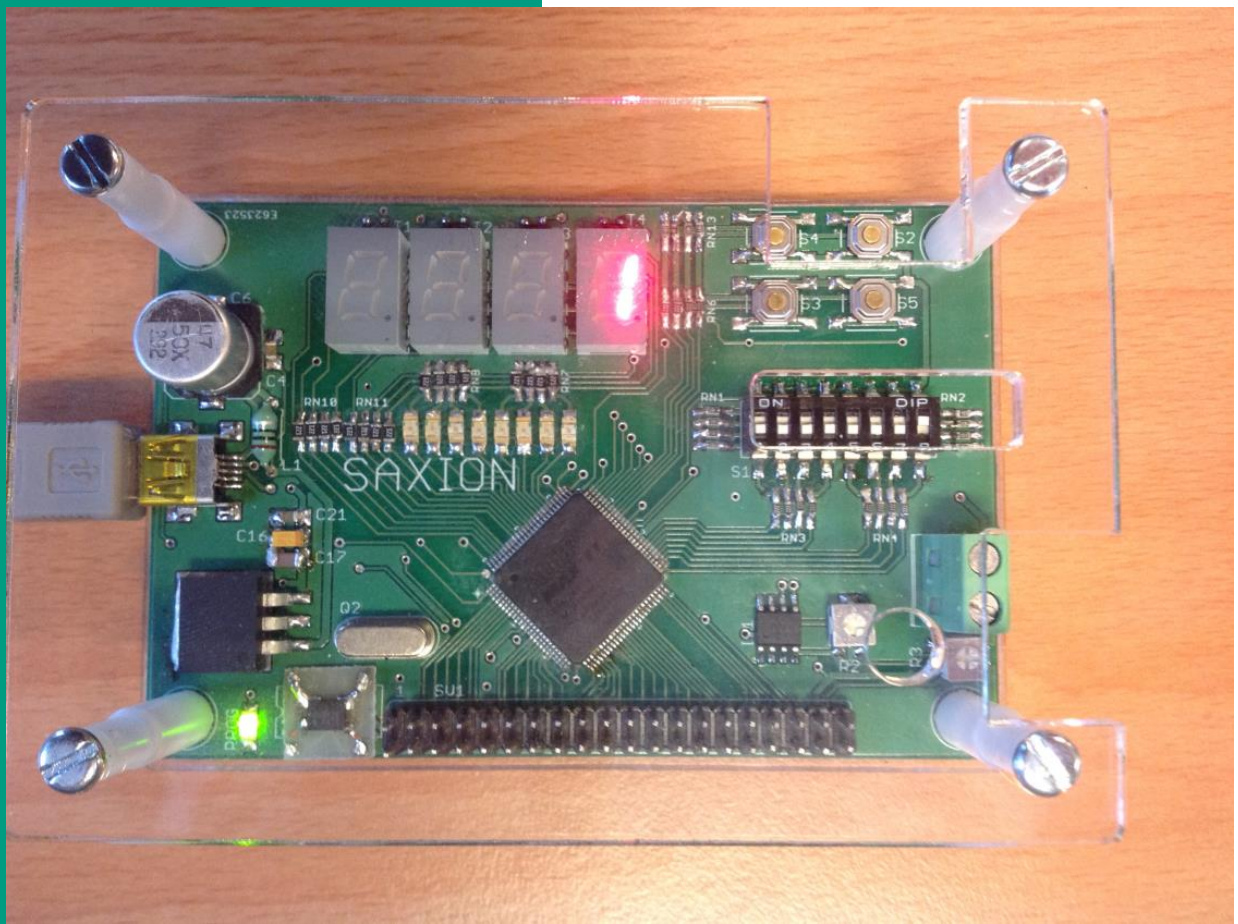
Very High Speed Integrated Circuits Hardware Description Language

