VHDL

Concurrent & Sequential Statements (cond...)

- What would happen if there is 'no 'else' part in your Sequential 'if-then-else' part ? in your Code ?

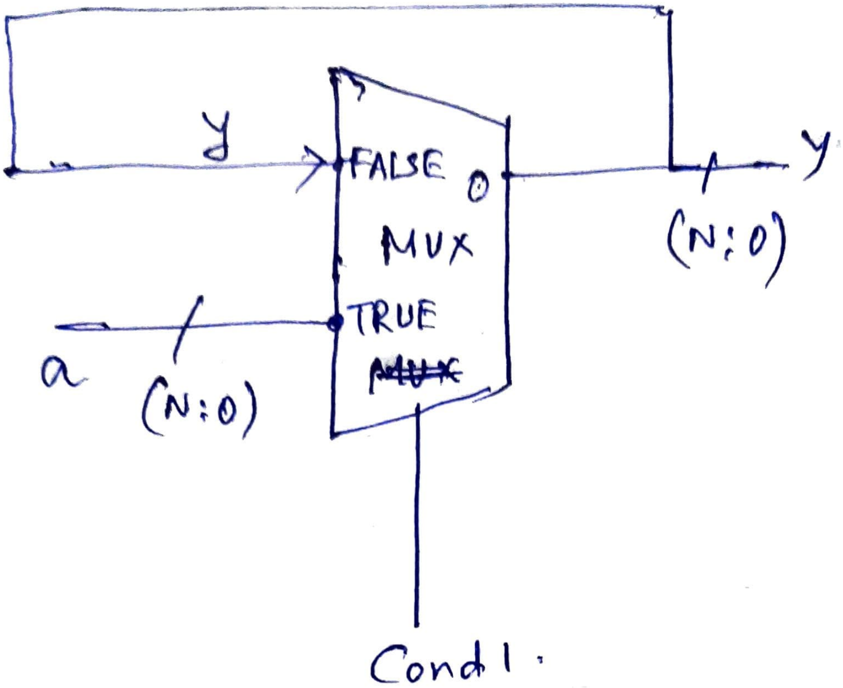
```
if Cond1 then
    y <= a;
end if
```

⇓

< Implied' memory / Inferred Latch >

⇒

```
if Cond1 then
    y <= a;
elsif
    y <= y;
end if;
```



This is also true for. Concurrent statements.

⇒ with 'enable select

y <= a when '1' ;

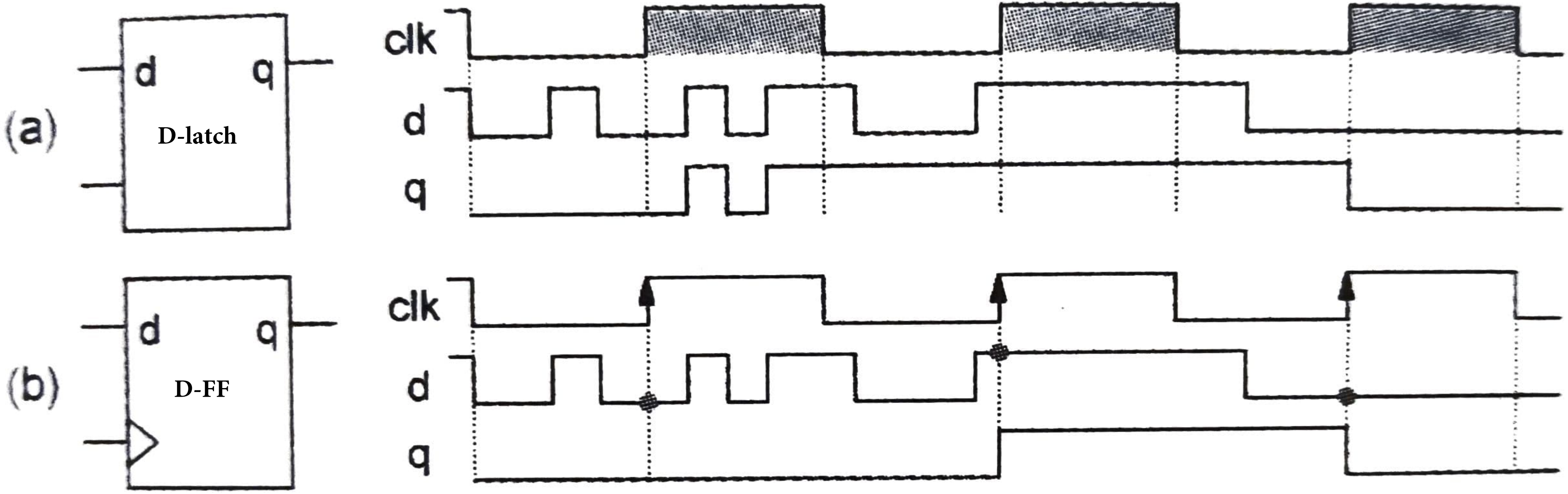
⇒ ~~with~~ y <= a when enable = '1' ;

