Продолжение. Начало в № 12 `2008

Разработка компонентов устройств ЦОС, реализуемых на базе аппаратных модулей DSP48E в ПЛИС FPGA серии Virtex-5, с помощью «мастера» Architecture Wizard САПР серии Xilinx ISE

Четвертая часть статьи продолжает ознакомление с процессом подготовки компонентов высокоскоростных устройств ЦОС, реализуемых на базе аппаратных секций DSP48E в ПЛИС FPGA [1] серии Virtex-5, с помощью «мастера» Architecture Wizard САПР серии Xilinx ISE [3, 4]. В этой части рассматривается процедура генерации описаний умножителей, обладающих высоким быстродействием, и приводятся примеры типовых элементов, выполняющих операцию умножения, с различной структурой. Здесь же приведены краткие сведения о разработке высокопроизводительных комплексных умножителей на основе элементов, формируемых с помощью «мастера» Architecture Wizard САПР серии Xilinx ISE.

Валерий ЗОТОВ
walerry@km.ru

В процессе подготовки данной публикации в составе серии ПЛИС Virtex-5 произошли изменения. В конце прошлого года фирма Xilinx приступила к серийному выпуску кристаллов нового семейства Virtex-5 TXT [2, 6, 7]. Следует обратить внимание на то, что в состав архитектуры ПЛИС этого семейства также входят аппаратные секции ЦОС DSP48E [5, 6]. Поэтому информация, представленная в предыдущих частях настоящей статьи, относится и к кристаллам семейства Virtex-5 TXT.

Подготовка описаний умножителей, реализуемых на базе аппаратных секций DSP48E в ПЛИС FPGA серии Virtex-5, с помощью «мастера» Architecture Wizard САПР серии Xilinx ISE

Процесс формирования описания умножителя, предназначенного для реализации на базе аппаратных модулей ЦОС DSP48E в ПЛИС FPGA семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 FXT и Virtex-5 TXT, при использовании автономного режима работы «мастера» Architecture Wizard начинается с выбора строки *Multiplier* в списке генерируемых элементов, который отображается во второй диалоговой панели с заголовком *Selection* (рис. 6, см. КиТ № 12 `2008, стр. 34). Если данный процесс осуществляется в рамках управляющей оболочки САПР серии Xilinx ISE *Навигатора проекта* (*Project Navigator*), то для генерации соответствующего описания нужно в диалоговой панели выбора типа параметризированного модуля *Select IP*, представленной на рис. 8 (см. КиТ № 12 `2008, стр. 35), выделить строку *Multiplier v9.1i*. После этого, независимо от выбранного режима, на экран выводится стартовая диалоговая панель «мастера» настройки параметров формируемого умножителя, которая имеет заголовок *Input/Output Data Setup – Multiplier*. С помощью этой диалоговой панели, вид которой представлен на рис. 24, нужно указать входные шины, используемые в качестве источников операндов в генерируемом умножителе, и определить их параметры.

Архитектура аппаратной секции DSP48E ПЛИС серии Virtex-5 предоставляет возможность выбора одного из четырех вариантов комбинации входных шин в формируемом умножителе. Требуемый вариант указывается в поле выбора *Select input data*. Для создания описания умножителя, в котором значения сомножителей определяются двоичными кодами, поступающими на входные шины данных A и B, в выпадающем списке этого поля следует выбрать вариант *B and A*. Если в разрабатываемом умножителе значение первого сомножителя должно определяться совокупностью сигналов на шине данных BCIN, предназначенной для подключения выходной шины BCOUT предшествующей секции DSP48E при каскадном соединении аппаратных модулей ЦОС, а второго — двоичным кодом, представленным на входной шине данных A, то в поле выбора *Select input data* нужно указать вариант *BCIN and A*. При подготовке описания умножителя, в котором источниками операндов являются входная шина данных B и шина ACIN, используемая для подключения выходной шины ACOUT предыдущей секции DSP48E при каскадном наращивании аппаратных модулей ЦОС, в выпадающем списке рассматриваемого поля выбора следует выделить строку *B and ACIN*. Чтобы сформировать описание умножителя, в котором входными портами данных являются шины BCIN и ACIN, предназначенные для каскадного соединения аппаратных секций DSP48E при наращивании разрядности формируемых элементов ЦОС, следует выбрать вариант *BCIN and ACIN*. По умолчанию в поле выбора *Select input data* предлагается вариант *B and A*.

Если в качестве входных портов генерируемого умножителя выбраны шины данных A, B или ACIN, то далее нужно определить разрядность этих портов, воспользовавшись полями редактирования *Width*, расположенными во встроенных панелях *Input Data A*, *Input Data B* или *Input Data ACIN* (рис. 24). При этом количество разрядов для входных

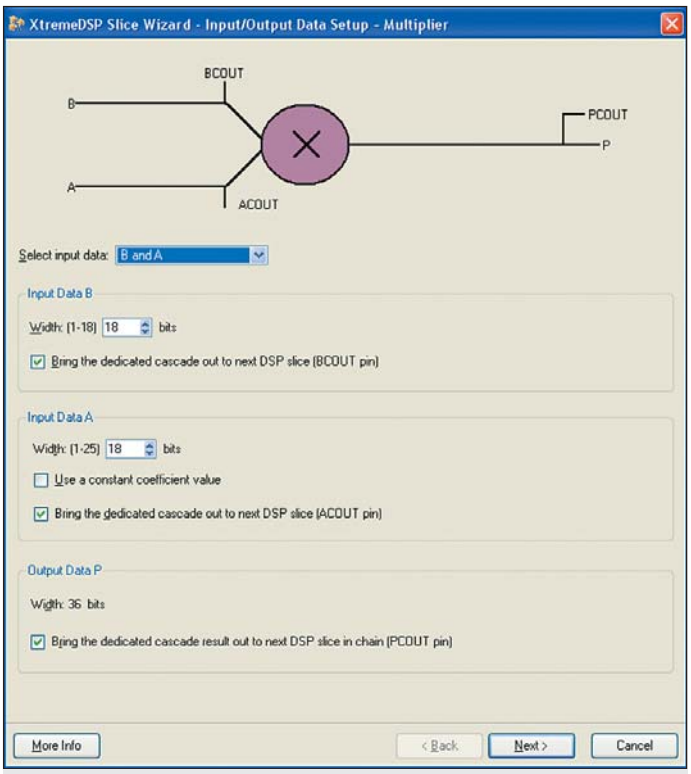

Рис. 24. Вид стартовой диалоговой панели Input/Output Data Setup – Multiplier «мастера» настройки параметров умножителей

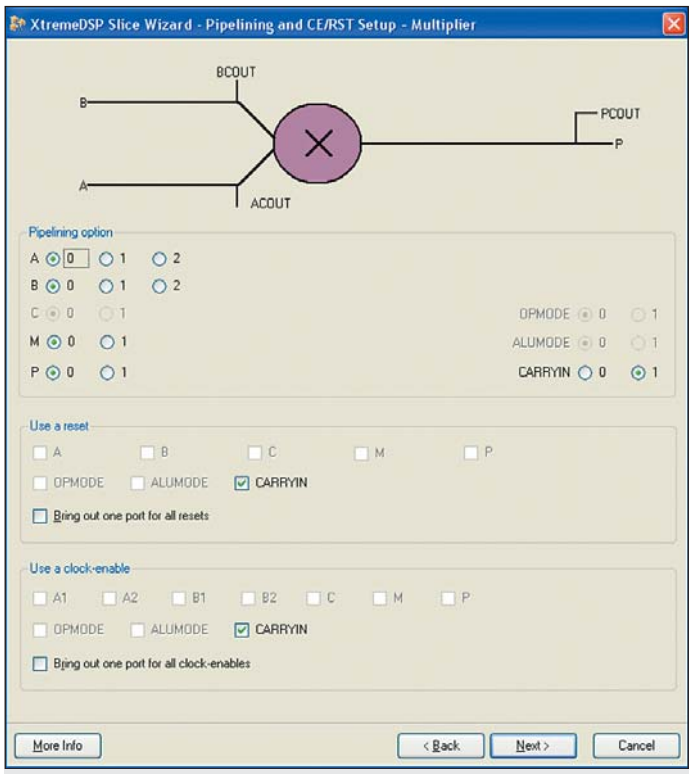

Рис. 25. Вид второй диалоговой панели Pipelining and CE/RST Setup – Multiplier «мастера» настройки параметров умножителей

шин данных A и ACIN можно выбирать в диапазоне от 1 до 25 бит. Для шины данных B допустимый диапазон значений разрядности составляет от 1 до 18 бит. По умолчанию в полях редактирования *Width* предлагается максимально допустимое значение. При использовании шины BCIN в качестве одного из входных портов формируемого умножителя значение параметра *Width* жестко зафиксировано и составляет 18 двоичных разрядов.

