Лабораторная работа 6

Синтаксические конструкции

Заготовка для двоичного синхронного суммирующего счетчика

```
process (clk) begin
      if rising_edge(clk) then
            if srst= '1' then
                  counter <= (others => '0');
            else
                  counter <= unsigned(counter) + 1;
            end if;
      end if;
end process;
```

Заготовка для двоичного синхронного реверсивного счетчика

```
process (clk) begin
      if rising_edge(clk) then
            if srst='1' then
                  counter <= (others => '0');
            elsif mode='0' then
                  counter <= unsigned(counter) + 1;
            else
                  counter <= unsigned(counter) - 1;
            end if;
      end if;
end process;
```

Заготовка для двоично-десятичного синхронного суммирующего счетчика от 6 до 9:

```
process (clk) begin
      if rising_edge(clk) then
            if srst= '1' then
                  counter <= "110";
            elsif unsigned(counter)=9 then
                  counter <= "110";
            else
                  counter <= unsigned(counter) + 1;
            end if;
      end if;
end process;
```

Задание к лабораторной работе 6

Реализовать различные типы счетчиков. В качестве входов использовать кнопки и/или слайдеры, в качестве выходов – светодиоды и семисегментный индикатор. Одним из входов разработанного устройства должен быть сигнал тактирования, корректно описанный в XDC файле.

Необходимо реализовать:

1. Реализовать 30-разрядный синхронный двоичный суммирующий счетчик по модулю 2^{30} с синхронным сбросом.
   Цифровое устройство должно иметь один вход для сброса счетчика (нижняя кнопка), один вход для сигнала тактирования и восемь информационных выходов (светодиоды). На выходные порты необходимо выводить восемь старших разрядов счетчика при проверке на плате и разряды с 15 по 8 для автопроверки.

Обратить внимание на то, что после сброса счетчик считает непрерывно. По достижении максимального значения (все единицы) он сам сбрасывается в ноль.

Использовать следующий шаблон для объявления интерфейса модуля и архитектуры:

```
-------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.std_logic_arith.all;

entity lab61 is
    Port ( clk : in STD_LOGIC;
           srst : in STD_LOGIC;
           dout : out STD_LOGIC_VECTOR (7 downto 0));
end lab61;

architecture a of lab61 is
    signal cntr : std_logic_vector(29 downto 0);
begin
--    dout <= cntr(29 downto 22); -- for implementation
    dout <= cntr(15 downto 8);   -- for auto_check
...
-------------------------------------
```

2. На основе счетчика реализовать делитель частоты. Получить сигнал с периодом примерно 1 с.

Цифровое устройство должно иметь один вход для сигнала тактирования и один выход (светодиод). Выходной светодиод должен мигать с периодом примерно 1 с. Сигнал сброса не нужен.

Так же, как и в предыдущем пункте, источником периодического сигнала будет двоичный суммирующий счетчик. Обратите внимание, что требование “примерно 1 с” позволяет не выполнять анализ значения счетчика. Т.к. на каждом разряде двоичного суммирующего счетчика частота делится в два раза по сравнению с предыдущим, для выполнения задания необходимо на светодиод вывести сигнал с нужного разряда. Нужного – в смысле имеющего необходимый период примерно 1 с.

Для автопроверки период выходного сигнала должен быть 256 тактов, а скважность равна 2.

Использовать следующий шаблон для объявления интерфейса модуля и архитектуры:

```
-------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.std_logic_arith.all;

entity lab62 is
    Port ( clk : in STD_LOGIC;
           dout : out STD_LOGIC);
end lab62;

architecture a of lab62 is
    signal cntr : std_logic_vector(26 downto 0):=(others => '0');
begin
    dout <= cntr(26);   -- for implementation
--    dout <= cntr(7);   -- for auto_check
...
-------------------------------------
```

3. *Реализовать 30-разрядный синхронный двоичный реверсивный счетчик по модулю 2^{30} с синхронным сбросом.