Both will infer a latch.

Sequential circuit

- Two main fundamental building blocks
  - Flip-flops ⇒ SR-FF , D-FF, T-FF , JK-FF
  - Latch ⇒ SR-Latch, D-Latch .

- What is the fundamental difference between Latch & FF ?


(a)
d
q
D-latch
clk
d
q
(b)
d
q
D-FF
clk
d
q

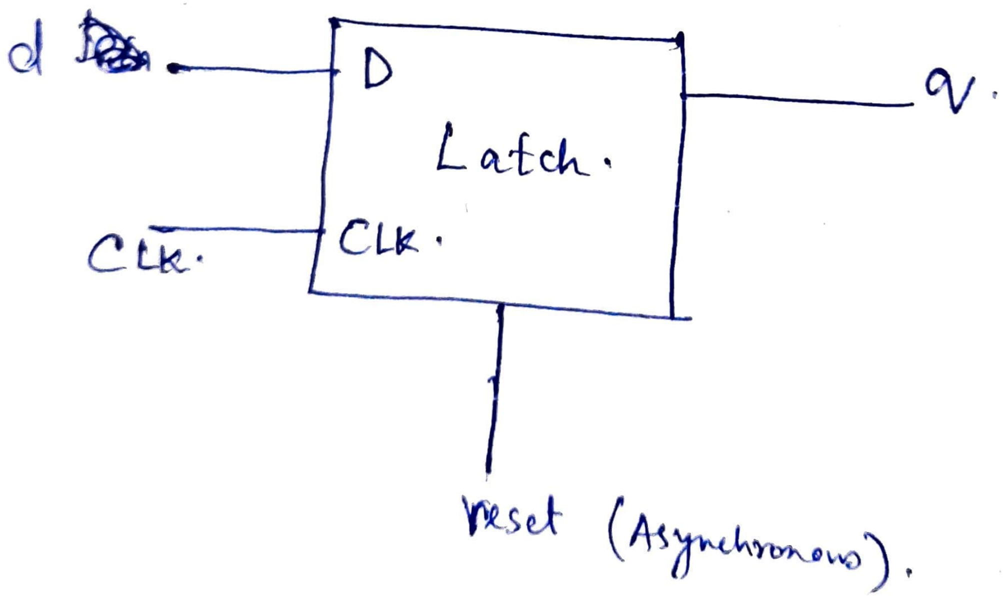
* Implementing Sequential circuits with VHDL Code.

=> The use of Concurrent code is in general ~~not recomm~~ not recommended.

