18-bit B cascade input
         C => C, -- 48-bit C data input
         CARRYCASCIN => CARRYCASCIN, -- 1-bit cascade carry
input
         CARRYIN => CARRYIN, -- 1-bit carry input signal
         CARRYINSEL => CARRYINSEL, -- 3-bit carry select input
         CEA1 => CEA1, -- 1-bit active high clock enable input for 1st
stage A registers
         CEA2 => CEA2, -- 1-bit active high clock enable input for 2nd
stage A registers
         CEALUMODE => CEALUMODE, -- 1-bit active high clock
enable input for ALUMODE registers
         CEB1 => CEB1, -- 1-bit active high clock enable input for 1st
stage B registers
         CEB2 => CEB2, -- 1-bit active high clock enable input for 2nd
stage B registers
         CEC => CEC, -- 1-bit active high clock enable input for C
registers
         CECARRYIN => CECARRYIN, -- 1-bit active high clock
enable input for CARRYIN register
         CECTRL => CECTRL, -- 1-bit active high clock enable input
for OPMODE and carry registers
         CEM => CEM, -- 1-bit active high clock enable input for
multiplier registers
         CEMULTCARRYIN => CEMULTCARRYIN, -- 1-bit active
high clock enable for multiplier carry in register
         CEP => CEP, -- 1-bit active high clock enable input for P
registers
         CLK => CLK, -- Clock input
         MULTSIGNIN => MULTSIGNIN, -- 1-bit multiplier sign input
         OPMODE => OPMODE, -- 7-bit operation mode input
         PCIN => PCIN, -- 48-bit P cascade input
         RSTA => RSTA, -- 1-bit reset input for A pipeline registers
         RSTALLCARRYIN => RSTALLCARRYIN, -- 1-bit reset input
for carry pipeline registers
         RSTALUMODE => RSTALUMODE, -- 1-bit reset input for
ALUMODE pipeline registers
         RSTB => RSTB, -- 1-bit reset input for B pipeline registers
         RSTC => RSTC, -- 1-bit reset input for C pipeline registers
         RSTCTRL => RSTCTRL, -- 1-bit reset input for OPMODE
pipeline registers
         RSTM => RSTM, -- 1-bit reset input for multiplier registers
         RSTP => RSTP -- 1-bit reset input for P pipeline registers
   );
-- End of DSP48E_inst instantiation
```

Для установки требуемых режимов работы и конфигурации формируемого экземпляра секции цифровой обработки сигналов в библиотечном примитиве ***DSP48E*** предусмотрены следующие параметры настройки:
- ACASCREG — указывает число конвейерных регистров, устанавливаемых между входами A/ACIN и выходом ACOUT.
- ALUMODEREG — задает количество конвейерных регистров на шине управления выбором функции, выполняемой АЛБ.
- AREG — определяет число конвейерных регистров, используемых на входной шине данных A.

- AUTORESET_PATTERN_DETECT — разрешает или запрещает автоматический сброс регистра на выходе детектора совпадения на следующем цикле после сравнения кода на выходной шине P и заданного значения.
- AUTORESET_PATTERN_DETECT_OPTINV — позволяет выбрать полярность сигнала на выходе детектора совпадения кода на выходной шине P и заданного значения.
- A_INPUT — позволяет выбрать источник данных, используемых в качестве первого сомножителя (операнда A).
- BCASCREG — определяет число конвейерных регистров, устанавливаемых между входами B/BCIN и выходом BCOUT.
- BREG — указывает количество конвейерных регистров, задействованных в составе формируемого элемента на входной шине данных B.
- B_INPUT — предоставляет возможность выбора источника данных, используемых в качестве второго сомножителя (операнда B).
- CARRYINREG — разрешает или запрещает применение буферного регистра в цепи сигнала входного переноса.
- CARRYINSELREG — управляет установкой буферного регистра на входах выбора источника сигнала входного переноса.
- CREG — определяет необходимость использования конвейерного регистра на дополнительной входной шине данных C сумматора/вычитающего устройства.
- MASK — задает значение маски для детектора совпадения кода на выходной шине P и заданного значения.
- MREG — предоставляет возможность использования в составе формируемого компонента конвейерного регистра на выходе аппаратного умножителя.
- MULTCARRYINREG — указывает количество конвейерных регистров в цепи внутреннего сигнала переноса, формируемого умножителем.
- OPMODEREG — позволяет задействовать конвейерный регистр на шине управления выбором источников входных данных АЛБ.