В состав интерфейса создаваемого умножителя при необходимости можно включить дополнительные выходные шины ACOUT и BCOUT, которые предназначены для соединения с входными шинами ACIN и BCIN следующей секции DSP48E при каскадном наращивании аппаратных модулей ЦОС. Для этого нужно перевести в состоянии «Включено» индикаторы *Bring the dedicated cascade out to next DSP slice (ACOUT pin)* и *Bring the dedicated cascade out to next DSP slice (BCOUT pin)* соответственно, находящиеся во встроенных панелях *Input Data A* (*Input Data ACIN*) и *Input Data B*.

Разрядность выходной шины данных генерируемого умножителя не зависит от выбранных значений разрядности входных портов и составляет 36 бит. Это значение отображается в строке *Width*, расположенной во встроенной панели *Output Data P* (рис. 24). Если формируемый умножитель предполагается использовать в составе каскадного соединения аппаратных модулей ЦОС, то в составе его интерфейса должна присутствовать дополнительная выходная шина PCOUT, которая предназначена для подключения к входной шине PCIN следующей секции DSP48E. Для этого индикатор *Bring the dedicated cascade result out to next DSP Slice in chain (PCOUT pin)*, представленный во встроенной панели *Output Data P*, должен находиться в состоянии «Включено».

С помощью «мастера» Architecture Wizard на базе аппаратных секций DSP48E можно создавать описания элементов, выполняющих операцию умножения входных значений на постоянный коэффициент. Для формирования описания умножителя на константу нужно в качестве входного порта, определяющего значение второго сомножителя, выбрать шину данных A. Таким образом, в поле выбора источников входных операндов *Select input data* следует указать вариант *B and A* или *BCIN and A*. Затем в поле редактирования *Width*, представленном во встроенной панели *Input Data A*, необходимо задать разрядность представления значения постоянного коэффициента и перевести индикатор *Use a constant coefficient value*, расположенный во встроенной панели *Input Data A*, в состояние «Включено». После этого в данной встроенной панели справа от индикатора состояния *Use a constant coefficient value* появляется поле редактирования, в котором следует указать значение постоянного коэффициента. Значение константы задается в десятичном формате со знаком. При этом указываемое число должно находиться в пределах допустимого диапазона значений, который составляет от $-2^{(n-1)}$ до $2^{(n-1)}-1$, где n — количество разрядов входной шины A, указанное в поле редактирования Width. Если в поле редактирования значения константы введено число, выходящее за рамки допустимого диапазона, то при переходе к следующей диалоговой панели «мастера» настройки параметров умножителей, реализуемых на базе аппаратного модуля DSP48E ПЛИС серии Virtex-5, на экран выводится соответствующее предупреждение. Это предупреждение представлено в виде информационной панели, содержащей сведения о границах допустимого диапазона значений постоянного коэффициента, которые соответствуют выбранному значению разрядности входной шины A.

Процедура выбора входных и выходных шин данных, включаемых в состав интерфейса формируемого умножителя, завершается нажатием клавиши «*Далее*» (*Next*), расположенной в стартовой диалоговой панели *Input/Output Data Setup – Multiplier* (рис. 24). При корректных значениях параметров, указанных в стартовой панели, на экран выводится вторая диалоговая панель «мастера» настройки параметров умножителей, реализуемых на базе аппаратных секций DSP48E в кристаллах семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 FXT и Virtex-5 TXT. Данная диалоговая панель имеет заголовок *Pipelining and CE/RST Setup – Multiplier* и используется для включения в состав структуры генерируемого умножителя входных и выходных регистров. Вид этой диалоговой панели при отсутствии буферизации входных и выходных данных изображен на рис. 25.

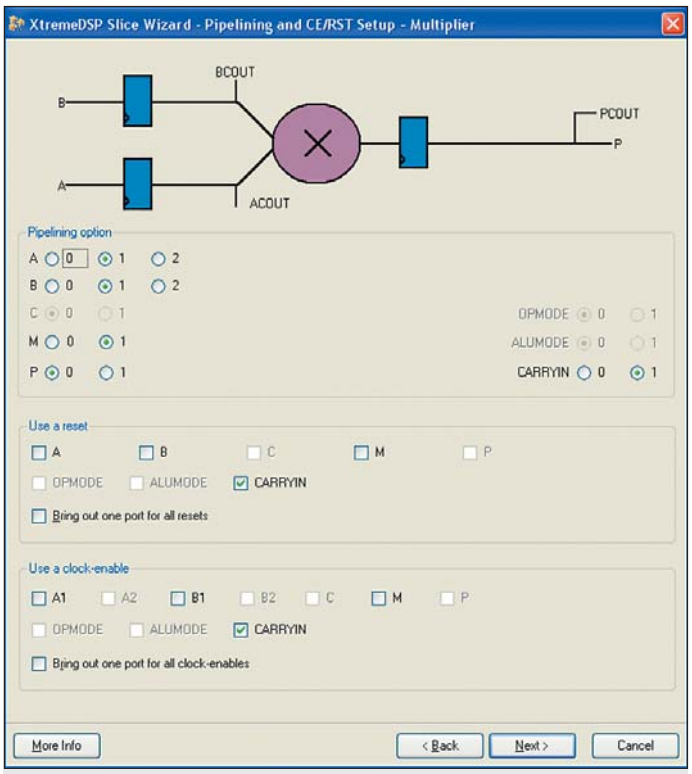

Рис. 26. Вид второй диалоговой панели Pipelining and CE/RST Setup — Multiplier «мастера» настройки параметров умножителей, предназначенных для реализации на базе аппаратных модулей DSP48E, при выборе одноступенчатой схемы буферизации входных и выходных данных

