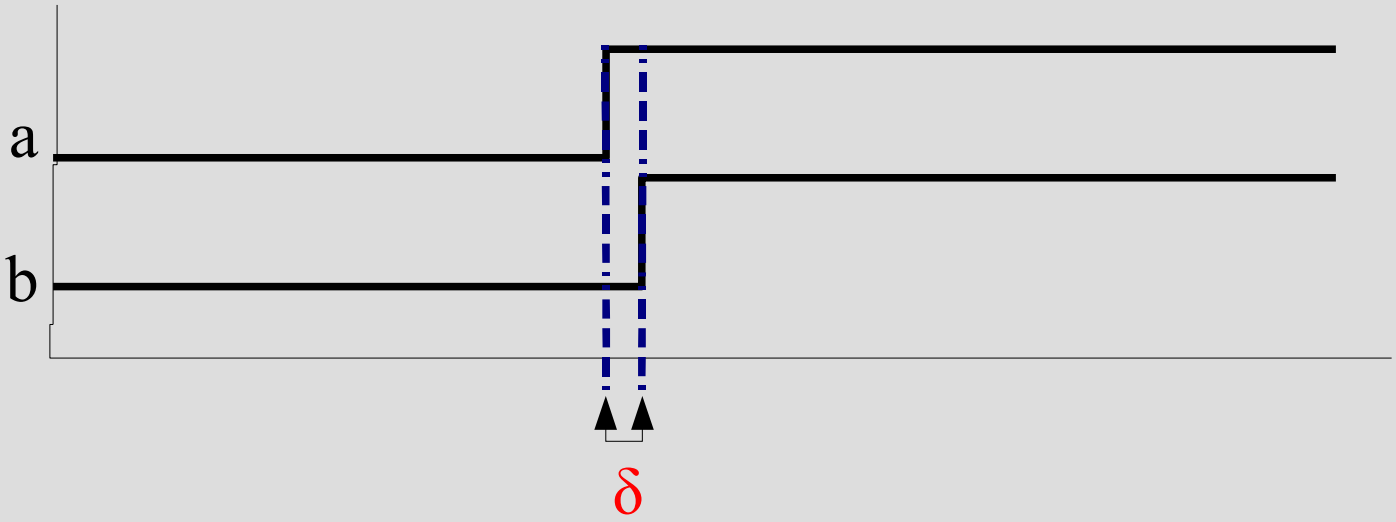
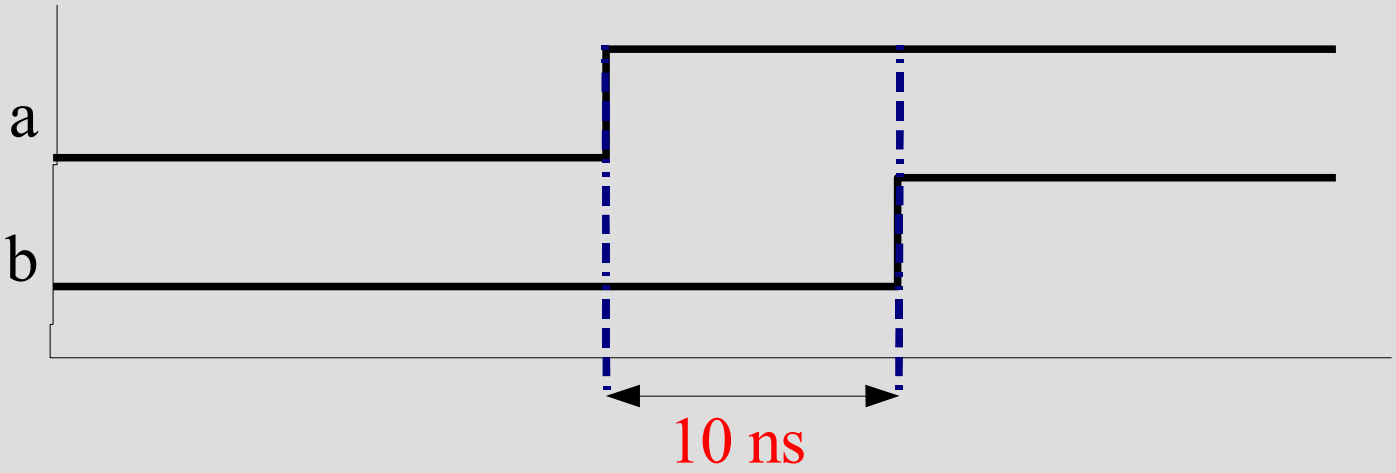
# Behavioral Modeling

