Comp 331: Intro to Computer Hardware
Today’s Overview

1. Combinatorial Logic ← Tuesday
   1. 2-to-1 MUX
   2. 4-to-1 MUX
   3. Quad 2-to-1 MUX
   4. VHDL implementations

2. n-input Multiplexers
   • 2-to-1 MUX
   • 4-to-1 MUX
   • Quad 2-to-1 MUX

3. Advanced VHDL Statements ← pick up here
   • if statement
   • procedural/sequential statements; process block
   • when statement
   • case statement

4. Generics and Generic Statements
   • Parameterizing component designs

---

2-to-1 MUX

VHDL Example 6.B

if statement

and

VHDL process stmt

sort of like a "procedure"

---

4-to-1 MUX

Example 7: 4-to-1 MUX via Cascading 2-to-1 MUXes

---

4-to-1 MUX

Case Statement

within process statement

when others required
4-to-1 MUX

**Entity**

```vhdl
entity mux41_top is
  port(
    sw : in STD_LOGIC_VECTOR(3 downto 0);
    btn : in STD_LOGIC_VECTOR(1 downto 0);
    ld : out STD_LOGIC_VECTOR(0 downto 0);
  )
end mux41_top;
```

**Architecture**

```vhdl
architecture mux41_top of mux42_top is
  component mux41x is
    port(
      c : in STD_LOGIC_VECTOR(3 downto 0);
      s : in STD_LOGIC_VECTOR(1 downto 0);
      z : out STD_LOGIC;
    )
  end component;

  begin
  QM4: mux41x port map
    (c => sw(3 downto 0), s => btn(1 downto 0),
    z => ld);
end
```

Quad 2-to-1 MUX

**Entity**

```vhdl
entity mux24x is
  port(
    a : in STD_LOGIC_VECTOR(3 downto 0);
    b : in STD_LOGIC_VECTOR(3 downto 0);
    s : in STD_LOGIC;
    y : out STD_LOGIC_VECTOR(3 downto 0);
  )
end mux24x;
```

**Architecture**

```vhdl
architecture mux24a of mux24x is
  signal s4: STD_LOGIC_VECTOR(3 downto 0);
  begin
  s4 <= s & s & s & s;
  y <= (not s4 and a) or (s4 and b);
end
```

Quad 4-to-1 MUX

**Alternate architectures**

**Generic Multiplexers: Parameterized Design**

**Library**

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

**Entity**

```vhdl
entity mux2g is
  generic (N: integer := 4);
  port(
    a : in STD_LOGIC_VECTOR(N-1 downto 0);
    b : in STD_LOGIC_VECTOR(N-1 downto 0);
    s : in STD_LOGIC;
    y : out STD_LOGIC_VECTOR(N-1 downto 0);
  )
end mux2g;
```

**Architecture**

```vhdl
architecture mux2g of mux2g is
  begin
  p1: process(a,b,s)
  begin
  if s = '0' then
    y <= a;
  else
    y <= b;
  end if;
  end process;
end
```

Generic Multiplexer: Parameterized Design

**Entity**

```vhdl
entity mux28 is
  port(
    a : in STD_LOGIC_VECTOR(7 downto 0);
    b : in STD_LOGIC_VECTOR(7 downto 0);
    s : in STD_LOGIC;
    y : out STD_LOGIC_VECTOR(7 downto 0);
  )
end mux28;
```

**Architecture**

```vhdl
architecture mux28 of mux28 is
  begin
  m8: mux2g generic map (N => 8)
    port map
    (a => sw(3 downto 0), b => sw(7 downto 4),
    s => btn(0), y => ld);
end
```