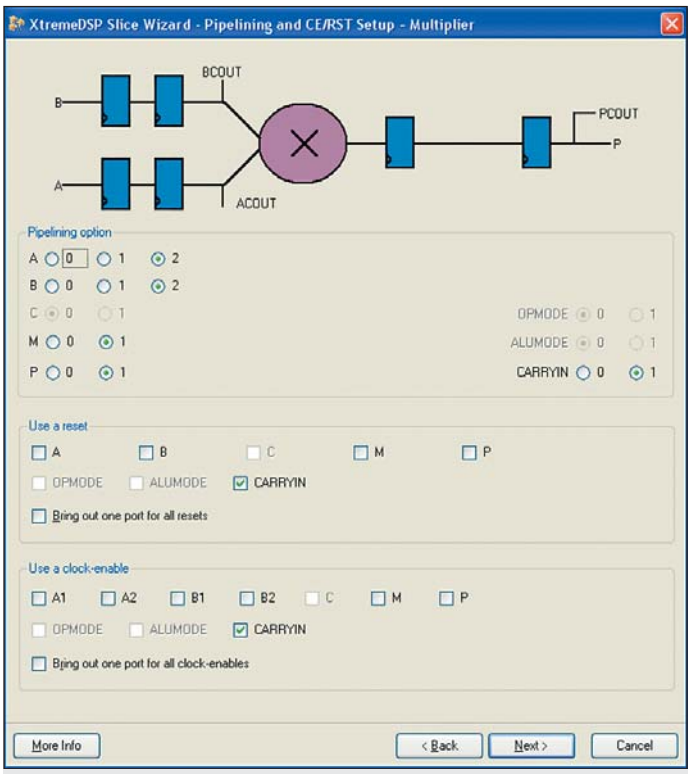

Рис. 27. Вид второй диалоговой панели Pipelining and CE/RST Setup – Multiplier «мастера» настройки параметров умножителей, предназначенных для реализации на базе аппаратных модулей DSP48E, при выборе двухступенчатой схемы конвейерной обработки входных и выходных данных

Архитектура аппаратных секций DSP48E позволяет реализовать одноступенчатую или двухступенчатую организацию конвейерной обработки данных в создаваемых умножителях. Количество конвейерных регистров, применяемых в составе структуры формируемого умножителя, указывается с помощью групп кнопок с зависимой фиксацией A, B, M и P, которые представлены во встроенной панели *Pipelining option*. Для генерации описания умножителя с одноступенчатой буферизацией входных и/или выходных данных следует переключить в нажатое состояние кнопку *1*. При этом в качестве выходного регистра можно выбрать регистр M, подключенный непосредственно к выходной шине умножителя, или регистр P, установленный на выходе арифметическо-логического блока (рис. 1, см. КиТ № 12 `2008, стр. 31), используя одноименную группу кнопок с зависимой фиксацией. В этом случае диалоговая панель *Pipelining and CE/RST Setup – Multiplier* автоматически преобразуется к виду, показанному на рис. 26.

После включения в структуру умножителя входных и выходных регистров становятся доступными соответствующие индикаторы состояния, расположенные во встроенных панелях *Use a reset* и *Use a clock-enable* (рис. 26). Эти индикаторы состояния позволяют добавить в состав интерфейса формируемого умножителя входы сброса и разрешения синхронизации для используемых буферных регистров. Для создания одного общего входа сброса всех входных и выходных регистров, включенных в состав структуры создаваемого элемента, нужно установить во включенное состояние индикатор *Bring out one port for all resets*, который находится во встроенной панели *Use a reset*. Чтобы задействовать один общий вход сигнала разрешения синхронизации для входных и выходных регистров, применяемых в составе умножителя, следует переключить в активное состояние индикатор *Bring out one port for all clock-enables*, расположенный во встроенной панели *Use a clock-enable*.

Для формирования описания умножителя с двухступенчатой конвейерной организацией обработки входных данных следует переключить в нажатое положение кнопки *2*. Чтобы реализовать двухступенчатую буферизацию выходных данных, нужно совместно задействовать регистры M и P, установив в нажатое состояние кнопки *1* в одноименных группах (рис. 25, 26). В результате этих действий диалоговая панель *Pipelining and CE/RST Setup – Multiplier* приобретает вид, представленный на рис. 27.

В этом случае во встроенных панелях *Use a reset* и *Use a clock-enable* в доступное состояние автоматически переводятся дополнительные индикаторы состояния, предоставляющие возможность выбора входов управления для соответствующих буферных регистров.

При генерации описаний элементов, выполняющих операцию умножения с учетом значения входного переноса, разработчик может задействовать также буферный регистр в цепи сигнала входного переноса, воспользовавшись группой кнопок с зависимой фиксацией *CARRYIN*, которая находится во встроенной панели *Pipelining option* (рис. 25–27). Для включения входов сброса и разрешения синхронизации этого регистра в состав интерфейса создаваемого умножителя нужно установить в состояние «Включено» индикаторы *CARRYIN*, которые становятся доступными во встроенных панелях *Use a reset* и *Use a clock-enable* соответственно.

Процедура выбора конвейерных регистров, включаемых в состав структуры формируемого умножителя, завершается нажатием клавиши «*Далее*» (*Next*) в нижней части диалоговой панели *Operation Mode Setup – Multiplier* (рис. 25–27). После этого открывается заключительная информационная панель «мастера» настройки параметров умножителя с заголовком *Summary – Multiplier*. Процесс автоматической генерации файлов описания умножителя осуществляется при нажатии клавиши «*Готово*» (*Finish*), находящейся в нижней части этой информационной панели. Типовые примеры описаний умножителей, реализуемых на основе аппаратных секций DSP48E в ПЛИС FPGA семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 FXT и Virtex-5 TXT, подготовленные с помощью «мастера» Architecture Wizard, представлены в следующих разделах.

Пример описания умножителя с одноступенчатой буферизацией входных и выходных данных, сформированного с помощью «мастера» Architecture Wizard, для реализации на базе аппаратных модулей DSP48E в ПЛИС FPGA серии Virtex-5