***a<=b;***

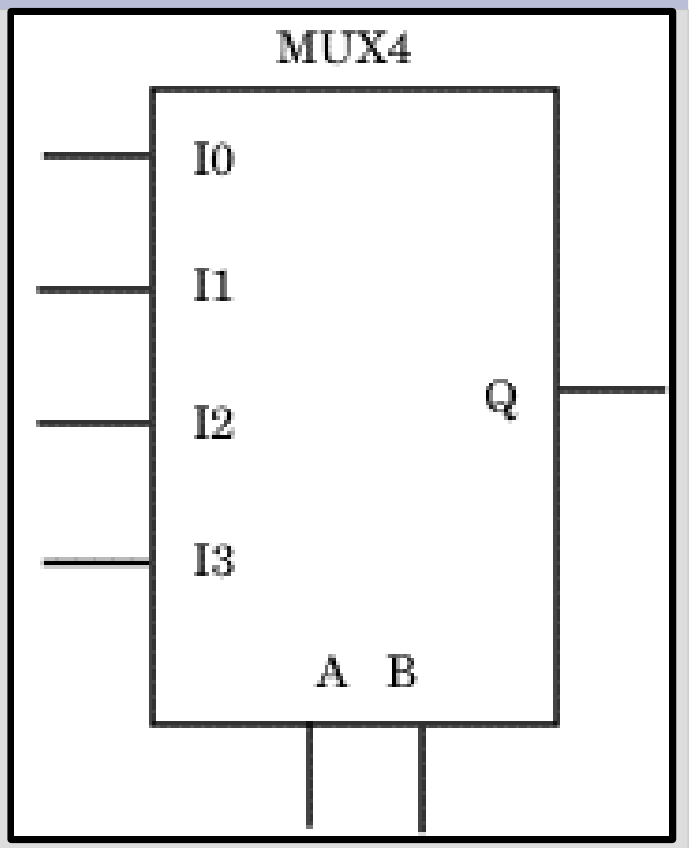



***a <= b AFTER 10 ns;***

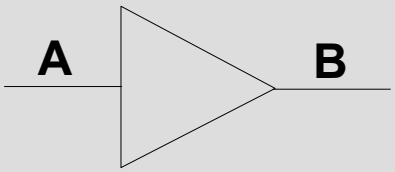


# Behavioral Modeling

```
ENTITY mux4 IS
PORT ( i0, i1, i2, i3, a, b : IN std_logic;
                             q : OUT std_logic);
END mux4;

ARCHITECTURE mux4 OF mux4 IS
SIGNAL sel: INTEGER;
BEGIN
WITH sel SELECT
 q <= i0 AFTER 10 ns WHEN 0,
      i1 AFTER 10 ns WHEN 1,
       i2 AFTER 10 ns WHEN 2,
       i3 AFTER 10 ns WHEN 3,
       'X' AFTER 10 ns WHEN OTHERS;

 sel <= 0 WHEN a = '0' AND b = '0' ELSE
        1 WHEN a = '1' AND b = '0' ELSE
        2 WHEN a = '0' AND b = '1' ELSE
        3 WHEN a = '1' AND b = '1' ELSE
        4 ;
 END mux4;
```