- PATTERN — определяет образцовое значение для детектора совпадения.
- PREG — позволяет включить выходной регистр в состав формируемого элемента.
- SIM_MODE — устанавливает режим моделирования конфигурируемого аппаратного модуля цифровой обработки сигналов.
- SEL_MASK — предоставляет возможность использования в качестве значения маски кода, представленного на входной шине данных C.
- SEL_PATTERN — позволяет выбрать в качестве образцового значения для детектора совпадения код, представленный на входной шине данных C.
- SEL_ROUNDING_MASK — применяется для выбора специальной маски, используемой в процессе округления.
- USE_MULT — управляет использованием аппаратного умножителя.
- USE_PATTERN_DETECT — разрешает или запрещает использование детектора совпадения кода на выходной шине P и заданного значения.
- USE_SIMD — устанавливает режим выполнения операций АЛБ.

Система условных обозначений входных и выходных портов, применяемых в описании интерфейса элементов, формируемых с помощью шаблона ***DSP Block (DSP48E)***, включает в себя следующие идентификаторы:

- ACOUT [29:0] — выходная 30-разрядная шина, предназначенная для подключения ко входной шине ACIN следующей секции DSP48E при каскадном сопряжении аппаратных модулей цифровой обработки сигналов.
- BCOUT — выходная 18-разрядная шина, предназначенная для подключения ко входной шине BCIN следующей секции DSP48E при каскадном соединении аппаратных модулей цифровой обработки сигналов.
- CARRYCASCOUT — выход сигнала переноса, используемый при каскадном наращивании секций DSP48E.
- CARRYOUT [3:0] — выходная 4-разрядная шина, на которую подаются сигналы переноса, формируемые 12-разрядными секциями АЛБ.
- MULTSIGNOUT — выход знакового разряда результата умножения, предназначенный для подключения ко входу MULTSIGNIN следующей секции DSP48E при каскадном соединении аппаратных модулей цифровой обработки сигналов.
- OVERFLOW — выход сигнала переполнения.
- P [47:0] — выходная 48-разрядная шина данных, на которую поступают значения результата обработки входных данных.
- PATTERNBDETECT — выход сигнала, информирующего о совпадении данных на выходной шине P и заданной строки.
- PATTERNDETECT — выход сигнала, информирующего о совпадении кода на выходной шине P и заданного значения.
- PCOUT [47:0] — выходная 48-разрядная шина данных, используемая при каскадном соединении секций DSP48E.
- UNDERFLOW — выход сигнала исчезновения значащих разрядов.
- A [29:0] — входная 30-разрядная шина данных, на которую поступает код операнда A.
- ACIN [29:0] — входная 30-разрядная шина, предназначенная для подключения выходной шины ACOUT предыдущей секции DSP48E при каскадном сопряжении аппаратных модулей цифровой обработки сигналов.
- ALUMODE [3:0] — входная 4-разрядная шина управления выбором функции, выполняемой АЛБ.
- B [17:0] — входная 18-разрядная шина данных, на которую поступает код операнда B.
- BCIN [17:0] — входная 18-разрядная шина, предназначенная для подключения выходной шины BCOUT предшествующей секции DSP48E при каскадном соединении аппаратных модулей цифровой обработки сигналов.
- C [47:0] — дополнительная 48-разрядная входная шина данных C АЛБ.
- CARRYCASCIN — вход сигнала переноса, формируемого предыдущим аппаратным модулем ЦОС при каскадном соединении секций DSP48E.
- CARRYIN — вход сигнала переноса.
- CARRYINSEL [2:0] — входная 3-разрядная шина управления выбором источника сигнала переноса.
- CEA1 и CEA2 — входы сигнала разрешения синхронизации для первого и второго конвейерных регистров, устанавливаемых на входе данных A аппаратного умножителя и АЛБ.
- CEALUMODE — вход сигнала разрешения синхронизации для регистра управления АЛБ, в который заносится код выполняемой функции.
- CEB1 и CEB2 — входы сигнала разрешения синхронизации для первого и второго конвейерных регистров, устанавливаемых на входе данных B аппаратного умножителя и АЛБ.
- CEC — вход сигнала разрешения синхронизации для буферного регистра, подключаемого к дополнительной шине данных C АЛБ.
- CECARRYIN — вход сигнала разрешения синхронизации для буферного регистра, устанавливаемого в цепи внешнего сигнала входного переноса.
- CECTRL — вход сигнала разрешения синхронизации регистров, предназначенных для хранения кода источников входных данных в АЛБ и сигналов входного переноса.