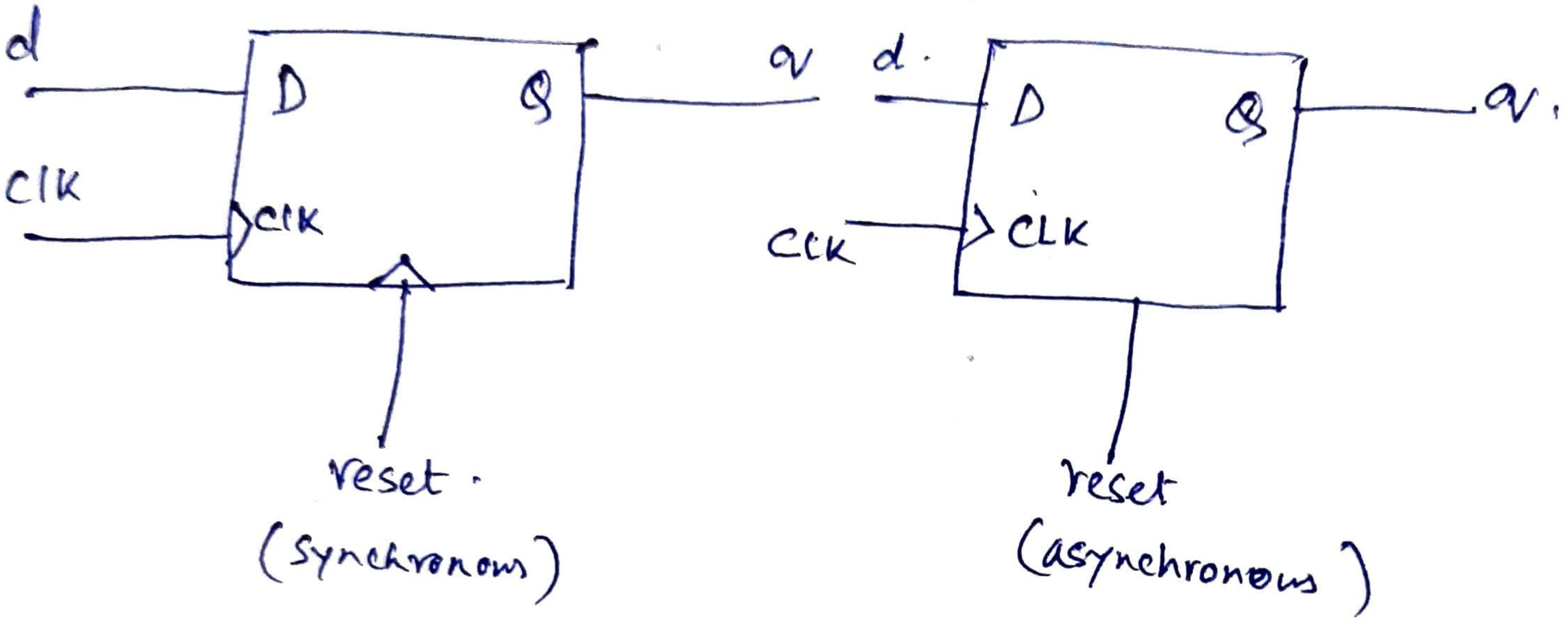
=> Use sequential Code

- Process with a Sensitivity list.

* How to design a D-Latch ?



* How to design a D-FF ?



