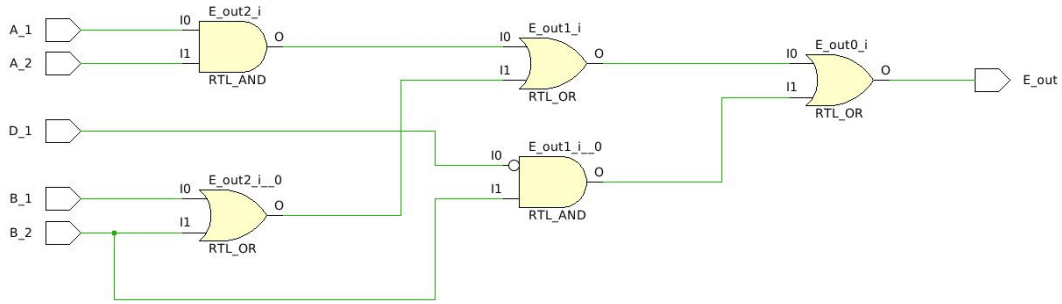
TIMOTHY LANGER
EMBEDDED SYSTEMS
RUID 189005424
HW #5
2)

IF STATEMENTS:

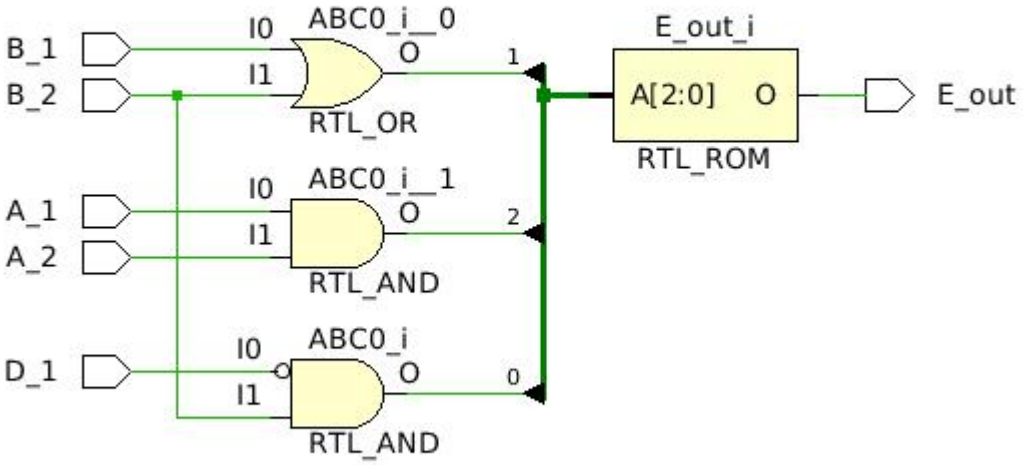
SCHEMATIC:



CODE:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity if_ckt is
      Port ( A_1 : in STD_LOGIC;
                  A_2 : in STD_LOGIC;
                  B_1 : in STD_LOGIC;
                  B_2 : in STD_LOGIC;
                  D_1 : in STD_LOGIC;
                  E_out : out STD_LOGIC);
end if_ckt;

