**Lab Task 1:**

**LED blinking**

**VHDL Code:**

library ieee**;**

**use** ieee**.**std\_logic\_1164**.all;**

**use** ieee**.**std\_logic\_unsigned**.all;**

entity ledBlinking **is**

port**(**

fpga\_clk **:** **in** std\_logic**;**

switch **:** **in** std\_logic\_vector**(**2 downto 0**);**

led **:** **out** std\_logic\_vector**(**3 downto 0**)**

**);**

**end** ledBlinking**;**

architecture bhv **of** ledBlinking **is**

--signals

signal **count** **:** std\_logic\_vector**(**31 downto 0**)** **:=** X"00000000"**;**

signal top\_clk **:** std\_logic**;**

signal ledout **:** std\_logic\_vector**(**3 downto 0**);**

**begin**

---

process **(**fpga\_clk**)**

**begin**

**if** **(**fpga\_clk 'event and fpga\_clk = '1') then

count <= count + 1;

end if;

end process;

---

process (fpga\_clk , count)

begin

if (fpga\_clk 'event **and** fpga\_clk **=** '1'**)** **then**

**case** switch **is**

**when** "000" **=>** top\_clk **<=** **count(**20**);**

**when** "100" **=>** top\_clk **<=** **count(**31**);**

**when** "010" **=>** top\_clk **<=** **count(**15**);**

**when** "001" **=>** top\_clk **<=** **count(**2**);**

**when** **others** **=>** top\_clk **<=** **count(**2**);**

**end** **case;**

**end** **if;**

**end** process**;**

---

process**(**top\_clk**)**

**begin**

**if** **(**top\_clk 'event' **and** top\_clk **=** '1'**)** **then**

ledout **<=** **not** ledout**;**

**end** **if;**

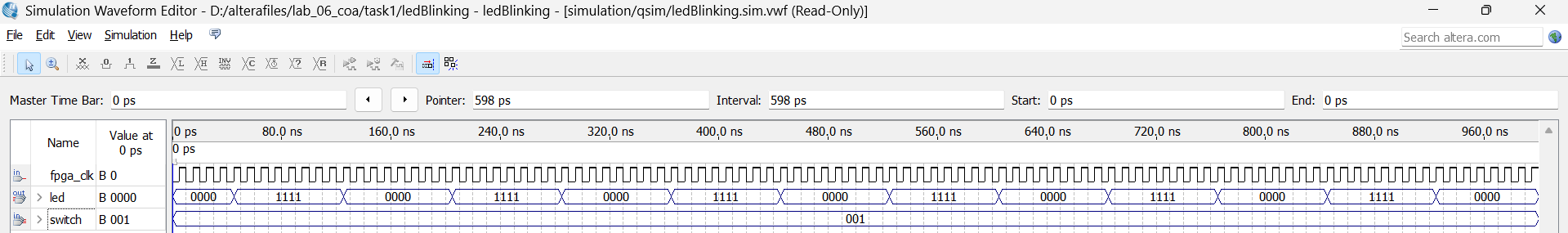
**end** process**;**

---

led **<=** ledout**;**

**end** bhv**;**

**Simulation:**



Simulation 1: Compute quotient and remainder after enabling positive edge trigger. The output is generated in 3 cycles.

**Lab Task 2:**

**Implement a 4-bit sequence detector(1011)**

**VHDL Code:**

**library** ieee**;**

**use** ieee**.**std\_logic\_1164**.all;**

**use** ieee**.**std\_logic\_unsigned**.all;**

**entity** FSM\_sequence\_detactor **is**

**port** **(**

clk **:** **in** std\_logic**;**

letstart **:** **in** std\_logic**;**

y **:** **out** std\_logic

**);**

**end** FSM\_sequence\_detactor**;**

**architecture** bhv **of** FSM\_sequence\_detactor **is**

--define state using one hot coding

**constant** state0 **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "00001"**;**

**constant** state1 **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "00010"**;**

**constant** state2 **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "00100"**;**

**constant** state3 **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "01000"**;**

**constant** state4 **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** "10000"**;**

-- State Memory

**signal** mystate **:** std\_logic\_vector**(**4 **downto** 0**)** **:=** state0**;** -- 00001

**begin**

**process** **(**clk**)**

**begin**

**if** **(**clk'**event** **and** clk **=** '1'**)** **then** -- positive edge sensitive

**case** mystate **is**

------

**when** state0 **=>**

y **<=** '0'**;**

**if** **(**letstart **=** '1'**)** **then**

mystate **<=** state1**;**

**else** -- 0

mystate **<=** state0**;**

**end** **if;**

------

**when** state1 **=>**

y **<=** '0'**;**

**if** **(**letstart **=** '0'**)** **then**

mystate **<=** state2**;**

**else** -- 1

mystate **<=** state1**;**

**end** **if;**

------

**when** state2 **=>**

y **<=** '0'**;**

**if** **(**letstart **=** '1'**)** **then**

mystate **<=** state3**;**

**else** --0

mystate **<=** state0**;**

**end** **if;**

------

**when** state3 **=>**

y **<=** '1'**;**

**if** **(**letstart **=** '1'**)** **then**

mystate **<=** state4**;**

**else** -- 0

mystate **<=** state2**;**

**end** **if;**

------

**when** state4 **=>**

y **<=** '1'**;**

**if** **(**letstart **=** '1'**)** **then**

mystate **<=** state0**;**

**else** -- 0

mystate **<=** state0**;**

**end** **if;**

------

**when** **others** **=>**

mystate **<=** state0**;**

------

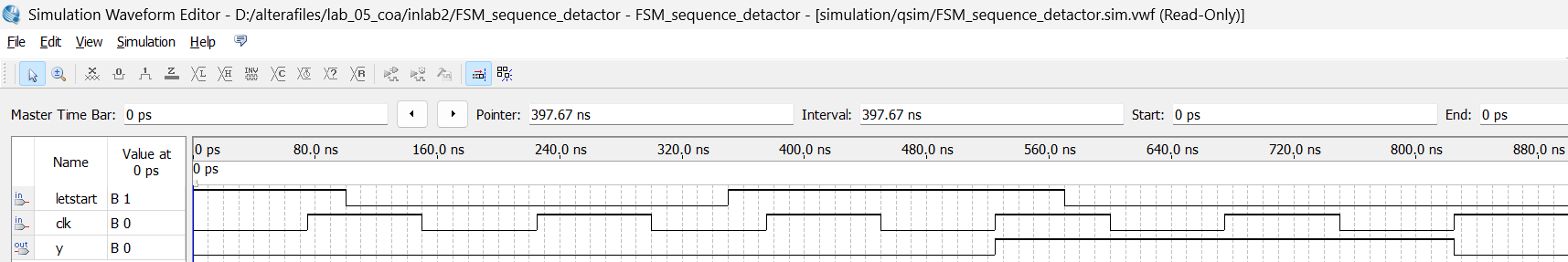
**end** **case;**

**end** **if;**

**end** **process;**

**end** bhv**;**

**Simulation:**



Simulation 2: detects the input(let start) pattern of 1011