VHDL Basics

Δαδαλιάρης Αντώνιος
dadaliaris@cs.uth.gr

VHDL: Combinational Circuits

- Full Adder (behavioral description)

test.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY fa IS
PORT(
   a, b, c : IN STD_LOGIC;
   s, cr : OUT STD_LOGIC);
END fa;

ARCHITECTURE behavioral OF fa IS
BEGIN
   PROCESS(a,b,c)
   BEGIN
     IF(a='0' AND b='0' AND c='0') THEN
       s<='0';
       cr<='0';
       ELSIF( a='0' AND b='0' AND c='1') THEN
```

test.vhd

```
ARCHITECTURE behavioral OF fa IS
BEGIN
   PROCESS(a,b,c)
   BEGIN
     IF(a='0' AND b='0' AND c='0') THEN
       s<='0';
       cr<='0';
       ELSIF( a='0' AND b='0' AND c='1') THEN
          s<='1';
          cr<='0';
          ....
       ELSE
          s<='1'; cr<='1';
       END IF;
   END PROCESS;
END behavioral;
```

~/Desktop/Ανάπτυξη Τηλεπικοινωνιακών Συστημάτων Σε Υλικό/test.vhd 0 0 0 26:1 LF UTF-8 VHDL

VHDL: Combinational Circuits

- Full Adder (dataflow description)

test.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY fa IS
PORT(
    a, b, c : IN STD_LOGIC;
    s, cr : OUT STD_LOGIC);
END fa;

ARCHITECTURE dataflow OF fa IS
BEGIN
    s<= a XOR b XOR c;
    cr<= (a AND b) OR (b AND cin) OR (c AND a);
END dataflow;
```

~/Desktop/Ανάπτυξη Τηλεπικοινωνιακών Συστημάτων Σε Υλικό/test.vhd 0 0 0 14:14 LF UTF-8 VHDL

VHDL: Combinational Circuits

- Full Adder (structural description)

and_gate.vhd

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY and_gate IS
PORT(
   a, b: IN STD_LOGIC;
   c: OUT STD_LOGIC);
END and_gate;

ARCHITECTURE arch OF and_gate IS
BEGIN
        c <= a AND b;
END arch;
```

or_gate.vhd

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY or_gate IS
PORT(
   a, b: IN STD_LOGIC;
   c: OUT STD_LOGIC);
END or_gate;

ARCHITECTURE arch OF or_gate IS
BEGIN
        c <= a OR b;
END arch;
```

xor_gate.vhd

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY xor_gate IS
PORT(
   a, b: IN STD_LOGIC;
   c: OUT STD_LOGIC);
END xor_gate;

ARCHITECTURE arch OF xor_gate IS
BEGIN
        c <= a XOR b;
END arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/or_gate.vhd 0 0 0 14:1 LF UTF-8 VHDL

VHDL: Combinational Circuits

- Full Adder (structural description) (cont.)

fa.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY fa IS
PORT (
   a, b, c : IN STD_LOGIC;
   s, cr : OUT STD_LOGIC);
END fa;

ARCHITECTURE structural OF fa IS
COMPONENT and_gate
PORT(
   a, b: IN STD_LOGIC;
   c: OUT STD_LOGIC);
END COMPONENT;

COMPONENT or_gate
PORT(
   a, b: IN STD_LOGIC;
   c: OUT STD_LOGIC);
```

fa.vhd

```
PORT(
   a, b: IN STD_LOGIC;
   c: OUT STD_LOGIC);
END COMPONENT;

SIGNAL xor1_to_xor2: STD_LOGIC;
SIGNAL and1_result: STD_LOGIC;
SIGNAL and2_result: STD_LOGIC;
SIGNAL and3_result: STD_LOGIC;
SIGNAL or1_to_or2: STD_LOGIC;