— Need to use process

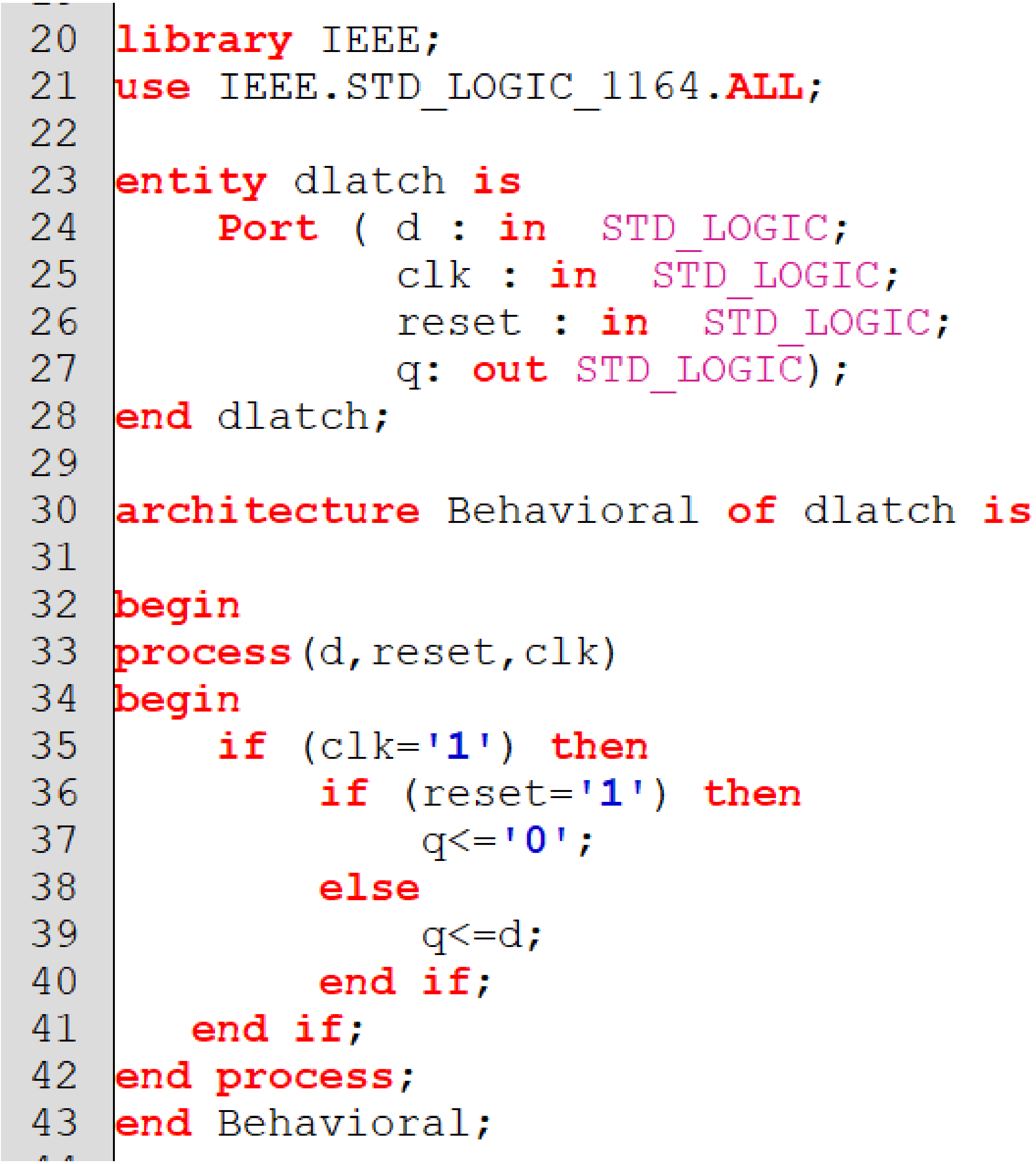
— Use 'EVENT' attribute

or

use 'rising-edge' or 'falling-edge' functions. defined in ieee.std-logic-1164 Package.