- CEM — вход сигнала разрешения синхронизации для конвейерного регистра, устанавливаемого на выходе аппаратного умножителя.
- CEMULTCARRYIN — вход сигнала разрешения синхронизации для буферного регистра, устанавливаемого в цепи внутреннего сигнала входного переноса аппаратного умножителя.
- CEP — вход сигнала разрешения синхронизации для выходного регистра.
- CLK — вход тактового сигнала.
- MULTSIGNIN — вход сигнала знакового разряда результата умножения, формируемого предшествующей секцией цифровой обработки сигналов при каскадном соединении модулей DSP48E.
- OPMODE [6:0] — входная 7-разрядная шина управления выбором источников входных данных для АЛБ.
- PCIN — входная 48-разрядная шина, предназначенная для подключения выходной шины PCOUT предыдущей секции DSP48E при каскадном наращивании аппаратных модулей цифровой обработки сигналов.

- RSTA — вход сигнала сброса конвейерных регистров, устанавливаемых на входе данных A аппаратного умножителя и АЛБ.
- RSTALLCARRYIN — вход сигнала сброса конвейерных регистров, применяемых в цепях входного переноса.
- RSTALUMODE — вход сигнала сброса для регистра управления АЛБ.
- RSTB — вход сигнала сброса конвейерных регистров, устанавливаемых на входе данных B аппаратного умножителя и АЛБ.
- RSTC — вход сигнала сброса буферного регистра, подключаемого к дополнительной шине данных C АЛБ.
- RSTCTRL — вход сигнала сброса регистров, предназначенных для хранения кода источников входных данных в АЛБ и сигналов входного переноса.
- RSTM — вход сигнала сброса конвейерного регистра, устанавливаемого на выходе аппаратного умножителя.
- RSTP — вход сигнала сброса выходного регистра.

При подготовке законченного описания элементов, реализуемых на базе аппаратных секций цифровой обработки сигналов DSP48E ПЛИС серии Virtex-5, вначале целесообразно указать, какие конвейерные регистры следует включить в состав формируемого компонента. Для этого нужно присвоить соответствующие значения параметрам настройки ACASCREG, ALUMODEREG, AREG, BCASCREG, BREG, CARRYINREG, CARRYINSELREG, CREG, MREG, MULTCARRYINREG, OPMODEREG и PREG. По умолчанию для всех перечисленных параметров предлагается единичное значение, при котором соответствующие конвейерные регистры задействуются в составе формируемого элемента. Чтобы исключить какой-либо регистр из структуры создаваемого элемента, следует присвоить нулевое значение соответствующему параметру настройки.

Затем нужно с помощью настраиваемых параметров A_INPUT и B_INPUT выбрать источники данных, определяющих значения первого и второго операндов. Эти параметры настройки могут принимать один из двух вариантов значений — “DIRECT” или “CASCADE”. По умолчанию для параметров A_INPUT и B_INPUT предлагается значение “DIRECT”, при котором в качестве операндов используются данные, представленные на входных шинах A и B соответственно. При выборе варианта “CASCADE” значения операндов определяются кодами, представленными на шинах, которые предназначены для организации каскадного соединения аппаратных модулей цифровой обработки сигналов (ACIN и/или BCIN).

Далее следует определить режим функционирования аппаратного умножителя в формируемом элементе, используя параметр настройки USE_MULT. Список возможных значений этого параметра содержит три варианта: “MULT_S”, “MULT” и “NONE”. Вариант “MULT_S”, предлагаемый по умолчанию, соответствует конвейерному режиму работы умножителя. При выборе значения “MULT” аппаратный умножитель конфигурируется без поддержки конвейерного режима функционирования. Если в создаваемом элементе не предполагается применение умножителя, то для параметра USE_MULT нужно указать значение “NONE”.

