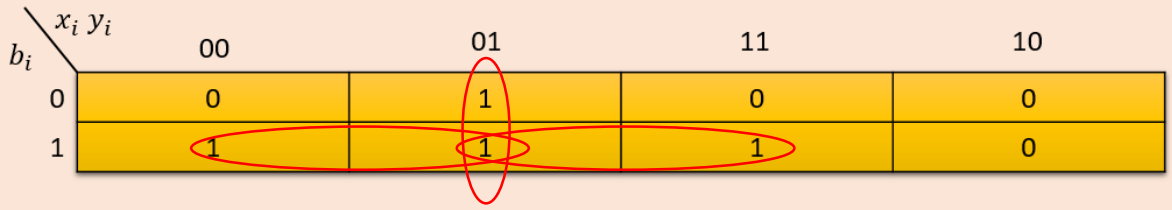
به نام خدا

جدول درستی مدار تفریق کننده ی کامل یک بیتی را می کشیم و داریم:

x_i	y_i	b_i	b_{i+1}	d_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

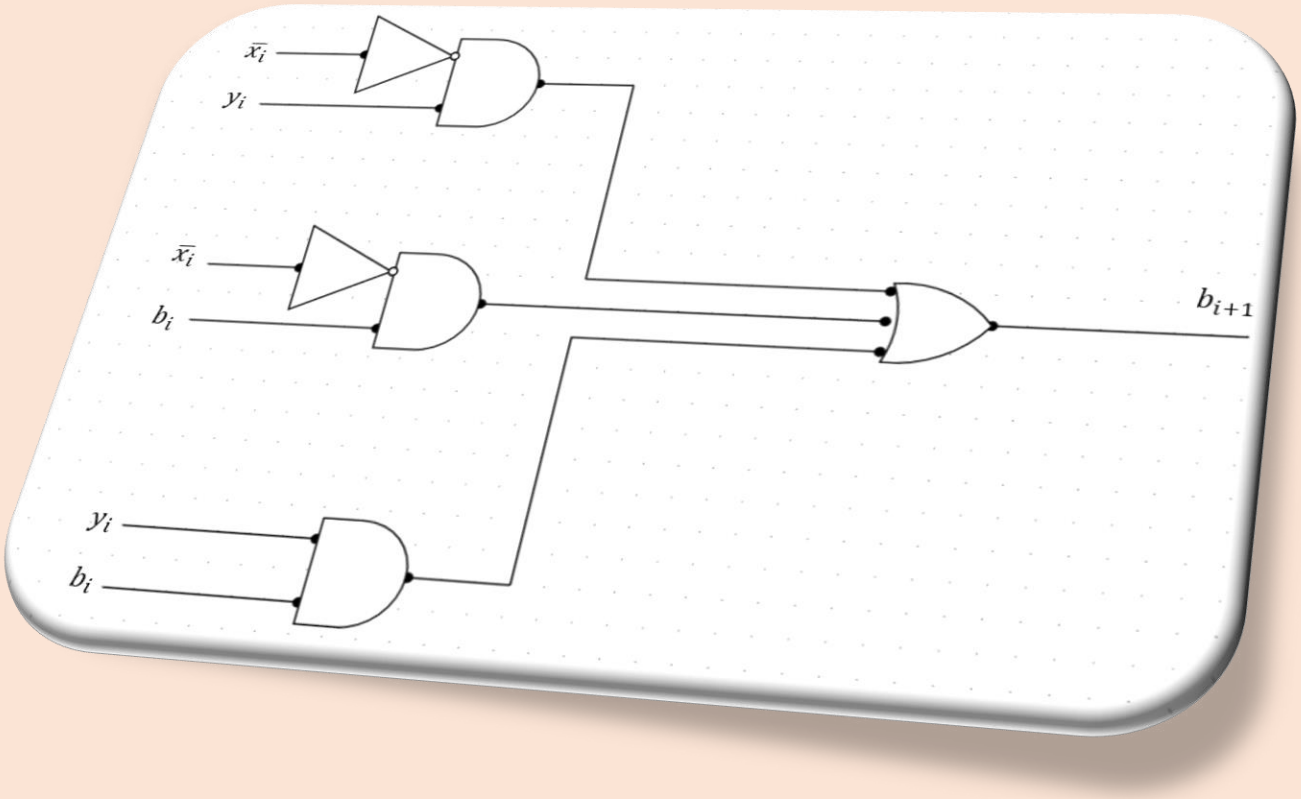
حال با استفاده از جدول کارنو برای مقدار b_{i+1} داریم:



بنابراین داریم:

$$b_{i+1} = \bar{x_i}y_i + \bar{x_i}b_i + y_i b_i$$

بنابراین برای طراحی گیت این قسمت داریم:



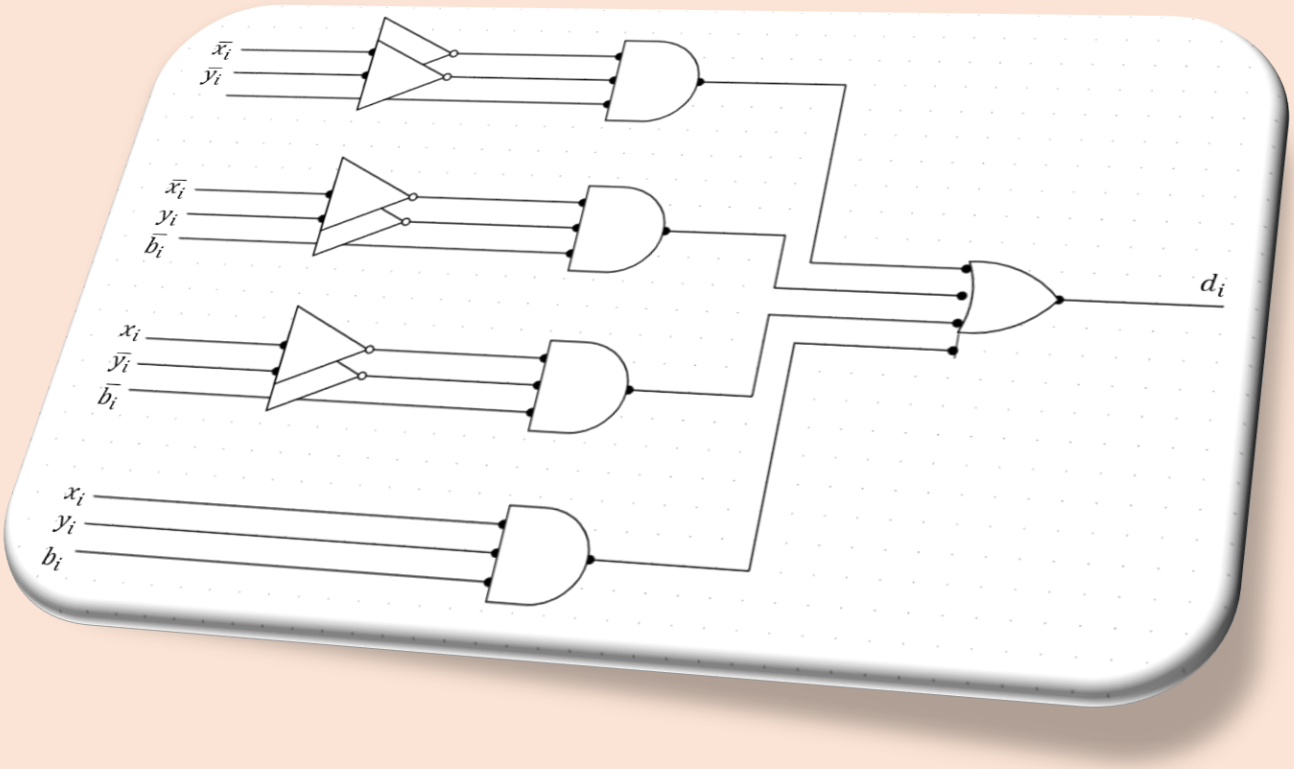
حال با استفاده از جدول کارنو برای مقدار d_i داریم:

b_i \ $x_i\ y_i$	00	01	11	10
0	0	1	0	1
1	1	0	1	0

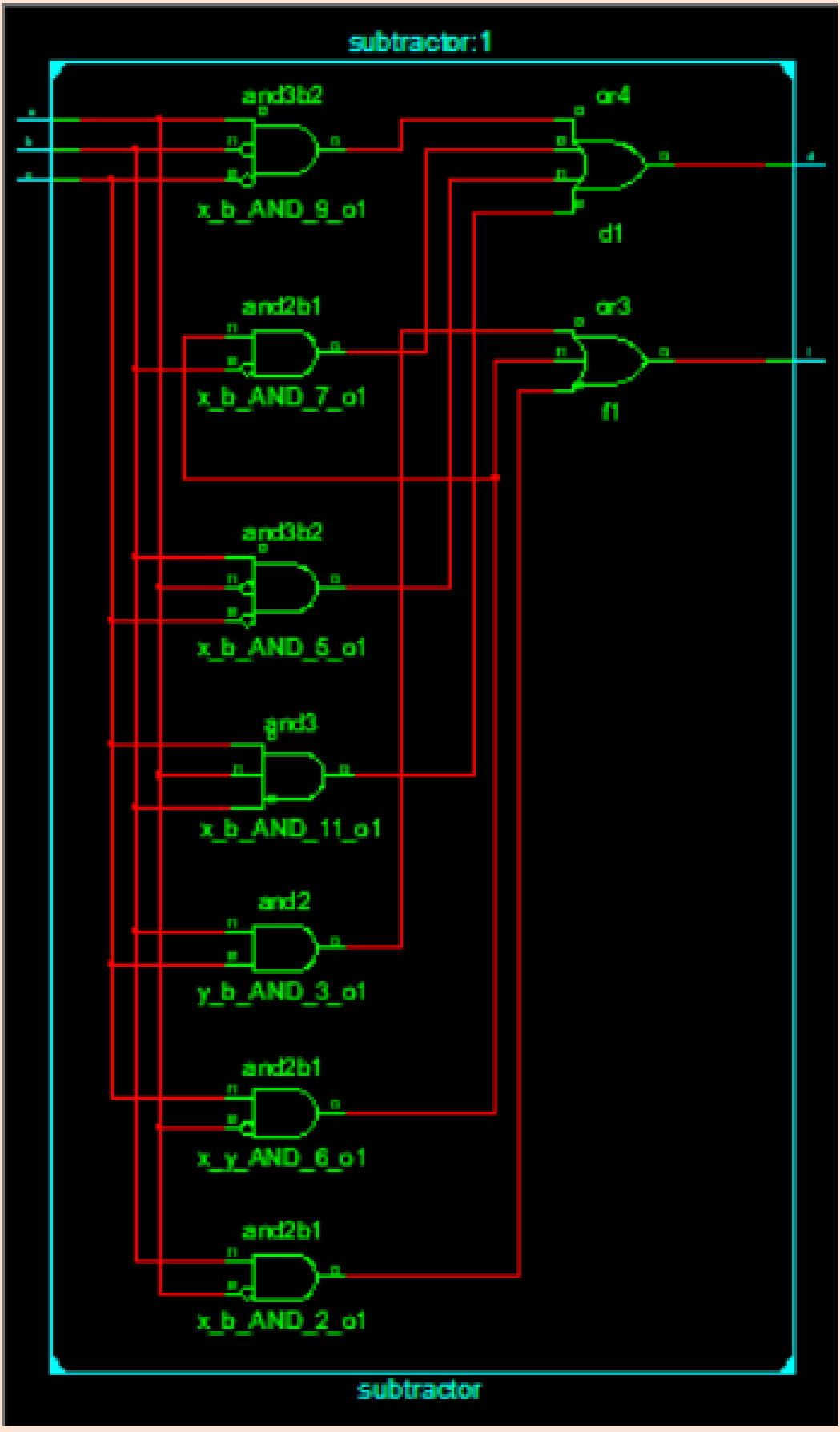
بنابراین داریم:

$$d_i = \overline{x_i}\,\overline{y_i}\,b_i + \overline{x_i}\,y_i\,\overline{b_i} + x_i\,\overline{y_i}\,\overline{b_i} + x_i\,y_i\,b_i$$

