Министерство науки и высшего образования Российской Федерации  
Пензенский государственный университет  
Кафедра вычислительная техника

**ОТЧЕТ**

по лабораторной работе №4

по дисциплине «Проектирование программного обеспечения для устройств на ПЛИС»

на тему «Разработка и отладка на языке VHDL специализированного процессора.»

Выполнили студенты группы 22ВВП1

Хоссейни Нежад С.А.С.М.

Беляев Д. И.

Демин М.

Приняли:

Федюнин Р.Н.

Гурин Е.И.

Пенза 2024

**Название**

Разработка и отладка на языке VHDL специализированного процессора.

**Цель работы**

Изучение принципов разработки специализированных процессоров на языке VHDL.

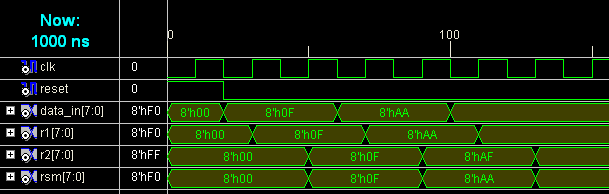
**Ход работы**

1. Разработать специализированное вычислительное устройство с заданным набором микроопераций на языке VHDL. Выполнить моделирование работы схемы.

|  |  |
| --- | --- |
| № | Набор микроопераций |
| 1,10 | R1:=ШД; R1:= R2; R3:= R1 V R2; RSM:= R1 + R2; RSM:= R3; |
| 2,11 | R1:=ШД; R1:= R2; R2:= R1; RSM:= R1 + R2; RSM:= R1 - R2; |
| 3,12 | R1:=ШД; R2:= ШД; R2:= R1 & R2; RSM:= R1 + R2; RSM:= R1; |
| 4 | R1:=ШД; R2:= R1; R2:= R1 V R2; RSM:= R1 + R2; RSM:= R1; |
| 5 | R1:=ШД; R2:= ШД; R3:= R1 & R2; RSM:= R1 + R3; RSM:= R2; |
| 6 | R1:=ШД; R2:= R1; R2:= R1 V R2; RSM:= R1 - R2; RSM:= ШД; |
| 7 | R1:=2; R2:= ШД; R2:= R1 V R2; RSM:= R1 - R2; RSM:= R2; |
| 8 | R1:= ШД; R2:= ШД; R3:= ШД; RSM:= R1 + R2; RSM:= R1 + R3; |
| 9 | R1:= ШД; R2:= R1 & R3; R3:= ШД; RSM:= R1 - R2; RSM:= R1 + R3; |

Разработать на языке VHDL файл внешних воздействий с последовательностью микроопераций, выполняющих заданные микрооперации. При необходимости добавить новые микрооперации. Выполнить моделирование работы схемы. Выполнить синтез схемы, просмотреть результаты синтеза.

Выполнить реализацию, просмотреть и включить в отчет результат размещения схемы в ПЛИС. Зафиксировать минимальный период работы схемы.



Листинг

**p1**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity p1 is

Port (

clk : in STD\_LOGIC; -- Тактовый сигнал

reset : in STD\_LOGIC; -- Сигнал сброса

data\_in : in STD\_LOGIC\_VECTOR(7 downto 0); -- Входные данные (ШД)

R1 : out STD\_LOGIC\_VECTOR(7 downto 0); -- Регистр R1

R2 : out STD\_LOGIC\_VECTOR(7 downto 0); -- Регистр R2

RSM : out STD\_LOGIC\_VECTOR(7 downto 0) -- Результат RSM

);

end p1;

architecture Behavioral of p1 is

signal reg\_R1 : STD\_LOGIC\_VECTOR(7 downto 0) := (others => '0');

signal reg\_R2 : STD\_LOGIC\_VECTOR(7 downto 0) := (others => '0');

signal reg\_RSM : STD\_LOGIC\_VECTOR(7 downto 0) := (others => '0');

begin

process(clk, reset)

begin

if reset = '1' then

reg\_R1 <= (others => '0');

reg\_R2 <= (others => '0');

reg\_RSM <= (others => '0');

elsif rising\_edge(clk) then

-- Пример последовательности микроопераций

-- 1. R1 := data\_in

reg\_R1 <= data\_in;