Lesson 5
VHDL
Statemachines

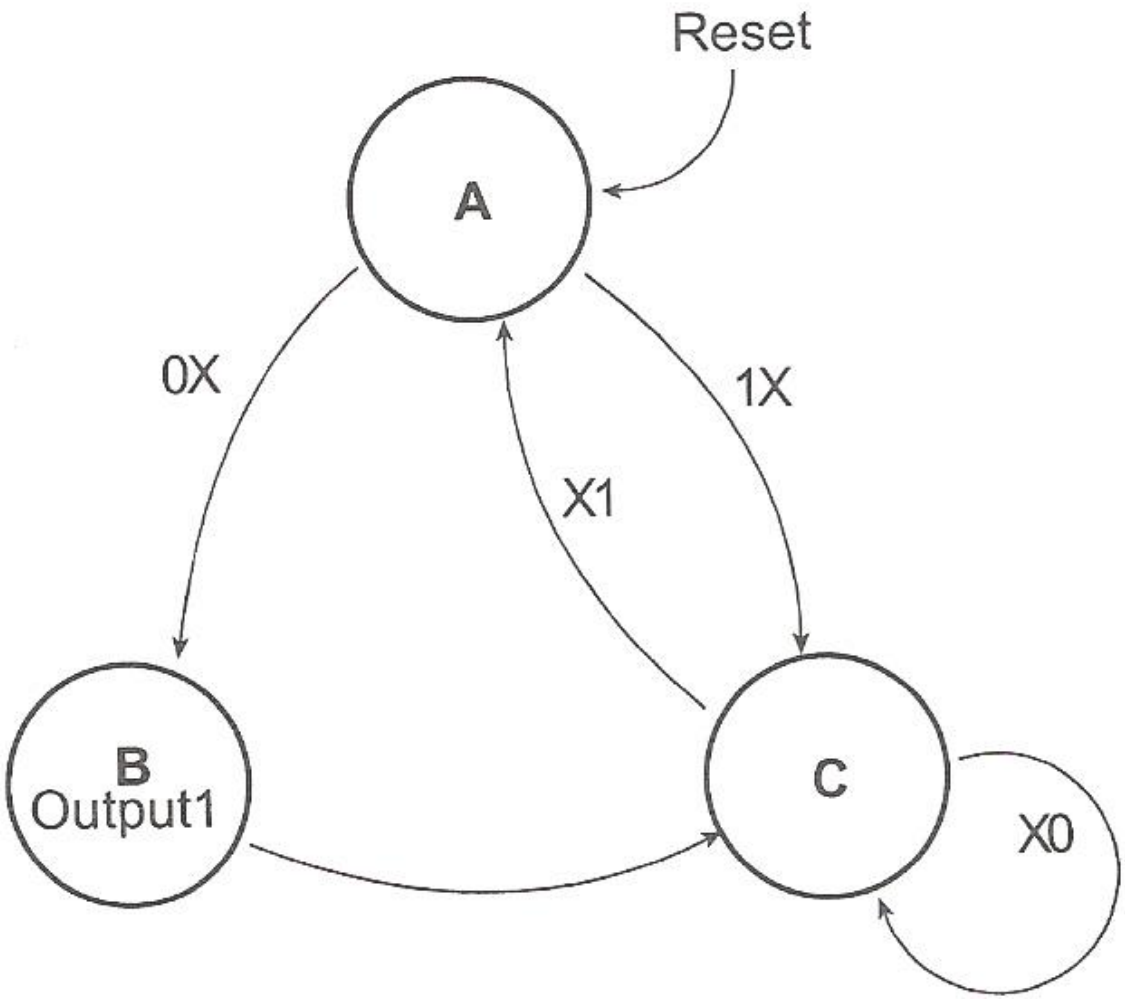


SAXION
UNIVERSITY OF APPLIED SCIENCES

Lesson 5 outline

- Statemachines(Chapter 11)
- ALU (Chapter 12)
- Multiply (Chapter 13)
- Memory (Chapter 14)
- Inferred latches(Chapter 9)
- Counters (Chapter 10)
- Simulating with Quartus simulator
- FPGA MAX II technology
- LUT
- I/O

State Machines



State Machines (flow)

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
ENTITY st_mach IS
        PORT(   clk, reset   : IN      STD_LOGIC;
                Input1, Input2       : IN      STD_LOGIC;
                Output1              : OUT     STD_LOGIC);
END st_mach;