В качестве примера описания умножителя с одноступенчатой структурой конвейерной обработки входных и выходных данных, сгенерированного с помощью «мастера» Architecture Wizard, в этом разделе представлено VHDL-описание элемента *multiplier_16_16_36*. Описание этого умножителя предназначено, в первую очередь, для использования в составе проектов устройств ЦОС, выполняемых на основе кристаллов FPGA семейства Virtex-5 LX. Элемент *multiplier_16_16_36* осуществляет операцию перемножения двух 16-разрядных значений входных данных. Входными портами сформированного умножителя являются шина данных B и шина ACIN, используемая для подключения выходной шины ACOUT предшествующей секции DSP48E при каскадном соединении аппаратных модулей ЦОС. Вычисляемое значение произведения отображается в виде 36-разрядного двоичного кода на выходной шине данных P. Дополнительные выходные шины ACOUT, BCOUT и PCOUT, добавленные в состав интерфейса рассматриваемого элемента, предоставляют возможность каскадного наращивания разрядности. Сгенерированный текст описания умножителя *multiplier_16_16_36* на языке VHDL имеет следующий вид:

```
--Command:  xaw2vhdl-st  D:\PRJ\multiplier_16_16_36.xaw
D:\PRJ\multiplier_16_16_36
--Design Name: multiplier_16_16_36
--Device: xc5vlx110-ff676-3
--
-- Module multiplier_16_16_36
-- Generated by Xilinx Architecture Wizard
-- Written for synthesis tool: XST
--
library ieee;
use ieee.std_logic_1164.ALL;
use ieee.numeric_std.ALL;
library UNISIM;
use UNISIM.Vcomponents.ALL;
--
entity multiplier_16_16_36 is
   port (
         ACIN_IN : in std_logic_vector (15 downto 0);
         B_IN : in std_logic_vector (15 downto 0);
         CARRYIN_IN : in std_logic;
         CEA1_IN : in std_logic;
         CEB1_IN : in std_logic;
         CECARRYIN_IN : in std_logic;
         CEMULTCARRYIN_IN : in std_logic;
         CEM_IN : in std_logic;
         CLK_IN : in std_logic;
         RSTALLCARRYIN_IN : in std_logic;
         RSTA_IN : in std_logic;
         RSTB_IN : in std_logic;
         RSTM_IN : in std_logic;
         ACOUT_OUT : out std_logic_vector (29 downto 0);
         BCOUT_OUT : out std_logic_vector (17 downto 0);
         PCOUT_OUT : out std_logic_vector (47 downto 0);
         P_OUT : out std_logic_vector (35 downto 0)
         );
end multiplier_16_16_36;
--
architecture BEHAVIORAL of multiplier_16_16_36 is
   signal GND_ACIN : std_logic_vector (4 downto 0);
   signal GND_BUS_3 : std_logic_vector (2 downto 0);
   signal GND_BUS_4 : std_logic_vector (3 downto 0);
   signal GND_BUS_18 : std_logic_vector (17 downto 0);
   signal GND_BUS_30 : std_logic_vector (29 downto 0);
   signal GND_BUS_48 : std_logic_vector (47 downto 0);
   signal GND_OPMODE : std_logic;
   signal P_float : std_logic_vector (11 downto 0);
   signal VCC_OPMODE : std_logic;
begin
   GND_ACIN(4 downto 0) <= «00000»;
   GND_BUS_3(2 downto 0) <= «000»;
   GND_BUS_4(3 downto 0) <= «0000»;
   GND_BUS_18(17 downto 0) <= «000000000000000000»;
   GND_BUS_30(29 downto 0) <= «000000000000000000000000000000»;
   GND_BUS_48(47 downto 0) <=
       «000000000000000000000000000000000000000000000000»;
   GND_OPMODE <= '0';
   VCC_OPMODE <= '1';
   DSP48E_INST : DSP48E
   generic map(
               ACASCREG => 1,
               ALUMODEREG => 0,
               AREG => 1,
               AUTORESET_PATTERN_DETECT => FALSE,
               AUTORESET_PATTERN_DETECT_OPTINV => «MATCH»,
               A_INPUT => «CASCADE»,
               BCASCREG => 1,
               BREG => 1,
               B_INPUT => «DIRECT»,
               CARRYINREG => 1,
               CARRYINSELREG => 0,
               CREG => 0,
               MASK => x«3FFFFFFFFFFF»,
               MREG => 1,
               MULTCARRYINREG => 1,
               OPMODEREG => 0,
               PATTERN => x«000000000000»,
               PREG => 0,
               SEL_MASK => «MASK»,
               SEL_PATTERN => «PATTERN»,
               SEL_ROUNDING_MASK => «SEL_MASK»,
               USE_MULT => «MULT_S»,
               USE_PATTERN_DETECT => «NO_PATDET»,
               USE_SIMD => «ONE48»
               )
       port map (
               A(29 downto 0)=>GND_BUS_30(29 downto 0),
               ACIN(29 downto 25)=>GND_ACIN(4 downto 0),
               ACIN(24)=>ACIN_IN(15),
               ACIN(23)=>ACIN_IN(15),
               ACIN(22)=>ACIN_IN(15),
               ACIN(21)=>ACIN_IN(15),
               ACIN(20)=>ACIN_IN(15),
               ACIN(19)=>ACIN_IN(15),
               ACIN(18)=>ACIN_IN(15),
               ACIN(17)=>ACIN_IN(15),
               ACIN(16)=>ACIN_IN(15),
               ACIN(15 downto 0)=>ACIN_IN(15 downto 0),
               ALUMODE(3 downto 0)=>GND_BUS_4(3 downto 0),
               B(17)=>B_IN(15),
               B(16)=>B_IN(15),
               B(15 downto 0)=>B_IN(15 downto 0),
               BCIN(17 downto 0)=>GND_BUS_18(17 downto 0),
               C(47 downto 0)=>GND_BUS_48(47 downto 0),
               CARRYCASCIN=>GND_OPMODE,
               CARRYIN=>CARRYIN_IN,
               CARRYINSEL(2 downto 0)=>GND_BUS_3(2 downto 0),
               CEALUMODE=>VCC_OPMODE,
               CEA1=>CEA1_IN,
               CEA2=>VCC_OPMODE,
               CEB1=>CEB1_IN,
               CEB2=>VCC_OPMODE,
               CEC=>VCC_OPMODE,
               CECARRYIN=>CECARRYIN_IN,
               CECTRL=>VCC_OPMODE,
               CEM=>CEM_IN,
               CEMULTCARRYIN=>CEMULTCARRYIN_IN,
               CEP=>VCC_OPMODE,
               CLK=>CLK_IN,
               MULTSIGNIN=>GND_OPMODE,
               OPMODE(6)=>GND_OPMODE,
               OPMODE(5)=>GND_OPMODE,
               OPMODE(4)=>GND_OPMODE,
               OPMODE(3)=>GND_OPMODE,
               OPMODE(2)=>VCC_OPMODE,
               OPMODE(1)=>GND_OPMODE,
               OPMODE(0)=>VCC_OPMODE,
               PCIN(47 downto 0)=>GND_BUS_48(47 downto 0),
               RSTA=>RSTA_IN,
               RSTALLCARRYIN=>RSTALLCARRYIN_IN,
               RSTALUMODE=>GND_OPMODE,
               RSTB=>RSTB_IN,
               RSTC=>GND_OPMODE,
               RSTCTRL=>GND_OPMODE,
               RSTM=>RSTM_IN,
               RSTP=>GND_OPMODE,
               ACOUT(29 downto 0)=>ACOUT_OUT(29 downto 0),
               BCOUT(17 downto 0)=>BCOUT_OUT(17 downto 0),
               CARRYCASCOUT=>open,
               CARRYOUT=>open,
               MULTSIGNOUT=>open,
               OVERFLOW=>open,
               P(47 downto 36)=>P_float(11 downto 0),
               P(35 downto 0)=>P_OUT(35 downto 0),
               PATTERNBDETECT=>open,
               PATTERNDETECT=>open,
               PCOUT(47 downto 0)=>PCOUT_OUT(47 downto 0),
               UNDERFLOW=>open
               );
end BEHAVIORAL;
```

Основу представленного VHDL-описания образует примитив DSP48E, который был подробно рассмотрен в первой части данной статьи (см. КиТ № 12 ’2008). В структуре умножителя *multiplier_16_16_36* кроме входных и выходного регистров используется буферный регистр в цепи сигнала входного переноса. В каждом из этих буферных регистров предусмотрены раздельные входы сброса и разрешения синхронизации, которые включены в состав интерфейса сформированного элемента.

Система условных обозначений входных и выходных портов сгенерированного умножителя включает в себя следующие идентификаторы:

- ACIN_IN[15:0] — 16-разрядная входная шина данных, совокупность сигналов которой определяет значение первого сомножителя;
- B_IN[15:0] — 16-разрядная входная шина данных, совокупность сигналов которой определяет значение второго сомножителя;
- CARRYIN_IN — вход сигнала переноса;
- CEA1_IN — вход сигнала разрешения синхронизации для входного регистра, установленного на шине данных ACIN_IN;
- CEB1_IN — вход сигнала разрешения синхронизации для входного регистра, установленного на шине данных B_IN;
- CECARRYIN_IN — вход сигнала разрешения синхронизации для буферного регистра, установленного в цепи внешнего сигнала входного переноса;
- CEMULTCARRYIN_IN — вход сигнала разрешения синхронизации для буферного регистра, установленного в цепи внутреннего сигнала входного переноса;
- CEM_IN — вход сигнала разрешения синхронизации для буферного регистра, установленного на входе арифметическо-логического блока;
- CLK_IN — вход тактового сигнала;
- RSTALLCARRYIN_IN — вход сигнала сброса буферных регистров, задействованных в цепях входного переноса;
- RSTA_IN и RSTB_IN — входы сигналов сброса буферных регистров, установленных на входных шинах данных ACIN_IN и B_IN соответственно;
- RSTM_IN — вход сигнала сброса буферного регистра, установленного на входе арифметическо-логического блока;
- ACOUT_OUT[29:0] — 30-разрядная выходная шина данных, предназначенная для подключения к входной шине ACIN_IN следующего модуля ЦОС при каскадном соединении;

- BCOUT_OUT[17:0] — 18-разрядная выходная шина данных, предназначенная для подключения к входной шине данных B_IN следующего модуля ЦОС при каскадном соединении;
- PCOUT_OUT[47:0] — 48-разрядная выходная шина данных, применяемая при каскадном соединении умножителей;
- P_OUT[35:0] — выходная 36-разрядная шина данных, представляющая значение произведения.