Цифровое устройство должно иметь один вход для сброса счетчика (нижняя кнопка), вход для сигнала тактирования, вход для выбора направления счета (mode, 0 – суммирующий счетчик, 1 – вычитающий счетчик) и восемь информационных выходов (светодиоды). На выходные порты

необходимо выводить восемь старших разрядов счетчика при проверке на плате и разряды с 8 по 1 для автопроверки.

Использовать следующий шаблон для объявления интерфейса модуля и архитектуры:

```
-------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.std_logic_arith.all;

entity lab63 is
    Port ( clk : in STD_LOGIC;
           srst : in STD_LOGIC;
           mode : in STD_LOGIC;-- 0 - count up, 1 - count down
           dout : out STD_LOGIC_VECTOR (7 downto 0));
end lab63;

architecture a of lab63 is
    signal cntr : std_logic_vector(29 downto 0);
begin
--    dout <= cntr(29 downto 22); -- for implementation
    dout <= cntr(8 downto 1);   -- for auto_check.
..
-------------------------------------
```

4. **На основе делителя частоты и 8-разрядного циклического сдвигающего регистра реализовать кольцевой счетчик с периодом сдвига примерно 1 с.

Инициализация регистра должна выполняться по нажатию кнопки сброса, синхронно.
Период счета должен быть равен примерно 1 с (использовать исходный код второго задания). Цифровое устройство должно иметь вход для сигнала тактирования, вход для сброса регистра и счетчика в начальное состояние (нижняя кнопка) и восемь выходов (светодиоды) для отображения содержимого сдвигающего регистра.
Начальное значение для счетчика – все нули. Начальное значение для сдвигающего регистра – “00000001”. Сдвиг выполнять вправо (от MSB к LSB).

Это задание можно выполнить разными способами. Нельзя использовать старший разряд счетчика в качестве сигнала тактирования сдвигающего регистра (т.е. в конструкции if rising_edge(shifter(MSB)). Такая реализация будет работать, но является плохим стилем программирования. Настойчиво рекомендуется для создания новых тактовых частот применять специальные блоки в FPGA.
Для хорошей наглядности и читаемости кода и хорошего быстродействия (высоких допустимых частот работы) рекомендуется выполнить задание следующим образом. Один процесс должен содержать сдвигающий регистр со сбросом и сигналом разрешения сдвига. Еще один процесс должен содержать только сбрасываемый счетчик. И еще в одном процессе должен формироваться собственно сигнал разрешения сдвига. Его период совпадает с периодом 26-го разряд счетчика (примерно 1 с), но длительность велика, тогда как сдвиг разрешается только один раз/такт за примерно 1 с.
Для формирования сигнала разрешения для сдвигающего регистра рекомендуется использовать схему формирования строба из последнего разряда счетчика, описанную в конце предыдущей лабораторной работы.

Для автопроверки использовать не 26-й разряд счетчика а 1-й.

Использовать следующий шаблон для объявления интерфейса модуля и архитектуры:

```
-------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.std_logic_arith.all;

entity lab64 is
    Port ( clk : in STD_LOGIC;
           srst : in STD_LOGIC;
           dout : out STD_LOGIC_VECTOR (7 downto 0));
end lab64;
```

```
architecture a of lab64 is
    signal cntr    : std_logic_vector(26 downto 0);
    signal shifter : std_logic_vector(7 downto 0);
    signal cntr26d : std_logic; -- delayed MSB of shifter
    signal en      : std_logic;
begin
...
--    en <= cntr(26) and (not cntr26d); --For implementation
    en <= cntr(1) and (not cntr26d); -- For auto_check
    delay_proc : process(clk) begin
        if rising_edge(clk) then
--            cntr26d <= cntr(26); --For implementation
            cntr26d <= cntr(1); -- For auto_check
        end if;
    end process;
...
    dout <= shifter;
end a;
---------------------------------------
```