VHDL Codes

D-Latch with synchronous reset



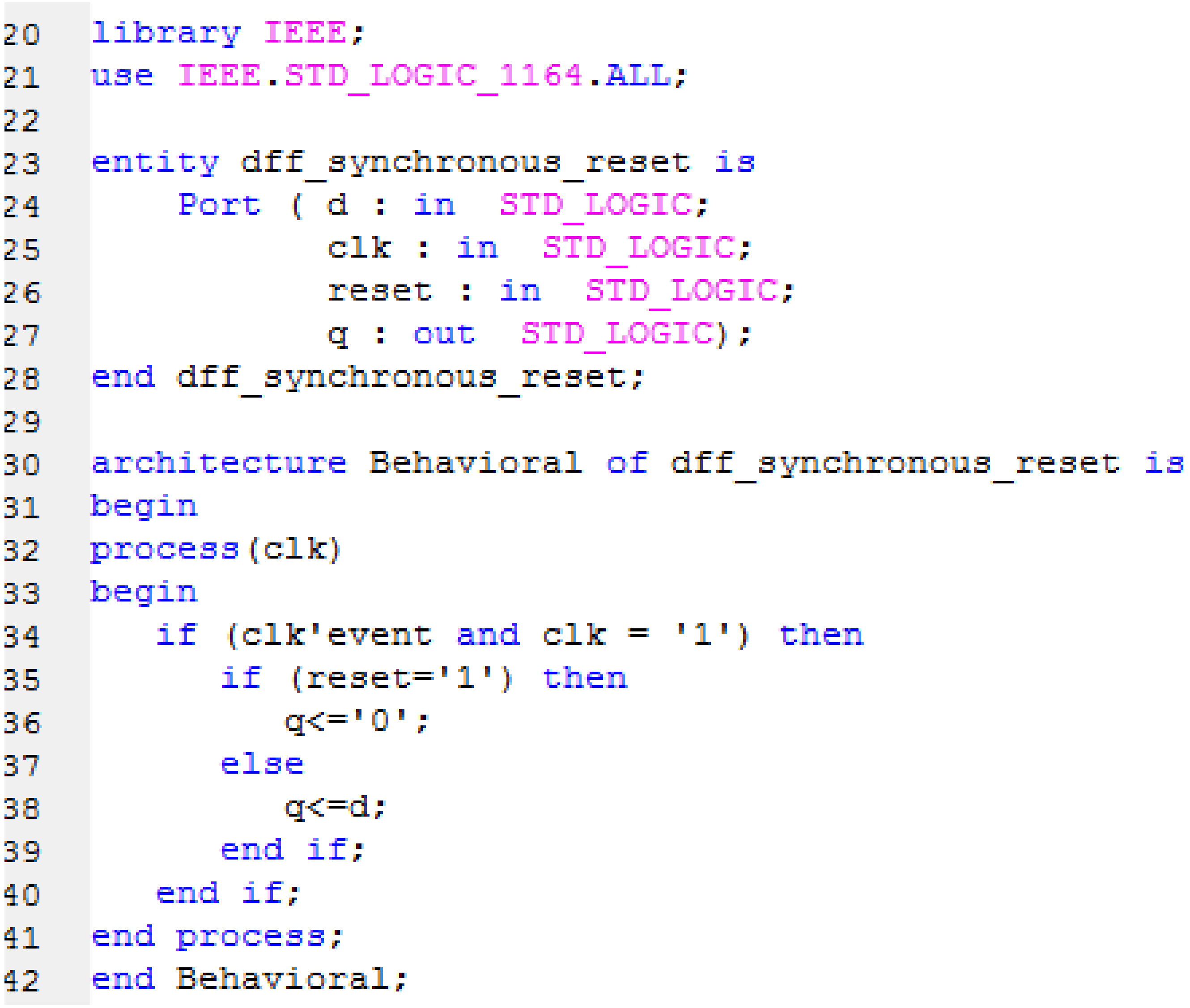
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity dlatch is
    Port ( d : in  STD_LOGIC;
           clk : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           q: out STD_LOGIC);
end dlatch;

