

1.5 second breathing light:

The image shows the Quartus II IDE interface with the following components:

- Project Navigator:** Shows the project hierarchy for 'Cyclone III: EP3C16F484C6' and 'bth5s'.
- Tasks:** A table showing the compilation tasks and their durations.

Task	Time
Compile Design	00:00:12
Analysis & Synthesis	00:00:02
Fitter (Place & Route)	00:00:06
Assembler (Generate programming files)	00:00:02
TimeQuest Timing Analysis	00:00:02
EDA Netlist Writer	
Program Device (Open Programmer)	
- Messages:** A list of timing warnings and messages.
 - 332123 Deriving Clock Uncertainty. Please refer to rep...
 - 332148 Timing requirements not met
 - 332146 Worst-case setup slack is -1.452
 - 332146 Worst-case hold slack is -0.221
 - 332140 No Recovery paths to report
 - 332140 No Removal paths to report
 - 332146 Worst-case minimum pulse width slack is -3.000
 - Analyzing Fast 1200mV OC Model
 - 332123 Deriving Clock Uncertainty. Please refer to rep...
 - 332148 Timing requirements not met
 - 332146 Worst-case setup slack is -0.533
 - 332146 Worst-case hold slack is -0.188
 - 332140 No Recovery paths to report
 - 332140 No Removal paths to report
 - 332146 Worst-case minimum pulse width slack is -3.000
 - 332102 Design is not fully constrained for setup requi...
 - 332102 Design is not fully constrained for hold requir...
 - Quartus II 64-Bit TimeQuest Timing Analyzer was
 - 293000 Quartus II Full Compilation was successful. 0 e...
- Source Code:** The VHDL code for 'bth5s.vhd' is displayed, showing the entity definition, port declarations, and the architecture logic.


```

1 library ieee;
2 use ieee.std_logic_1164.all;
3 use ieee.std_logic_unsigned.all;
4 use ieee.std_logic_arith.all;
5
6 entity bth5s is
7 port(
8   clk: in std_logic;
9   reset: in std_logic;
10  led: out std_logic_vector(9 downto 0));
11 end bth5s;
12
13 architecture bth5s of bth5s is
14   signal cnt: std_logic_vector(7 downto 0);
15   signal divcnt: std_logic_vector(5 downto 0);
16   signal i, msl0: std_logic;
17   signal r: std_logic_vector(7 downto 0);
18   signal c10s: std_logic_vector(13 downto 0);
19   signal k: std_logic;
20 begin
21   process(clk)
22   begin
23     if rising_edge(clk) then
24       if divcnt = 49 then
25         divcnt <= "000000";
26       else
27         divcnt <= divcnt + 1;
28       end if;
29     end if;
30   end process;
31   i <= divcnt(5);
32
33   process(i)
34   begin
35     if rising_edge(i) then
36       cnt <= cnt + 1;
37     end if;
38   end process;
39
40   process(i)
41   begin
42     if rising_edge(i) then
43       if (c10s = 9999) then
44         c10s <= (others => '0');
45       else
46         c10s <= c10s + 1;
47       end if;
48     end if;
49   end process;
50   msl0 <= c10s(13);
51   process(msl0, reset)
52   begin
53     if reset = '0' then
54       r <= "00000000";
55     elsif rising_edge(msl0) then
56       if r = "11111110" then
57         k <= '1';
58       elsif r = "00000001" then
59         k <= '0';
60       end if;
61       if k = '0' then
62         r <= r + 1;
63       elsif k = '1' then
64         r <= r - 1;
65       end if;
66     end if;
67   end process;
68   LED <= (others=>'1')when cnt < r else
69     (others=>'0');
70 end bth5s;
      
```