VLSI LAB REPORT

Name: Anuran Chakraborty

Roll No: 20

Section: A1

Index

1. Design a 4-to-2 encoder.

Description

An encoder in digital electronics is a one-hot to binary converter. That is, if there are 2^n input lines, and at most only one of them will ever be high, the binary code of this 'hot' line is produced on the n-bit output lines. A 4-to-2 encoder has 4 input lines and 2 output lines.

Truth Table

Input				Output	
i_3	i_2	i_1	i_0	o_1	o_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Block Diagram

(a) Using structural modelling.

```
----------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    19:26:09 06/07/2020
-- Design Name:
-- Module Name:    foutto2enc_str - 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 foutto2enc_str is
    Port ( i : in  STD_LOGIC_VECTOR (3 downto 0);
           o : out  STD_LOGIC_VECTOR (1 downto 0));
end foutto2enc_str;

architecture Behavioral of foutto2enc_str is

begin

    o(0)<=i(1) or i(3);
    o(1)<=i(2) or i(3);

end Behavioral;
```

(b) Using behavioral modelling using concurrent statements.

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    19:38:03 06/07/2020
-- Design Name:
-- Module Name:    foutto2enc_beh - 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 foutto2enc_beh is
    Port ( i : in  STD_LOGIC_VECTOR (3 downto 0);
           o : out  STD_LOGIC_VECTOR (1 downto 0));
end foutto2enc_beh;

architecture Behavioral of foutto2enc_beh is

begin

with i select o<=
    "00" when "0001",
    "01" when "0010",
    "10" when "0100",
    "11" when "1000",
    "ZZ" when others;

end Behavioral;
```

(c) Using behavioral modelling using sequential statements.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    19:44:18 06/07/2020
-- Design Name:
-- Module Name:    fourto2enc_beh2 - 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 fourto2enc_beh2 is
    Port ( i : in  STD_LOGIC_VECTOR (3 downto 0);
           o : out STD_LOGIC_VECTOR (1 downto 0));
end fourto2enc_beh2;

architecture Behavioral of fourto2enc_beh2 is

begin

Process(i)
begin
    case i is
        when "0001" => o<="00";
        when "0010" => o<="01";
        when "0100" => o<="10";
        when "1000" => o<="11";
        when others => o<="ZZ";
    end case;

end Process;

end Behavioral;
```

2. Design an 8-to-3 encoder.

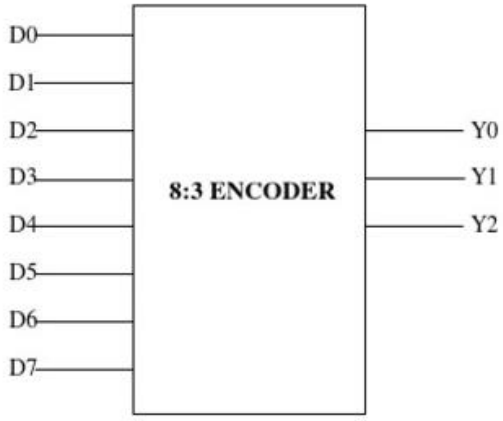
Description

An encoder in digital electronics is a one-hot to binary converter. That is, if there are 2^n input lines, and at most only one of them will ever be high, the binary code of this 'hot' line is produced on the n-bit output lines. An 8-to-3 encoder has 8 input lines and 3 output lines.

Truth Table

Input								Output		
i_7	i_6	i_5	i_4	i_3	i_2	i_1	i_0	o_2	o_1	o_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

Block Diagram



(a) Using structural modelling.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    00:29:28 01/27/2020
-- Design Name:
-- Module Name:    eighttothree - 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 eighttothree is
    Port ( i : in  STD_LOGIC_VECTOR (7 downto 0);
           o : out  STD_LOGIC_VECTOR (2 downto 0));
end eighttothree;

architecture Behavioral of eighttothree is

begin
      o(0)<= i(1) or i(3) or i(5) or i(7);
      o(1)<= i(2) or i(3) or i(6) or i(7);
      o(2)<= i(4) or i(5) or i(6) or i(7);

end Behavioral;
```

(b) **Using behavioral modelling using select statements.**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    00:47:07 01/27/2020
-- Design Name:
-- Module Name:    eighttothree_beh_c - 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 eighttothree_beh_c is
    Port ( i : in  STD_LOGIC_VECTOR (7 downto 0);
           o : out  STD_LOGIC_VECTOR (2 downto 0));
end eighttothree_beh_c;

architecture Behavioral of eighttothree_beh_c is

begin
with i select o<=
    "000" when "00000001",
    "001" when "00000010",
    "010" when "00000100",
    "011" when "00001000",

    "100" when "00010000",
    "101" when "00100000",
    "110" when "01000000",
    "111" when "10000000",

    "ZZZ" when others ;

end Behavioral;
```