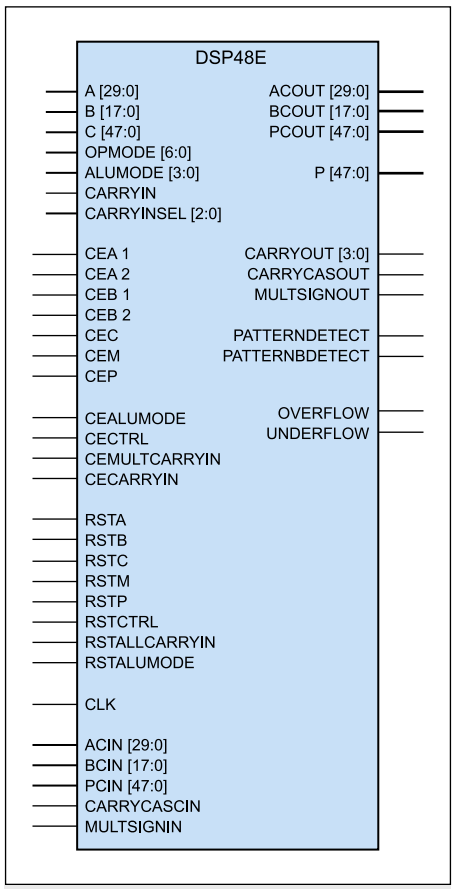

Рис. 367. Условный графический образ элементов, формируемых на основе шаблона DSP Block (DSP48E)

Условный графический образ элементов, описание которых создается с помощью шаблона ***DSP Block (DSP48E)*** для последующей реализации на основе аппаратных модулей цифровой обработки сигналов в кристаллах программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT, показан на рис. 367.

Global Clock Buffer MUX (BUFGMUX_CTRL) включает в себя образец описания варианта конфигурирования глобального буферного элемента с функцией мультиплексирования двух входных сигналов синхронизации в ПЛИС серии Virtex-5. В качестве основы этого шаблона используется оператор создания экземпляра библиотечного примитива ***BUFGMUX_CTRL***, который представляет указанный вариант конфигурирования глобального буферного элемента тактового сигнала в кристаллах программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT:

```
-- BUFGMUX_CTRL : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the body of the design code. The instance name
-- declaration : (BUFGMUX_CTRL_inst) and/or the port declarations after the
-- code : "=>" assignment maybe changed to properly reference and
-- : connect this function to the design. All inputs
-- : and outputs must be connect.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
   -- BUFGMUX_CTRL: Global Clock Buffer 2-to-1 MUX
   -- Virtex-5
   -- Xilinx HDL Language Template, version 13.2
--
   BUFGMUX_CTRL_inst : BUFGMUX_CTRL
   port map (
            O => O, -- Clock MUX output
            I0 => I0, -- Clock0 input
            I1 => I1, -- Clock1 input
            S => S -- Clock select input
            );
   -- End of BUFGMUX_CTRL_inst instantiation
```

В состав системы условных обозначений входов и выходов, используемых в описании интерфейса глобальных буферных элементов, формируемых с помощью шаблона ***Global Clock Buffer MUX (BUFGMUX_CTRL)***, входят следующие идентификаторы:

- O — выход глобального сигнала синхронизации;
- I0 — вход первого тактового сигнала;
- I1 — вход второго тактового сигнала;
- S — вход выбора тактового сигнала.

На рис. 368 изображен условный графический образ глобального буферного элемента с функцией мультиплексирования двух входных сигналов синхронизации, описание которого создается на основе шаблона ***Global Clock Buffer MUX (BUFGMUX_CTRL)*** для последующего использования в составе цифровых устройств, реализуемых на базе ПЛИС серии Virtex-5.

Boundary Scan (BSCAN_VIRTEX5) является шаблоном VHDL-описания компонента, предназначенного для выполнения операций периферийного сканирования в кристаллах программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT,

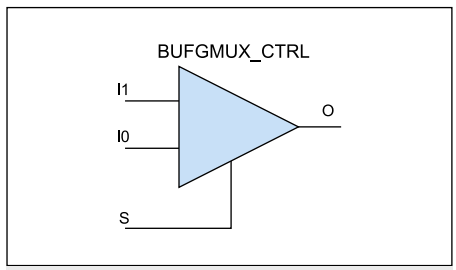

Рис. 368. Условный графический образ глобального буферного элемента, формируемого с помощью шаблона Global Clock Buffer MUX (BUFGMUX_CTRL)

Virtex-5 TXT и Virtex-5 FXT. Основу этого шаблона образует оператор создания экземпляра библиотечного примитива ***BSCAN_VIRTEX5***:

```
-- BSCAN_VIRTEX5 : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the architecture body of the design code. The instance name
-- declaration : (BSCAN_VIRTEX5_inst) and/or the port declarations after the
-- code : "=>" assignment maybe changed to properly reference and connect this
-- : function to the design. All inputs must be connected.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
   -- BSCAN_VIRTEX5: Boundary Scan primitive for connecting internal logic to
   -- JTAG interface.
