DECODER 3 TO 8

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    10:52:26 08/31/2016
-- Design Name:
-- Module Name:    DECODER_FPGA - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
--library IEEE;
--use IEEE.STD_LOGIC_1164.ALL;
--
---- Uncomment the following library declaration if using
```

```
---- arithmetic functions with Signed or Unsigned values
----use IEEE.NUMERIC_STD.ALL;
--
---- Uncomment the following library declaration if instantiating
---- any Xilinx primitives in this code.
----library UNISIM;
----use UNISIM.VComponents.all;
--
--entity DECODER_FPGA is
--    Port ( A0,A1,A2 : in  STD_LOGIC;
--           D0,D1,D2,D3,D4,D5,D6,D7 : out  STD_LOGIC);
--end DECODER_FPGA;
--
--architecture Behavioral of DECODER_FPGA is
--
--begin
--
--
--end Behavioral;
--
library ieee;
use ieee.std_logic_1164.all;


entity andGate is
```

```
  port(        A, B, C : in std_logic;
           F : out std_logic);
end andGate;

architecture func of andGate is
begin
  F <= A and B and C;
end func;
--*============================
                --    This is the NOT gate
library ieee;
use ieee.std_logic_1164.all;

entity notGate is
  port(        inPort : in std_logic;
       outPort : out std_logic);
end notGate;
--
architecture func of notGate is
begin
  outPort <= not inPort;
end func;
--*======================*=====================
```

```
--      Now we write the definition for the 3-to-8 Decoder
library ieee;
use ieee.std_logic_1164.all;
--
entity Decoder_3to8 is
  port( A0, A1, A2 : in std_logic;
       D0, D1, D2, D3, D4, D5, D6, D7 : out std_logic);
end Decoder_3to8;
--
architecture func of Decoder_3to8 is
  component andGate is        --import AND Gate entity
    port( A, B, C : in std_logic;
             F : out std_logic);
  end component;

  component notGate is        --import NOT Gate entity
    port( inPort  : in std_logic;
          outPort : out std_logic);
  end component;

  signal invA0, invA1, invA2  : std_logic;

begin
--notice that there are as many concurrent statements
```

```
--here as there are gates in the physical circuit shown
-- on Teahlab.com.
  GI1: notGate  port map(A0, invA0);
  GI2: notGate  port map(A1, invA1);
  GI3: notGate  port map(A2, invA2);
  --the outputs
  GA1: andGate  port map(invA0, invA1, invA2, D0);
  GA2: andGate  port map(   A0, invA1, invA2, D1);
  GA3: andGate  port map(invA0,    A1, invA2, D2);
  GA4: andGate  port map(   A0,    A1, invA2, D3);
  GA5: andGate  port map(invA0, invA1,    A2, D4);
  GA6: andGate  port map(   A0, invA1,    A2, D5);
  GA7: andGate  port map(invA0,    A1,    A2, D6);
  GA8: andGate  port map(   A0,    A1,    A2, D7);

end func;
-------------------------------------------------------END
-------------------------------------------------------END


%%%%%%%%%FPGA%%%%%%%%%%%%%
NET "A0" LOC= "N17" ;
NET "A1" LOC= "H18" ;
NET "A2" LOC= "L14" ;
```

NET "D0" LOC= "F12" ;

NET "D1" LOC= "E12" ;

NET "D2" LOC= "E11" ;

NET "D3" LOC= "F11" ;

NET "D4" LOC= "C11" ;

NET "D5" LOC= "D11" ;

NET "D6" LOC= "E9" ;

NET "D7" LOC= "F9" ;

RTL SCHEMATIC