(c) Using behavioral modelling using case statements.

```
----------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    00:55:27 01/27/2020
-- Design Name:
-- Module Name:    eighttothreee_bhe_s - 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 eighttothreee_bhe_s is
    Port ( i : in  STD_LOGIC_VECTOR (7 downto 0);
           o : out  STD_LOGIC_VECTOR (2 downto 0));
end eighttothreee_bhe_s;

architecture Behavioral of eighttothreee_bhe_s is

begin

Process(i)
begin
      case i is
            when "00000001" => o<="000";
            when "00000010" => o<="001";
            when "00000100" => o<="010";
            when "00001000" => o<="011";

            when "00010000" => o<="100";
            when "00100000" => o<="101";
            when "01000000" => o<="110";
            when "10000000" => o<="111";

            when others => o<="ZZZ";
      end case;
end Process;

end Behavioral;
```

3. Design a decimal-to-BCD encoder.

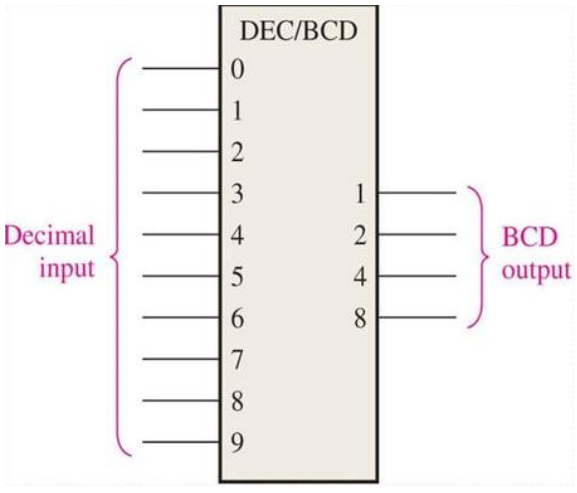
Description

An encoder in digital electronics is a one-hot to binary converter. That is, if there are 2^n input lines, and at most only one of them will ever be high, the binary code of this 'hot' line is produced on the n-bit output lines. A decimal to BCD encoder has 10 input lines and 4 output lines. If i_x is set to 1 then the output is the binary equivalent of x.

Truth Table

Input										Output			
i_9	i_8	i_7	i_6	i_5	i_4	i_3	i_2	i_1	i_0	o_3	o_2	o_1	o_0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0	0	1

Block Diagram



Code

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    01:21:23 01/27/2020
-- Design Name:
-- Module Name:    dectoBCD_str - 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 dectoBCD_str is
    Port ( i : in  STD_LOGIC_VECTOR (9 downto 0);
           o : out  STD_LOGIC_VECTOR (3 downto 0));
end dectoBCD_str;

architecture Behavioral of dectoBCD_str is

begin
      o(3) <= i(9) or i(8);
      o(2) <= i(7) or i(6) or i(5) or i(4);
      o(1) <= i(7) or i(6) or i(3) or i(2);
      o(0) <= i(9) or i(7) or i(5) or i(3) or i(1);

end Behavioral;
```

4. Design a 1-to-2 decoder.

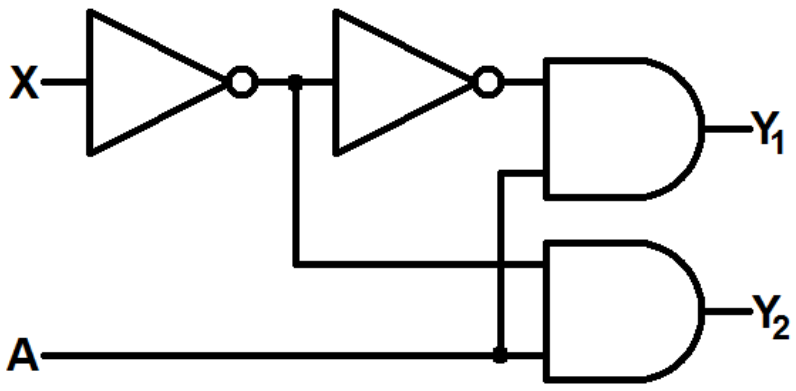
Description

Decoder is a combinational circuit that has 'n' input lines and maximum of 2^n output lines. One of these outputs will be active High based on the combination of inputs present, when the decoder is enabled. That means decoder detects a particular code. A 1-to-2 decoder has 1 input lines and 2 output lines. An enable input is provided to switch the decoder on and off.

Truth Table

Input		Output	
e	**i_0**	**o_1**	**o_0**
0	x	0	0
1	0	0	1
1	1	1	0

Gate Diagram



(a) Using structural modelling.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    00:29:56 02/03/2020
-- Design Name:
-- Module Name:    onetotwodecoder - 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 onetotwodecoder is
    Port ( e : in  BIT;
           i : in  BIT;
           o : out BIT_VECTOR (1 downto 0));
end onetotwodecoder;

architecture Behavioral of onetotwodecoder is
begin

o(0)<= e and not(i);
o(1)<= e and i;

end Behavioral;
```