   -- Virtex-5
   -- Xilinx HDL Language Template, version 13.2
   --
   BSCAN_VIRTEX5_inst : BSCAN_VIRTEX5
   generic map (
         JTAG_CHAIN => 1) -- Value for USER command. Possible values: (1,2,3 or 4)
   port map (
         CAPTURE => CAPTURE, -- CAPTURE output from TAP controller
         DRCK => DRCK, -- Data register output for USER functions
         RESET => RESET, -- Reset output from TAP controller
         SEL => SEL, -- USER active output
         SHIFT => SHIFT, -- SHIFT output from TAP controller
         TDI => TDI, -- TDI output from TAP controller
         UPDATE => UPDATE, -- UPDATE output from TAP controller
         TDO => TDO -- Data input for USER function
         );
   -- End of BSCAN_VIRTEX5_inst instantiation
```

В библиотечном примитиве ***BSCAN_VIRTEX5*** используется тот же параметр настройки и система условных обозначений входных и выходных портов, что и в шаблоне ***JTAG Boundary Scan Logic Control Circuit (BSCAN_SPARTAN6)***.

На рис. 369 показан условный графический образ компонента, формируемого с помощью шаблона ***Boundary Scan (BSCAN_VIRTEX5)*** для осуществления операций периферийного сканирования в ПЛИС серии Virtex-5.

Clear Encryption Key Value (KEY_CLEAR) содержит образец описания элемента, который предоставляет возможность очистки регистра, содержащего значение ключа, используемого в процессе кодирования/декодирования конфигурационной последовательности в кристаллах программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT. Основой этого шаблона является оператор создания экземпляра библиотечного примитива ***KEY_CLEAR***:

```
-- KEY_CLEAR : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the body of the design code. The instance name
-- declaration : (KEY_CLEAR_inst) and/or the port declarations after the
-- code : "=>" assignment maybe changed to properly reference and
-- : connect this function to the design. All inputs
-- : must be connected.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
   -- KEY_CLEAR: Startup primitive for GSR, GTS or startup sequence control
   -- Virtex-5
   -- Xilinx HDL Language Template, version 13.2
--
   KEY_CLEAR_inst : KEY_CLEAR
      port map (
            KEYCLEARB => KEYCLEARB -- Active low key reset 1-bit input
            );
   -- End of KEY_CLEAR_inst instantiation
```

В составе интерфейса библиотечного примитива ***KEY_CLEAR*** представлен единственный вход сброса KEYCLEARB, для которого активным уровнем сигнала является уровень логического нуля.

На рис. 370 приведен условный графический образ элемента, обеспечивающего возможность удаления ключа шифрования/дешифрации конфигурационных данных в ПЛИС серии Virtex-5, для подготовки описания которого используется шаблон ***Clear Encryption Key Value (KEY_CLEAR)***.

ConfigDataAccess (USR_ACCESS_VIRTEX5) включает в себя шаблон VHDL-описания 32-разрядного регистра, предоставляющего прямой доступ к данным конфигурационной последовательности в кристаллах программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT. В качестве основы этого шаблона используется экземпляр библиотечного примитива ***USR_ACCESS_VIRTEX5***:

```
--USR_ACCESS_VIRTEX5: In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the architecture body of the design code. The instance name
-- declaration : (USR_ACCESS_VIRTEX5_inst) and/or the port declarations after the
-- code : "=>" assignment maybe changed to properly reference and connect this
-- : function to the design. All inputs must be connected.
-- : that are not necessary.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
      -- USR_ACCESS_VIRTEX5: Configuration Data Memory Access Port
      -- Virtex-5
      -- Xilinx HDL Language Template, version 13.2
--
      USR_ACCESS_VIRTEX5_inst : USR_ACCESS_VIRTEX5
      port map (
            CFGCLK => CFGCLK, -- 1-bit configuration clock output
            DATA => DATA, -- 32-bit config data output
            DATAVALID => DATAVALID -- 1-bit data valid output
            );
      -- End of USR_ACCESS_VIRTEX5_inst instantiation
```

В состав системы условных обозначений выходных портов, применяемых в описании интерфейса библиотечного примитива ***USR_ACCESS_VIRTEX5***, входят следующие идентификаторы:

- CFGCLK — выход тактового сигнала, используемого в процессе конфигурирования ПЛИС;
- DATA — 32-разрядная выходная шина данных;
- DATAVALID — выход сигнала, информирующего о достоверности конфигурационных данных, представленных на шине DATA.

