

**ECE 331**

**Testbenches - 2**



High Performance

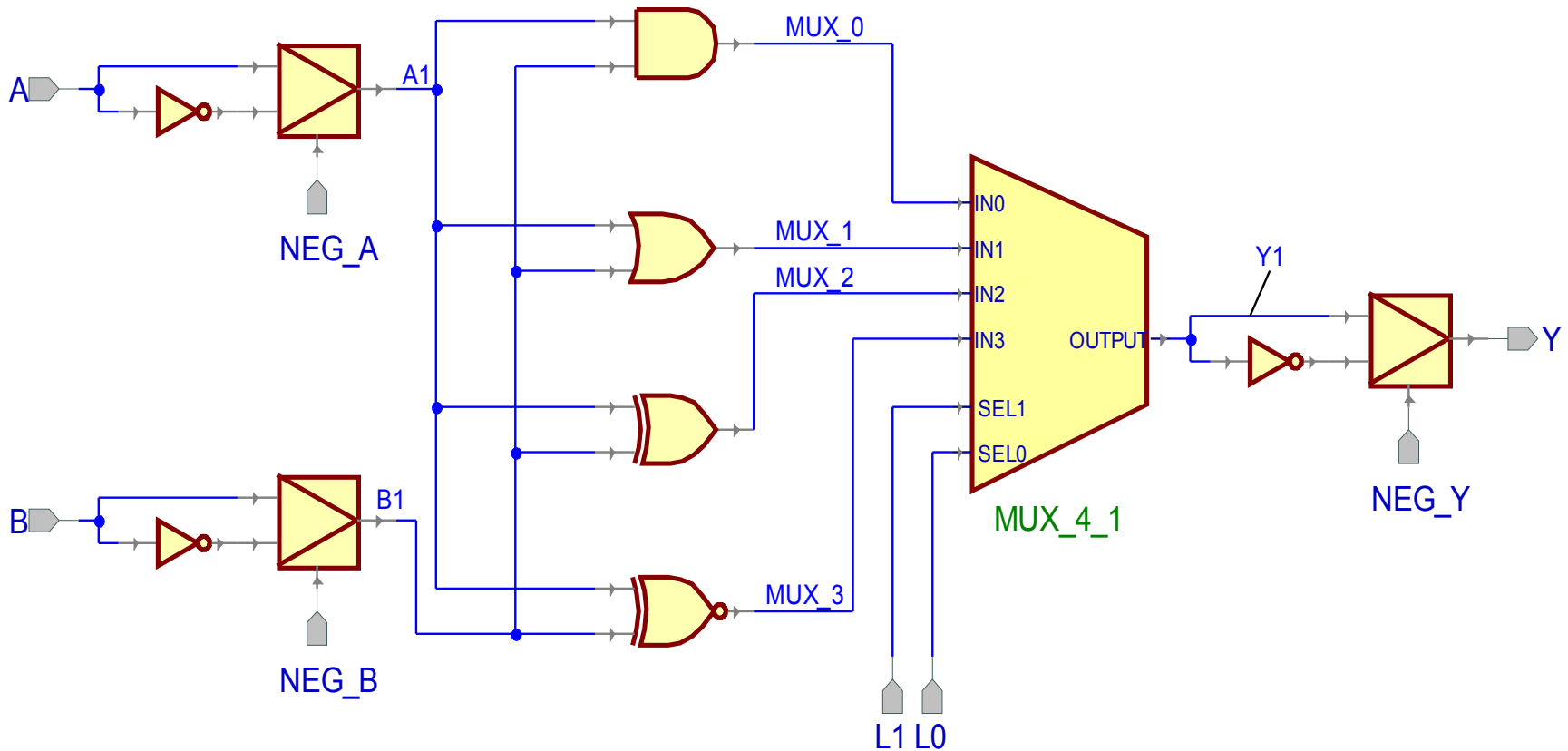
CoolClock

Low Power

CoolCache



# MLU Block Diagram



# MLU: Entity Declaration

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mlu IS
    PORT(
        NEG_A : IN STD_LOGIC;
        NEG_B : IN STD_LOGIC;
        NEG_Y : IN STD_LOGIC;
        A :     IN STD_LOGIC;
        B :     IN STD_LOGIC;
        L1 :    IN STD_LOGIC;
        L0 :    IN STD_LOGIC;
        Y :     OUT STD_LOGIC
    );
END mlu;
```

# MLU: Architecture Declarative Section

```
ARCHITECTURE mlu_dataflow OF mlu IS
```

```
SIGNAL A1 : STD_LOGIC;
```

```
SIGNAL B1 : STD_LOGIC;
```

```
SIGNAL Y1 : STD_LOGIC;
```

```
SIGNAL MUX_0 : STD_LOGIC;
```

```
SIGNAL MUX_1 : STD_LOGIC;
```

```
SIGNAL MUX_2 : STD_LOGIC;
```

```
SIGNAL MUX_3 : STD_LOGIC;
```

```
SIGNAL L: STD_LOGIC_VECTOR(1 DOWNTO 0);
```

# MLU - Architecture Body

```
BEGIN
  A1<= NOT A WHEN (NEG_A='1') ELSE
    A;
  B1<= NOT B WHEN (NEG_B='1') ELSE
    B;
  Y <= NOT Y1 WHEN (NEG_Y='1') ELSE
    Y1;

