# C6 – Délais, Delta Délais et variables

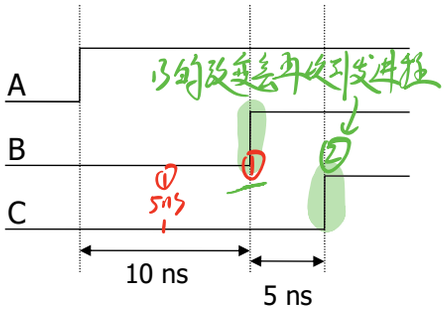
Yann DOUZE



# Les délais

chronogramme de A ,B et C ?

```
process (A,B)
begin
  B <= A after 10 ns;
  C <= B after 5 ns;
end process;
```



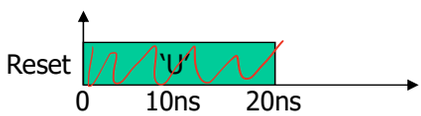

- Les délais sont utilisés pour :
  - Générer des stimuli pendant la simulation
  - Modéliser les retards dû à la technologie
- Les délais sont ignorés à la synthèse.

# Génération d'une impulsion (1)

1.没有敏感信号列表（只用于仿真），只执行一遍；
2. process结束后完成一次赋值；
3. 多重赋值，只执行最后一次赋值语句。

```
process
begin
   reset <= '0';
   reset <= '1' after 10 ns;
   reset <= '0' after 20 ns;
   wait;
end process;
```

Mauvaise syntaxe !



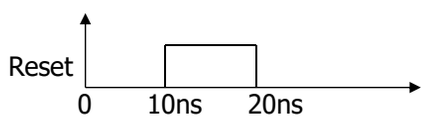



# Génération d'une impulsion (2)

```
Reset <= '0', '1' after 10 ns, '0' after 20 ns;
```

```
process
begin
   reset <= '0';
   wait for 10 ns;
   reset <= '1'
   wait for 10 ns;
   reset <= '0';
   wait;
end process;
```

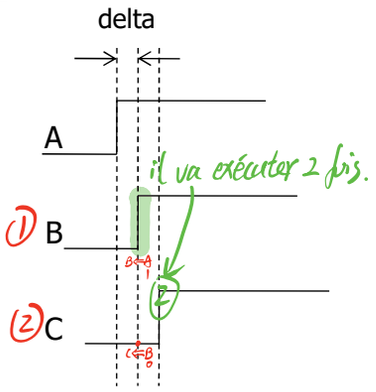
Bonne syntaxe !



δ延时 (temps de propagation dans les portes logiques)

# Delta Délais

```
process (A,B)
begin
   B <= A;①
   C <= B;②
end process;
```




- En jargon VHDL, un délai delta = délai infinitésimal non nul
- Un signal prend sa nouvelle valeur après un délai delta.
- Une variable prend sa nouvelle valeur immédiatement.

信号在δ延时后更新值. (即结束进程后的δ延时)
变量会立即更新值.



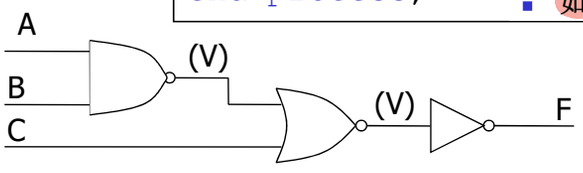
# Les Variables

变量

- **Les variables ne peuvent être déclarées et n'existent que dans un process.** 变量只能在进程中使用.
- **L'affectation d'une variable est immédiate : la valeur affectée à V à la première ligne peut directement être réutilisée à la deuxième.** 变量的赋值是立即的. 第1行对值给V的值 可在第二行直接用.

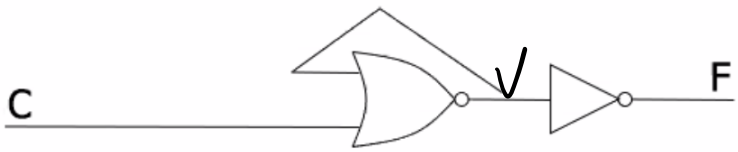
```
process (A, B, C)
-- zone de déclaration d'une variable
   variable V: STD_LOGIC;
begin
   V := A nand B;
   V := V nor C;
   F <= not V;
end process;
```

- 变量只能声明，并且只存在于进程中。
- 变量的赋值是立即的：在第一行赋值给v的值，可以在第二行直接重用。
- 如果V表示为信号，第一行赋值不起作用。执行第二行赋值。




Redessinez le schéma si on considère V comme un signal ?

```
signal V: STD_LOGIC;
process (A, B, C)
-- zone de déclaration d'une variable
   begin
   V <= A nand B;
   V <= V nor C;
   F <= not V;
end process;
```




Redessinez le schéma si on considère V comme un signal ?

# Exemple : parité impaire

检验偶数。 初始P为1；
来一个1，设为0；
来第二个1，设为1；

```
Entity parite is
PORT ( a   : IN std_logic_vector(0 TO 3) ;
       s   : OUT std_logic );
END enttity;
architecture behaviour of parite is
begin
   process(a)
      variable parite : std_logic ;
   begin
      parite := '1' ;
      FOR i in 0 to 3 LOOP
         if a(i) = '1' then
            parite := not parite;
         end if;
      END LOOP;
      s <= parite;
   end process;
END architecture;
```



# Exercice

```
process(A, S)
   variable V: STD_LOGIC;
begin
   V := A;
   S <= V;
   V := S;
   T <= V;
end process;
```

变量立即赋值。

当在同一进程中，同一信号赋值目标有多个赋值源时，信号赋值目标获得的是最后一个赋值 源的赋值，其前面相同的赋值 目标不作任何变化。

本应Process结束赋值。
最终V=S，S再次被赋给自己。

~~立即赋值，使得之前V：=A无用。~~

process结束时，T被赋为V。

- Supposer que S vaut '0', et A change de l'état '0' à l'état '1'.

# Questions

1. Qu'elle est la valeur de S à la fin du process, avant le delta délai ? 0
2. Qu'elle est la valeur de V à la fin du process, avant le delta délai ? 0
3. Après l'exécution du process et après le delta délai, S et T prennent leurs nouvelles valeurs, que valent S et T ? 1 0
4. Après la première exécution du process et après le delta délai, que se passe-t-il ?