Условный графический образ 32-разрядного регистра, обеспечивающего возможность прямого доступа к данным конфигурационной последовательности в ПЛИС серии Virtex-5, описание которого создается с помощью шаблона ***Config Data Access (USR_ACCESS_VIRTEX5)***, показан на рис. 371.

Configuration Simulation Model (SIM_CONFIG_V5) представляет собой обра-

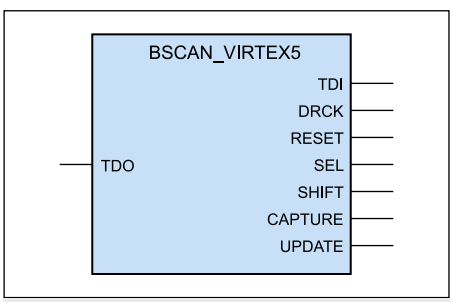

Рис. 369. Условный графический образ компонента, предназначенного для организации периферийного сканирования в ПЛИС серии Virtex-5

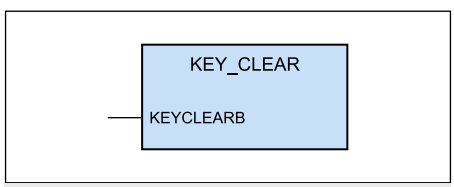

Рис. 370. Условный графический образ элемента, предоставляющего возможность удаления ключа кодирования/декодирования конфигурационных данных

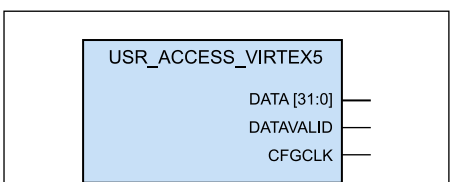

Рис. 371. Условный графический образ компонента, формируемого с помощью шаблона Config Data Access (USR_ACCESS_VIRTEX5)

зец VHDL-описания компонента, предназначенного для моделирования параллельного конфигурационного интерфейса SelectMap в кристаллах программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT. Этот шаблон выполнен на основе экземпляра библиотечного примитива ***SIM_CONFIG_V5***:

```
-- SIM_CONFIG_V5 : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the architecture body of the design code. The
-- declaration : (SIM_CONFIG_V5_inst) and/or the port declarations
-- code : after the "=>" assignment maybe changed to properly
-- : reference and connect this function to the design.
-- : All inputs and outputs must be connected.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
      -- SIM_CONFIG_V5: Behavioral Simulation-only Model of FPGA
SelectMap Configuration
      -- Virtex-5
      -- Xilinx HDL Language Template, version 13.2
--
      SIM_CONFIG_V5_inst : SIM_CONFIG_V5
      generic map (
            DEVICE_ID => X"00000000") -- Specifies the Pre-
programmed Device ID value
      port map (
            BUSY => BUSY, -- 1-bit output Busy pin
            CSOB => CSOB, -- 1-bit output chip select pin
            DONE => DONE, -- 1-bit bi-directional Done pine
            CCLK => CCLK, -- 1-bit input configuration clock
            D => D, -- 8-bit bi-directional configuration data
            DCMLOCK => DCMLOCK, -- 1-bit input DCM Lock
            INITB =>INITB, -- 1-bit bi-directional INIT status pin
            M => M, -- 3-bit input Mode pins
            PROGB => PROGB, -- 1-bit input Program pin
            RDWRB => RDWRB -- 1-bit input Read/Write pin
        );
      -- End of SIM_CONFIG_V5_inst instantiation
```

В библиотечном примитиве ***SIM_CONFIG_V5*** предусмотрен единственный параметр настройки DEVICE_ID, которому присваивается значение идентификационного кода применяемой ПЛИС серии Virtex-5. Значение этого настраиваемого параметра указывается в форме 8-разрядного шестнадцатеричного числа.

Система условных обозначений, используемых в описании интерфейса компонента, формируемого с помощью шаблона ***Configuration Simulation Model (SIM_CONFIG_V5)***, включает в себя следующие идентификаторы входных и выходных портов:

- BUSY — выход одноименного сигнала, применяемого в процессе обратного чтения конфигурационных данных из кристалла программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT.
- CSOB — выход сигнала Chip Select параллельной daisy-цепочки ПЛИС серии Virtex-5.
- DONE — двунаправленный вывод (вход/выход) одноименного сигнала, информирующего о состоянии процесса конфигурирования кристалла программируемой логики.
