Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

ОТЧЕТ

по лабораторной работе №1

Вариант № 1

Выполнил Проверил:

студент группы 150504: Шеменков В.В.

Горбачевский К.В.

МИНСК 2024

# ЦЕЛЬ РАБОТЫ И ОПИСАНИЕ УСТРОЙСТВА

Целью данной лабораторной работы является описание комбинационного устройства на языке VHDL.

Устройство реализует алгоритм, который позволяет перейти от двоично-десятичного представления числа перейти к изображению этого числа в десятичном виде на семисегментном индикаторе. (bcd-to-seven-segment).

На рисунке 1.1 изображена упрощенная схема устройства. На рисунке 1.2 приведена таблица истинности для данного устройства.

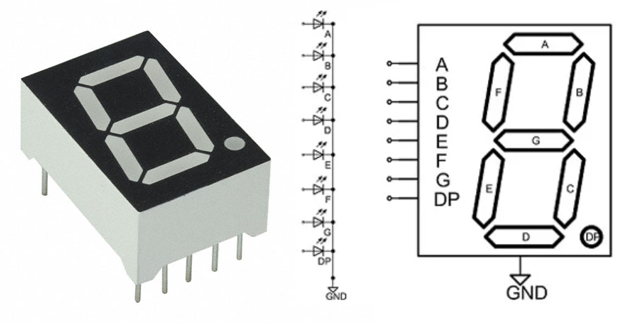


Рисунок 1.1 – семисегментный индикатор

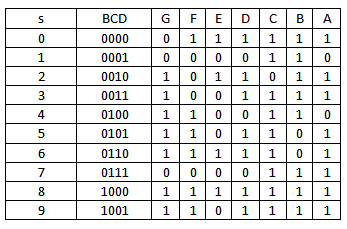


Рисунок 1.2 – таблица истиности устройства

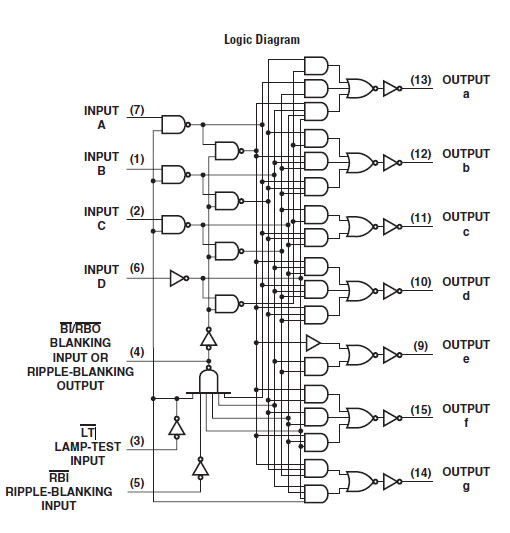


Рисунок 1.3 – схема устройства

# ОПИСАНИЕ НА ЯЗЫКЕ VHDL

Используя таблицу истинности можно легко описать устройство на языке описания аппаратуры VHDL.

* 1. Листинг кода параллельного устройства

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity labwork1ParallelOperator is

Port (

aIn, bIn, cIn, dIn: in std\_logic;

notBI: in std\_logic;

notRBI: in std\_logic;

notLT: in std\_logic;

notRBO: out std\_logic;

aOut, bOut, cOut, dOut, eOut, fOut, gOut: out std\_logic

);

end labwork1ParallelOperator;

architecture Behavioral of labwork1ParallelOperator is

signal temp: std\_logic\_vector (7 downto 0);

signal LT, pass: std\_logic;

begin

temp(0) <= not aIn when pass = '1' else '0';

temp(1) <= not(temp(0));

temp(2) <= not bIn when pass = '1' else '0';

temp(3) <= not(temp(2));

temp(4) <= not cIn when pass = '1' else '0';

temp(5) <= not(temp(4));

temp(6) <= not dIn when pass = '1' else '0';

temp(7) <= not(temp(6));