architecture Behavioral of dlatch is

begin
process(d,reset,clk)
begin
    if (clk='1') then
        if (reset='1') then
            q<='0';
        else
            q<=d;
        end if;
   end if;
end process;
end Behavioral;
```

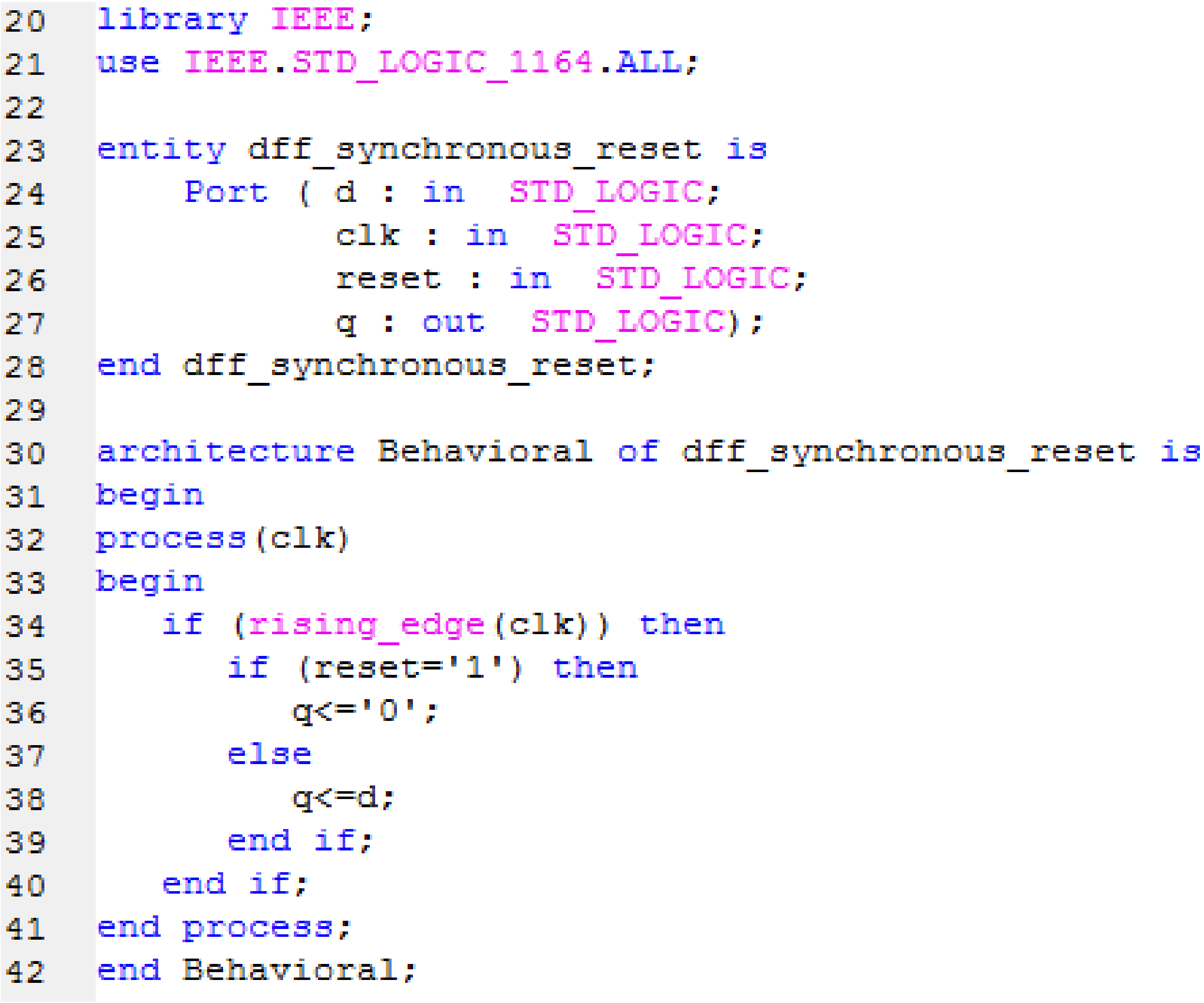
D-FF with synchronous reset



```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity dff_synchronous_reset is
    Port ( d : in  STD_LOGIC;
           clk : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           q : out  STD_LOGIC);
end dff_synchronous_reset;

