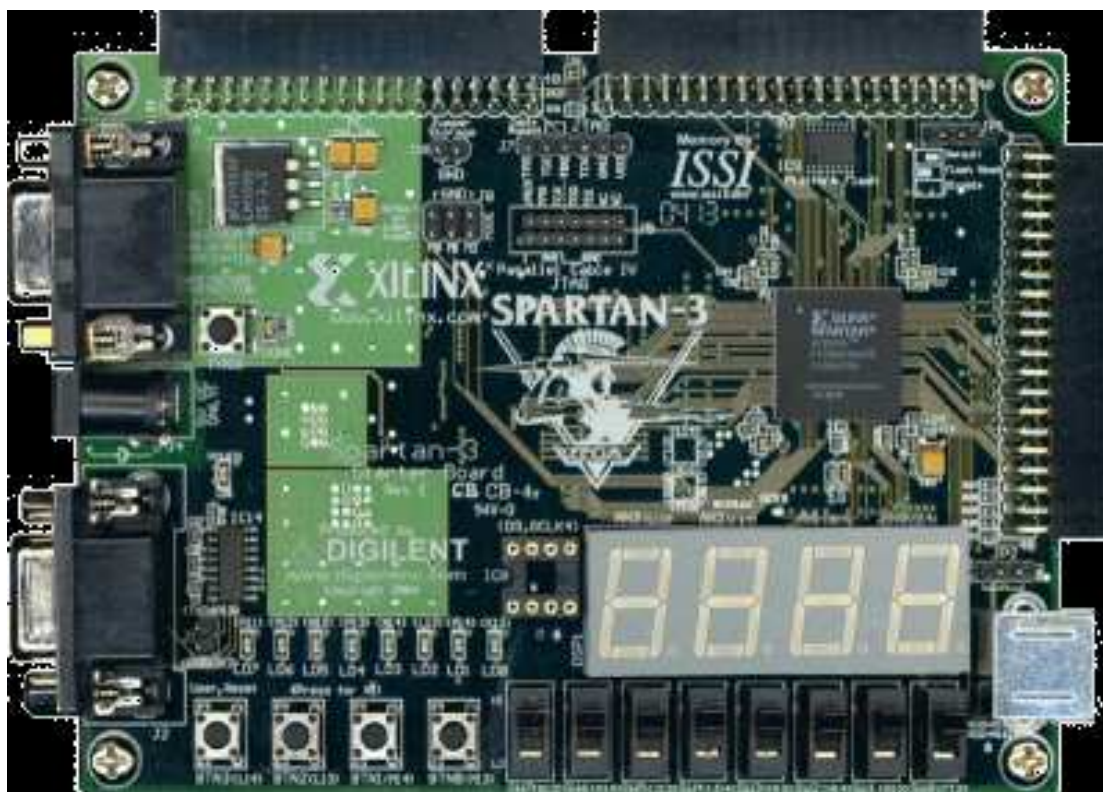


Выполненный тестовый проект

В качестве примера далее будет рассмотрен выполненный автором тестовый проект «**Мигающие цифровые часы**» на базе **Digilent Spartan-3 Starter Kit Board**, содержащей в качестве ПЛИС ядро Xilinx Spartan-3 XC3-S400 FT256.



Проект выполнен на базе программного обеспечения Xilinx ISE 10.1 DIE и Digilent Adept SW

ExPort, с использованием оригинальной документации производителя тестовой платы и пособия

по быстрому старту от Xilinx. Код проекта указан ниже:

```
-----  
-- Design Name: Мигающие цифровые часы  
-- Module Name: display - Behavioral  
-- Project Name: SevenSegTest  
-- Target Devices: Xilinx Spartan III XC3-S400  
-- Tool versions: ISE 10.1  
-- Description: Digilent Spartan-3 Starter Kit Board в качестве мигающих цифровых часов  
-- Назначение контактов Digilent Spartan-3 Starter Kit Board:  
-- data(7) => E14 a  
-- data(6) => G13 b  
-- data(5) => N15 c
```

```

-- data(4) => P15    d
-- data(3) => R16    e
-- data(2) => F13    f
-- data(1) => N16    g
-- data(0) => P16    dp
-- clk    => T9      50 MHz
-- reset  => M13     BTN0
-- anode(3) => E13
-- anode(2) => F14
-- anode(1) => G14
-- anode(0) => D14