BEGIN
   label0: and_gate PORT MAP(a, b, and1_result);
   label1: and_gate PORT MAP(b, c, and2_result);
   label2: and_gate PORT MAP(c, a, and3_result);
   label3: xor_gate PORT MAP(a, b, xor1_to_xor2);
   label4: xor_gate PORT MAP(xor1_to_xor2, c, s);
   label5: or_gate PORT MAP(and1_result, and2_result, or1_to_or2);
   label6: or_gate PORT MAP(or1_to_or2, and3_result, cr);
END structural;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/fa.vhd 0 0 0 44:1 LF UTF-8 VHDL

VHDL: Testbench

- A testbench is a VHDL code that is used to perform simulations.
- A testbench consists of:
  - The design we want to simulate.
  - The inputs that are going to be fed to the design.
  - A procedure that checks the generated outputs.

VHDL: Testbench (cont.)

testbench.vhd

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY even_detector_testbench IS
END even_detector_testbench;

ARCHITECTURE tb_arch OF even_detector_testbench IS
COMPONENT even_detector
PORT(
        a: IN STD_LOGIC_VECTOR(2 DOWNTO 0);
        even: OUT STD_LOGIC);
END COMPONENT;

SIGNAL test_in: STD_LOGIC_VECTOR(2 DOWNTO 0);
SIGNAL test_out: STD_LOGIC;

BEGIN
dut: even_detector PORT MAP(a=>test_in, even =>test_out);

PROCESS
  BEGIN
    test_in <= "0000";
```

testbench.vhd

```vhdl
  BEGIN
    test_in <= "0000";
    WAIT FOR 100 ns;
      ....
END PROCESS;

PROCESS
  VARIABLE error_status: BOOLEAN;
  BEGIN
    WAIT ON test_in;
    WAIT FOR 100 ns;

    IF ((test_in="0000" AND test_out='1') OR
        (....) OR (....)) THEN error_status := FALSE;
    ELSE
      error_status := TRUE;
    END IF;

    ASSERT NOT error_status REPORT "failed" SEVERITY note;
  END PROCESS;
END tb_arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/testbench.vhd 0 0 0 39:49 LF UTF-8 VHDL

VHDL: Sequential Circuits

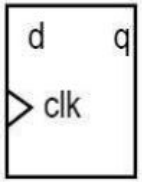
- Combinational Circuits:
  - Outputs depend solely on current inputs.
- Sequential Circuits:
  - Outputs depend on current and past inputs (memory).

- Basic Memory Elements:
  - D Latch
  - D Flip Flop
  - RAM

VHDL: Sequential Circuits (cont.)

- DFF:
  - Clock input
  - Data input
  - Negative/Positive Edge Triggered
  - Synchronous/Asynchronous Reset
  - Enable signal
  - Reset signal

VHDL: Positive Edge Triggered DFF



clk	q*
0	q
1	q
↑	d

ex.vhd

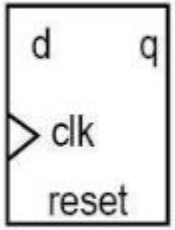
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY dff IS
PORT(
   clk, d: IN STD_LOGIC;
   q: OUT STD_LOGIC);
END dff;

ARCHITECTURE arch OF dff IS
BEGIN
PROCESS(clk)
   BEGIN
      IF (clk'EVENT AND clk='1') THEN q <= d;
      END IF;
END PROCESS;
END arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 18:1 LF UTF-8 VHDL