LT <= not notLT when notBI = '1' else '0';

notRBO <= '0' when notBI = '0' else '1' when LT = '1' else not(not(aIn or bIn or cIn or dIn) and (not notRBI));

pass <= notBI and not(not(aIn or bIn or cIn or dIn) and (not notRBI));

aOut <= not((temp(3) and temp(7)) or (temp(0) and temp(5)) or (temp(1) and temp(2) and temp(4) and temp(6))) or LT;

bOut <= not((temp(3) and temp(7)) or (temp(1) and temp(2) and temp(5)) or (temp(0) and temp(3) and temp(5))) or LT;

cOut <= not((temp(5) and temp(7)) or (temp(0) and temp(3) and temp(4))) or LT;

dOut <= not((temp(1) and temp(2) and temp(4)) or (temp(0) and temp(2) and temp(5)) or (temp(1) and temp(3) and temp(5))) or LT;

eOut <= not(temp(1) or (temp(2) and temp(5))) or LT;

fOut <= not((temp(1) and temp(3)) or (temp(3) and temp(4)) or (temp(1) and temp(4) and temp(6))) or LT;

gOut <= not((temp(1) and temp(3) and temp(5)) or (temp(2) and temp(4) and temp(6) and notLT)) or LT;

end Behavioral;

* 1. Листинг кода тест-борда параллельного устройства

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity labwork1ParallelOperatorSimulation is

end labwork1ParallelOperatorSimulation;

architecture Behavioral of labwork1ParallelOperatorSimulation is

component labwork1ParallelOperator

Port (

aIn, bIn, cIn, dIn: in std\_logic;

notBI: in std\_logic;

notRBI: in std\_logic;

notLT: in std\_logic;

notRBO: out std\_logic;

aOut, bOut, cOut, dOut, eOut, fOut, gOut: out std\_logic

);

end component;

SIGNAL aIn, bIn, cIn, dIn: std\_logic;

SIGNAL notBI: std\_logic;

SIGNAL notRBO: std\_logic;

SIGNAL notRBI: std\_logic;

SIGNAL notLT: std\_logic;

SIGNAL aOut, bOut, cOut, dOut, eOut, fOut, gOut: std\_logic;

begin

UUT : labwork1ParallelOperator port map(

aIn => aIn,

bIn => bIn,

cIn => cIn,

dIn => dIn,

notBI => notBI,

notRBO => notRBO,

notRBI => notRBI,

notLT => notLT,

aOut => aOut,

bOut => bOut,

cOut => cOut,

dOut => dOut,

eOut => eOut,

fOut => fOut,

gOut => gOut);

stimulus: process

begin

--0

notBI <= '1';

aIn <= '0';

bIn <= '0';

cIn <= '0';

dIn <= '0';

notRBI <= '1';

notLT <= '1';

--1

wait for 10 ns;

aIn <= '1';

notRBI <= 'X';

--2

wait for 10 ns;

aIn <= '0';

bIn <= '1';

--3

wait for 10 ns;

aIn <= '1';

--4

wait for 10 ns;

aIn <= '0';

bIn <= '0';

cIn <= '1';

--5

wait for 10 ns;

aIn <= '1';

--6

wait for 10 ns;

aIn <= '0';

bIn <= '1';

--7

wait for 10 ns;

aIn <= '1';

--8

wait for 10 ns;

aIn <= '0';

bIn <= '0';

cIn <= '0';

dIn <= '1';

--9

wait for 10 ns;

aIn <= '1';

--10

wait for 10 ns;

aIn <= '0';

bIn <= '1';

--11

wait for 10 ns;

aIn <= '1';

--12

wait for 10 ns;

aIn <= '0';

bIn <= '0';

cIn <= '1';

--13

wait for 10 ns;

aIn <= '1';

--14

wait for 10 ns;

aIn <= '0';

bIn <= '1';

--15

wait for 10 ns;

aIn <= '1';

--BI

wait for 10 ns;

notBI <= '0';

aIn <= 'X';

bIn <= 'X';

cIn <= 'X';

dIn <= 'X';

notRBI <= 'X';

notLT <= 'X';