Для установки требуемых значений всех необходимых атрибутов библиотечного примитива DSP48E, на основе которого выполнено VHDL-описание умножителя *multiplier_16_16_36*, «мастером» Architecture Wizard сгенерирован файл с расширением ucf, содержащий следующие выражения:

```
# Generated by Xilinx Architecture Wizard
# --- UCF Template Only ---
# Cut and paste these attributes into the project's UCF file, if desired
INST DSP48E_INST ACASCREG = 1;
INST DSP48E_INST ALUMODEREG = 0;
INST DSP48E_INST AREG = 1;
INST DSP48E_INST AUTORESET_PATTERN_DETECT = FALSE;
INST DSP48E_INST AUTORESET_PATTERN_DETECT_OPTINV =
MATCH;
INST DSP48E_INST A_INPUT = CASCADE;
INST DSP48E_INST BCASCREG = 1;
INST DSP48E_INST BREG = 1;
INST DSP48E_INST B_INPUT = DIRECT;
INST DSP48E_INST CARRYINREG = 1;
INST DSP48E_INST CARRYINSELREG = 0;
INST DSP48E_INST CREG = 0;
INST DSP48E_INST MASK = 3FFFFFFFFFFF;
INST DSP48E_INST MREG = 1;
INST DSP48E_INST MULTCARRYINREG = 1;
INST DSP48E_INST OPMODEREG = 0;
INST DSP48E_INST PATTERN = 000000000000;
INST DSP48E_INST PREG = 0;
INST DSP48E_INST SEL_MASK = MASK;
INST DSP48E_INST SEL_PATTERN = PATTERN;
INST DSP48E_INST SEL_ROUNDING_MASK = SEL_MASK;
INST DSP48E_INST USE_MULT = MULT_S;
INST DSP48E_INST USE_PATTERN_DETECT = NO_PATDET;
INST DSP48E_INST USE_SIMD = ONE48;
```

Пример описания умножителя с двухступенчатой структурой конвейерной обработки входных и выходных данных, сформированного с помощью «мастера» Architecture Wizard, для реализации на базе аппаратных модулей DSP48E в ПЛИС FPGA серии Virtex-5

Примером умножителя с двухступенчатой структурой конвейерной обработки входных и выходных данных, сформированного с помощью «мастера» Architecture Wizard, для последующей реализации на базе аппаратных модулей DSP48E в ПЛИС FPGA серии Virtex-5 является описание элемента *multiplier_d_24_18_36*. Этот умножитель предназначен для вычисления произведения двух сомножителей, один из которых представлен в виде 24-разрядного двоичного кода, а второй — 18-разрядного. В качестве входных портов, определяющих значения сомножителей в элементе *multiplier_d_24_18_36*, используются входные шины данных A и B. В состав интерфейса сформированного умножителя включены раздельные входы сброса и разрешения синхронизации для каждого входного, выходного и буферного регистра. В архитектуре элемента *multiplier_d_24_18_36* предусмотрены дополнительные выходные шины ACOUT, BCOUT и PCOUT, позволяющие применять этот умножитель в качестве одного из компонентов каскадного соединения аппаратных модулей ЦОС.

Текст сгенерированного VHDL-описания умножителя *multiplier_d_24_18_36* выглядит следующим образом:

```
--Command:  xaw2vhdl-st  D:\PRJ\multiplier_d_24_18_36.xaw
D:\PRJ\multiplier_d_24_18_36
--Design Name: multiplier_d_24_18_36
--Device: xc5vlx110t-ff1136-3
--
-- Module multiplier_d_24_18_36
-- Generated by Xilinx Architecture Wizard
-- Written for synthesis tool: XST
--
library ieee;
use ieee.std_logic_1164.ALL;
use ieee.numeric_std.ALL;
library UNISIM;
use UNISIM.Vcomponents.ALL;
--
entity multiplier_d_24_18_36 is
   port (
         A_IN : in std_logic_vector (23 downto 0);
         B_IN : in std_logic_vector (17 downto 0);
         CARRYIN_IN : in std_logic;
         CEA1_IN : in std_logic;
         CEA2_IN : in std_logic;
         CEB1_IN : in std_logic;
         CEB2_IN : in std_logic;
         CECARRYIN_IN : in std_logic;
         CEMULTCARRYIN_IN : in std_logic;
         CEM_IN : in std_logic;
         CEP_IN : in std_logic;
         CLK_IN : in std_logic;
         RSTALLCARRYIN_IN : in std_logic;
         RSTA_IN : in std_logic;
         RSTB_IN : in std_logic;
         RSTM_IN : in std_logic;
         RSTP_IN : in std_logic;
         ACOUT_OUT : out std_logic_vector (29 downto 0);
         BCOUT_OUT : out std_logic_vector (17 downto 0);
         PCOUT_OUT : out std_logic_vector (47 downto 0);
         P_OUT : out std_logic_vector (35 downto 0)
         );
end multiplier_d_24_18_36;
--
architecture BEHAVIORAL of multiplier_d_24_18_36 is
   signal GND_A : std_logic_vector (4 downto 0);
   signal GND_BUS_3 : std_logic_vector (2 downto 0);
   signal GND_BUS_4 : std_logic_vector (3 downto 0);
   signal GND_BUS_18 : std_logic_vector (17 downto 0);
   signal GND_BUS_30 : std_logic_vector (29 downto 0);
   signal GND_BUS_48 : std_logic_vector (47 downto 0);
   signal GND_OPMODE : std_logic;
   signal P_float : std_logic_vector (11 downto 0);
   signal VCC_OPMODE : std_logic;
begin
   GND_A(4 downto 0) <= «00000»;
   GND_BUS_3(2 downto 0) <= «000»;
   GND_BUS_4(3 downto 0) <= «0000»;
   GND_BUS_18(17 downto 0) <= «000000000000000000»;
   GND_BUS_30(29 downto 0) <= «00000000000000000000000000000»;
   GND_BUS_48(47 downto 0) <=
         «000000000000000000000000000000000000000000000000»;
   GND_OPMODE <= '0';
   VCC_OPMODE <= '1';
   DSP48E_INST : DSP48E
   generic map(
               ACASCREG => 1,
               ALUMODEREG => 0,
               AREG => 2,
               AUTORESET_PATTERN_DETECT => FALSE,
               AUTORESET_PATTERN_DETECT_OPTINV => «MATCH»,
               A_INPUT => «DIRECT»,
               BCASCREG => 1,
               BREG => 2,
               B_INPUT => «DIRECT»,
               CARRYINREG => 1,
               CARRYINSELREG => 0,
               CREG => 0,
               MASK => x»3FFFFFFFFFFF»,
               MREG => 1,
               MULTCARRYINREG => 1,
               OPMODEREG => 0,
               PATTERN => x»000000000000»,
               PREG => 1,
               SEL_MASK => «MASK»,
               SEL_PATTERN => «PATTERN»,
               SEL_ROUNDING_MASK => «SEL_MASK»,
               USE_MULT => «MULT_S»,
               USE_PATTERN_DETECT => «NO_PATDET»,
               USE_SIMD => «ONE48»
               )
      port map (
               A(29 downto 25)=>GND_A(4 downto 0),
               A(24)=>A_IN(23),
               A(23 downto 0)=>A_IN(23 downto 0),
               ACIN(29 downto 0)=>GND_BUS_30(29 downto 0),
               ALUMODE(3 downto 0)=>GND_BUS_4(3 downto 0),
               B(17 downto 0)=>B_IN(17 downto 0),
               BCIN(17 downto 0)=>GND_BUS_18(17 downto 0),
               C(47 downto 0)=>GND_BUS_48(47 downto 0),
               CARRYCASCIN=>GND_OPMODE,
               CARRYIN=>CARRYIN_IN,
               CARRYINSEL(2 downto 0)=>GND_BUS_3(2 downto 0),
               CEALUMODE=>VCC_OPMODE,
               CEA1=>CEA1_IN,
               CEA2=>CEA2_IN,
               CEB1=>CEB1_IN,
               CEB2=>CEB2_IN,
               CEC=>VCC_OPMODE,
               CECARRYIN=>CECARRYIN_IN,
               CECTRL=>VCC_OPMODE,
               CEM=>CEM_IN,
               CEMULTCARRYIN=>CEMULTCARRYIN_IN,
               CEP=>CEP_IN,
               CLK=>CLK_IN,
               MULTSIGNIN=>GND_OPMODE,
               OPMODE(6)=>GND_OPMODE,
               OPMODE(5)=>GND_OPMODE,
               OPMODE(4)=>GND_OPMODE,
               OPMODE(3)=>GND_OPMODE,
               OPMODE(2)=>VCC_OPMODE,
               OPMODE(1)=>GND_OPMODE,
               OPMODE(0)=>VCC_OPMODE,
               PCIN(47 downto 0)=>GND_BUS_48(47 downto 0),
               RSTA=>RSTA_IN,
               RSTALLCARRYIN=>RSTALLCARRYIN_IN,
               RSTALUMODE=>GND_OPMODE,
               RSTB=>RSTB_IN,
               RSTC=>GND_OPMODE,
               RSTCTRL=>GND_OPMODE,
               RSTM=>RSTM_IN,
               RSTP=>RSTP_IN,
               ACOUT(29 downto 0)=>ACOUT_OUT(29 downto 0),
               BCOUT(17 downto 0)=>BCOUT_OUT(17 downto 0),
               CARRYCASCOUT=>open,
               CARRYOUT=>open,
               MULTSIGNOUT=>open,
               OVERFLOW=>open,
               P(47 downto 36)=>P_float(11 downto 0),
               P(35 downto 0)=>P_OUT(35 downto 0),
               PATTERNBDETECT=>open,
               PATTERNDETECT=>open,
               PCOUT(47 downto 0)=>PCOUT_OUT(47 downto 0),
               UNDERFLOW=>open
               );
--
end BEHAVIORAL;
```