```

```

-----
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

```

```

---- Uncomment the following library declaration if instantiating
---- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

```

```

entity display is
    Port ( clk: in  STD_LOGIC;
          reset : in  STD_LOGIC;
          data : out STD_LOGIC_VECTOR (7 downto 0);
          anode : out STD_LOGIC_VECTOR (3 downto 0));
end display;

```

architecture Behavioral of display is

```

-- FSM используется для обновления сегментов дисплея
type state_type is (d0, d1, d2, d3);
type char is (v, y, E, r, I, L, o, G);
signal next_state, curr_state: state_type;

-- slow_clk: 1 Hz, mid_clk: 5 Hz, показывает надпись
signal slow_clk, mid_clk, start: std_logic;
-- one counter for one 7-segment display
signal counter0, counter1, counter2, counter3: integer range 0 to 9;

```



```

-- декодирует " VErIloG " для 7-seg сигналов
function char_to_led(c: char) return std_logic_vector is
begin

```

```

    case c is
        when v => return "1101100";
        when y => return "1011010";
        when E => return "0110000";
        when r => return "1111010";
        when I => return "1111001";
        when L => return "1110001";
        when o => return "1100010";
        when G => return "0100001";
        when Z => return "1111111";
        when others => return "1111111";
    end case;

```

```

        end case;
    end char_to_led;
    -- декодирует число для 7-seg сигналов
    function num_to_led(digit: integer) return std_logic_vector is
    begin
        case digit is
            when 0 => return "0000001";
            when 1 => return "1001111";
            when 2 => return "0010010";
            when 3 => return "0000110";
            when 4 => return "1001100";
            when 5 => return "0100100";
            when 6 => return "1100000";
            when 7 => return "0001111";
            when 8 => return "0000000";
            when 9 => return "0001100";
            when others => return "1111111";
        end case;
        --return "1111111";
    end num_to_led;
begin
    -- показывает время для 1 Hz (slow_clk) и 5 Hz (mid_clk)
    p1: process(clk, reset)
        variable cnt, cnt1: integer;
    begin
        if reset = '1' then
            cnt := 0;
            cnt1 := 0;
            slow_clk <= '0';
            mid_clk <= '0';
        elsif clk'event and clk='1' then
            cnt := cnt + 1;
            cnt1 := cnt1 + 1;
            if cnt = 25000000 then
                cnt := 0;
                slow_clk <= not slow_clk;
            end if;
            if cnt1 = 500000 then
                cnt1 := 0;
                mid_clk <= not mid_clk;
            end if;
        end if;
    end process;

    -- считает process, counter1 и counter3 только до 5!
    p2: process(slow_clk, start)
    begin
        if start = '1' then
            counter0 <= 0;
            counter1 <= 0;
            counter2 <= 0;
            counter3 <= 0;

```

```

    elsif slow_clk'event and slow_clk='1' then
        if counter0 < 9 then
            counter0 <= counter0 + 1;
        else
            counter0 <= 0;
            -- counter1 counts to 6
            if counter1 < 5 then
                counter1 <= counter1 + 1;
            else
                counter1 <= 0;
                if counter2 < 9 then
                    counter2 <= counter2 + 1;
                else
                    counter2 <= 0;
                    if counter3 < 5 then
                        counter3 <= counter3 + 1;
                    else
                        counter3 <= 0;
                    end if;
                end if;
            end if;
        end if;
    end if;
end process;

-- выходные данные для для 7-seg дисплея
-- точка DP для второго знака мерцает 1 Hz (slow_clk)
p3: process(curr_state, next_state, counter0, counter1, counter2, counter3, slow_clk)
begin
    case curr_state is
        when d0 =>
            next_state <= d1;
            if start = '1' then
                if slow_clk = '1' then
                    data <= char_to_led(r)&"1";
                else
                    data <= char_to_led(G)&"1";
                end if;
            else
                data <= num_to_led(counter0)&"1";
            end if;
            anode <= "1110";
        when d1 =>
            next_state <= d2;
            if start = '1' then
                if slow_clk = '1' then
                    data <= char_to_led(E)&"1";
                else
                    data <= char_to_led(o)&"1";
                end if;
            else
                data <= num_to_led(counter1)&"1";
            end if;
    end case;
end process;