- CCLK — вход тактового сигнала конфигурационного интерфейса ПЛИС.
- D — двунаправленная 8-разрядная шина конфигурационных данных.
- DCMLOCK — вход сигнала блокировки модуля DCM.
- INITB — двунаправленный вывод (вход/выход) одноименного сигнала конфигурационного интерфейса ПЛИС.
- M — входная 3-разрядная шина, комбинация значений сигналов которой определяет режим конфигурирования кристалла программируемой логики.
- PROGB — вход сигнала Program конфигурационного интерфейса.
- RDWRB — вход сигнала выбора режима записи или чтения конфигурационных данных (направления передачи данных по шине D).

На рис. 372 изображен условный графический образ компонента, используемого для моделирования параллельного конфигурационного интерфейса SelectMap ПЛИС серии Virtex-5, описание которого формируется с помощью шаблона ***Configuration Simulation Model (SIM_CONFIG_V5)***.

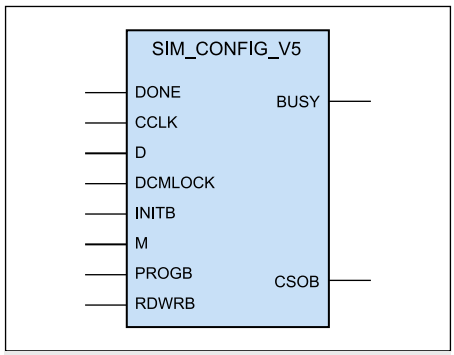

Рис. 372. Условный графический образ компонента, формируемого с помощью шаблона Configuration Simulation Model (SIM_CONFIG_V5)

Internal Config Access Port (ICAP_VIRTEX5) является шаблоном описания внутреннего порта конфигурационного интерфейса кристаллов программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT, обеспечивающего возможность доступа к конфигурационным регистрам ПЛИС и обратного чтения данных. Основу этого шаблона образует оператор создания экземпляра библиотечного примитива ***ICAP_VIRTEX5***, представляющего внутренний порт конфигурационного интерфейса, подобного SelectMAP [48]:

```
-- ICAP_VIRTEX5 : In order to incorporate this function into the design,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the architecture body of the design code. The
-- declaration : instance name (ICAP_VIRTEX5_inst) and/or the port
declarations
-- code : after the "=>" assignment maybe changed to properly
-- : connect this function to the design. All inputs must be
-- : connected.
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
      -- ICAP_VIRTEX5: Internal Configuration Access Port
      -- Virtex-5
      -- Xilinx HDL Language Template, version 13.2
--
      ICAP_VIRTEX5_inst : ICAP_VIRTEX5
      generic map (
            ICAP_WIDTH => "X8") -- "X8", "X16" or "X32"
      port map (
            BUSY => BUSY, -- Busy output
            O => O, -- 32-bit data output
            CE => CE, -- Clock enable input
            CLK => CLK, -- Clock input
            I => I, -- 32-bit data input
            WRITE => WRITE -- Write input
        );
      -- End of ICAP_VIRTEX5_inst instantiation
```

В библиотечном примитиве ***ICAP_VIRTEX5*** используются те же условные обозначения интерфейсных портов и параметр настройки, что и в шаблоне ***Internal Config Access Port (ICAP_VIRTEX4)***.

Условный графический образ компонента, предоставляющего возможность выполнения операций чтения и записи данных в конфигурационные регистры ПЛИС серии Virtex-5, для подготовки описания которого используется шаблон ***Internal Config Access Port (ICAP_VIRTEX5)***, представлен на рис. 373.

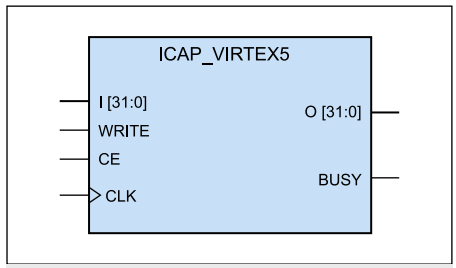

Рис. 373. Условный графический образ компонента, создаваемого с помощью шаблона Internal Config Access Port (ICAP_VIRTEX5)

JTAG Simulation Model (JTAG_SIM_VIRTEX5) содержит образец VHDL-описания компонента, предназначенного для функционального моделирования контроллера порта JTAG-интерфейса в составе разрабатываемого устройства, реализуемого на базе кристаллов программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT:

```
-- JTAG_SIM_VIRTEX5 : In order to incorporate this simulation model,
-- VHDL : the following instance declaration needs to be placed
-- instance : in the architecture body of the testbench code.