architecture Behavioral of dff_synchronous_reset is
begin
process(clk)
begin
   if (clk'event and clk = '1') then
      if (reset='1') then
         q<='0';
      else
         q<=d;
      end if;
   end if;
end process;
end Behavioral;
```

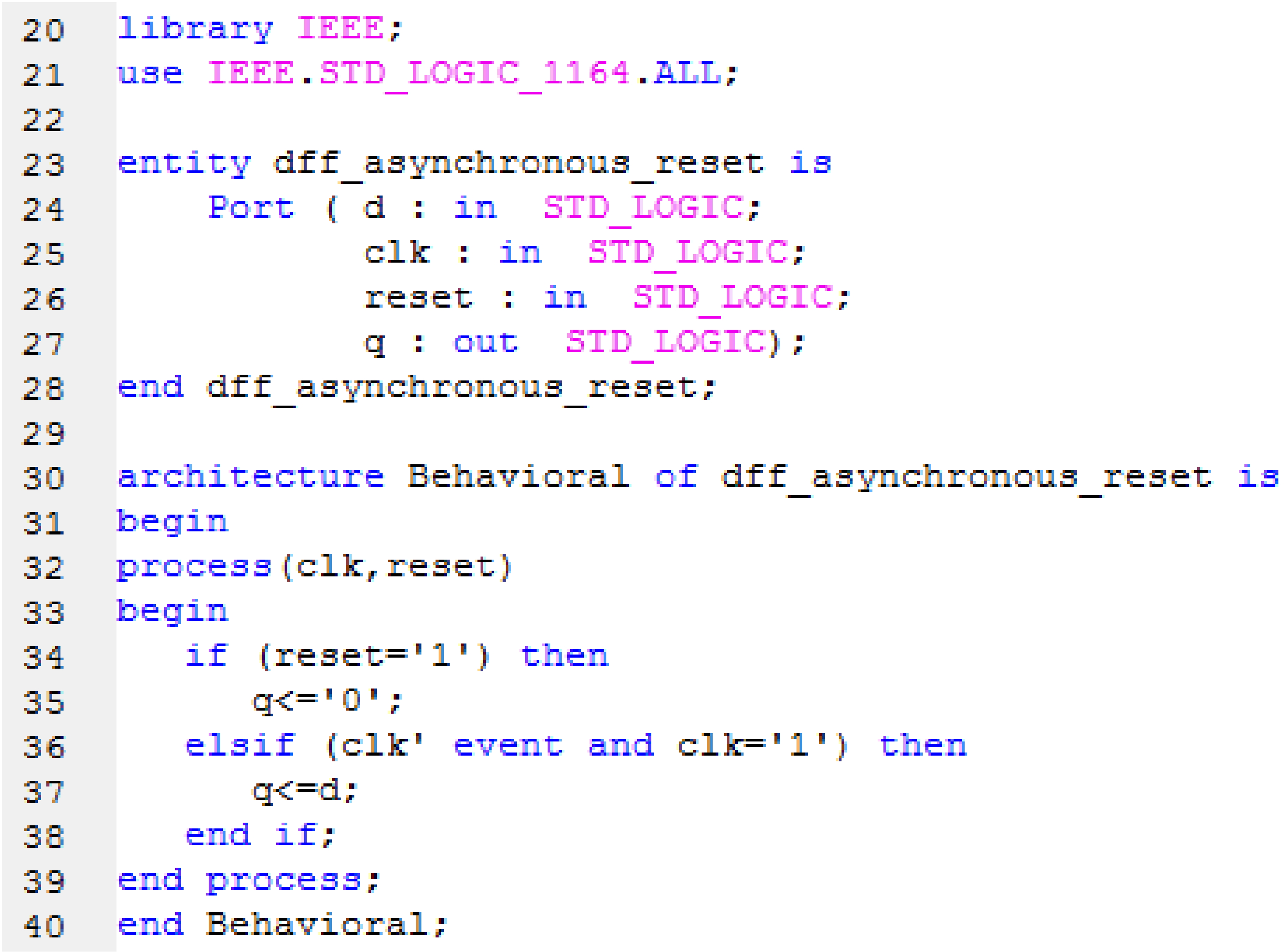


```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity dff_synchronous_reset is
    Port ( d : in  STD_LOGIC;
           clk : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           q : out  STD_LOGIC);
end dff_synchronous_reset;