architecture Behavioral of if_ckt is
begin
process (A_1, A_2, B_1, B_2, D_1)
begin
if( (A_1 and A_2) ='1' or (B_1 or B_2)='1' or ((not D_1) and B_2)='1' )then
E_out <='1';
else
E_out <='0';
end if;
end process;
end Behavioral;
```

CASE STATEMENTS



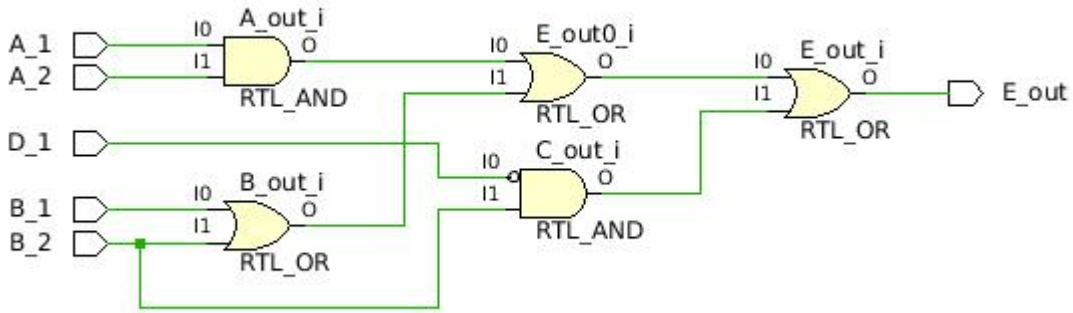
CODE:

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

entity case_ckt is
    Port ( A_1 : in STD_LOGIC;
           A_2 : in STD_LOGIC;
           B_1 : in STD_LOGIC;
           B_2 : in STD_LOGIC;
           D_1 : in STD_LOGIC;
           E_out : out STD_LOGIC);
end case_ckt;

architecture Behavioral of case_ckt is
signal A_out, b_out, C_out : std_logic;
signal ABC : std_logic_vector (2 downto 0);


begin
```

```
ABC <= A_out & B_out & C_out;

process (A_1, A_2, B_1, B_2, D_1)
begin

A_out <= A_1 and A_2;
B_out <= B_1 or B_2;
C_out <= (not D_1) and B_2;


case (ABC) is
when "000" => E_out <= '0';
when "001" => E_out <= '1';
when "010" => E_out <= '1';
when "011" => E_out <= '1';
when "100" => E_out <= '1';
when "101" => E_out <= '1';
when "110" => E_out <= '1';
when "111" => E_out <= '1';

end case;

end process;
end Behavioral;
```

3) Using Concurrent signal assignment to implement the same circuit

SCHEMATIC:



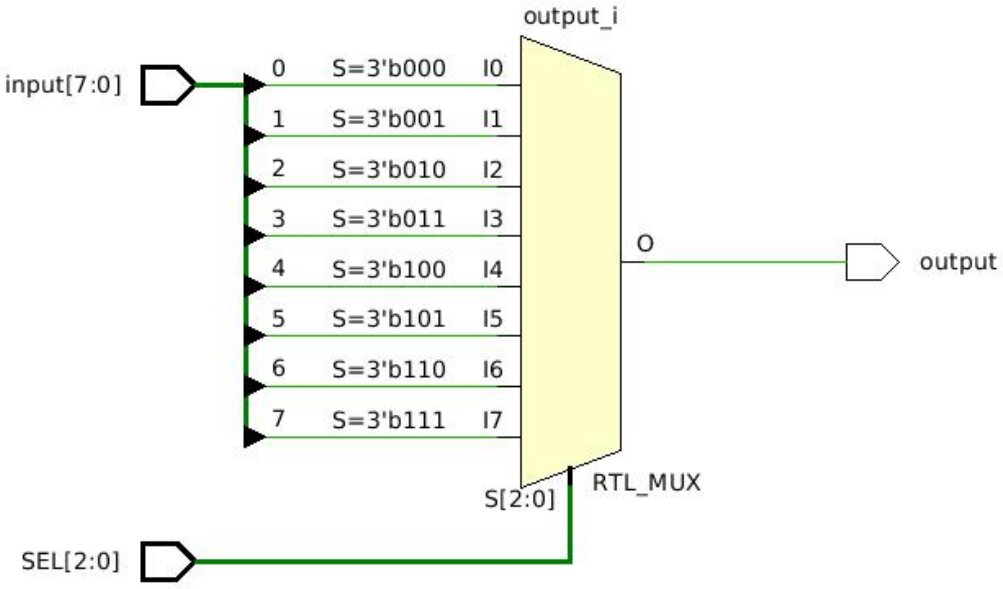
CODE:

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

entity concurrent_design is
    Port ( A_1 : in STD_LOGIC;
           A_2 : in STD_LOGIC;
           B_1 : in STD_LOGIC;
           B_2 : in STD_LOGIC;
           D_1 : in STD_LOGIC;
           E_out : out STD_LOGIC);
end concurrent_design;

architecture Behavioral of concurrent_design is

signal A_out, B_out, C_out : std_logic;

begin


A_out <= A_1 and A_2;
B_out <= B_1 or B_2;
C_out <= (not D_1) and B_2;
E_out <= A_out or B_out or C_out;

end Behavioral;
```

6) 8 to 1 Multiplexer

USING CASE STATEMENTS:

SCHEMATIC:



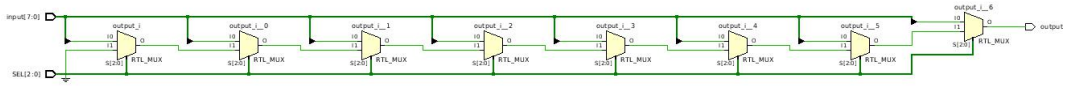
(VHDL CODE ON NEXT PAGE)

VHDL CODE:

```
library IEEE;
use IEEE.std_logic_1164.all;

entity mux is
port (
input : in std_logic_vector (7 downto 0);
SEL : in std_logic_vector (2 downto 0);
output : out std_logic);
end mux;

architecture behavioral of mux is
begin

process (SEL,input)
begin

case (SEL) is
when "000" => output <= input(0);
when "001" => output <= input(1);
when "010" => output <= input(2);
when "011" => output <= input(3);
when "100" => output <= input(4);
when "101" => output <= input(5);
when "110" => output <= input(6);
when "111" => output <= input(7);
when others => output <= '0';
end case;
end process;
end behavioral;
```

8 to 1 mux, USING IF STATEMENTS:

SCHEMATIC:

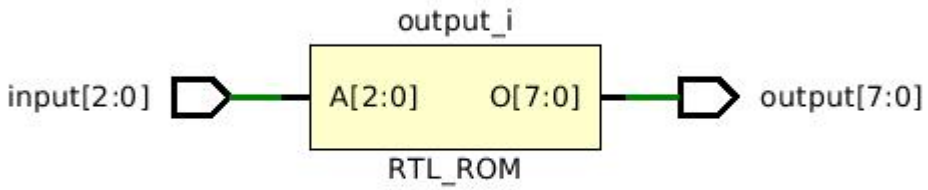


CODE:

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

entity if_mux is
      Port ( input : in STD_LOGIC_VECTOR (7 downto 0);
                SEL : in STD_LOGIC_VECTOR (2 downto 0);
                output : out STD_LOGIC);
end if_mux;

architecture Behavioral of if_mux is

begin

process (SEL,input)
begin

    if SEL = "000" then output<= input(0);
elsif SEL = "001" then output<= input(1);
elsif SEL = "010" then output<= input(2);
elsif SEL = "011" then output<= input(3);
elsif SEL = "100" then output<= input(4);
elsif SEL = "101" then output<= input(5);
elsif SEL = "110" then output<= input(6);
elsif SEL = "111" then output<= input(7);
else
output<='0';
end if;

end process;

end Behavioral;
```

8) DECODER USING CASE:

SCHEMATIC:



CODE:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity case_decoder is
      Port ( input : in STD_LOGIC_VECTOR (2 downto 0);
                output : out STD_LOGIC_VECTOR (7 downto 0));
end case_decoder;

architecture Behavioral of case_decoder is

begin
process(input)
begin
case (input) is
when "000" => output <= "11111110";
when "001" => output <= "11111101";
when "010" => output <= "11111011";
when "011" => output <= "11110111";
when "100" => output <= "11101111";
when "101" => output <= "11011111";
when "110" => output <= "10111111";
when "111" => output <= "01111111";
when others =>output<= "11111111";
end case;
end process;
end Behavioral;
```

DECODER USING IF STATEMENTS:

CODE:

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


entity decoder_if is
      Port ( input : in STD_LOGIC_VECTOR (3 downto 0);
                output : out STD_LOGIC_VECTOR (7 downto 0));
end decoder_if;

architecture Behavioral of decoder_if is

begin

process(input)
begin
    if input = "000" then output <="11111110";
elsif input ="001" then output <= "11111101";
elsif input ="010" then output <= "11111011";
elsif input ="011" then output <= "11110111";
elsif input ="100" then output <= "11101111";
elsif input ="101" then output <= "11011111";
elsif input ="110" then output <= "10111111";
elsif input ="111" then output <= "01111111";
else output <="11111111";
end if;

end process;


end Behavioral;
```