CS 232 Computer Organization

VHDL

- Library: import useful packets
  - library ieee;
  - use ieee.std_logic_1164.all; (provide enhanced signal types)
  - use ieee.numerics_std.all; (unsigned/signed arrays of std_logic for signals)
- VHDL programs have two parts: entity and architecture
  - entity:
    - entity name has to match the filename (thing.vhd)
    - contains a port statement that describe the name of the circuit and its inputs and outputs. All in/outputs are signals, which you can think of as wires.
    - each input/output must be labeled as to its usage (IN, OUT, INOUT) and type

```
entity andgate is
  port( I1: in std_logic;
      I2: in std_logic;
      Q: out std_logic);
end andgate;
```

- Signals: IEEE standard types
  - std_logic: represent one bit
    - a signal or variable of this type can take on the value 0 or 1.

```
a: in std_logic;
signal s1, s2: std_logic;
```

  - std_logic_vector: an array of type std_logic;
    - 4 bit output, indexed from 3 to 0 with 3 being the index of the leftmost bit or the most significant bit; 8-bit register, indexed from 7 to 0

```
b: out std_logic_vector (3 downto 0);
signal register: std_logic_vector (7 down 0);
```

  - unsigned/signed: vector types for which addition and subtraction are defined

```
c: in unsigned (7 downto 0);
signal counter: unsigned (7 down 0);
```

- architecture:
  - The circuit definition. Code describes the circuit goes here.
  - Declaration section (before the begin)
    - Temporary local variables
Component declarations: define other circuits to be included

- Code section (between \texttt{begin} and \texttt{end})
  - Concurrent statements: all statements in a VHDL architecture act simultaneously

```vhdl
architecture architecture_name of entity_name is

architecture test of boxtest is
  signal A: std_logic_vector(3 downto 0);
  signal Aout: std_logic_vector(3 downto 0);

  component boxdriver
    port(
      A: in std_logic_vector(3 downto 0);
      result: out std_logic_vector(3 downto 0)
    );
  end component;

  begin
    A(3) <= '0', '1' after 200 ns, '0' after 400 ns;
    A(2) <= '0', '1' after 100 ns, '0' after 200 ns, '1' after 300 ns;
    A(1) <= '0', '1' after 50 ns, '0' after 100 ns, '1' after 150 ns, '0' after 200 ns, '1' after 250 ns, '0' after 300 ns, '1' after 350 ns;
    A(0) <= '0', '1' after 25 ns, '0' after 50 ns, '1' after 75 ns, '0' after 100 ns, '1' after 125 ns, '0' after 150 ns, '1' after 175 ns, '0' after 200 ns, '1' after 225 ns, '0' after 250 ns, '1' after 275 ns, '0' after 300 ns, '1' after 325 ns, '0' after 350 ns, '1' after 375 ns;

    C0: boxdriver port map( A, Aout );
  end test;
end architecture_name
```

- Statements:
  - Assignments: a wire connecting two nodes of the circuit

```vhdl
-- connects the value of B to the value of A with a wire
A <= B;

-- connects the value of B to the value of A with a wire and a time delay
A <= B after 5 ns;

-- selects B or (not B) to connect to A depending upon signal S
A <= B when S = '0' else (not B);

-- combines a switch and delay
A <= B after 5 ns when S = '0' else (not B) after 5 ns
```
- Port maps: creates an instance of an entity and connects the inputs and outputs of the entity with local signals

```
'label' : <entity> port map ( <argument list> );

-- example
A0: andgate port map (I1 => A, I2 => B, Q => C);
```

- Processes: permit the programmer to write a functional description of the circuit using more traditional control flow techniques

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

entity driver is
  port( I: in std_logic;
       Q: out std_logic);
end driver;

architecture behavior of driver is begin

  process(I)
  begin
    -- compare to truth table
    if (I='1') then
      Q <= '1';
    else
      Q <= '0';
    end if;
  end process;
end behavior;
```

Useful Combinational Circuits

- Encoder

```
def ENC (I):
    if I == "0001": return [0, 0]
    elif I == "0010": return [0, 1]
    elif I == "0100": return [1, 0]
    elif I == "1000": return [1, 1]
```
• Priority Encoder
  - $2^N$ inputs and $N + 1$ output. The $N$ output lines whose 2-bit binary value indicate which of $2^N$ input lines is high, the rest output indicating all are low.
  - Draw the truth table

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>Q0</th>
<th>Q1</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>d</td>
<td>d</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>d</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>d</td>
<td>d</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>d</td>
<td>d</td>
<td>d</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>