(b) Using behavioral modelling using concurrent statements.

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    00:34:38 02/03/2020
-- Design Name:
-- Module Name:    onetotwodecoder2 - 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 onetotwodecoder2 is
    Port ( e : in  BIT;
           i : in  BIT;
           o : out  BIT_VECTOR (1 downto 0));
end onetotwodecoder2;

architecture Behavioral of onetotwodecoder2 is

begin

      with (e & i) select o<=
            "01" when "10",
            "10" when "11",
            "00" when others;
end Behavioral;
```

(c) Using behavioral modelling using sequential statement.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    00:45:09 02/03/2020
-- Design Name:
-- Module Name:    onetotwodecoder3 - 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 onetotwodecoder3 is
    Port ( e : in  STD_LOGIC;
           i : in  STD_LOGIC;
           o : out  STD_LOGIC_VECTOR (1 downto 0));
end onetotwodecoder3;

architecture Behavioral of onetotwodecoder3 is

begin

process(e,i)
begin

      if(e='0') then
            o<="00";
      elsif(i='0') then
            o<="01";
      elsif(i='1') then
            o<="10";
      end if;

end process;
end Behavioral;
```

5. Design a 2-to-4 decoder.

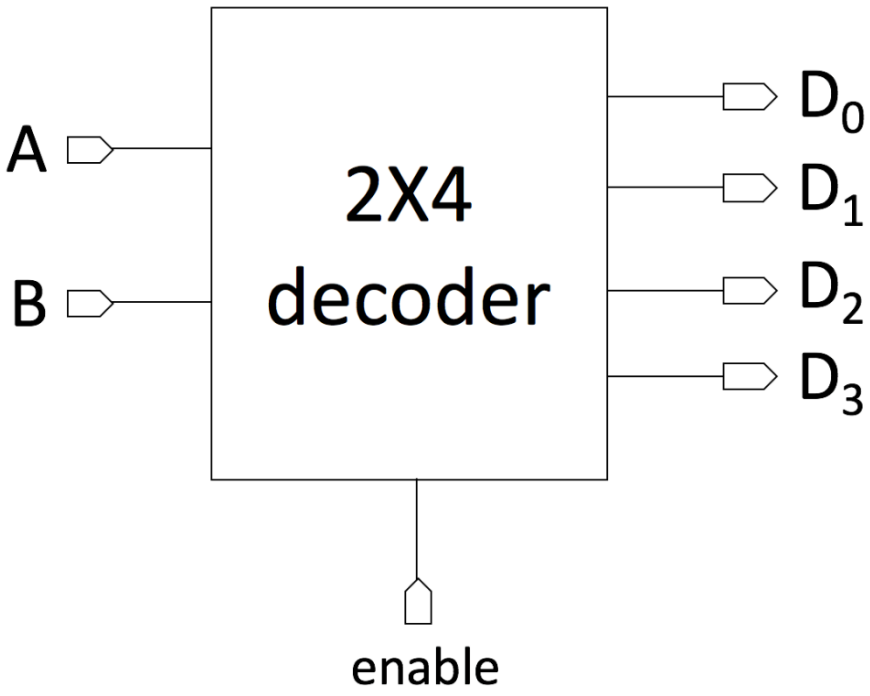
Description

Decoder is a combinational circuit that has 'n' input lines and maximum of 2^n output lines. One of these outputs will be active High based on the combination of inputs present, when the decoder is enabled. That means decoder detects a particular code. A 2-to-4 decoder has 2 input lines and 4 output lines. An enable input is provided to switch the decoder on and off.