VHDL: DFF with Asynchronous Reset



reset	clk	q*
1	-	0
0	0	q
0	1	q
0	↑	d

ex.vhd

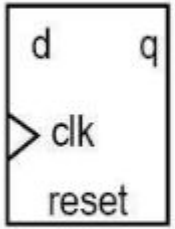
```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY dffr IS
PORT(
   clk, reset, d: IN STD_LOGIC;
   q: OUT STD_LOGIC);
END dffr;

ARCHITECTURE arch OF dffr IS
BEGIN
PROCESS(clk,reset)
   BEGIN
      IF (reset='1') THEN q <='0';
      ELSIF RISING_EDGE(clk) THEN q <= d;
      END IF;
END PROCESS;
END arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 19:1 LF UTF-8 VHDL

VHDL: DFF with Synchronous Reset



reset	clk	q*
1	-	0
0	0	q
0	1	q
0	↑	d

ex.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY dffr IS
PORT(
   clk, reset, d: IN STD_LOGIC;
   q: OUT STD_LOGIC);
END dffr;

ARCHITECTURE arch OF dffr IS
BEGIN
PROCESS(clk)
   BEGIN
      IF (clk'EVENT AND clk='1') THEN
         IF (reset = '1') THEN q <='0';
         ELSE q <= d;
         END IF;
      END IF;
END PROCESS;
END arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 21:1 LF UTF-8 VHDL

VHDL: Registers

- Register:
  - Multiple DFFs
  - Same Clock
  - Same Reset

ex.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY reg8 IS
PORT(
   clk, reset: IN STD_LOGIC;
   d: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
   q: OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END reg8;

ARCHITECTURE arch OF reg8 IS
BEGIN
PROCESS(clk, reset)
   BEGIN
      IF (reset='1') THEN q <= (OTHERS=>'0');
      ELSIF (clk'EVENT AND clk='1') THEN q <= d;
      END IF;
END PROCESS;
END arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 20:1 LF UTF-8 VHDL

VHDL: Shift Registers

- Categorized based on the way inputs are inserted and outputs are generated:
  - Serial In - Serial Out (SISO)
  - Serial In - Parallel Out (SIPO)
  - Parallel In - Parallel Out (PIPO)
  - Parallel In - Serial Out (PISO)

VHDL: Serial In - Serial Out

ex.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY siso IS
PORT(
   clk  : IN STD_LOGIC;
   si   : IN STD_LOGIC;
   so   : OUT STD_LOGIC);
END siso;

ARCHITECTURE arch OF siso IS
SIGNAL temp : STD_LOGIC_VECTOR(7 DOWNTO 0);
BEGIN
PROCESS (clk)
BEGIN
   IF (clk'EVENT AND clk='1') THEN
      temp(7 DOWNTO 1) <= temp(6 DOWNTO 0);
      temp(0) <= si;
   END IF;
```

ex.vhd

```
PORT(
   clk  : IN STD_LOGIC;
   si   : IN STD_LOGIC;
   so   : OUT STD_LOGIC);
END siso;

ARCHITECTURE arch OF siso IS
SIGNAL temp : STD_LOGIC_VECTOR(7 DOWNTO 0);
BEGIN
PROCESS (clk)
BEGIN
   IF (clk'EVENT AND clk='1') THEN
      temp(7 DOWNTO 1) <= temp(6 DOWNTO 0);
      temp(0) <= si;
   END IF;
END PROCESS;
so <= temp(7);
END arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 23:1 LF UTF-8 VHDL

VHDL: Serial In - Parallel Out

ex.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY sipo IS
PORT(
   clk, si: IN STD_LOGIC;
   po     : INOUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END sipo;

ARCHITECTURE arch OF sipo IS
BEGIN
PROCESS(clk)
   BEGIN
      IF (clk='1' AND clk'EVENT) THEN
         po(7 DOWNTO 1) <= po(6 DOWNTO 0);
         po(0) <= si;
      END IF;
END PROCESS;
END arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 20:1 LF UTF-8 VHDL

VHDL: Parallel In - Parallel Out

ex.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY pipo IS
PORT(
   clk  : IN STD_LOGIC;
   pi   : IN STD_LOGIC_VECTOR(7 DOWNTO 0);
   po   : OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END pipo;

ARCHITECTURE arch OF pipo IS
BEGIN
PROCESS (clk)
   BEGIN
      IF (clk'EVENT AND clk='1') THEN po <= pi;
   END IF;
END PROCESS;
END arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 19:1 LF UTF-8 VHDL

VHDL: Parallel In - Serial Out

ex.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY piso IS
PORT(
   clk, load  : IN STD_LOGIC;
   pi         : IN STD_LOGIC_VECTOR(7 DOWNTO 0)
   so         : OUT STD_LOGIC);
END piso;

ARCHITECTURE arch OF piso IS
SIGNAL t   : STD_LOGIC;
SIGNAL temp: STD_LOGIC_VECTOR(7 DOWNTO 0);
BEGIN
PROCESS (clk,pi,load)
   BEGIN
      IF (load='1') THEN
         temp(7 DOWNTO 0) <= pi(7 DOWNTO 0);
      ELSIF (clk'EVENT AND clk='1') THEN
```