-- 2. R2 := R1

reg\_R2 <= reg\_R1;

-- 3. R2 := R1 OR R2

reg\_R2 <= reg\_R1 or reg\_R2;

-- 4. RSM := R1 + R2

reg\_RSM <= reg\_R1 + reg\_R2;

-- 5. RSM := R1

reg\_RSM <= reg\_R1;

end if;

end process;

-- Вывод значений регистров

R1 <= reg\_R1;

R2 <= reg\_R2;

RSM <= reg\_RSM;

end Behavioral;

**Test\_p1**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity Test\_p1 is

end Test\_p1;

architecture Behavioral of Test\_p1 is

-- Компонент устройства

component p1

Port (

clk : in STD\_LOGIC;

reset : in STD\_LOGIC;

data\_in : in STD\_LOGIC\_VECTOR(7 downto 0);

R1 : out STD\_LOGIC\_VECTOR(7 downto 0);

R2 : out STD\_LOGIC\_VECTOR(7 downto 0);

RSM : out STD\_LOGIC\_VECTOR(7 downto 0)

);

end component;

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

signal reset : STD\_LOGIC := '0';

signal data\_in : STD\_LOGIC\_VECTOR(7 downto 0) := (others => '0');

signal R1 : STD\_LOGIC\_VECTOR(7 downto 0);

signal R2 : STD\_LOGIC\_VECTOR(7 downto 0);

signal RSM : STD\_LOGIC\_VECTOR(7 downto 0);

begin

-- Инстанцирование устройства

UUT: p1

Port map (

clk => clk,

reset => reset,

data\_in => data\_in,

R1 => R1,

R2 => R2,

RSM => RSM

);

-- Генерация тактового сигнала

clk\_process: process

begin

clk <= '0';

wait for 10 ns;

clk <= '1';

wait for 10 ns;

end process;

-- Стимулы для моделирования

stimulus\_process: process

begin

-- Инициализация и сброс устройства

reset <= '1';

wait for 20 ns;

reset <= '0';

-- Подача данных на вход data\_in после сброса

data\_in <= "00001111"; -- Ввод данных 1

wait for 40 ns;

data\_in <= "10101010"; -- Ввод данных 2

wait for 40 ns;

data\_in <= "11110000"; -- Ввод данных 3

wait for 40 ns;

wait;

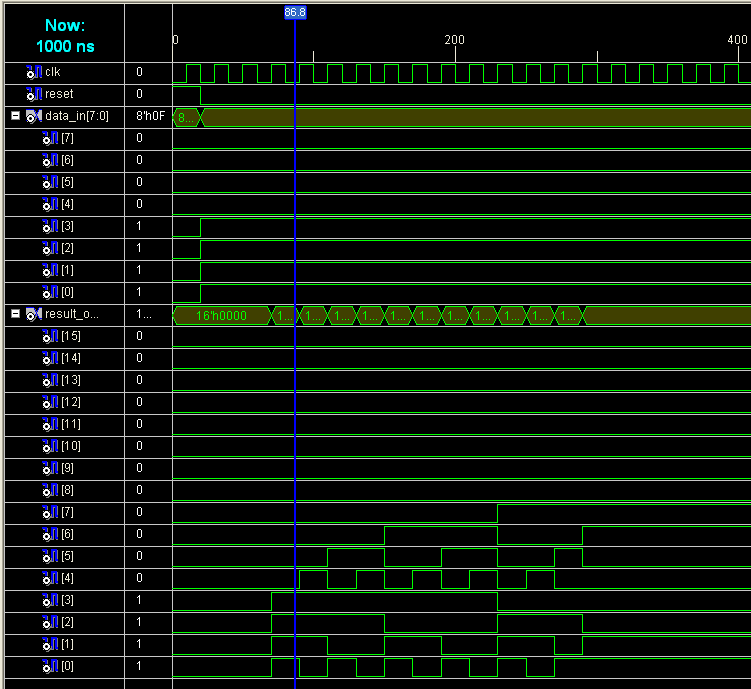
end process;

end Behavioral;