Система условных обозначений входных и выходных портов элемента *multiplier_d_24_18_36* отличается от совокупности идентификаторов, используемых в описании интерфейса умножителя *multiplier_16_16_36*, который был рассмотрен в предыдущем разделе, наличием дополнительных наименований входов. В описании интерфейса элемента *multiplier_d_24_18_36* дополнительно применяются следующие идентификаторы:

- A_IN[23:0] — 24-разрядная входная шина данных, совокупность сигналов которой определяет значение первого сомножителя;
- CEA2_IN — вход сигнала разрешения синхронизации для второго буферного регистра, установленного на входной шине данных A;
- CEB2_IN — вход сигнала разрешения синхронизации для второго буферного регистра, установленного на входной шине данных B;

- CEP_IN — вход сигнала разрешения синхронизации для регистра, установленного на выходе арифметическо-логического блока;
- RSTP_IN — вход сигнала сброса для регистра, установленного на выходе арифметическо-логического блока.

При использовании представленного VHDL-описания умножителя *multiplier_d_24_18_36* в составе разрабатываемого проекта САПР серии Xilinx ISE следует включить в файл временных и топологических ограничений этого проекта приведенную далее совокупность выражений:

```
# Generated by Xilinx Architecture Wizard
# --- UCF Template Only ---
# Cut and paste these attributes into the project's UCF file, if desired
INST DSP48E_INST ACASCREG = 1;
INST DSP48E_INST ALUMODEREG = 0;
INST DSP48E_INST AREG = 2;
INST DSP48E_INST AUTORESET_PATTERN_DETECT = FALSE;
INST DSP48E_INST AUTORESET_PATTERN_DETECT_OPTINV =
MATCH;
INST DSP48E_INST A_INPUT = DIRECT;
INST DSP48E_INST BCASCREG = 1;
INST DSP48E_INST BREG = 2;
INST DSP48E_INST B_INPUT = DIRECT;
INST DSP48E_INST CARRYINREG = 1;
INST DSP48E_INST CARRYINSELREG = 0;
INST DSP48E_INST CREG = 0;
INST DSP48E_INST MASK = 3FFFFFFFFFFF;
INST DSP48E_INST MREG = 1;
INST DSP48E_INST MULTCARRYINREG = 1;
INST DSP48E_INST OPMODEREG = 0;
INST DSP48E_INST PATTERN = 000000000000;
INST DSP48E_INST PREG = 1;
INST DSP48E_INST SEL_MASK = MASK;
INST DSP48E_INST SEL_PATTERN = PATTERN;
INST DSP48E_INST SEL_ROUNDING_MASK = SEL_MASK;
INST DSP48E_INST USE_MULT = MULT_S;
INST DSP48E_INST USE_PATTERN_DETECT = NO_PATDET;
INST DSP48E_INST USE_SIMD = ONE48;
```

Пример описания умножителя значений входных данных на константу, подготовленного с помощью «мастера» Architecture Wizard, для реализации на базе аппаратных модулей DSP48E в ПЛИС FPGA серии Virtex-5

Элементы, выполняющие операцию умножения значений входных данных на постоянный коэффициент, широко применяются в устройствах цифровой обработки сигналов. В частности, умножители на константу являются наиболее часто используемыми компонентами цифровых фильтров. «Мастер» Architecture Wizard предоставляет возможность быстрой подготовки необходимого количества описаний таких умножителей с заданными коэффициентами, реализуемых на базе аппаратных секций DSP48E в ПЛИС FPGA семейств Virtex-5 LX, Virtex-5 LXT, Virtex-5 SXT, Virtex-5 FXT и Virtex-5 TXT. При необходимости значения коэффициентов можно оперативно изменять в процессе разработки проекта устройства ЦОС. Для этого следует повторно запустить «мастер» Architecture Wizard и указать новое значение коэффициента. Можно также задать новое значение константы непосредственно в сформированном описании умножителя в текстовом режиме, используя встроенный HDL-редактор САПР серии Xilinx ISE.

Результат применения «мастера» Architecture Wizard для формирования описаний умножителей входных данных на постоянное значение, предназначенных для реализации на основе аппаратных секций DSP48E в ПЛИС серии Virtex-5, продемонстрирован в настоящем разделе на примере элемента *constant_multiplier_14_18_36*. Этот умножитель вычисляет значение произведения 18-разрядного двоичного кода, поступающего на входную шину данных BCIN, и заданного постоянного коэффициента, равного 15. Архитектура сгенерированного умножителя выполнена с применением двухступенчатой структуры конвейерной обработки входных и выходных данных. При этом входы управления всех задействованных конвейерных регистров представлены в виде соответствующих портов в составе интерфейса элемента *constant_multiplier_14_18_36*.