Truth Table

Input			Output			
e	**i_1**	**i_0**	**o_3**	**o_2**	**o_1**	**o_0**
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Block Diagram



(a) Using structural modelling.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    01:09:16 02/03/2020
-- Design Name:
-- Module Name:    twotofourdecoder2 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
-----
library IEEE;
use IEEE.BIT_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 twotofourdecoder2 is
    Port ( e : in  BIT;
           i : in  BIT_VECTOR (1 downto 0);
           o : out  BIT_VECTOR (3 downto 0));
end twotofourdecoder2;

architecture Behavioral of twotofourdecoder2 is

begin

      o(0)<= e and not(i(1)) and not (i(0));
      o(1)<= e and not(i(1)) and (i(0));
      o(2)<= e and(i(1)) and not (i(0));
      o(3)<= e and (i(1)) and (i(0));


end Behavioral;
```

(b) Using behavioral modelling sequential statements.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    01:48:03 02/03/2020
-- Design Name:
-- Module Name:    twotofourdecoder4 - 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 twotofourdecoder4 is
    Port ( e : in  STD_LOGIC;
           i : in  STD_LOGIC_VECTOR (1 downto 0);
           o : out  STD_LOGIC_VECTOR (3 downto 0));
end twotofourdecoder4;

architecture Behavioral of twotofourdecoder4 is

begin

process(e,i)
begin

      if(e='0') then
            o<="0000";
      elsif(i="00") then
            o<="0001";
      elsif(i="01") then
            o<="0010";
      elsif(i="10") then
            o<="0100";
      elsif(i="11") then
```

```
            o<="1000";
      end if;
end process;
end Behavioral;
```

(c) Using behavioral modelling using concurrent statements.

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    01:43:50 02/03/2020
-- Design Name:
-- Module Name:    twotofourdecoder3 - 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 twotofourdecoder3 is
    Port ( e : in  STD_LOGIC;
           i : in  STD_LOGIC_VECTOR (1 downto 0);
           o : out  STD_LOGIC_VECTOR (3 downto 0));
end twotofourdecoder3;

architecture Behavioral of twotofourdecoder3 is

begin

with (e & i) select o<=
      "0001" when "100",
      "0010" when "101",
      "0100" when "110",
      "1000" when "111",
      "0000" when others;
end Behavioral;
```

6. Design a 3-to-8 decoder.

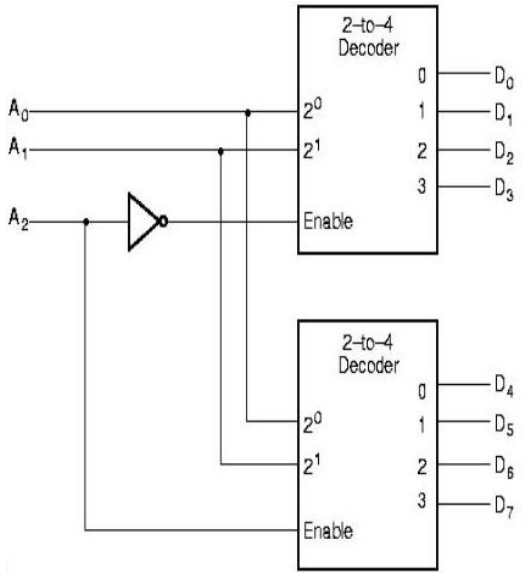
Description

Decoder is a combinational circuit that has 'n' input lines and maximum of 2^n output lines. One of these outputs will be active High based on the combination of inputs present, when the decoder is enabled. That means decoder detects a particular code. A 3-to-8 decoder has 3 input lines and 8 output lines. An enable input is provided to switch the decoder on and off.

Truth Table

Input				Output							
e	**i_2**	**i_1**	**i_0**	**o_7**	**o_6**	**o_5**	**o_4**	**o_3**	**o_2**	**o_1**	**o_0**
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

Block Diagram



(a) Using component instantiate.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    21:10:22 06/07/2020
-- Design Name:
-- Module Name:    threeto8dec_comp - 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 threeto8dec_comp is
    Port ( inp : in  STD_LOGIC_VECTOR (2 downto 0);
                    en : in  STD_LOGIC;
           op : out  STD_LOGIC_VECTOR (7 downto 0));
end threeto8dec_comp;

architecture Behavioral of threeto8dec_comp is

component twotofourdecoder4 is
    Port ( e : in  STD_LOGIC;
           i : in  STD_LOGIC_VECTOR (1 downto 0);
           o : out  STD_LOGIC_VECTOR (3 downto 0));
end component;
signal notinp:STD_LOGIC;

begin
      notinp<=not inp(2);
      dec1:twotofourdecoder4 port map(inp(2),inp(1 downto 0),op(7 downto 4));
      dec2:twotofourdecoder4 port map(notinp,inp(1 downto 0),op(3 downto 0));

end Behavioral;
```

