SINKRONI T BISTABIL

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

ENTITY sintff IS PORT (
clk, t, clr, st: IN std_logic;
Q, Qn: OUT std_logic);
END sintff;

ARCHITECTURE arch OF sintff IS
BEGIN
PROCESS (clk)
VARIABLE mem: std_logic:='0';
BEGIN
IF falling_edge(clk) THEN
IF clr='1' THEN mem:='0';
ELSIF st='1' THEN mem:='1';
ELSIF t='1' THEN mem:=not mem;
END IF;
END IF;
Q<=mem after 10 ns;
Qn<=not mem after 10 ns;
END PROCESS;
END arch;
```

ASINKRONI T BISTABIL

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

ENTITY asintff IS PORT(
clk, t, clr, st: IN std_logic;
Q, Qn: OUT std_logic);
END asintff;

ARCHITECTURE arch OF asintff IS
BEGIN
PROCESS (clr, st, clk)
VARIABLE mem: std_logic:='0';
BEGIN
IF clr='1' THEN mem:='0';
ELSIF st='1' THEN mem:=not mem;
ELSIF falling_edge(clk) THEN
IF t='1' THEN mem:=not mem;
END IF;
END IF;
Q<=mem after 10 ns;
Qn<=not mem after 10 ns;
END PROCESS;
END arch;
```

SEKVENCIJSKI SKLOP

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

ENTITY sekvencijski_sklop IS PORT (
clk, reset, p: IN std_logic;
Q: OUT std_logic_vector (4 downto 0));
END sekvencijski_sklop;

ARCHITECTURE arch OF sekvencijski_sklop IS
SIGNAL i1, i2, i3, i4, i5: std_logic;
SIGNAL Qp: std_logic_vector (4 downto 0);
BEGIN
i1<=reset AND p;
i2<=reset AND not p;
i3<=Qp(0) AND Qp(1);
i4<=i3 AND Qp(2);
i5<=i4 AND Qp(3);
B0: ENTITY work.sintff PORT MAP (clk, '1', i2, i1, Qp(0), open);
B1: ENTITY work.sintff PORT MAP (clk, Qp(0), reset, '0', Qp(1), open);
B2: ENTITY work.sintff PORT MAP (clk, i3, reset, '0', Qp(2), open);
B3: ENTITY work.sintff PORT MAP (clk, i4, reset, '0', Qp(3), open);
B4: ENTITY work.sintff PORT MAP (clk, i5, reset, '0', Qp(4), open);
Q<=Qp;
END arch;
```