```

```

        anode <= "1101";
    when d2 =>
        next_state <= d3;
        if start = '1' then
            if slow_clk = '1' then
                data <= char_to_led(y)&"1";
            else
                data <= char_to_led(L)&"1";
            end if;
        else
            data <= num_to_led(counter2)&slow_clk;
        end if;
        anode <= "1011";
    when d3 =>
        next_state <= d0;
        if start = '1' then
            if slow_clk = '1' then
                data <= char_to_led(v)&"1";
            else
                data <= char_to_led(I)&"1";
            end if;
        else
            data <= num_to_led(counter3)&"1";
        end if;
        anode <= "0111";
    end case;
end process;

-- обновляет экран с частотой 5 Hz - эффект мерцания
p4: process(mid_clk, reset)
begin
    if reset = '1' then
        curr_state <= d0;
    elsif mid_clk'event and mid_clk = '1' then
        curr_state <= next_state;
    end if;
end process;

-- считает 5 раз и отобразит " VErIlloG ", затем часы
p5: process(slow_clk, reset)
    variable cnt: integer;
begin
    if reset = '1' then
        cnt := 0;
        start <= '1';
    elsif slow_clk'event and slow_clk='1' then
        cnt := cnt + 1;
        if cnt = 10 then
            start <= '0';
        end if;
    end if;
end process;
end Behavioral;
```

File Edit View Project Source Process Window Help

Sources for: Implementation

- SevenSegTest
- xc3s400-4ft256
 - display - Behavioral (display.vhd)
 - display.ucf (display.ucf)

Processes for: display - Behavioral

- Add Existing Source
- Create New Source
- View Design Summary
- Design Utilities
 - Create Schematic Symbol
 - View Command Line Log File
 - View HDL Instantiation Template
- User Constraints
 - Create Timing Constraints
 - Floorplan IO - Pre-Synthesis
 - Floorplan Area / IO / Logic - Post-Synthesis
- Synthesize - XST
 - View Synthesis Report
 - View RTL Schematic
 - View Technology Schematic
 - Check Syntax
 - Generate Post-Synthesis Simulation Model
- Implement Design
 - Translate
 - Map
 - Place & Route
- Generate Programming File
 - Programming File Generation Report
- Configure Target Device
 - Generate Target PROM/ACE File
 - Manage Configuration Project (IMPACT)

```

42 --use UNISIM.VComponents.all;
43
44 entity display is
45     Port ( clk: in  STD_LOGIC;
46           reset : in  STD_LOGIC;
47           data : out STD_LOGIC_VECTOR (7 downto 0);
48           anode : out STD_LOGIC_VECTOR (3 downto 0));
49 end display;
50
51 architecture Behavioral of display is
52     -- a FSM is used to update each digit in the 7-Segment displays
53     type state_type is (d0, d1, d2, d3);
54     type char is (v, y, E, r, I, L, o, G, z);
55     signal next_state, curr_state: state_type;
56     -- slow_clk: 1 Hz, mid_clk: 5 Hz, start is used to display d&n Lo at beginning
57     signal slow_clk, mid_clk, start: std_logic;
58     -- one counter for one 7-segment display
59     signal counter0, counter1, counter2, counter3: integer range 0 to 9;
60
61     -- decode "d&n Lo" to 7-segment signals
62     function char_to_led(c: char) return std_logic_vector is
63     begin
64         case c is
65             when v => return "1101100";
66             when y => return "1011010";
67             when E => return "0110000";
68             when r => return "1111010";
69             when I => return "1111001";
70             when L => return "1110001";
71             when o => return "1100010";
72             when G => return "0100001";
73             when Z => return "1111111";
74             when others => return "1111111";
75         end case;
76     end char_to_led;
77     -- decode number to 7-segment signals
78     function num_to_led(digit: integer) return std_logic_vector is
79     begin
80         case digit is
81             when 0 => return "0000001";
82             when 1 => return "1001111";

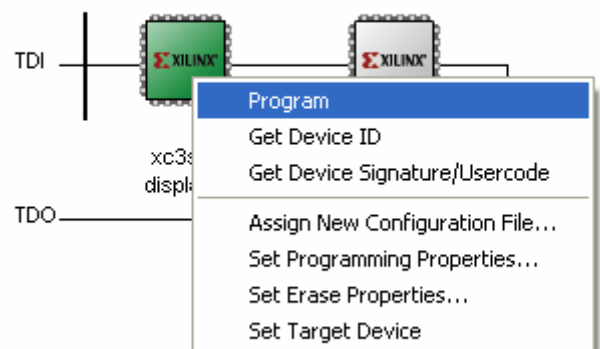
```