--RBI

wait for 10 ns;

notBI <= '1';

aIn <= '0';

bIn <= '0';

cIn <= '0';

dIn <= '0';

notRBI <= '0';

notLT <= '1';

--LT

wait for 10 ns;

aIn <= 'X';

bIn <= 'X';

cIn <= 'X';

dIn <= 'X';

notRBI <= 'X';

notLT <= '0';

wait for 10 ns;

end process;

end Behavioral;

* 1. Листинг кода последовательного устройства

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity labwork1SerialOperator is

Port (

aIn, bIn, cIn, dIn: in std\_logic;

notBI: in std\_logic;

notRBI: in std\_logic;

notLT: in std\_logic;

notRBO: out std\_logic;

aOut, bOut, cOut, dOut, eOut, fOut, gOut: out std\_logic

);

end labwork1SerialOperator;

architecture Behavioral of labwork1SerialOperator is

begin

process(aIn, bIn, cIn, dIn, notBI, notRBI, notLT)

begin

if(notBI = '0') then aOut <= '0'; bOut <= '0'; cOut <= '0'; dOut <= '0'; eOut <= '0'; fOut <= '0'; gOut <= '0'; notRBO <= '0';

elsif (notLT = '0') then aOut <= '1'; bOut <= '1'; cOut <= '1'; dOut <= '1'; eOut <= '1'; fOut <= '1'; gOut <= '1'; notRBO <= '1';

elsif (notLT = '1' and notRBI = '0' and dIn = '0' and cIn = '0' and bIn = '0' and aIn = '0' and notBI = '0') then aOut <= '0'; bOut <= '0'; cOut <= '0'; dOut <= '0'; eOut <= '0'; fOut <= '0'; gOut <= '0'; notRBO <= '0';

--0

elsif (notLT = '1' and notRBI = '1' and dIn = '0' and cIn = '0' and bIn = '0' and aIn = '0') then aOut <= '1'; bOut <= '1'; cOut <= '1'; dOut <= '1'; eOut <= '1'; fOut <= '1'; gOut <= '0'; notRBO <= '1';

--1

elsif (notLT = '1' and dIn = '0' and cIn = '0' and bIn = '0' and aIn = '1') then aOut <= '0'; bOut <= '1'; cOut <= '1'; dOut <= '0'; eOut <= '0'; fOut <= '0'; gOut <= '0'; notRBO <= '1';

--2

elsif (notLT = '1' and dIn = '0' and cIn = '0' and bIn = '1' and aIn = '0') then aOut <= '1'; bOut <= '1'; cOut <= '0'; dOut <= '1'; eOut <= '1'; fOut <= '0'; gOut <= '1'; notRBO <= '1';

--3

elsif (notLT = '1' and dIn = '0' and cIn = '0' and bIn = '1' and aIn = '1') then aOut <= '1'; bOut <= '1'; cOut <= '1'; dOut <= '1'; eOut <= '0'; fOut <= '0'; gOut <= '1'; notRBO <= '1';

--4

elsif (notLT = '1' and dIn = '0' and cIn = '1' and bIn = '0' and aIn = '0') then aOut <= '0'; bOut <= '1'; cOut <= '1'; dOut <= '0'; eOut <= '0'; fOut <= '1'; gOut <= '1'; notRBO <= '1';

--5

elsif (notLT = '1' and dIn = '0' and cIn = '1' and bIn = '0' and aIn = '1') then aOut <= '1'; bOut <= '0'; cOut <= '1'; dOut <= '1'; eOut <= '0'; fOut <= '1'; gOut <= '1'; notRBO <= '1';

--6

elsif (notLT = '1' and dIn = '0' and cIn = '1' and bIn = '1' and aIn = '0') then aOut <= '0'; bOut <= '0'; cOut <= '1'; dOut <= '1'; eOut <= '1'; fOut <= '1'; gOut <= '1'; notRBO <= '1';