ex.vhd

```
ARCHITECTURE arch OF piso IS
SIGNAL t   : STD_LOGIC;
SIGNAL temp: STD_LOGIC_VECTOR(7 DOWNTO 0);
BEGIN
PROCESS (clk,pi,load)
   BEGIN
      IF (load='1') THEN
         temp(7 DOWNTO 0) <= pi(7 DOWNTO 0);
      ELSIF (clk'EVENT AND clk='1') THEN
         t <= temp(7);
         temp(7 DOWNTO 1) <= temp(6 DOWNTO 0);
         temp(0) <= '0';
      END IF;
END PROCESS;

so <= t;
END arch;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 27:4 LF UTF-8 VHDL

VHDL: Register Characteristics

- Enable Signals
- Clear Signals
- Reset Signals
- Set Signals

VHDL: 8-bit BCD Counter

ex.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY counter IS
PORT(
   clock_enable, clock, reset: IN STD_LOGIC;
   outpt: OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END counter;

ARCHITECTURE behavioral OF counter IS
SIGNAL temp: STD_LOGIC_VECTOR(7 DOWNTO 0);
BEGIN
PROCESS(clock, reset)
   BEGIN
   IF (reset = '1') THEN
      temp <= "00000000";
   ELSIF (RISING_EDGE(clock)) THEN
      IF (clock_enable = '0') THEN
         IF (temp = "10000001") THEN
```

ex.vhd

```
SIGNAL temp: STD_LOGIC_VECTOR(7 DOWNTO 0);
BEGIN
PROCESS(clock, reset)
   BEGIN
   IF (reset = '1') THEN
      temp <= "00000000";
   ELSIF (RISING_EDGE(clock)) THEN
      IF (clock_enable = '0') THEN
         IF (temp = "10000001") THEN
            temp <= "00000000";
         ELSE temp <= temp + 1;
         END IF;
      END IF;
   END IF;
END PROCESS;

outpt<=temp;
END behavioral;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 27:13 LF UTF-8 VHDL

VHDL: 8-bit Binary Counter (with load enable)

ex.vhd

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY count8 IS
PORT(
   din: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
   clk, load: IN STD_LOGIC;
   dout: OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END count8;

ARCHITECTURE behavioral OF count8 IS
BEGIN
PROCESS(clk)
   VARIABLE count: UNSIGNED(7 DOWNTO 0):= "000
   BEGIN
      IF (clk'EVENT AND clk = '1') THEN
         IF load = '1' THEN count := din;
         ELSE count := count + 1;
         END IF;
```

ex.vhd

```
   clk, load: IN STD_LOGIC;
   dout: OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END count8;

ARCHITECTURE behavioral OF count8 IS
BEGIN
PROCESS(clk)
   VARIABLE count: UNSIGNED(7 DOWNTO 0):= "000
   BEGIN
      IF (clk'EVENT AND clk = '1') THEN
         IF load = '1' THEN count := din;
         ELSE count := count + 1;
         END IF;
      END IF;

      dout <= count;
   END PROCESS;
END behavioral;
```

~/Desktop/cs.uth Εργαστήριο Οργάνωση Ηλεκτρονικών Υπολογιστών/ex.vhd 0 0 0 24:4 LF UTF-8 VHDL