2. Разработать конвейерныйсумматор на языке VHDL. Выполнить моделирование работы схемы.

|  |  |  |
| --- | --- | --- |
| № | Разр данных | К-во слагаемых |
| 1,10 | 8 | 12 |
| 2,11 | 8 | 11 |
| 3,12 | 8 | 10 |
| 4 | 8 | 13 |
| 5 | 8 | 14 |
| 6 | 8 | 15 |
| 7 | 8 | 9 |
| 8 | 12 | 7 |
| 9 | 12 | 6 |

Выполнить синтез схемы, просмотреть результаты синтеза. Выполнить реализацию, просмотреть и включить в отчет результат размещения схемы в ПЛИС. Зафиксировать минимальный период работы схемы.



Листинг

**p2**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity p2 is

Port (

clk : in STD\_LOGIC; -- Тактовый сигнал

reset : in STD\_LOGIC; -- Сигнал сброса

data\_in : in STD\_LOGIC\_VECTOR(7 downto 0); -- Входные данные (8 бит)

result\_out: out STD\_LOGIC\_VECTOR(15 downto 0) -- Результат суммы (16 бит)

);

end p2;

architecture Behavioral of p2 is

-- Параметры для хранения промежуточных сумм на каждом этапе конвейера

type pipeline\_type is array (0 to 12) of STD\_LOGIC\_VECTOR(15 downto 0);

signal pipeline: pipeline\_type := (others => (others => '0'));

begin

process(clk, reset)

begin

if reset = '1' then

-- Сброс всех регистров конвейера

for i in 0 to 12 loop

pipeline(i) <= (others => '0');

end loop;

result\_out <= (others => '0');

elsif rising\_edge(clk) then

-- Заполнение первого регистра значением data\_in

pipeline(0) <= ("00000000" & data\_in); -- Расширение до 16 бит

-- Конвейерное суммирование

for i in 1 to 12 loop

pipeline(i) <= pipeline(i - 1) + pipeline(0);

end loop;

-- Вывод результата из последнего этапа конвейера

result\_out <= pipeline(12);

end if;

end process;

end Behavioral;

Test\_p2

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity Test\_p2 is

end Test\_p2;

architecture Behavioral of Test\_p2 is

-- Компонент устройства

component p2

Port (

clk : in STD\_LOGIC;

reset : in STD\_LOGIC;

data\_in : in STD\_LOGIC\_VECTOR(7 downto 0);

result\_out : out STD\_LOGIC\_VECTOR(15 downto 0)

);

end component;

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

signal reset : STD\_LOGIC := '0';

signal data\_in : STD\_LOGIC\_VECTOR(7 downto 0) := (others => '0');

signal result\_out : STD\_LOGIC\_VECTOR(15 downto 0);

begin

-- Инстанцирование устройства

UUT: p2

Port map (

clk => clk,

reset => reset,

data\_in => data\_in,

result\_out => result\_out

);

-- Генерация тактового сигнала

clk\_process: process

begin

clk <= '0';

wait for 10 ns;

clk <= '1';

wait for 10 ns;

end process;

-- Стимулы для моделирования

stimulus\_process: process

begin

-- Инициализация и сброс устройства

reset <= '1';

wait for 20 ns;

reset <= '0';