# Inertial vs Transport Delays



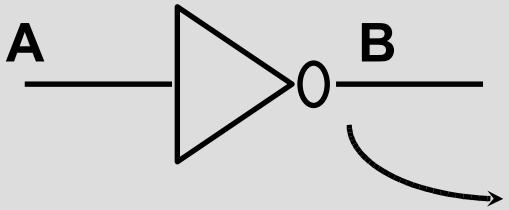

**Inertial Delay**

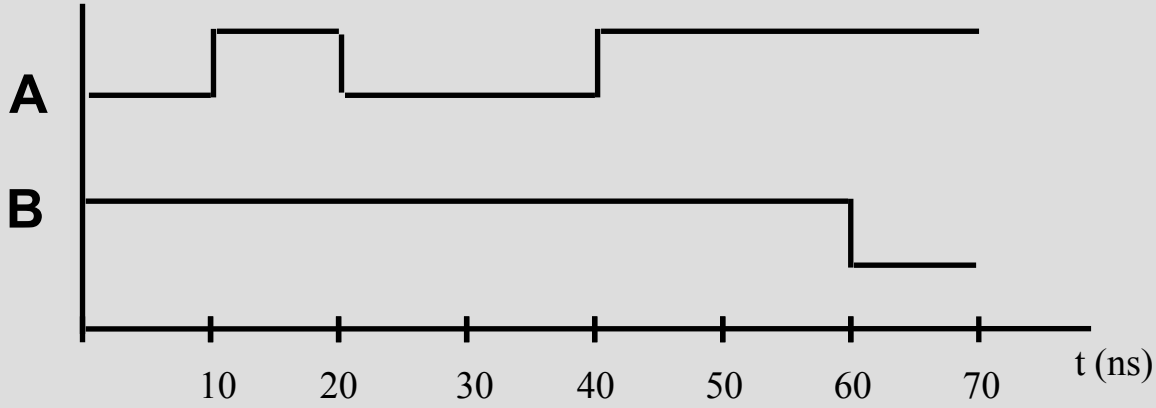
```
ENTITY my_buffer IS
    PORT( A: IN BIT; B : OUT BIT);
END my_buffer;

ARCHITECTURE behavior OF my_buffer IS
BEGIN
  B <= A AFTER 20 ns;
END behavior;
```

**Transport Delay**

```
ENTITY my_buffer IS
    PORT( A: IN BIT; B : OUT BIT);
END my_buffer;

ARCHITECTURE behavior OF my_buffer IS
BEGIN
  B <= TRANSPORT A AFTER 20 ns;
END behavior;
```

# Inertial Delay

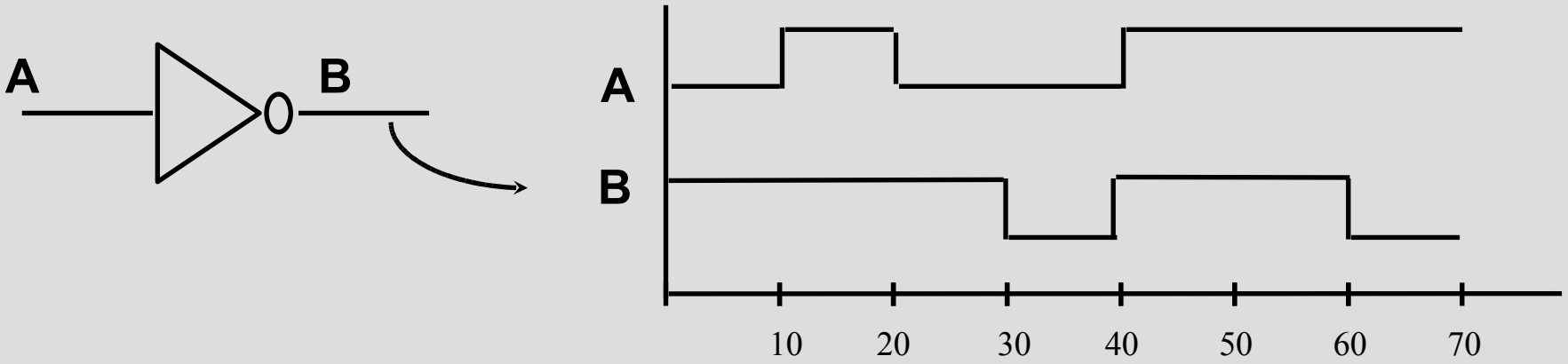



```
ENTITY my_not IS
    PORT( A: IN BIT; B : OUT BIT);
END my_not;

ARCHITECTURE behavior OF my_not IS
BEGIN
  B <= not(A) AFTER 20 ns;
END behavior;
```