Текст VHDL-описания умножителя входных данных на заданную константу, сформированный с помощью «мастера» Architecture Wizard, имеет следующий вид.

```
--Command: xaw2vhdl-st D:\PRJ\constant_multiplier_14_18_36.xaw
D:\PRJ\constant_multiplier_14_18_36
--Design Name: constant_multiplier_14_18_36
--Device: xc5vfx100t-ff1136-3
--
-- Module constant_multiplier_14_18_36
-- Generated by Xilinx Architecture Wizard
-- Written for synthesis tool: XST
--
library ieee;
use ieee.std_logic_1164.ALL;
use ieee.numeric_std.ALL;
library UNISIM;
use UNISIM.Vcomponents.ALL;
--
entity constant_multiplier_14_18_36 is
   port (
         BCIN_IN : in std_logic_vector (17 downto 0);
         CARRYIN_IN : in std_logic;
         CEA1_IN : in std_logic;
         CEA2_IN : in std_logic;
         CEB1_IN : in std_logic;
         CEB2_IN : in std_logic;
         CECARRYIN_IN : in std_logic;
         CEMULTCARRYIN_IN : in std_logic;
         CEM_IN : in std_logic;
         CEP_IN : in std_logic;
         CLK_IN : in std_logic;
         RSTALLCARRYIN_IN : in std_logic;
         RSTA_IN : in std_logic;
         RSTB_IN : in std_logic;
         RSTM_IN : in std_logic;
         RSTP_IN : in std_logic;
         ACOUT_OUT : out std_logic_vector (29 downto 0);
         BCOUT_OUT : out std_logic_vector (17 downto 0);
         PCOUT_OUT : out std_logic_vector (47 downto 0);
         P_OUT : out std_logic_vector (35 downto 0)
         );
end constant_multiplier_14_18_36;
--
architecture BEHAVIORAL of constant_multiplier_14_18_36 is
   signal GND_A : std_logic_vector (4 downto 0);
   signal GND_BUS_3 : std_logic_vector (2 downto 0);
   signal GND_BUS_4 : std_logic_vector (3 downto 0);
   signal GND_BUS_18 : std_logic_vector (17 downto 0);
   signal GND_BUS_30 : std_logic_vector (29 downto 0);
   signal GND_BUS_48 : std_logic_vector (47 downto 0);
   signal GND_OPMODE : std_logic;
   signal P_float : std_logic_vector (11 downto 0);
   signal VCC_A : std_logic;
   signal VCC_OPMODE : std_logic;
begin
   GND_A(4 downto 0) <= «00000»;
   GND_BUS_3(2 downto 0) <= «000»;
   GND_BUS_4(3 downto 0) <= «0000»;
   GND_BUS_18(17 downto 0) <= «000000000000000000»;
   GND_BUS_30(29 downto 0) <= «000000000000000000000000000000»;
   GND_BUS_48(47 downto 0) <=
         «000000000000000000000000000000000000000000000000»;
   GND_OPMODE <= '0';
   VCC_A <= '1';
   VCC_OPMODE <= '1';
   DSP48E_INST : DSP48E
   generic map(
            ACASCREG => 1,
            ALUMODEREG => 0,
            AREG => 2,
            AUTORESET_PATTERN_DETECT => FALSE,
            AUTORESET_PATTERN_DETECT_OPTINV => «MATCH»,
            A_INPUT => «DIRECT»,
            BCASCREG => 1,
            BREG => 2,
            B_INPUT => «CASCADE»,
            CARRYINREG => 1,
            CARRYINSELREG => 0,
            CREG => 0,
            MASK => x«3FFFFFFFFFFF»,
            MREG => 1,
            MULTCARRYINREG => 1,
            OPMODEREG => 0,
            PATTERN => x«000000000000»,
            PREG => 1,
            SEL_MASK => «MASK»,
            SEL_PATTERN => «PATTERN»,
            SEL_ROUNDING_MASK => «SEL_MASK»,
            USE_MULT => «MULT_S»,
            USE_PATTERN_DETECT => «NO_PATDET»,
            USE_SIMD => «ONE48»
            )
      port map (
            A(29 downto 25)=>GND_A(4 downto 0),
            A(24)=>GND_A(0),
            A(23)=>GND_A(0),
            A(22)=>GND_A(0),
            A(21)=>GND_A(0),
            A(20)=>GND_A(0),
            A(19)=>GND_A(0),
            A(18)=>GND_A(0),
            A(17)=>GND_A(0),
            A(16)=>GND_A(0),
            A(15)=>GND_A(0),
            A(14)=>GND_A(0),
            A(13)=>GND_A(0),
            A(12)=>GND_A(0),
            A(11)=>GND_A(0),
            A(10)=>GND_A(0),
            A(9)=>GND_A(0),
            A(8)=>GND_A(0),
            A(7)=>GND_A(0),
            A(6)=>GND_A(0),
            A(5)=>GND_A(0),
            A(4)=>GND_A(0),
            A(3)=>VCC_A,
            A(2)=>VCC_A,
            A(1)=>VCC_A,
            A(0)=>VCC_A,
            ACIN(29 downto 0)=>GND_BUS_30(29 downto 0),
            ALUMODE(3 downto 0)=>GND_BUS_4(3 downto 0),
            B(17 downto 0)=>GND_BUS_18(17 downto 0),
            BCIN(17 downto 0)=>BCIN_IN(17 downto 0),
            C(47 downto 0)=>GND_BUS_48(47 downto 0),
            CARRYCASCIN=>GND_OPMODE,
            CARRYIN=>CARRYIN_IN,
            CARRYINSEL(2 downto 0)=>GND_BUS_3(2 downto 0),
            CEALUMODE=>VCC_OPMODE,
            CEA1=>CEA1_IN,
            CEA2=>CEA2_IN,
            CEB1=>CEB1_IN,
            CEB2=>CEB2_IN,
            CEC=>VCC_OPMODE,
            CECARRYIN=>CECARRYIN_IN,
            CECTRL=>VCC_OPMODE,
            CEM=>CEM_IN,
            CEMULTCARRYIN=>CEMULTCARRYIN_IN,
            CEP=>CEP_IN,
            CLK=>CLK_IN,
            MULTSIGNIN=>GND_OPMODE,
            OPMODE(6)=>GND_OPMODE,
            OPMODE(5)=>GND_OPMODE,
            OPMODE(4)=>GND_OPMODE,
            OPMODE(3)=>GND_OPMODE,
            OPMODE(2)=>VCC_OPMODE,
            OPMODE(1)=>GND_OPMODE,
            OPMODE(0)=>VCC_OPMODE,
            PCIN(47 downto 0)=>GND_BUS_48(47 downto 0),
            RSTA=>RSTA_IN,
            RSTALLCARRYIN=>RSTALLCARRYIN_IN,
            RSTALUMODE=>GND_OPMODE,
            RSTB=>RSTB_IN,
            RSTC=>GND_OPMODE,
            RSTCTRL=>GND_OPMODE,
            RSTM=>RSTM_IN,
            RSTP=>RSTP_IN,
            ACOUT(29 downto 0)=>ACOUT_OUT(29 downto 0),
            BCOUT(17 downto 0)=>BCOUT_OUT(17 downto 0),
            CARRYCASCOUT=>open,
            CARRYOUT=>open,
            MULTSIGNOUT=>open,
            OVERFLOW=>open,
```