ARCHITECTURE A OF st_mach IS
-- Enumerated Data Type for State
        TYPE STATE_TYPE IS (state_A, state_B, state_C);
        SIGNAL state: STATE_TYPE;
BEGIN
        PROCESS (reset, clk)
        BEGIN
                IF reset = '1' THEN
                        state <= state_A;
                ELSIF clk'EVENT AND clk = '1' THEN
                        CASE state IS
```

State Machines

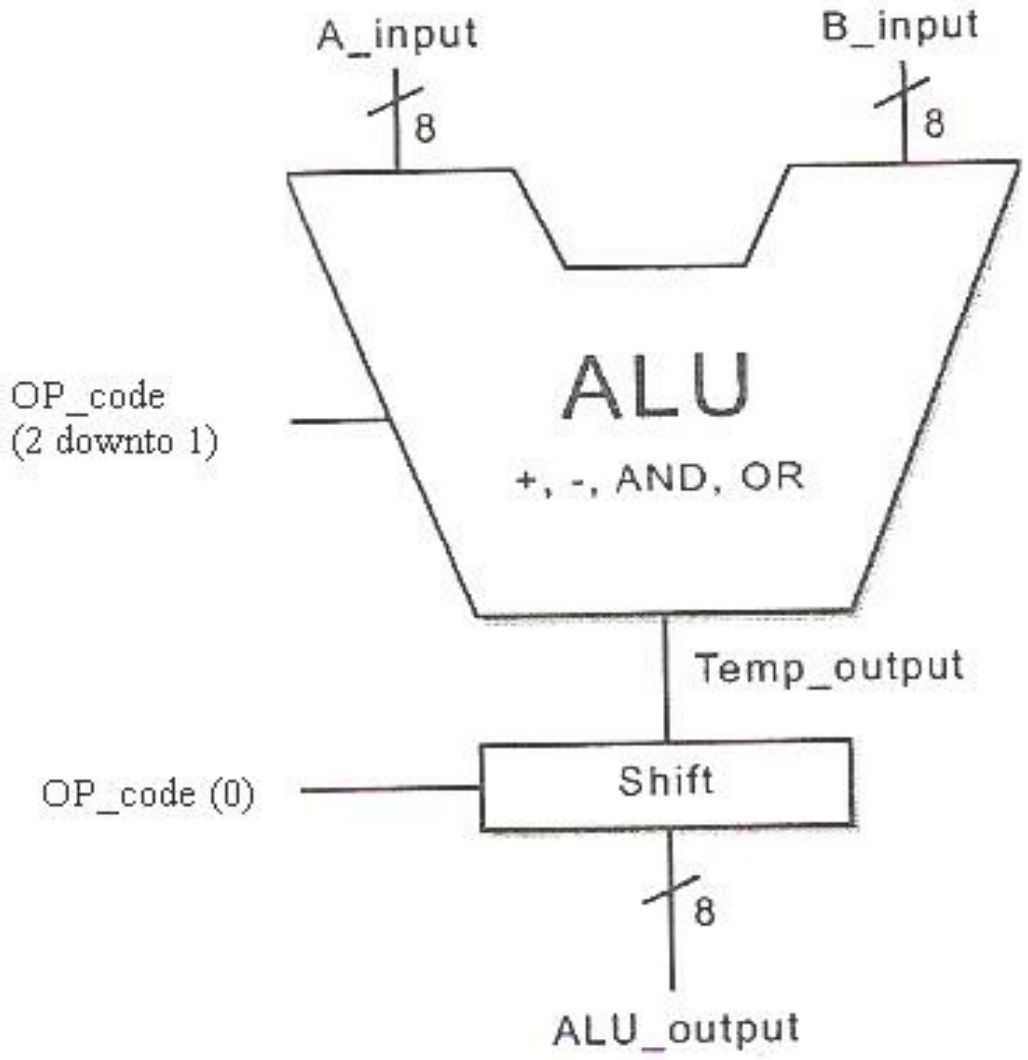
```
ELSIF clk'EVENT AND clk = '1' THEN
                CASE state IS
                        WHEN state_A =>
                                IF Input1 = '0' THEN
                                        state <= state_B;
                                ELSE
                                        state <= state_C;
                                END IF;
                        WHEN state_B =>
                                        state <= state_C;
                        WHEN state_C =>
                                IF Input2 = '1' THEN
                                        state <= state_A;
                                END IF;
                        WHEN OTHERS =>
                                state <= state_A;
                END CASE;
        END IF;

END PROCESS;
```

State Machines (output decoder)