(b) Using procedural statement.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    21:23:36 06/07/2020
-- Design Name:
-- Module Name:    threeto8dec_proc - 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 threeto8dec_proc is
    Port ( inp : in  STD_LOGIC_VECTOR (2 downto 0);
           op : out  STD_LOGIC_VECTOR (7 downto 0));
end threeto8dec_proc;

architecture Behavioral of threeto8dec_proc is


procedure twotofourdecoder3 ( e : in  STD_LOGIC;
          i : in  STD_LOGIC_VECTOR (1 downto 0);
          o : out  STD_LOGIC_VECTOR (3 downto 0)) is
begin

      with (e & i) select o:=
            "0001" when "100",
            "0010" when "101",
            "0100" when "110",
            "1000" when "111",
            "0000" when others;

end procedure;
```

```
begin

	process(inp)
	variable varop:STD_LOGIC_VECTOR (7 downto 0);
	begin

		dec1:twotofourdecoder3(inp(2),inp(1 downto 0),varop(7 downto 4));
		dec2:twotofourdecoder3(not inp(2),inp(1 downto 0),varop(3 downto
0));
		op<=varop;
	end process;


end Behavioral;
```

7. Design Half Adder using gate level modelling

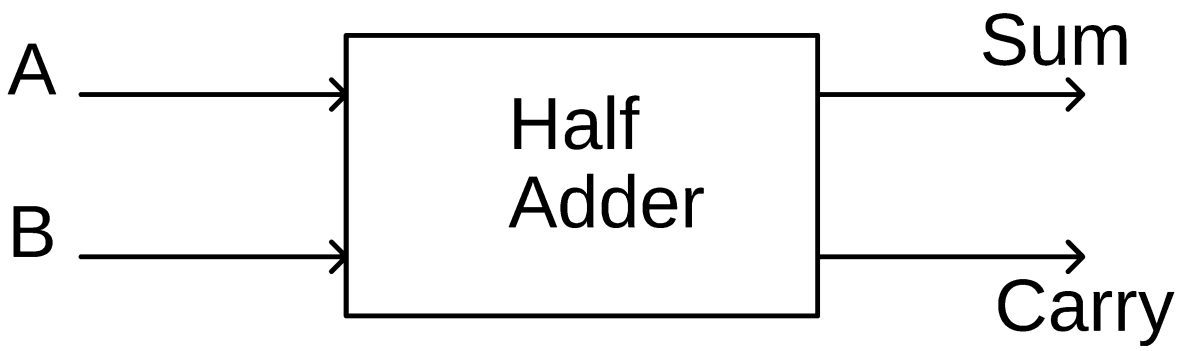
Description

Half adder is a combinational circuit that adds two bits and produces a sum bit (S) and carry bit (C) as the output. If A and B are the input bits, then sum bit (S) is the X-OR of A and B and the carry bit (C) will be the AND of A and B.

Truth Table

Input		Output	
i_1	i_0	sum	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Block Diagram



Code

```
----------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    23:52:26 02/09/2020
-- Design Name:
-- Module Name:    halfadder - 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 halfadder is
    Port ( a : in  BIT;
           b : in  BIT;
           s : out  BIT;
           c : out  BIT);
end halfadder;

architecture Behavioral of halfadder is

begin

      s<=a xor b;
      c<=a and b;

end Behavioral;
```

8. Design Full Adder

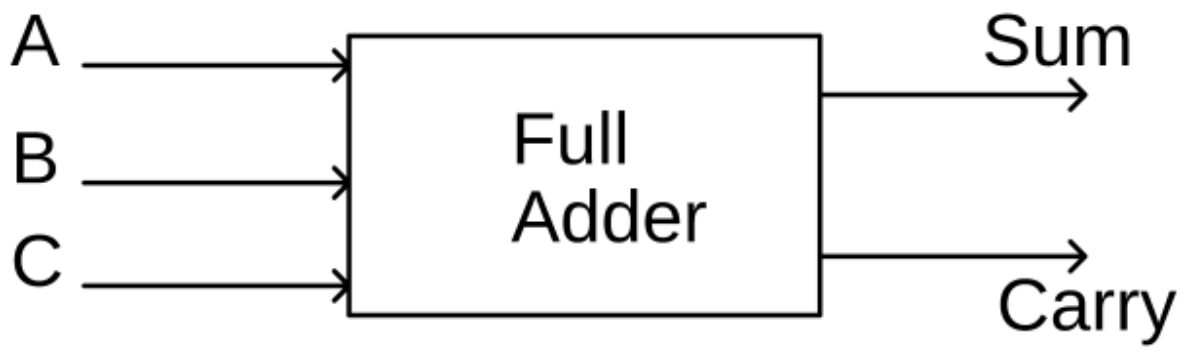
Description

Full Adder is a combinational circuit which adds three inputs and produces two outputs. The first two inputs are A and B and the third input is an input carry as C-IN. The output carry is designated as C-OUT and the normal output is designated as S which is SUM.

Truth Table

	Input		Output	
i_2	i_1	i_0	sum	carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Block Diagram



(a) Using component instantiate.

