МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего профессионального образования

**Национальный исследовательский ядерный университет «МИФИ»**

Институт Интеллектуальных Кибернетических Систем

Отчет по лабораторной работе

Учебная дисциплина : «Схемотехника»

Тема: Реверсивный счётчик, 4 бит

Выполнили:

студенты группы С22-501

Тертышников Иван

Воробьев Егор

Ярошевский Михаил

Москва 2024

[1. Постановка задачи 3](#_Toc1)

[2. Спецификация 4](#_Toc2)

[2.1 Условное графическое обозначение и список портов ввода-вывода 4](#_Toc3)

[2.2 Описание рабочего режима 5](#_Toc4)

[3. Тестирование 6](#_Toc5)

[4. Результаты синтеза 7](#_Toc6)

5. Заключение………………………………………………………………………………………….8

1. Постановка задачи

В рамках данной работы требовалось реализовать на языке описания интегральных схем VHDL устройство четырёхбитного реверсивного счётчика с выбором начального значения.

Данное устройство представляет собой комбинационную схему, служащую для выполнения операции счёта как в прямом, так и в обратном направлении.

2. Спецификация

2.1 Условное графическое обозначение и список портов ввода-вывода

Условное графическое обозначение разрабатываемого реверсивного счётчика представлено на рисунке 1.

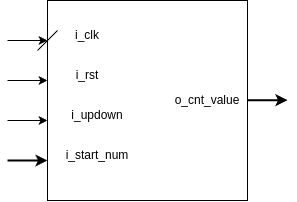


Рисунок 1 – Условное графического обозначение реверсивного счётчика.

Список портов ввода-вывода разрабатываемого устройства представлена в таблице 1.

Таблица 1 – Список портов ввода-вывода в разработанном реверсивном счётчике.

|  |  |
| --- | --- |
| Наименование | Описание |
| i\_clk | Тактовый сигнал |
| i\_rst | Сигнал сброса текущего значения счётчика (активный сигнал “1”) |
| i\_updown | Выбор унарной операции уменьшения или увеличения |
| i\_start\_num [3...0] | Задаваемое начальное значение |
| o\_cnt\_value [3...0] | Вывод текущего значения счётчика |

2.2 Описание рабочего режима

Разработанный реверсивный счётчик работает следующим образом:

Сначала осуществляется проверка на наличие сигнала сброса для текущего значения. Если сигнал сброса получен, текущее значение счётчика обновляется на новое, поступившее от устройства. В случае отсутствия сигнала сброса текущее значение изменяется на единицу в зависимости от выбранной унарной операции при каждом фронте тактового сигнала — оно может увеличиваться или уменьшаться. В завершение выводится текущее значение счётчика.

Предполагается, что в процессе работы устройства можно как выбрать унарную операцию, так и задать новое начальное значение.

Выбранная унарная операция начнёт действовать лишь при следующем фронте тактового сигнала.

3. Тестирование

В рамках данной работы требовалось провести тестирование разработанного устройства.

Были сформулированы следующие сценарии тестирования:

1. Задание начального значения в момент начала работы реверсивного счётчика.
2. Работы реверсивного счётчика в целом.
3. Изменение выбора унарной операции во время работы реверсивного счётчика.
4. Ввод нового значения во время работы реверсивного счётчика.

Задание начального значение проиллюстрировано на рисунке 2.

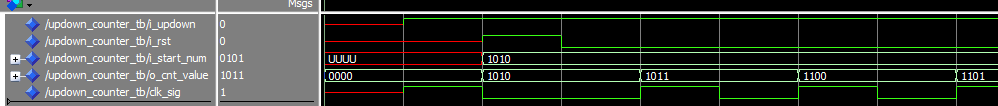


Рисунок 2 – Задание начального значения реверсивного счётчика.

Работа реверсивного счётчика в целом и изменение унарной операции во время его работы проиллюстрированы на рисунке 3.

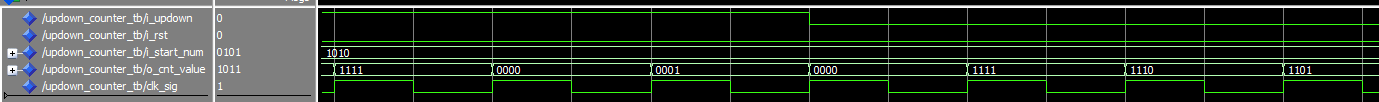


Рисунок 3 – Работа и изменение унарной операции.

Ввод нового значения во время работы проиллюстрирован на рисунке 4.