```
PROCESS (state)
   BEGIN
     CASE state IS
        WHEN A =>
                        output1         <=      ‘0';
         WHEN B =>
                        output1         <=      '1';
         WHEN C =>
                        output1         <=      ‘0';
END CASE;
```

VHDL Synthesis Model of an ALU



12. VHDL Synthesis Model of an ALU library and entity

```
LIBRARY IEEE;
USE  IEEE.STD_LOGIC_1164.all;
USE  IEEE.STD_LOGIC_ARITH.all;
USE  IEEE.STD_LOGIC_UNSIGNED.all;

ENTITY ALU IS
  PORT(
        Op_code          : in std_logic_vector(2 DOWNTO 0);
        A_input, B_input  : in std_logic_vector(7 DOWNTO 0);
        ALU_output       : out std_logic_vector(7 DOWNTO 0));

END ALU;
```

VHDL Synthesis Model of an ALU A architecture

```
ARCHITECTURE behavior OF ALU IS

SIGNAL temp_output: std_logic_vector(7 DOWNTO 0);
BEGIN
  PROCESS (Op_code, A_input, B_input)
            BEGIN
                        -- Select Arithmetic/Logical Operation
            CASE Op_Code (2 DOWNTO 1) IS
                        WHEN "00" =>
                           temp_output <= A_input + B_input;
                        WHEN "01" =>
                           temp_output <= A_input - B_input;
                        WHEN "10" =>
                           temp_output <= A_input AND B_input;
                        WHEN "11" =>
                           temp_output <= A_input OR B_input;
                        WHEN OTHERS =>
                           temp_output <= "00000000";
            END CASE;
                                    -- Select Shift Operation
            IF Op_Code(0) = '1' THEN
                                    -- Shift bits left with zero fill using concatination operator
                                    -- can also use VHDL 1076-1993 shift operator such as SLL
                  Alu_output <= temp_output(6 DOWNTO 0) & '0';
            ELSE
                  Alu_output <= temp_output;
            END IF;
   END PROCESS;
  END behavior;
```

VHDL Synthesis Model of multiply and Divide hardware

-Efficient design requires vendor specific library function

-Use MegaWizard in Quartus

- Foating point operations needs big FPGA, Special functions are available , for example Digital filtering

VHDL Synthesis Model of multiply and Divide hardware ALU

```
LIBRARY IEEE;
USE  IEEE.STD_LOGIC_1164.all;
USE IEEE.STD_LOGIC_ARITH.all;
USE IEEE.STD_LOGIC_UNSIGNED.all;
LIBRARY lpm;
USE lpm.lpm_components.ALL;

ENTITY  mult IS
            PORT(       A,B             : IN        STD_LOGIC_VECTOR(7  DOWNTO  0);
                        Product         : OUT       STD_LOGIC_VECTOR(15  DOWNTO  0));
END mult;

ARCHITECTURE  a OF mult IS
BEGIN
 multiply: lpm_mult
  GENERIC  MAP(         LPM_WIDTHA  => 8,
                                LPM_WIDTHB  => 8,
                                LPM_WIDTHS  => 16,
                                LPM_WIDTHP  => 16,
                                LPM_REPRESENTATION  => "UNSIGNED")
   PORT MAP (           dataa => A,
                        datab => B,
                        result => Product);
   END a;
```

VHDL Synthesis models for memory

```
USE IEEE.STD_LOGIC1164.ALL;

ENTITY amemory IS
        PORT(read_data          :OUT STD_LOGIC_VECTOR( 7 DOWNTO 0 );
             memory_address:    IN STD_LOGIC_VECTOR( 2 DOWNTO 0 );
             write_data         :IN STD_LOGIC_VECTOR( 7 DOWNTO 0 );
             Memwrite           :IN STD_LOGIC;
             Clock,reset    :   IN STD_LOGIC );
END amemory;

ARCHITECTURE behavior OF amemory IS
BEGIN
Data_memory:lpm_ram_dq          --LPM memory function

GENERIC MAP (lpm_widthad        =>3,
             lpm_outdata        =>"UNREGISTERED" ,
             lpm_indata         =>"REGISTERED",
             lpm_address_control =>"UNREGISTERED" ,

        PORT MAP (data => write_data, address => memory_address( 2 DOWNTO 0 ),
                  We   => Memwrite, inclock => clock, q =>read_data );
END behavior;
```

Assignment week 5

- Do self test chapter 11,12
- Do the exercises chapter 11,12,14