--7

elsif (notLT = '1' and dIn = '0' and cIn = '1' and bIn = '1' and aIn = '1') then aOut <= '1'; bOut <= '1'; cOut <= '1'; dOut <= '0'; eOut <= '0'; fOut <= '0'; gOut <= '0'; notRBO <= '1';

--8

elsif (notLT = '1' and dIn = '1' and cIn = '0' and bIn = '0' and aIn = '0') then aOut <= '1'; bOut <= '1'; cOut <= '1'; dOut <= '1'; eOut <= '1'; fOut <= '1'; gOut <= '1'; notRBO <= '1';

--9

elsif (notLT = '1' and dIn = '1' and cIn = '0' and bIn = '0' and aIn = '1') then aOut <= '1'; bOut <= '1'; cOut <= '1'; dOut <= '0'; eOut <= '0'; fOut <= '1'; gOut <= '1'; notRBO <= '1';

--10

elsif (notLT = '1' and dIn = '1' and cIn = '0' and bIn = '1' and aIn = '0') then aOut <= '0'; bOut <= '0'; cOut <= '0'; dOut <= '1'; eOut <= '1'; fOut <= '0'; gOut <= '1'; notRBO <= '1';

--11

elsif (notLT = '1' and dIn = '1' and cIn = '0' and bIn = '1' and aIn = '1') then aOut <= '0'; bOut <= '0'; cOut <= '1'; dOut <= '1'; eOut <= '0'; fOut <= '0'; gOut <= '1'; notRBO <= '1';

--12

elsif (notLT = '1' and dIn = '1' and cIn = '1' and bIn = '0' and aIn = '0') then aOut <= '0'; bOut <= '1'; cOut <= '0'; dOut <= '0'; eOut <= '0'; fOut <= '1'; gOut <= '1'; notRBO <= '1';

--13

elsif (notLT = '1' and dIn = '1' and cIn = '1' and bIn = '0' and aIn = '1') then aOut <= '1'; bOut <= '0'; cOut <= '0'; dOut <= '1'; eOut <= '0'; fOut <= '1'; gOut <= '1'; notRBO <= '1';

--14

elsif (notLT = '1' and dIn = '1' and cIn = '1' and bIn = '1' and aIn = '0') then aOut <= '0'; bOut <= '0'; cOut <= '0'; dOut <= '1'; eOut <= '1'; fOut <= '1'; gOut <= '1'; notRBO <= '1';

--15

elsif (notLT = '1' and dIn = '1' and cIn = '1' and bIn = '1' and aIn = '1') then aOut <= '0'; bOut <= '0'; cOut <= '0'; dOut <= '0'; eOut <= '0'; fOut <= '0'; gOut <= '0'; notRBO <= '1';

end if;

end process;

end Behavioral;

* 1. Листинг кода тест-борда последовательного устройства

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity labwork1SerialOperatorSimulation is

end labwork1SerialOperatorSimulation;

architecture Behavioral of labwork1SerialOperatorSimulation is

component labwork1SerialOperator

Port (

aIn, bIn, cIn, dIn: in std\_logic;

notBI: in std\_logic;

notRBI: in std\_logic;

notLT: in std\_logic;

notRBO: out std\_logic;

aOut, bOut, cOut, dOut, eOut, fOut, gOut: out std\_logic

);

end component;

SIGNAL aIn, bIn, cIn, dIn: std\_logic;

SIGNAL notBI: std\_logic;

SIGNAL notRBO: std\_logic;

SIGNAL notRBI: std\_logic;

SIGNAL notLT: std\_logic;

SIGNAL aOut, bOut, cOut, dOut, eOut, fOut, gOut: std\_logic;

begin

UUT : labwork1SerialOperator port map(

aIn => aIn,

bIn => bIn,

cIn => cIn,

dIn => dIn,

notBI => notBI,

notRBO => notRBO,

notRBI => notRBI,

notLT => notLT,

aOut => aOut,

bOut => bOut,

cOut => cOut,

dOut => dOut,

eOut => eOut,

fOut => fOut,

gOut => gOut);

stimulus: process

begin

--0

notBI <= '1';

aIn <= '0';

bIn <= '0';

cIn <= '0';

dIn <= '0';

notRBI <= '1';

notLT <= '1';