```
----------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    23:36:06 02/09/2020
-- Design Name:
-- Module Name:    fulladder - 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 fulladder is
    Port ( x : in  BIT;
           y : in  BIT;
           z : in  BIT;
           s : out BIT;
           c : out BIT);
end fulladder;

architecture Behavioral of fulladder is

component halfadder
    Port ( a : in  BIT;
           b : in  BIT;
           s : out BIT;
           c : out BIT);
end component;

signal ins:BIT;
signal inc:BIT;
signal inc2:BIT;

begin

      h1:halfadder port map(x,y,ins,inc);
      h2:halfadder port map(z,ins,s,inc2);
      c<=inc2 or inc;

end Behavioral;
```

(b) Using function or procedure.

```
----------------------------------------------------------------------------------
```

```
-- Company:
-- Engineer:
--
-- Create Date:    00:36:27 02/10/2020
-- Design Name:
-- Module Name:    fulladder2 - 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 fulladder2 is
    Port ( i : in  BIT_VECTOR (2 downto 0);
           sum : out  BIT;
           carry : out  BIT);
end fulladder2;

architecture Behavioral of fulladder2 is

procedure halfadder (signal inp:in BIT_VECTOR(1 downto 0);
                                    signal s:out BIT;
                                    signal c:out BIT) is
begin

      s<=inp(1) xor inp(0);
      c<=inp(1) and inp(0);
end procedure;

signal a,b,d: BIT;
signal in1,in2:BIT_VECTOR(1 downto 0);
begin
      in1<=i(1 downto 0);
      in2<=i(2)& a;
      ha1:halfadder(in1,a,b);
      ha2:halfadder(in2,sum,d);
      carry<=b or d;
end Behavioral;
```

9. Design 4-bit Ripple-Carry-Adder

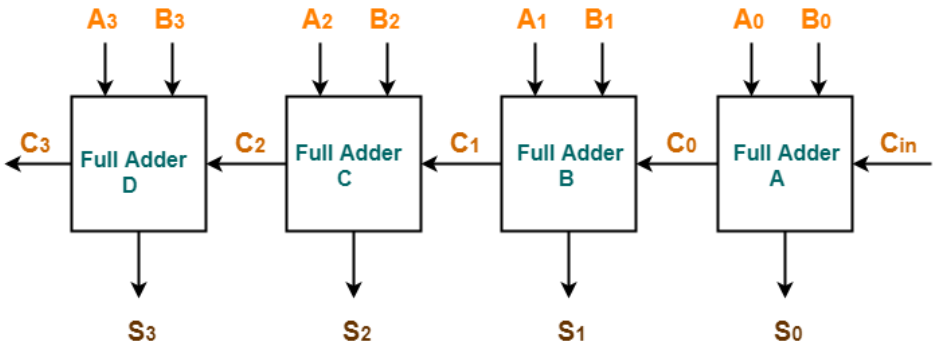
Description

Multiple full adder circuits can be cascaded in parallel to add an N-bit number. For an N- bit parallel adder, there must be N number of full adder circuits. A ripple carry adder is a logic circuit in which the carry-out of each full adder is the carry in of the succeeding next most significant full adder. It is called a ripple carry adder because each carry bit gets rippled into the next stage.

Truth Table

A_3	A_2	A_1	A_0	B_3	B_2	B_1	B_0	C_{out}	S_3	S_2	S_1	S_0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	0	0	0	1	0	1
0	1	0	0	0	1	0	1	0	1	0	0	1
0	1	1	0	1	0	0	0	0	1	1	1	0
1	0	0	1	0	1	1	1	1	0	0	0	0
1	0	1	0	1	0	0	1	1	0	0	1	1
1	1	0	1	1	0	1	0	1	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	0

Block Diagram



(a) Using generate statement.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    20:16:24 06/07/2020
-- Design Name:
-- Module Name:    fourbitripple_gen - 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 fourbitripple_gen is
    Port ( a : in  STD_LOGIC_VECTOR (3 downto 0);
           b : in  STD_LOGIC_VECTOR (3 downto 0);
           cin : in  STD_LOGIC;
           s : out  STD_LOGIC_VECTOR (3 downto 0);
           cout : out  STD_LOGIC);
end fourbitripple_gen;

architecture Behavioral of fourbitripple_gen is

component fulladder
    Port ( x : in  STD_LOGIC;
           y : in  STD_LOGIC;
           z : in  STD_LOGIC;
           s : out STD_LOGIC;
           c : out STD_LOGIC);
end component;

signal c:STD_LOGIC_VECTOR (4 downto 0);

begin

    c(0)<='0';

    gen1: for i in 0 to 3 generate
        fa:fulladder port map(a(i),b(i),c(i),s(i),c(i+1));
    end generate;
    cout<=c(4);


end Behavioral;
```