architecture Behavioral of dff_synchronous_reset is
begin
process(clk)
begin
   if (rising_edge(clk)) then
      if (reset='1') then
         q<='0';
      else
         q<=d;
      end if;
   end if;
end process;
end Behavioral;
```

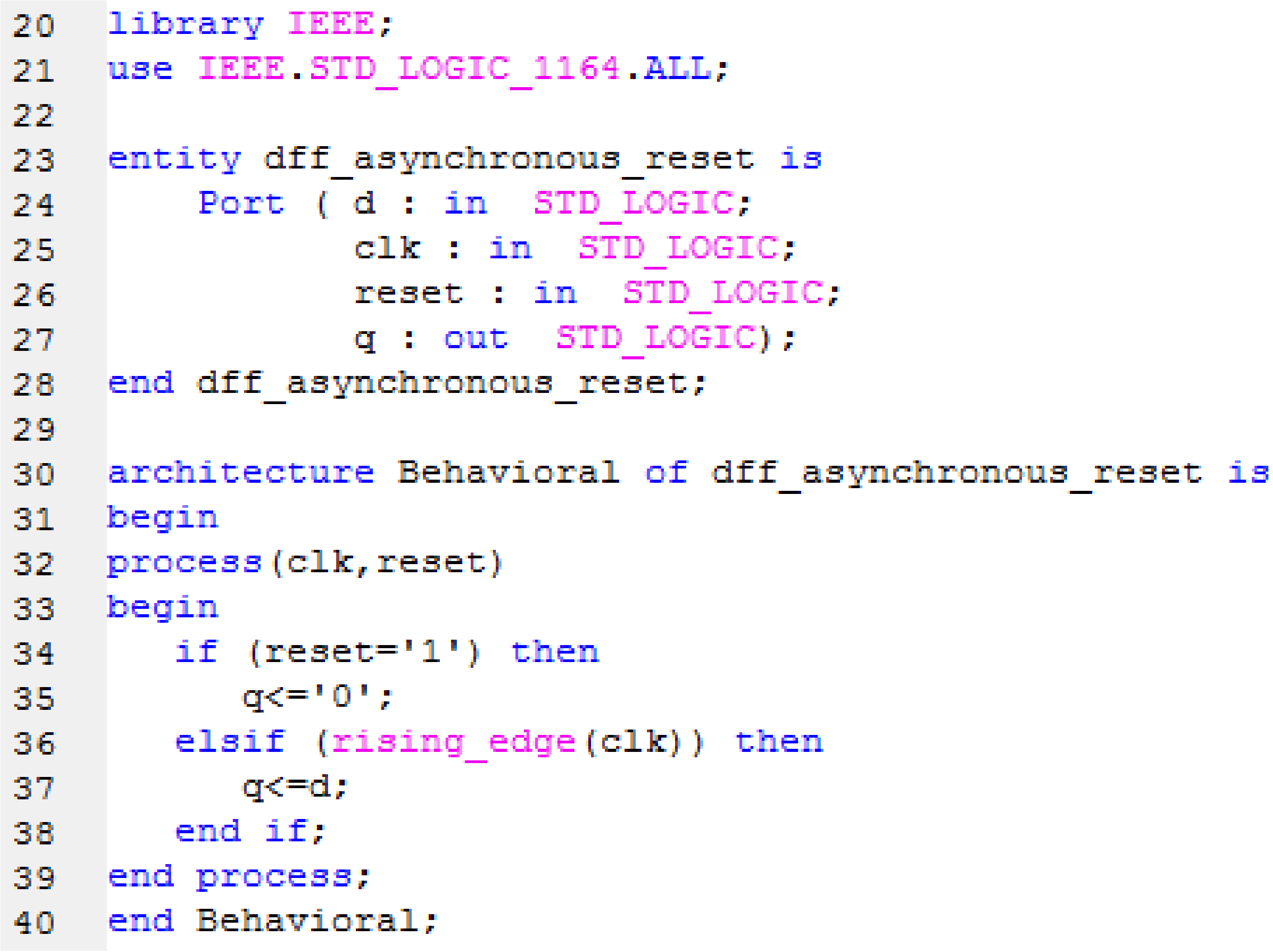
D-FF with asynchronous reset



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity dff_asynchronous_reset is
    Port ( d : in  STD_LOGIC;
           clk : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           q : out  STD_LOGIC);
end dff_asynchronous_reset;

architecture Behavioral of dff_asynchronous_reset is
begin
process(clk,reset)
begin
   if (reset='1') then
      q<='0';
   elsif (clk' event and clk='1') then
      q<=d;
   end if;
end process;
end Behavioral;
```



```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity dff_asynchronous_reset is
    Port ( d : in  STD_LOGIC;
           clk : in  STD_LOGIC;
           reset : in  STD_LOGIC;
           q : out  STD_LOGIC);
end dff_asynchronous_reset;

architecture Behavioral of dff_asynchronous_reset is
begin
process(clk,reset)
begin
   if (reset='1') then
      q<='0';
   elsif (rising_edge(clk)) then
      q<=d;
   end if;
end process;
end Behavioral;
```