What's New in ISE Design Suite 10.1 Design Summary display.vhd

Started : "Launching ISE Text Editor to edit display.vhd".

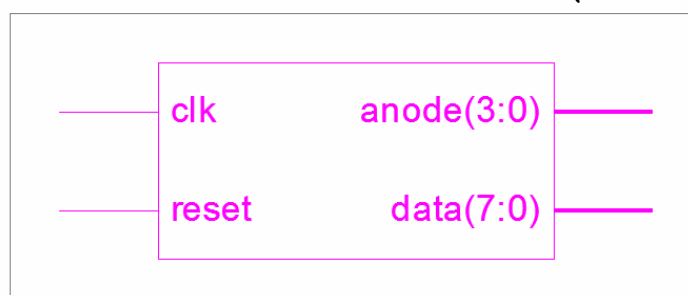
Property Name	Value
Product Category	All
Family	Spartan3
Device	XC3S400
Package	FT256
Speed	-4
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	Modelsim-SE Mixed
Preferred Language	Verilog
Enable Enhanced Design Summary	<input checked="" type="checkbox"/>
Enable Message Filtering	<input type="checkbox"/>

Настройки проекта:



Программирование устройств
(RAM FPGA, ROM FLASH)

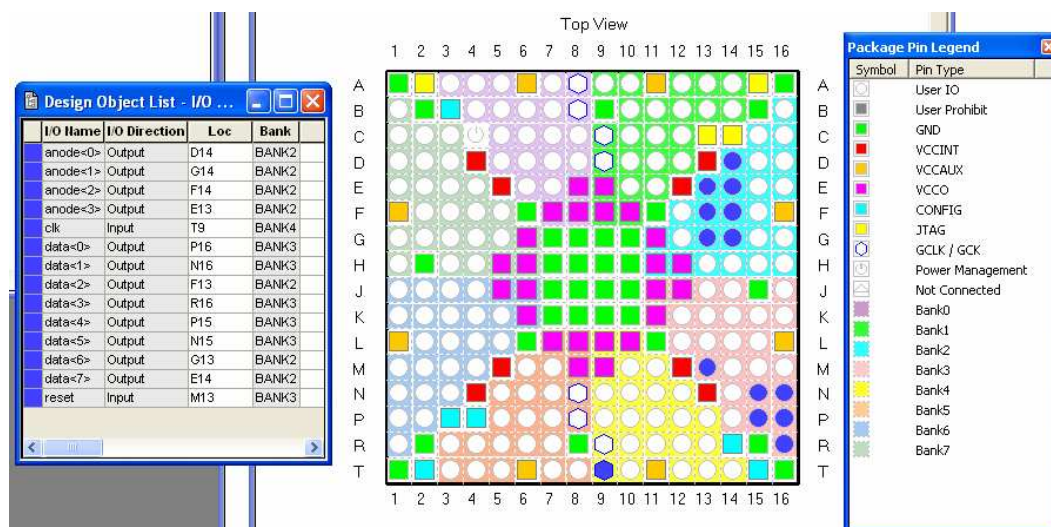
Условная схема:



SevenSegTest Project Status			
Project File:	SevenSegTest.isc	Current State:	Programming File Generated
Module Name:	display	• Errors:	No Errors
Target Device:	xc3s400-4ft256	• Warnings:	2 Warnings
Product Version:	ISE 10.1 - Foundation	• Routing Results:	All Signals Completely Routed
Design Goal:	Balanced	• Timing Constraints:	All Constraints Met
Design Strategy:	Xilinx Default (unlocked)	• Final Timing Score:	0 (Timing Report)

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	115	7,168	1%	
Number of 4 input LUTs	176	7,168	2%	
Logic Distribution				
Number of occupied Slices	186	3,584	5%	
Number of Slices containing only related logic	186	186	100%	
Number of Slices containing unrelated logic	0	186	0%	
Total Number of 4 input LUTs	362	7,168	5%	
Number used as logic	176			
Number used as a route-thru	186			
Number of bonded IOBs	14	173	8%	
Number of BUFGMUXs	2	8	25%	

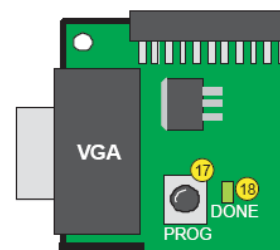
Обзорные данные проекта



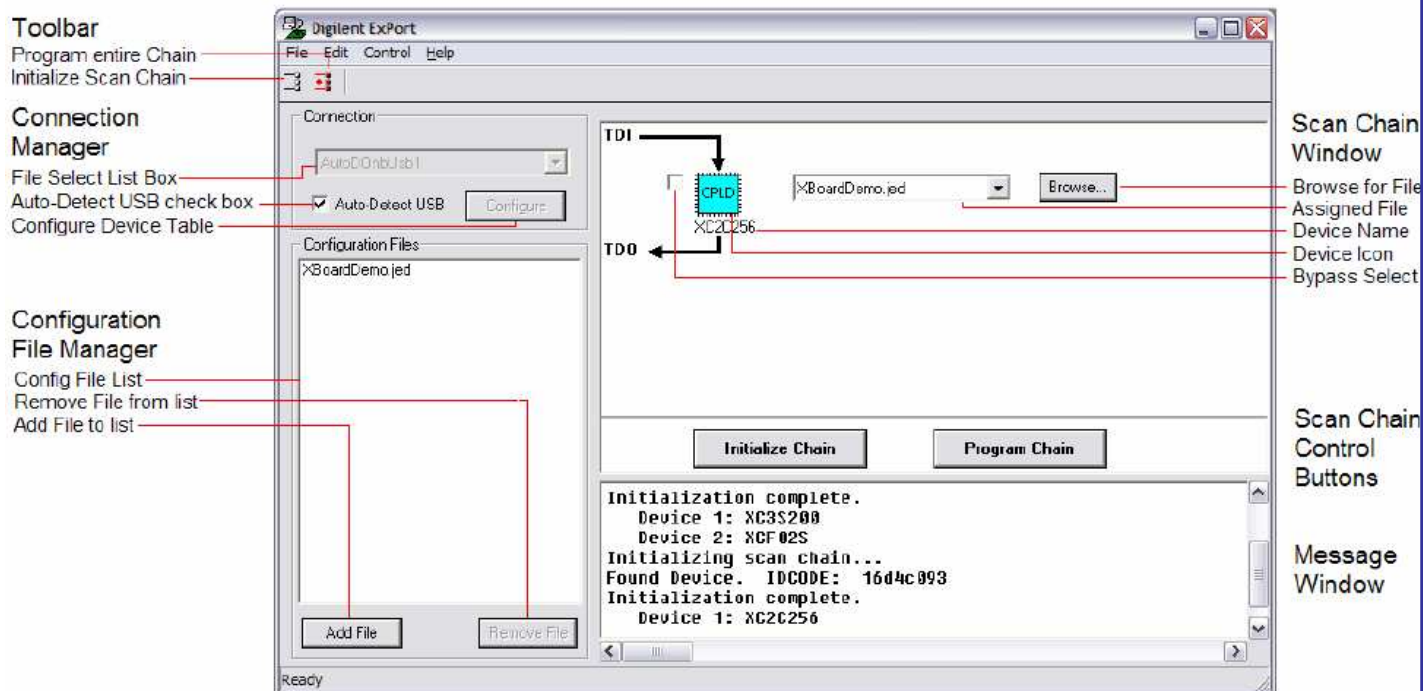
Карта внутренних соединений входов-выходов

Table 10-1: Jumper JP1 Controls the Platform Flash Options

Option	Jumper JP1 Setting	Description
Default	JP1	The FPGA boots from Platform Flash. No additional data storage is available.
Flash Read	JP1	The FPGA boots from Platform Flash, which is permanently enabled. The FPGA can read additional data from Platform Flash.
Disable	JP1	Jumper removed. Platform Flash is disabled. Other configuration data source provides FPGA boot data.