(b) Using Loop statement.

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    00:09:50 03/02/2020
-- Design Name:
-- Module Name:    fbrcaprocess - 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 fbrcaprocess is
    Port ( a : in  BIT_VECTOR (3 downto 0);
           b : in  BIT_VECTOR (3 downto 0);
           sum : out  BIT_VECTOR (3 downto 0);
           cout : out  BIT;
           cin : in  BIT);
end fbrcaprocess;

architecture Behavioral of fbrcaprocess is

procedure halfadder(ha,hb:in bit;
                                          hs,hc:out bit) is
begin
      hs:= ha xor hb;
      hc:= ha and hb;
end procedure;

procedure fulladder(fa,fb,fc:in bit;
                                          fss,fcc:out bit) is
variable ints1,ints2,intc1,intc2:bit;
begin
      h1:halfadder(fb,fc,ints1,intc1);
      h2:halfadder(fa,ints1,ints2,intc2);
      fss:=ints2;
```

```
      fcc:=intc1 or intc2;
end procedure;


begin
      process(a,b,cin)
   variable c:bit_vector(4 downto 0);
      variable ss:bit_vector(3 downto 0);
      begin
            c(0):=cin;

            for i in 0 to 3 loop
                  f:fulladder(a(i),b(i),c(i),ss(i),c(i+1));
            end loop;
            sum<=ss;
            cout<=c(4);
      end process;



end Behavioral;
```

10. Design BCD Adder.

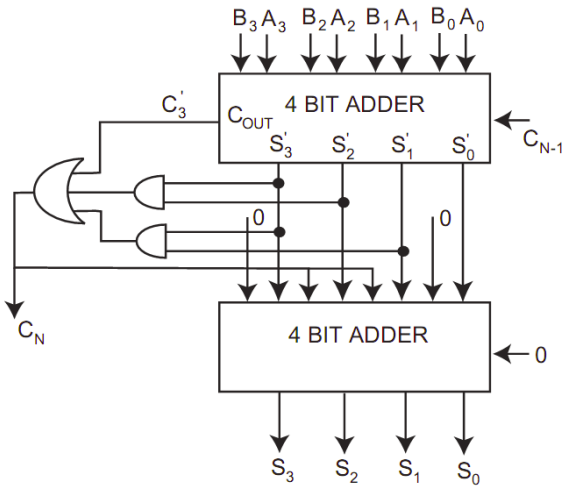
Description

A BCD adder is used to add two numbers and then get the BCD output instead of the Binary Output.

Truth Table

A_3	A_2	A_1	A_0	B_3	B_2	B_1	B_0	C_{out}	S_3	S_2	S_1	S_0	Decimal
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	0	0	0	1	1	3
0	1	0	0	0	0	1	1	0	0	1	1	1	7
0	0	0	1	1	0	0	0	0	1	0	0	1	9
0	1	1	0	0	1	0	1	1	0	0	0	1	11
1	0	0	0	0	1	1	0	1	0	1	0	0	14
0	1	1	1	1	0	0	0	1	0	1	0	1	15
1	0	0	1	1	0	0	1	1	1	0	0	0	18

Block Diagram



Code

```
----------------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    20:06:54 06/07/2020
-- Design Name:
-- Module Name:    BCDAdder - 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 BCDAdder is
    Port ( an : in  STD_LOGIC_VECTOR (3 downto 0);
           bn : in  STD_LOGIC_VECTOR (3 downto 0);
           sum : out  STD_LOGIC_VECTOR (3 downto 0);
           k : inout  STD_LOGIC);
end BCDAdder;

architecture Behavioral of BCDAdder is

component fourbitripple_gen is
    Port ( a : in  STD_LOGIC_VECTOR (3 downto 0);
           b : in  STD_LOGIC_VECTOR (3 downto 0);
           cin : in  STD_LOGIC;
           s : out  STD_LOGIC_VECTOR (3 downto 0);
           cout : out  STD_LOGIC);
end component;

signal binarySum: STD_LOGIC_VECTOR (3 downto 0);
signal carry: STD_LOGIC;
signal condition: STD_LOGIC;
signal toAdd: STD_LOGIC_VECTOR (3 downto 0);
signal nouse: STD_LOGIC;
begin

      rc1:fourbitripple_gen port map(an,bn,'0',binarySum,carry);
      condition<= carry or (binarySum(3) and binarySum(2)) or (binarySum(3)
and binarySum(1));