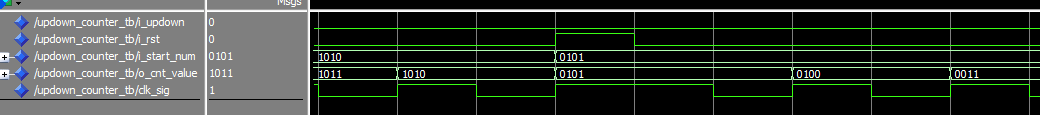


Рисунок 4 – Ввод нового значения во время работы.

4. Результаты синтеза

RTL схема по результатам синтеза проиллюстрирована на рисунке 5.

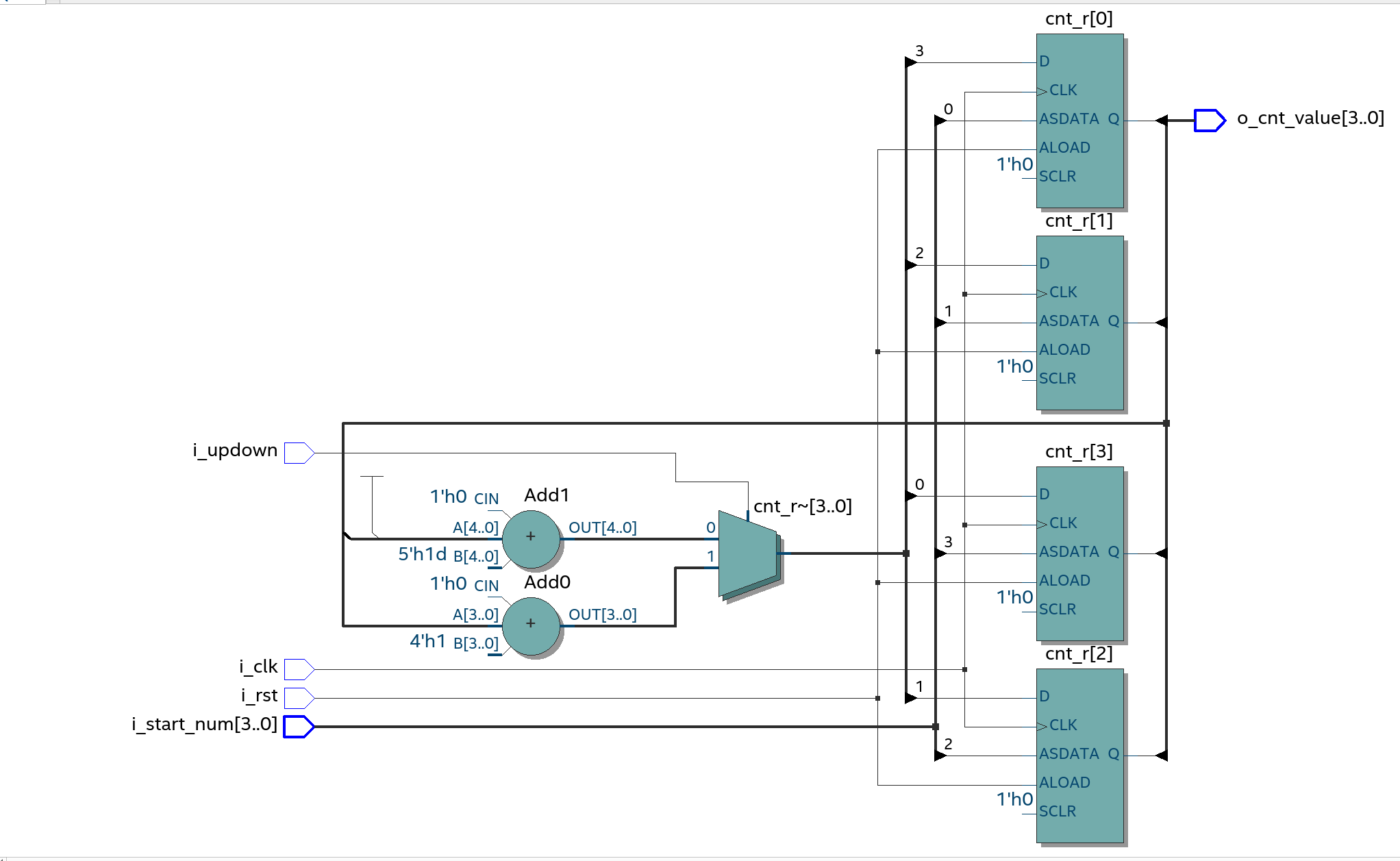


Рисунок 5 – RTL схема разработанного устройства.