  MUX_0 <= A1 AND B1;
  MUX_1 <= A1 OR B1;
  MUX_2 <= A1 XOR B1;
  MUX_3 <= A1 XNOR B1;

  L <= L1 & L0;

  with (L) select
    Y1 <= MUX_0 WHEN "00",
          MUX_1 WHEN "01",
          MUX_2 WHEN "10",
          MUX_3 WHEN OTHERS;

END mlu_dataflow;
```

# MLU: Testbench (1)

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity MLU_TB is
end MLU_TB;

architecture MLU_TB_ARCHITECTURE of MLU_TB is
    -- Component declaration of the tested unit
    component mlu
    port(
        NEG_A : in std_logic;
        NEG_B : in std_logic;
        NEG_Y : in std_logic;
        A : in std_logic;
        B : in std_logic;
        L1 : in std_logic;
        L0 : in std_logic;
        Y : out std_logic );
    end component;
```

# MLU: Testbench (2)

-- Stimulus signals - signals mapped to the input and inout ports of tested entity

```
signal TEST_NEG_A : std_logic;
```

```
signal TEST_NEG_B : std_logic;
```

```
signal TEST_NEG_Y : std_logic;
```

-- Observed signals - signals mapped to the output ports of tested entity

```
signal TEST_Y : std_logic;
```

```
signal TEST_AB: std_logic_vector(1 downto 0);
```

```
signal TEST_SEL: std_logic_vector(1 downto 0);
```

-- Unit Under Test port map

```
UUT : mlu
```

```
    port map (
```

```
        NEG_A => TEST_NEG_A,
```

```
        NEG_B => TEST_NEG_B,
```

```
        NEG_Y => TEST_NEG_Y,
```

```
        A => TEST_AB(1),
```

```
        B => TEST_AB(0),
```

```
        L1 => TEST_SEL(1),
```

```
        L0 => TEST_SEL(0),
```

```
        Y => TEST_Y);
```

# MLU: Testbench (3)

```
TESTING: process  
begin
```

```
    TEST_NEG_A<='0';  
    TEST_NEG_B<='0';  
    TEST_NEG_Y<='0';
```

```
    TEST_AB<="00";  
    TEST_SEL<="00";
```

```
    for I in 0 to 3 loop
```

```
        for J in 0 to 3 loop
```

```
            wait for 10 ns;
```

```
            TEST_AB<=TEST_AB+"01";
```

```
        end loop;
```

```
        TEST_SEL<=TEST_SEL+"01";
```

```
    end loop;
```

# MLU: Testbench (4)

---

-- Inputs Negated

---

```
TEST_NEG_A<='1';  
TEST_NEG_B<='1';  
TEST_NEG_Y<='0';
```

```
TEST_AB<="00";  
TEST_SEL<="00";
```

```
for I in 0 to 3 loop  
    for J in 0 to 3 loop  
        wait for 10 ns;  
        TEST_AB<=TEST_AB+"01";  
    end loop;  
    TEST_SEL<=TEST_SEL+"01";  
end loop;
```

```
end process TESTING;
```

# Asserts & Reports



High Performance

CoolClock

Low Power

Low Latency



# Assert

Assert is a **non-synthesizable** statement whose purpose is to write out messages on the screen when problems are found during simulation.

Depending on the **severity of the problem**, The simulator is instructed to continue simulation or halt.

# Assert - syntax

```
ASSERT condition  
[REPORT "message"  
[SEVERITY severity_level ];
```

**The message is written when the condition is FALSE.**

Severity\_level can be:

**Note, Warning, Error (default), or Failure.**

# Assert - Examples

```
assert initial_value <= max_value  
    report "initial value too large"  
    severity error;
```

```
assert packet_length /= 0  
    report "empty network packet received"  
    severity warning;
```

```
assert false  
    report "Initialization complete"  
    severity note;
```

# Report - syntax

```
REPORT "message"  
[SEVERITY severity_level ];
```

**The message is always written.**

Severity\_level can be:

**Note (default), Warning, Error, or Failure.**

# Report - Examples

```
report "Initialization complete";
```

```
report "Current time = " & time'image(now);
```

```
report "Incorrect branch" severity error;
```

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity example_1_tb is
end example_1_tb;

architecture behavioral of example_1_tb is
    signal clk : std_logic := '0';
begin
    clk <= not clk after 100 ns;
    process
        begin
            wait for 1000 ns;

            report "Initialization complete";
            report "Current time = " & time'image(now);

            wait for 1000 ns;
            report "SIMULATION COMPLETED" severity failure;
        end process;
end behavioral;
```