E' il comportamento di default del simulatore VHDL in quanto più simile a quello dei dispositivi reali

# Transport Delay

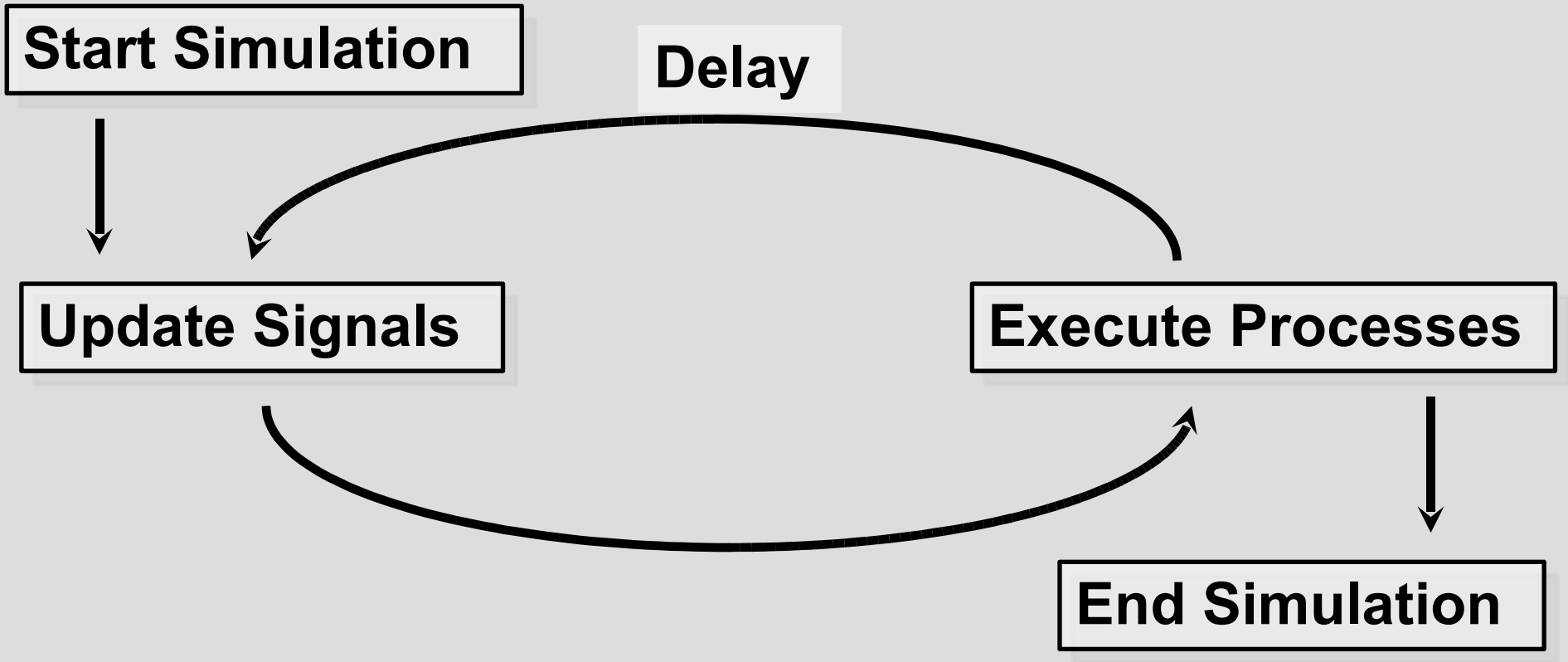



```
ENTITY my_not IS
    PORT( A: IN BIT; B : OUT BIT);
END my_not;

ARCHITECTURE behavior OF my_not IS
BEGIN
  B <= transport not(A) AFTER 20 ns;
END behavior;
```