Список затраченных ресурсов среды проиллюстрирован на рисунке 6.

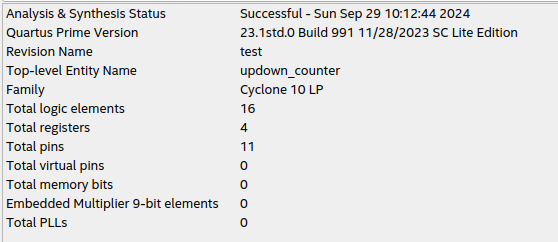


Рисунок 6 – Список затраченных ресурсов.

**5. Заключение**

В рамках данной работы были изучены основы языка описания интегральных схем VHDL, а также разработан полностью функционирующий реверсивный счетчик с выбором начального значения.

* Произведено тестирование устройства. Полученные результаты сошлись с ожидаемые.

Приложение

1.1 Файл updown\_counter.vhd

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity updown\_counter is

Port ( i\_clk, i\_rst, i\_updown : in STD\_LOGIC;

i\_start\_num : in STD\_LOGIC\_VECTOR (3 downto 0);

o\_cnt\_value : out STD\_LOGIC\_VECTOR (3 downto 0));

end updown\_counter;

architecture Behavioral of updown\_counter is

signal cnt\_r : std\_logic\_vector (3 downto 0) := "0000";

begin

process(i\_clk, i\_rst)

begin

if (i\_rst = '1') then

cnt\_r <= i\_start\_num;

elsif (rising\_edge(i\_clk)) then

if (i\_updown = '1') then

cnt\_r <= cnt\_r + 1;

else

cnt\_r <= cnt\_r - 1;

end if;

end if;

end process;

o\_cnt\_value <= cnt\_r;

end Behavioral;

1.2 Файл updown\_counter\_tester.vhd

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity updown\_counter\_tester is

port (

i\_clk, i\_rst, i\_updown : out STD\_LOGIC;

i\_start\_num : out STD\_LOGIC\_VECTOR (3 downto 0)

);

end updown\_counter\_tester;

architecture updown\_counter\_tester\_arch of updown\_counter\_tester is

constant num\_of\_clocks : integer := 50;

constant clk\_period : time := 10 ns;

signal clk\_sig : STD\_LOGIC := '0';

-- signal i : integer := 0;

procedure wait\_clk(constant j: in integer) is

variable ii: integer := 0;

begin

while ii < j loop

if (rising\_edge(clk\_sig)) then

ii := ii + 1;

end if;

wait for 10 ps;

end loop;

end;

begin

clk\_sig <= not clk\_sig after clk\_period / 2;

i\_clk <= clk\_sig;

process

begin

i\_updown <= '1';

wait\_clk(1);

i\_rst <= '1';

i\_start\_num <= "1010";

wait\_clk(1);

i\_rst <= '0';

wait\_clk(20);

i\_updown <= '0';

wait\_clk(1);

i\_rst <= '1';

i\_start\_num <= "0101";

wait\_clk(1);

i\_rst <= '0';

wait\_clk(20);

wait;

end process;

end updown\_counter\_tester\_arch;

1.3 Файл updown\_counter\_tb.vhd

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity updown\_counter\_tb is

end entity;

architecture updown\_counter\_tb\_arch of updown\_counter\_tb is

signal i\_updown : STD\_LOGIC;

signal i\_rst : STD\_LOGIC;

signal i\_start\_num : STD\_LOGIC\_VECTOR (3 downto 0);

signal o\_cnt\_value : STD\_LOGIC\_VECTOR (3 downto 0);

signal clk\_sig : STD\_LOGIC;

component updown\_counter is

port (i\_clk, i\_rst, i\_updown : in STD\_LOGIC;

i\_start\_num : in STD\_LOGIC\_VECTOR (3 downto 0);

o\_cnt\_value : out STD\_LOGIC\_VECTOR (3 downto 0));

end component;

component updown\_counter\_tester is

port (i\_clk, i\_rst, i\_updown : out STD\_LOGIC;

i\_start\_num : out STD\_LOGIC\_VECTOR (3 downto 0));

end component;

begin

t1: updown\_counter\_tester port map(

i\_clk => clk\_sig,

i\_rst => i\_rst,

i\_updown => i\_updown,

i\_start\_num => i\_start\_num

);

t2: updown\_counter port map(

i\_clk => clk\_sig,

i\_rst => i\_rst,

i\_updown => i\_updown,

i\_start\_num => i\_start\_num,

o\_cnt\_value => o\_cnt\_value

);

end updown\_counter\_tb\_arch;