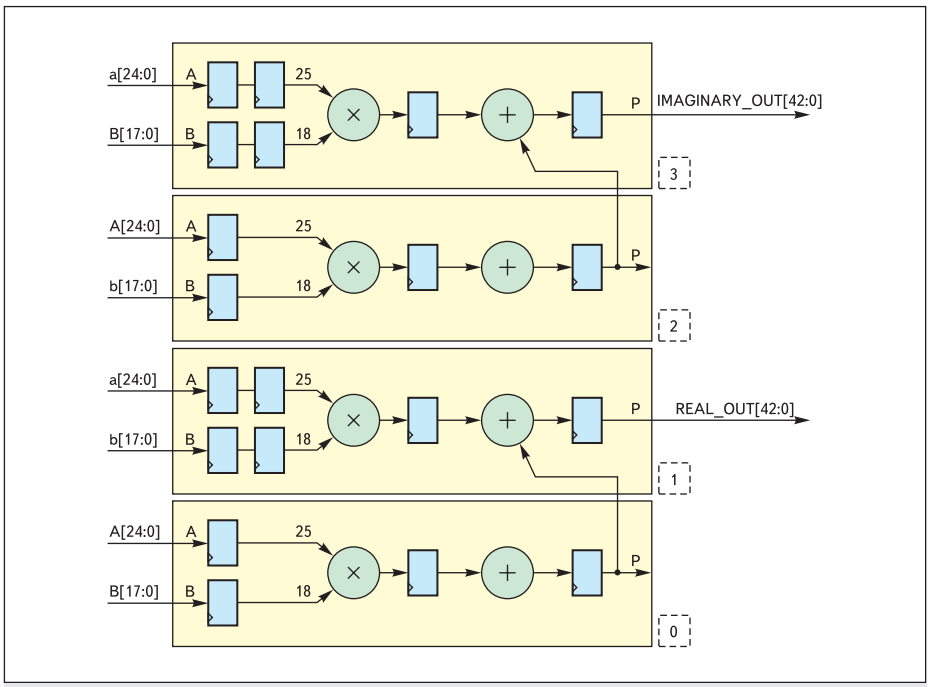

Рис. 28. Структурная схема комплексного умножителя, предназначенного для реализации на базе аппаратных секций DSP48E, выполненная с использованием элементов, формируемых с помощью «мастера» Architecture Wizard

```
            P(47 downto 36)=>P_float(11 downto 0),
            P(35 downto 0)=>P_OUT(35 downto 0),
            PATTERNBDETECT=>open,
            PATTERNDETECT=>open,
            PCOUT(47 downto 0)=>PCOUT_OUT(47 downto 0),
            UNDERFLOW=>open
            );
end BEHAVIORAL;
```

В описании интерфейса элемента *constant_multiplier_14_18_36* применяется та же система условных обозначений входных и выходных портов, что и для умножителей *multiplier_16_16_36* и *multiplier_d_24_18_36*, которые были рассмотрены в предыдущих разделах. Кроме того, при описании интерфейса умножителя на постоянный коэффициент *constant_multiplier_14_18_36* используется идентификатор BCIN_IN[17:0] для обозначения входной 18-разрядной шины данных.

Значение постоянного коэффициента в описании сгенерированного умножителя задается в двоичном формате в виде высокого и низкого логических уровней сигнала на соответствующих линиях входной шины A библиотечного примитива DSP48E. Значение логической единицы формируется путем подключения соответствующего проводника этой шины к цепи сигнала VCC_A. Для формирования значения логического нуля в каком-либо разряде входного порта A соответствующая линия данной шины должна быть присоединена к цепи сигнала GND_A(0).

В случае включения элемента *constant_multiplier_14_18_36* в качестве одного из компонентов в состав описания разрабатываемого устройства ЦОС следует поместить в файл временных и топологических ограничений проекта САПР серии Xilinx ISE приведенную далее последовательность выражений, которые определяют значения основных атрибутов используемого примитива DSP48E:

```
# Generated by Xilinx Architecture Wizard
# --- UCF Template Only ---
# Cut and paste these attributes into the project's UCF file, if desired
INST DSP48E_INST ACASCREG = 1;
INST DSP48E_INST ALUMODEREG = 0;
INST DSP48E_INST AREG = 2;
INST DSP48E_INST AUTORESET_PATTERN_DETECT = FALSE;
INST DSP48E_INST AUTORESET_PATTERN_DETECT_OPTINV =
MATCH;
INST DSP48E_INST A_INPUT = DIRECT;
INST DSP48E_INST BCASCREG = 1;
INST DSP48E_INST BREG = 2;
INST DSP48E_INST B_INPUT = CASCADE;
INST DSP48E_INST CARRYINREG = 1;
INST DSP48E_INST CARRYINSELREG = 0;
INST DSP48E_INST CREG = 0;
INST DSP48E_INST MASK = 3FFFFFFFFFFF;
INST DSP48E_INST MREG = 1;
INST DSP48E_INST MULTCARRYINREG = 1;
INST DSP48E_INST OPMODEREG = 0;
INST DSP48E_INST PATTERN = 000000000000;
INST DSP48E_INST PREG = 1;
INST DSP48E_INST SEL_MASK = MASK;
INST DSP48E_INST SEL_PATTERN = PATTERN;
INST DSP48E_INST SEL_ROUNDING_MASK = SEL_MASK;
INST DSP48E_INST USE_MULT = MULT_S;
INST DSP48E_INST USE_PATTERN_DETECT = NO_PATDET;
INST DSP48E_INST USE_SIMD = ONE48;
```

Разработка высокоскоростных комплексных умножителей, реализуемых на базе аппаратных секций DSP48E в ПЛИС FPGA серии Virtex-5, с использованием элементов, формируемых с помощью «мастера» Architecture Wizard САПР серии Xilinx ISE

Наряду с обычными (вещественными) умножителями при разработке устройств цифровой обработки сигналов широко применяются элементы, выполняющие операцию вычисления произведения двух комплексных значений. Для достижения максимального быстродействия устройств ЦОС, реализуемых на базе ПЛИС FPGA серии Virtex-5, комплексные умножители наиболее эффективно выполняются на основе аппаратных секций DSP48E. При этом в качестве компонентов таких умножителей целесообразно использовать описания элементов, генерируемых с помощью «мастера» Architecture Wizard.

Операция умножения двух комплексных значений в общем случае может быть представлена в следующем виде:

$$(A+ja)\times(B+jb) = (AB-ab)+j(Ab+Ba), \quad (1)$$

где A и a — значения вещественной и мнимой части первого сомножителя соответственно; B и b — значения вещественной и мнимой части второго сомножителя соответственно.

Таким образом, для реализации устройства, вычисляющего произведение двух комплексных значений, необходимы два обычных умножителя, процесс подготовки которых был рассмотрен в предыдущих разделах, и элементы, выполняющие операцию умножения со сложением и вычитанием. Структурная схема комплексного умножителя, предназначенного для реализации на базе аппаратных секций DSP48E, которая выполнена с использованием элементов, формируемых с помощью «мастера» Architecture Wizard, приведена на рис. 28. Данный умножитель вычисляет значение произведения двух комплексных значений, представленных в 25- и 18-разрядном двоичном коде.

Процесс формирования описаний элементов, осуществляющих операцию умножения с накоплением, для последующей реализации на базе аппаратных секций DSP48E будет рассмотрен в следующей части данной статьи.

Окончание следует

Литература

1. Кузелин М. О., Кнышев Д. А., Зотов В. Ю. Современные семейства ПЛИС фирмы Xilinx. Справочное пособие. М.: Горячая линия – Телеком, 2004.
2. Зотов В. Инструментальный модуль компании Avnet для отладки проектов встраиваемых систем, разрабатываемых на базе нового семейства ПЛИС FPGA фирмы Xilinx Virtex-5 FXT // Компоненты и технологии. 2008. № 9.
3. Зотов В. Ю. Проектирование цифровых устройств на основе ПЛИС фирмы Xilinx в САПР WebPack ISE. М.: Горячая линия – Телеком, 2003.
4. Зотов В. Ю. Проектирование встраиваемых микропроцессорных систем на основе ПЛИС фирмы Xilinx. М.: Горячая линия – Телеком, 2006.
5. Virtex-5 FPGA XtremeDSP Design Considerations. User Guide. Xilinx, 2009.
6. Virtex-5 Family Overview. Xilinx, 2009.
7. Virtex-5 FPGA User Guide. Xilinx, 2009.