Consente di simulare wires e delay lines. Ogni variazione del segnale di ingresso viene trasportata in uscita

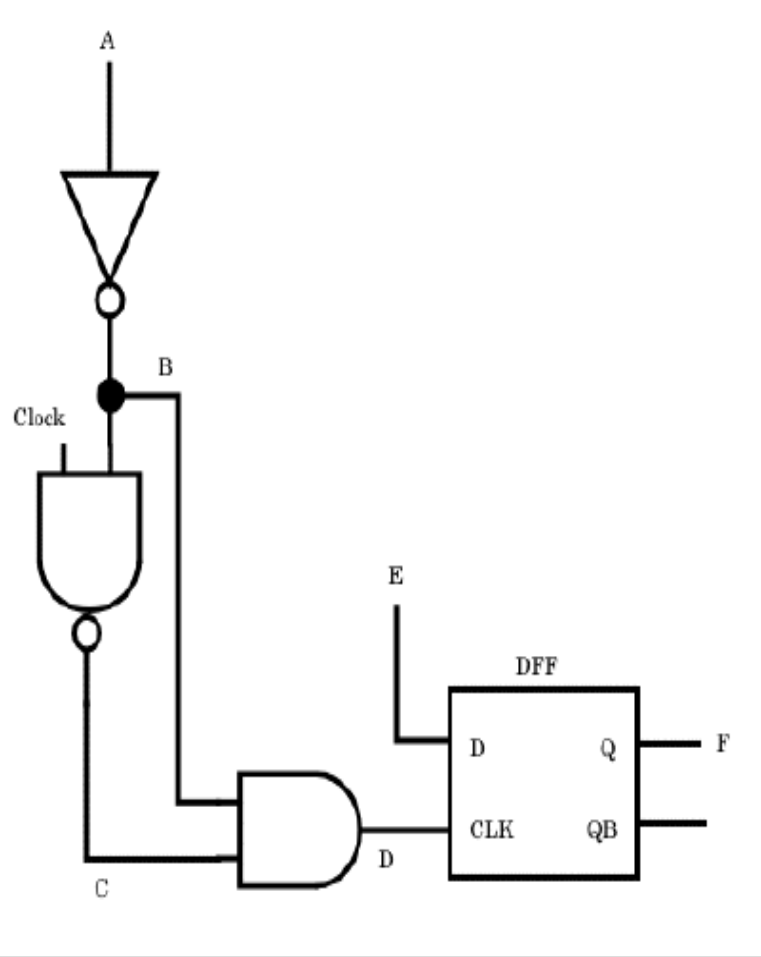
# Timing Model

Ciclo di simulazione degli stimoli e delle risposte di un circuito digitale descritto in VHDL



# Delta Delay

Primo problema:
come simulo questo?