      Process(toAdd,condition)
      begin

            if(condition='0') then
                  toAdd<="0000";
            elsif(condition='1') then
                  toAdd<="0110";
            end if;
      end process;
      rc2:fourbitripple_gen port map(binarySum,toAdd,'0',sum,k);
end Behavioral;
```

11. Design Adder/Subtractor circuit.

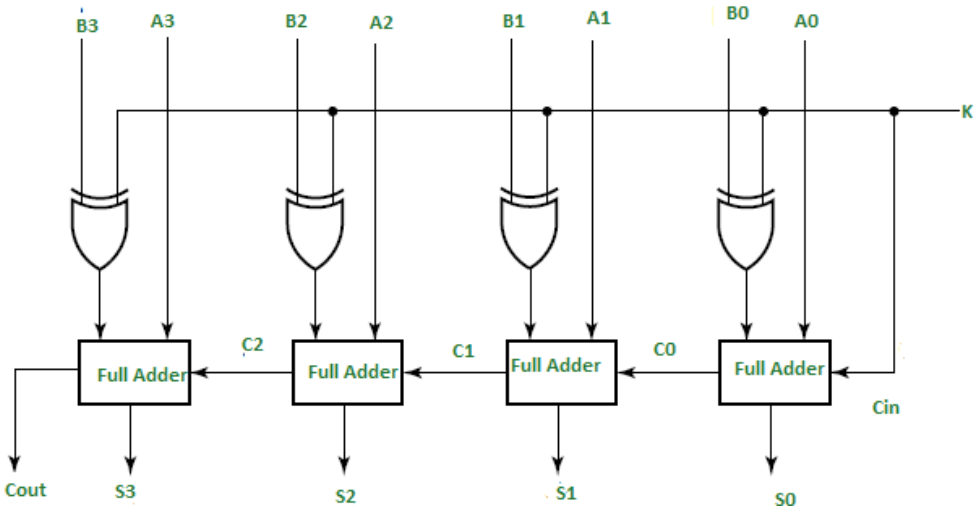
Description

A Binary Adder-Subtractor is one which is capable of both addition and subtraction of binary numbers in one circuit itself. The operation being performed depends upon the binary value the control signal holds. It is one of the components of the ALU (Arithmetic Logic Unit).

Truth Table

C_{in}	A_3	A_2	A_1	A_0	B_3	B_2	B_1	B_0	C_{out}	S_3	S_2	S_1	S_0
0	0	0	0	1	0	0	1	0	0	0	0	1	1
1	0	0	0	1	0	0	1	0	0	0	1	0	1
0	1	0	0	0	0	1	0	1	0	1	1	0	1
1	1	0	0	0	0	1	0	1	0	0	0	1	1
0	0	1	1	1	1	0	1	0	1	0	0	0	1
1	0	1	1	1	1	0	1	0	0	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	0	0	0	0	0

Block Diagram



Code

```
----------------------------------------------------------------------------
-----
-- Company:
-- Engineer:
--
-- Create Date:    00:09:23 02/17/2020
-- Design Name:
-- Module Name:    addersub - 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 addersub is
    Port ( a : in  BIT_VECTOR (3 downto 0);
           b : in  BIT_VECTOR (3 downto 0);
           cin : in  BIT;
           s : out  BIT_VECTOR (3 downto 0);
           cout : out  BIT);
end addersub;

architecture Behavioral of addersub is

component fulladder
    Port ( x : in  BIT;
           y : in  BIT;
           z : in  BIT;
           s : out BIT;
           c : out BIT);
end component;

signal c:BIT_VECTOR (4 downto 0);
signal inputb:BIT_VECTOR (3 downto 0);

begin

      inputb<=b when cin='0' else not(b);
      c(0)<=cin;
      gen1:for i in 0 to 3 generate
            fa:fulladder port map(a(i),inputb(i),c(i),s(i),c(i+1));
      end generate;

      cout<='0' when cin='1' and c(4)='1' else c(4);

end Behavioral;
```