-- Подача данных на вход data\_in после сброса

data\_in <= "00001111"; -- Пример значения входных данных (15 в десятичной системе)

wait for 260 ns; -- Подождем, пока данные пройдут через все этапы конвейера

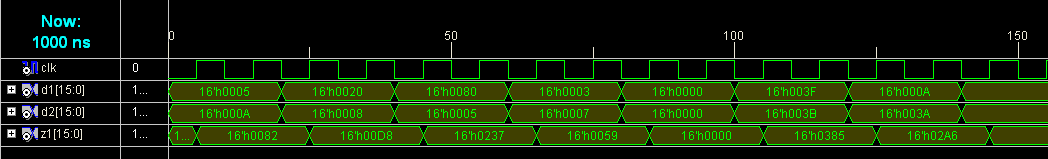
wait;

end process;

end Behavioral;

3. Разработать на языке процессор VHDL Z1 <= M\*D1 + (15-M)\*D2, где M – номер варианта. Операцию умножения заменить сложениями со сдвигом. Выполнить моделирование работы схемы для 5-6 значений, в том числе отрицательных.

Выполнить реализацию, просмотреть и включить в отчет результат размещения схемы в ПЛИС. Зафиксировать минимальный период работы схемы.



Листинг

**p3**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_SIGNED.ALL;

use IEEE.NUMERIC\_STD.ALL;

entity p3 is

Port ( clk : in STD\_LOGIC;

D1 : in STD\_LOGIC\_VECTOR(15 downto 0);

D2 : in STD\_LOGIC\_VECTOR(15 downto 0);

Z1 : out STD\_LOGIC\_VECTOR(15 downto 0));

end p3;

architecture Behavioral of p3 is

begin

process(clk)

begin

if rising\_edge(clk) then

-- Преобразуем STD\_LOGIC\_VECTOR в signed для сдвига

Z1 <= std\_logic\_vector(signed(D1) sll 2) +

std\_logic\_vector(signed(D2) sll 3) +

std\_logic\_vector(signed(D2)) +

std\_logic\_vector(signed(D2)) +

std\_logic\_vector(signed(D2));

end if;

end process;

end Behavioral;

**Test\_p3**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_SIGNED.ALL;

use IEEE.NUMERIC\_STD.ALL;

entity Test\_p3 is

end Test\_p3;

architecture behavior of Test\_p3 is

-- Сигналы для тестирования

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

signal D1, D2, Z1 : STD\_LOGIC\_VECTOR(15 downto 0);

-- Период тактового сигнала

constant clk\_period : time := 10 ns; -- Период такта

begin

-- Установка компонента (процессора)

uut: entity work.p3

port map ( clk => clk, D1 => D1, D2 => D2, Z1 => Z1 );

-- Генерация тактового сигнала

clk\_process: process

begin

-- Генерация тактового сигнала

while true loop

clk <= '0';

wait for clk\_period / 2; -- Половина периода для генерации такта

clk <= '1';

wait for clk\_period / 2; -- Половина периода для генерации такта

end loop;

end process;

-- Генерация тестов

stim\_proc: process

begin

-- Пример 1: положительные значения

D1 <= "0000000000000101";

D2 <= "0000000000001010";

wait for 20 ns; -- Задержка для моделирования

-- Пример 2: отрицательные значения

D1 <= "0000000000100000";

D2 <= "0000000000001000";

wait for 20 ns;

-- Пример 3: D1 > D2

D1 <= "0000000010000000";

D2 <= "0000000000000101";

wait for 20 ns;

-- Пример 4: D1 < D2

D1 <= "0000000000000011";

D2 <= "0000000000000111";

wait for 20 ns;

-- Пример 5: нулевые значения

D1 <= "0000000000000000";

D2 <= "0000000000000000";

wait for 20 ns;

-- Пример 6: максимальные значения

D1 <= "0000000000111111";

D2 <= "0000000000111011";

wait for 20 ns;

-- Пример 7: сочетание положительных и отрицательных значений

D1 <= "0000000000001010";

D2 <= "0000000000111010";

wait for 20 ns;

-- Пример 8: D1 и D2 отрицательные

D1 <= "0000000000110000";

D2 <= "0000000001101100";

wait for 20 ns;

wait;

end process;

end behavior;

**Вывод**

Изучили принципы разработки специализированных процессоров на языке VHDL.