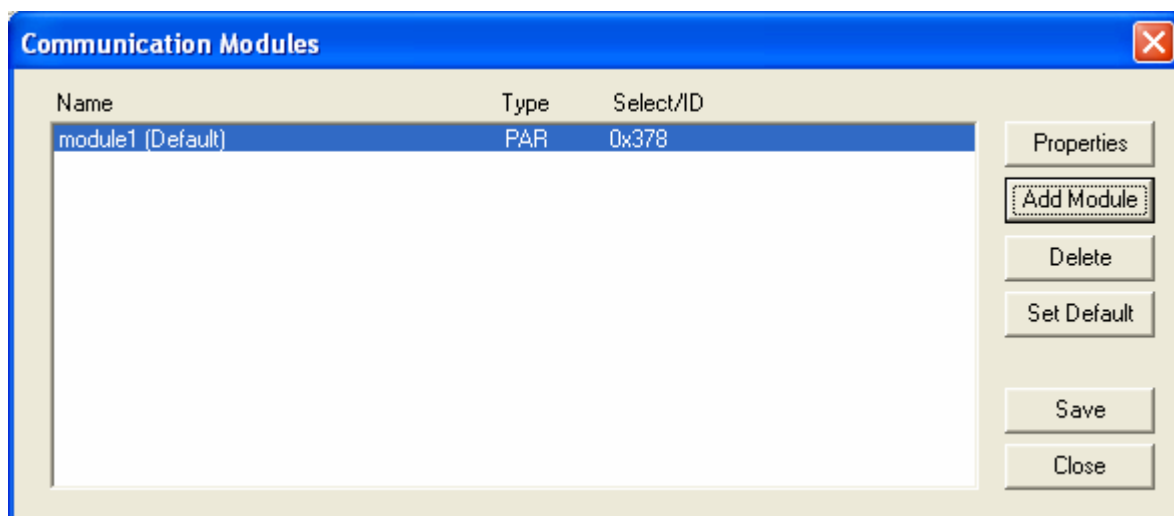


● Для работы прошивки платы после перезагрузки, следует установить джампер JP1 в положение Default/Flash Read. В положении Disable используются данные, загруженные в ПЛИС при программировании, **исчезающие** при выключении питания или нажатия кнопки **PROG!!!**



Digilent ExPort Main Window

Основное окно программного обеспечения Digilent Adept ExPort (программирование устройств возможно только после установки параллельного порта (Add Module -> Parallel -< Add)).



Выводы.

В результате данной работы были изучены возможности использования языка Verilog в применении на практике. На базе тестовой платы был разработан модуль цифровых часов, отображающий надпись Verilog 10 раз при включении, далее переходящий в режим цифровых часов.

Список литературы:

1. Verilog HDL: A Guide to Digital and Synthesis.
2. Advanced Asic Chip Synthesis, Himanshu Bhatnagar.
3. Verilog HDL Synthesis J.Bhasker.
4. Writing Testbenches Functional Verification of HDL Models, Janick Bergeron.
5. Tai-Ran Hsu, MEMS and Microsystems Design and Manufacture, McGraw-Hill
Science/Engineering, 2001
6. R. A. ADEY, Simulation and Design of Microsystems and Microstructures Wessex Institute of
Technology, UK and P. RENAUD, Swiss Federal Institute of Technology, Switzerland, 1997.
7. Digilent Adept Suite User's Manual® www.digilentinc.com Revision: 11/30/06
8. Spartan-3 Starter Kit Board User Guide www.xilinx.com UG130 (v1.1) May 13, 2005