ECE 545
Lecture 10b
Non-synthesizable VHDL
Poor Design Practices
Required reading

• P. Chu, *RTL Hardware Design using VHDL*

*Chapter 9.1, Poor design practices and their remedies*
Non-synthesizable VHDL
Delays

Delays are not synthesizable

Statements, such as

```
   wait for 5 ns
   a <= b after 10 ns
```

will not produce the required delay, and should not be used in the code intended for synthesis.
Initializations

Declarations of signals (and variables) with initialized values, such as

```vhdl
SIGNAL a : STD_LOGIC := '0';
```

cannot be synthesized, and thus should be avoided.

If present, they will be ignored by the synthesis tools.

**Use set and reset signals instead.**
Dual-edge triggered register/counter (1)

In FPGAs register/counter can change only at either rising (default) or falling edge of the clock.

Dual-edge triggered clock is not synthesizable correctly, using either of the descriptions provided below.
Dual-edge triggered register/counter (2)

PROCESS (clk)
BEGIN
   IF (clk’EVENT AND clk=‘1’ ) THEN
      counter <= counter + 1;
   ELSIF (clk’EVENT AND clk=‘0’ ) THEN
      counter <= counter + 1;
   END IF;
END PROCESS;
Dual-edge triggered register/counter (3)

PROCESS (clk)
BEGIN
  IF (clk'EVENT) THEN
    counter <= counter + 1;
  END IF;
END PROCESS;

PROCESS (clk)
BEGIN
  counter <= counter + 1;
END PROCESS;
Poor Design Practices
1. Poor design practice and remedy

- Synchronous design is the most important methodology
- Poor practice in the past (to save chips)
  - Misuse of asynchronous reset
  - Misuse of gated clock
  - Misuse of derived clock
Misuse of Asynchronous Reset
(a) Block diagram

(b) Timing diagram
• Problem
  – Glitches in transition 1001 (9) => 0000 (0)
  – Glitches in aync_clr can reset the counter
  – How about timing analysis? (maximal clock rate)

• Asynchronous reset should only be used for power-on initialization
Decade (mod-10) counter

architecture two_seg_arch of mod10_counter is
constant TEN: integer := 10;
signal r_reg: unsigned(3 downto 0);
signal r_next: unsigned(3 downto 0);
begin
  -- register
  process (clk,reset)
  begin
    if (reset='1') then
      r_reg <= (others=>'0');
    elsif (clk'event and clk='1') then
      r_reg <= r_next;
    end if;
  end process;
  -- next-state logic
  r_next <= (others=>'0') when r_reg=(TEN-1) else
    r_reg + 1;
  -- output logic
  q <= std_logic_vector(r_reg);
end two_seg_arch;
Misuse of Gated Clock
Figure 9.2  Disabling FF with gated clock
• Problem
  – Gated clock width can be narrow
  – Gated clock may pass glitches of en
  – Difficult to design the clock distribution network
Dedicated Clock Tree Network – H Tree
Misuse of Derived Clock
Misuse of derived clock

- Subsystems may run at different clock rate
- Poor design: use a derived slow clock for slow subsystem
• Problem
  – Multiple clock distribution network
  – How about timing analysis? (maximal clock rate)
Better use a synchronous one-clock enable pulse
• E.g., second and minutes counter
  – Input: 1 MHz clock
  – Poor design:

(a) Design with derived clock
– Better design

(b) Design with a single synchronous clock