**Se valuto prima il NAND:**

**A: 1->0**

**B: 0->1**

**C: 1->0**

**D: 0->0**

**Se valuto prima l'AND:**

**A: 1->0**

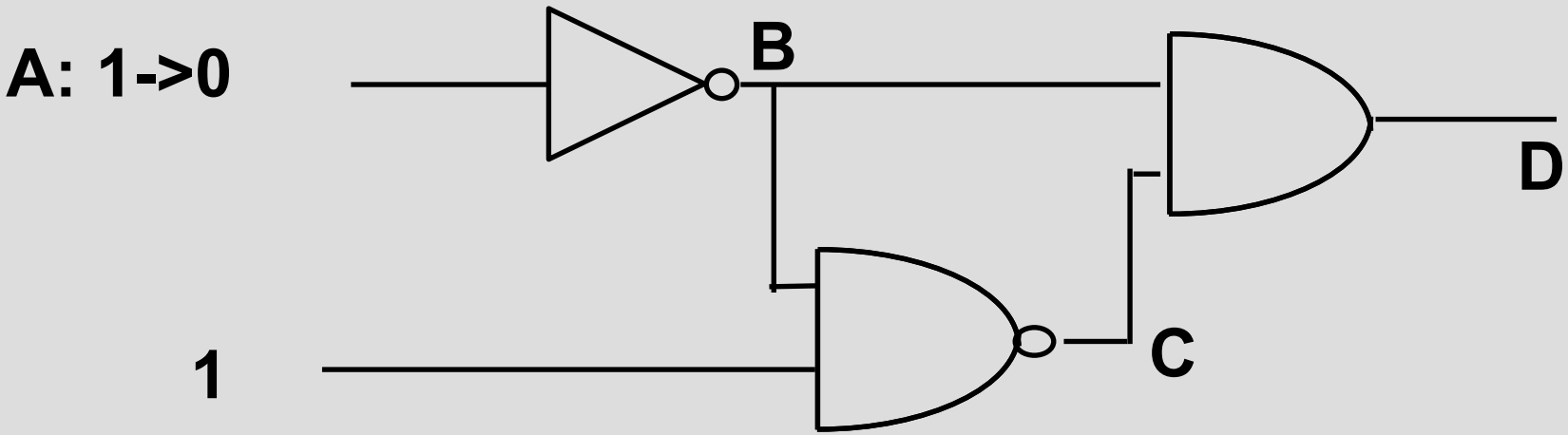
**B: 0->1**

**D: 0->1**

**C: 1->0**

**D: 1->0**

# La soluzione con Delta Delay




## Using delta delay scheduling

| Time | Delta | Event |
|---|---|---|
| 0 ns | 1 | A: 1->0<br>eval INVERTER |
| | 2 | B: 0->1<br>eval NAND, AND |
| | 3 | C: 1->0<br>D: 0->1<br>eval AND |
| | 4 | D: 1->0 |
| 1 ns | | |

# Drivers

- I *drivers* vengono creati attraverso assegnazione di segnale.
- *Driver Multipli*: molto utili per gestire data-bus, bus bi-direzionali, vengono “risolti” in un unico valore dalle cosiddette *resolution functions*

## Errori nella gestione dei drivers

```
ENTITY mio_mux IS
    PORT( i0,i1,i2,i3,a,b:IN std_logic;
          q : OUT std_logic);
END mio_mux;
```

```
ARCHITECTURE sbagliata OF mio_mux IS
BEGIN
  q <= i0 when a='0' AND b='0' else '0';
  q <= i1 when a='1' AND b='0' else '0';
  q <= i2 when a='0' AND b='1' else '0';
  q <= i3 when a='1' AND b='1' else '0';
END sbagliata;
```

**Sono stati creati 4 drivers, ma ci sono ambiguità:una eventuale resolution function, come potrebbe gestire questa situazione?**

```
ARCHITECTURE giusta OF mio_mux IS
BEGIN
  q <= i0 when a='0' AND b='0'else
       i1 when a='1' AND b='0' else
       i2 when a='0' AND b='1' else
       i3 when a='1' AND b='1' else
    'X';
END giusta;
```

**In questo modo abbiamo un unico driver !**

# Generics

- Parametri utilizzati per passare informazioni nell'"*istance*" di una entity (ritardi, lunghezze di parole ecc.)
- Posso essere "passati" come *generics* tutti i *tipi* consentiti dal VHDL

```
ENTITY my_reg IS
GENERIC(data_width:INTEGER); -- numero di bit di in/out
PORT(
   clock,reset: IN std_logic;
   din:  IN  std_logic_vector(data_width-1 downto 0);
  dout: OUT std_logic_vector(data_width-1 downto 0));
END my_reg;

ARCHITECTURE behavior OF my_buffer IS
   BEGIN
   PROCESS(clock,reset)
      BEGIN
            IF reset='1' then
                  dout<=(OTHERS=>'0');
            ELSIF clock='1' and clock'event then
                  dout<=din;
            END IF;
   END PROCESS;
END behavior;
```