-- declaration : This should not be instantiated into the synthesizable design
-- code : code.
-- : The instance name
-- : (JTAG_SIM_VIRTEX5_inst) and/or the port declarations after
the : "=>" assignment maybe changed to properly reference and
connect this
-- : function to the design.
-- : Only one JTAG_SIM_VIRTEX5 should be
```

```
-- : instantiated per design.
-- : All inputs and output should be connected
--
-- Library : In addition to adding the instance declaration, a use
-- declaration : statement for the UNISIM.vcomponents library needs to be
-- for : added before the entity declaration. This library
-- Xilinx : contains the component declarations for all Xilinx
-- primitives : primitives and points to the models that will be used
-- : for simulation.
--
-- Copy the following two statements and paste them before the
-- Entity declaration, unless they already exist.
--
Library UNISIM;
use UNISIM.vcomponents.all;
--
-- <---Cut code below this line and paste into the architecture body--->
--
      -- JTAG_SIM_VIRTEX5: JTAG Interface Simulation Model
      -- Virtex-5
      -- Xilinx HDL Language Template, version 13.2
      --
      JTAG_SIM_VIRTEX5_inst : JTAG_SIM_VIRTEX5
      generic map (
               PART_NAME => "LX30") -- Specify target V5 device.
Possible values are "LX30", "LX50", "LX85", "LX110", "LX220",
"LX330"
      port map (
               TDO => TDO, -- JTAG data output (1-bit)
               TCK => TCK, -- Clock input (1-bit)
               TDI => TDI, -- JTAG data input (1-bit)
               TMS => TMS -- JTAG command input (1-bit)
        );
      -- End of JTAG_SIM_VIRTEX5_inst instantiation
```

Основой приведенной конструкции является оператор создания экземпляра библиотечного примитива ***JTAG_SIM_VIRTEX5***, в котором используется та же система условных обозначений интерфейсных портов и параметр настройки, что и в шаблоне ***JTAG Simulation Model (JTAG_SIM_VIRTEX4)***. Но при этом следует учитывать, что список возможных вариантов значений параметра настройки ***PART_NAME*** в библиотечном примитиве ***JTAG_SIM_VIRTEX5*** отличается от указанного шаблона. Этот список содержит варианты сокращенных условных обозначений ПЛИС серии Virtex-5.

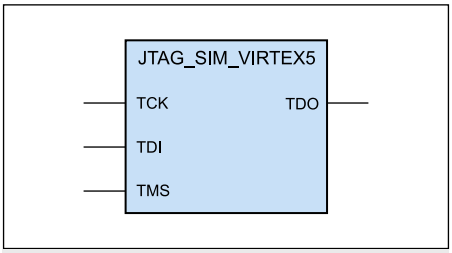

Рис. 374. Условный графический образ компонента, формируемого с помощью шаблона JTAG Simulation Model (JTAG_SIM_VIRTEX5)

На рис. 374 приведен условный графический образ компонента, формируемого на основе шаблона ***JTAG Simulation Model (JTAG_SIM_VIRTEX5)*** для функционального моделирования контроллера порта JTAG-интерфейса в кристаллах программируемой логики семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 TXT и Virtex-5 FXT. ■

Продолжение следует

Литература

44. Virtex-5 Family Overview. Xilinx, 2009.
45. Virtex-5 FPGA Data Sheet: DC and Switching Characteristics. Xilinx, 2010.
46. Virtex-5 FPGA User Guide. Xilinx, 2010.
47. Virtex-5 FPGA XtremeDSP Design Considerations. Xilinx, 2010.
48. Virtex-5 FPGA Configuration Guide. Xilinx, 2010.
49. Virtex-5 FPGA RocketIO GTP Transceiver User Guide. Xilinx, 2009.
50. Virtex-5 FPGA RocketIO GTX Transceiver User Guide. Xilinx, 2009.
51. Virtex-5 FPGA Embedded Tri-Mode Ethernet Media Access Controller User Guide. Xilinx, 2011.
52. Virtex-5 FPGA Integrated Endpoint Block for PCI Express Designs User Guide. Xilinx, 2011.
53. Virtex-5 FPGA System Monitor User Guide. Xilinx, 2011.
54. Virtex-5 FPGA Embedded Processor Block Reference Guide. Xilinx, 2010.