--1

wait for 10 ns;

aIn <= '1';

notRBI <= 'X';

--2

wait for 10 ns;

aIn <= '0';

bIn <= '1';

--3

wait for 10 ns;

aIn <= '1';

--4

wait for 10 ns;

aIn <= '0';

bIn <= '0';

cIn <= '1';

--5

wait for 10 ns;

aIn <= '1';

--6

wait for 10 ns;

aIn <= '0';

bIn <= '1';

--7

wait for 10 ns;

aIn <= '1';

--8

wait for 10 ns;

aIn <= '0';

bIn <= '0';

cIn <= '0';

dIn <= '1';

--9

wait for 10 ns;

aIn <= '1';

--10

wait for 10 ns;

aIn <= '0';

bIn <= '1';

--11

wait for 10 ns;

aIn <= '1';

--12

wait for 10 ns;

aIn <= '0';

bIn <= '0';

cIn <= '1';

--13

wait for 10 ns;

aIn <= '1';

--14

wait for 10 ns;

aIn <= '0';

bIn <= '1';

--15

wait for 10 ns;

aIn <= '1';

--BI

wait for 10 ns;

notBI <= '0';

aIn <= 'X';

bIn <= 'X';

cIn <= 'X';

dIn <= 'X';

notRBI <= 'X';

notLT <= 'X';

--RBI

wait for 10 ns;

notBI <= '1';

aIn <= '0';

bIn <= '0';

cIn <= '0';

dIn <= '0';

notRBI <= '0';

notLT <= '1';

--LT

wait for 10 ns;

aIn <= 'X';

bIn <= 'X';

cIn <= 'X';

dIn <= 'X';

notRBI <= 'X';

notLT <= '0';

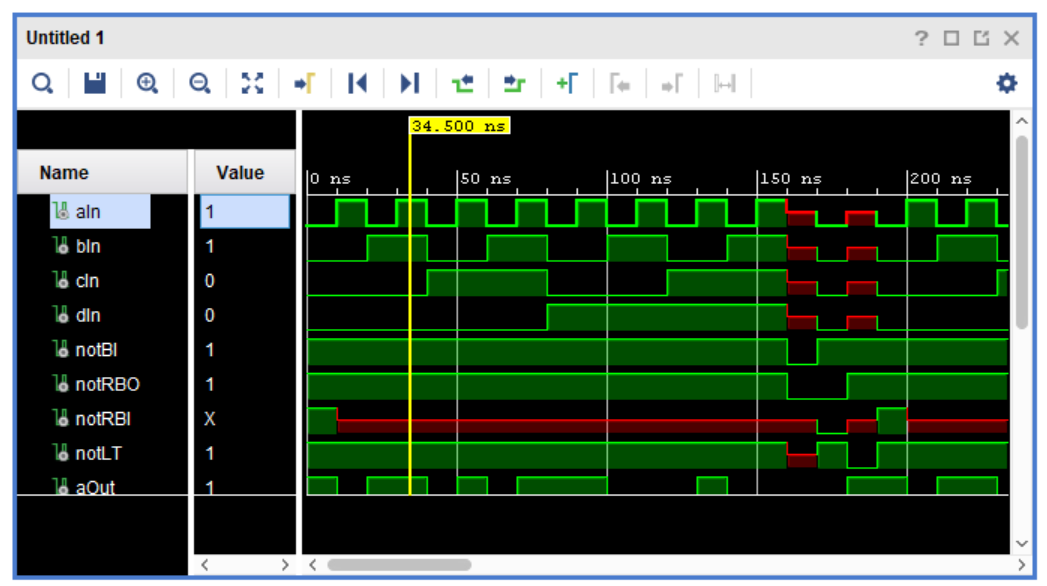
wait for 10 ns;

end process;

end Behavioral;

# ВРЕМЕННАЯ ДИАГРАММА СИМУЛЯЦИИ

Временная диаграмма последовательного оператора:



Временная диаграмма параллельного оператора:

