Testbench

- 考试要求：
    1. **能画出测试结果的波形**
    2. **写Testbench程序**
- ~~Testbench is a kind of VHDL Program~~
- ~~Procedures using testbench~~
    - ~~Design Under Test instantiation~~
    - ~~Stimulus generation~~
    - ~~Apply the stimulus on DUT and observe its response~~
    - ~~Compare the response with the expected result~~
- Non-synthesisable codes: cannot be translated to circuit, but are useful for testbench
    - `Wait`
    - `While Loop`
    - Infinite Loop
    - File input/output

Structure of Tenstbench

1. Lib calling
2. Package calling
3. Entity
    - **the entity of testbench is empty**
4. Architecture
    1. Component(DUT) Instantiation: Use `component` statements
    2. Signal Declaration
    3. DUT Port Mapping
    4. Stimulus Generations: Use `process` statements

Process Without Sensitivity Lists

- A process without a sensitivity list will keep executing in a loop, but its execution can be controlled using `wait` statements
- The `wait` statement suspends a process for a given perios of time(limited or forever) after which the process wakes up with status where it left off

Stimulus Generation

Clock Signal

- Types:
    - symmetric(duty ratio = 50%)
    - asymmetric

```
SIGNAL clk1,clk2: std_logic;
clk1_gen:PROCESS
    CONSTANT clk_period: TIME:= 40 ns;
BEGIN
    clk1 <= '1';
    WAIT FOR clk_period/2;
    clk1 <= '0';
    WAIT FOR clk_period/2;
END PROCESS;
```

Reset Signal

```
SIGNAL reset1,reset2: std_logic;
--reset1使用绝对时间，里面的时间表示绝对时刻
    reset1 <= '0','1' AFTER 60 ns,'0' AFTER 100 ns;
reset2_gen:PROCESS
--reset2使用相对时间，里面的时间表示每一段时间的长度
BEGIN
   reset2 <= '0';
   WAIT FOR 20 ns;
   reset2 <= '1';
   WAIT FOR 40 ns;
   reset2 <= '0';
   WAIT:
END PROCESS;
```

Complex Periodic Signal

- Same as the reset signal

Correlated Signals

- Use attributes `delay`

```
--period2相比period1有10ns的延时
SIGNAL period1,period2: std_logic;
...
period1 <= '0','1' AFTER 60 ns,'0' AFTER 100 ns;
period2 <= period1'delayed(10ns);
```

General Stimulus

- Use `wait` and `process`
- Aims to cover all possibilites of inputs
- Make use of `for` loop

Typical Errors

- Assignments to the same signal in different processes are not recommended. It may introduce assignment conflicts, and cause uncertain states.