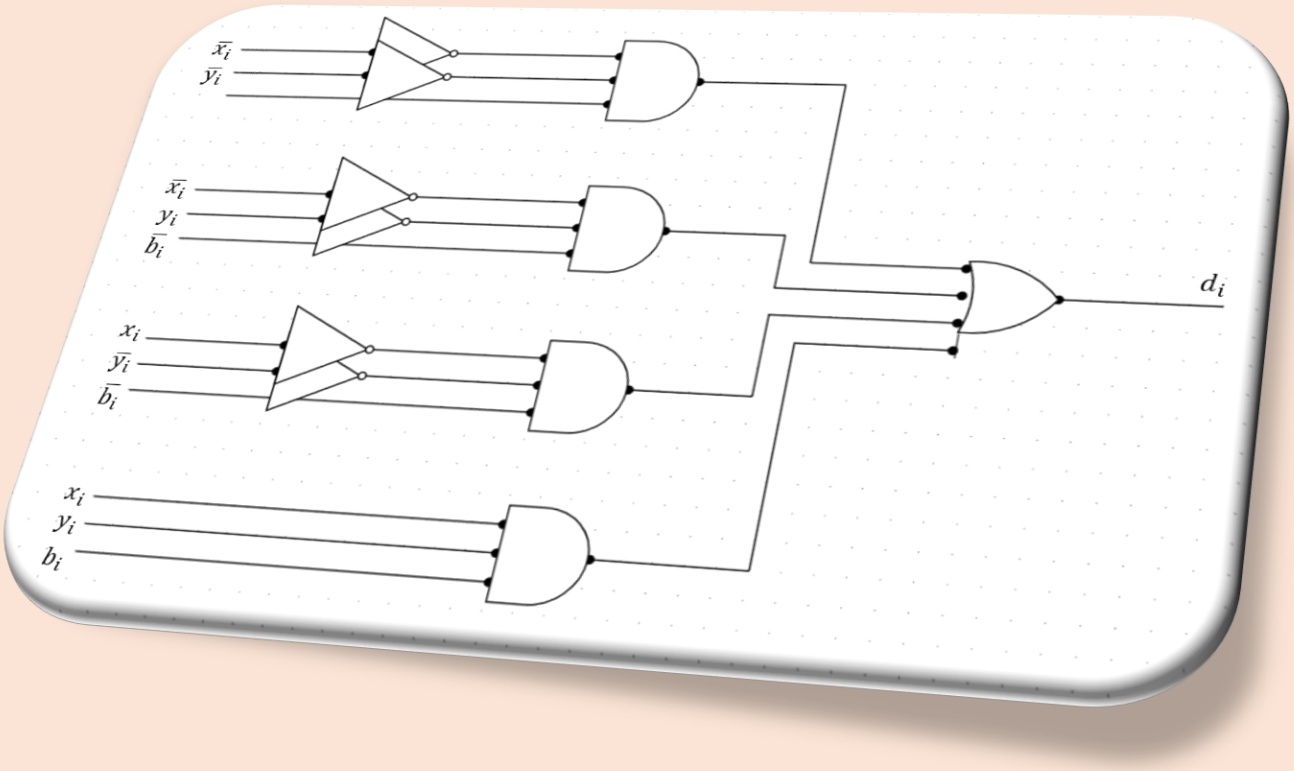
بنابراین برای طراحی گیت این قسمت داریم:



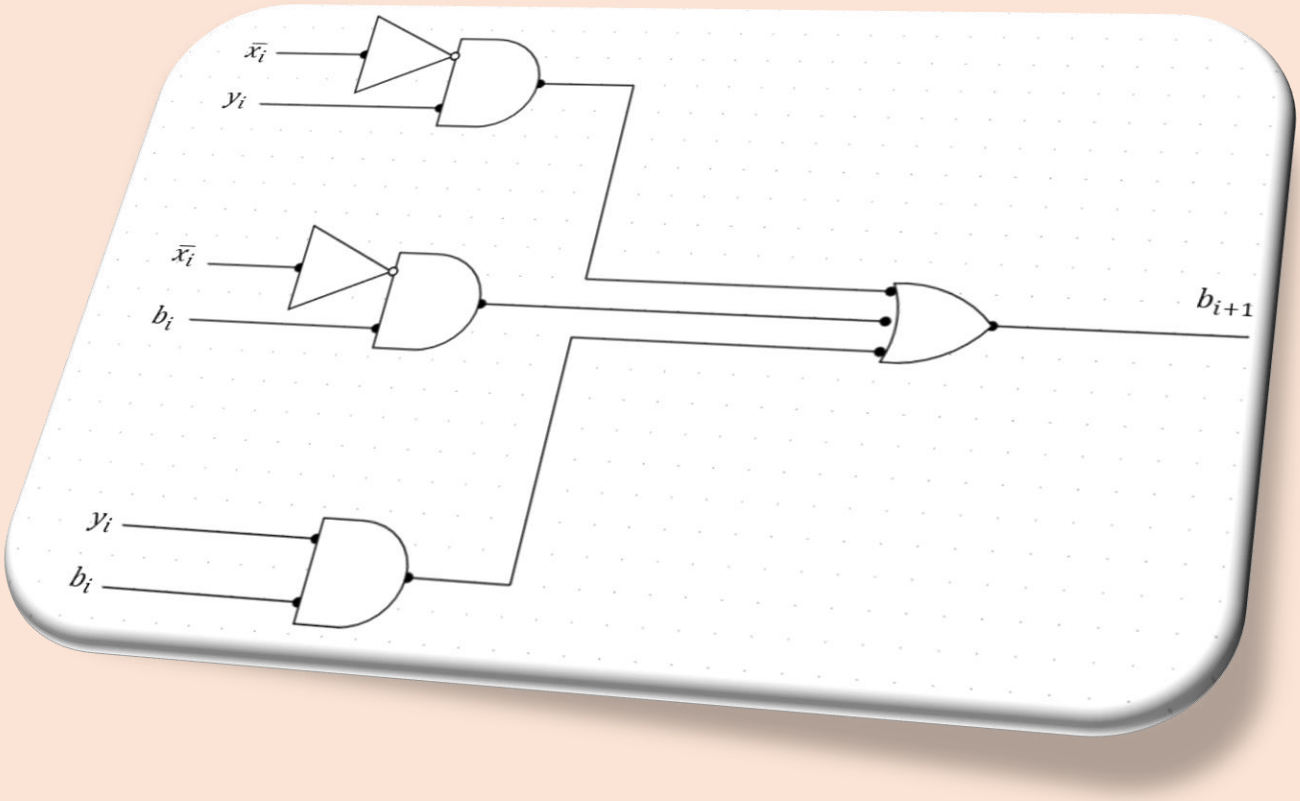
حال اگر بخواهیم هر دو تابع را در یک مدار با نام مدار تفریق کننده ی کامل دو بیتی جمع کنیم، به شکل زیر در سطح گیت ها می رسیم:


subtractor:1
and3b2
x_b_AND_9_o1
or4
d1
and2b1
x_b_AND_7_o1
or3
f1
and3b2
x_b_AND_5_o1
gnd3
x_b_AND_11_o1
and2
y_b_AND_3_o1
and2b1
x_y_AND_6_o1
and2b1
x_b_AND_2_o1
subtractor

در بخش بعد از این سؤال، قرار است با قرار دادن یک مدار پیش بینی رقم نقلی در کنار چندین مدار تفریق کننده یک بیتی بدون محاسبه ی رقم نقلی، به یک مدار تفریق کننده ی n بیتی برسیم. مدار حاصل از تفریق کننده ی یک بیتی بدون محاسبه رقم نقلی به شکل زیر و مشابه آنچه که پیش از این به دست آورده بودیم، می باشد:



برای طراحی مدار پیش بینی بیت قرضی نیز کافی است مدار زیر را گسترش دهیم و برای محاسبه ی بیت قرضی بعدی از بیت قرضی در مرحله ی قبل کمک بگیریم:



بنابراین قسمت پیش بینی بیت قرضی به این صورت است که به تعداد n، از این ساختارها کنار یکدیگر قرار می گیرند و ... نهایتا کل مدار را مطابق با شکل صورت سؤال طراحی کرده و به کد زیر می رسیم:

```
----------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    10:28:29 08/22/2020
-- Design Name:
-- Module Name:    debt - 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 debt is
        generic ( n: integer := 64 );
```

```
        port ( x, y: in std_logic_vector(n-1 downto 0);
                    b: in std_logic;
                    d: out std_logic_vector(n downto 0)
                   );
end debt;

architecture Behavioral of debt is
        component COMP1 is
                port ( x, y, b: in std_logic;
                             d: out std_logic
                            );
        end component;

        component COMP2 is
                port ( x, y, b: in std_logic;
                             d: out std_logic
                            );
        end component;

        for all: COMP1 use entity work.my_subtractor(borrow_bit);
        for all: COMP2 use entity work.my_subtractor(out_bit);
--      for rest: COMP use entity work.my_subtractor(borrow_bit);
--      for fulls: COMP use entity work.my_subtractor(out_bit);
        constant number: integer := 64;
```

```
    signal im: std_logic_vector(number downto 0);
begin

    im(0) <= '0';

    c_1: for i in 0 to number-1 generate
        deb: COMP1 port map ( x(i), y(i), im(i), im(i+1) );
    end generate;

    c_2: for i in 0 to number-1 generate
        ou: COMP2 port map ( x(i), y(i), im(i), d(i) );
    end generate;

    d(number) <= im(number);

end Behavioral;
```

البته دقت شود که از قبل کد تفریق کننده ی دوبیتی با دو معماری مختلف نوشته شده است. یکی از معماری ها که فقط بیت قرضی را محاسبه می کند، برای محاسبه ی قسمت پیش بینی بیت قرضی استفاده شده است و یک معماری نیز فقط مقدار همان بیت را با توجه به دو بیت ورودی و بیت قرضی قبلی که محاسبه می شود، نوشته شده است. کد این تفریق کننده ی کامل یک بیتی نیز به شکل زیر می باشد:

```
----------------------------------------------------------------------------
-- Company:
-- Engineer:
```

```
--
-- Create Date:    10:14:48 08/22/2020
-- Design Name:
-- Module Name:    my_subtractor - 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 my_subtractor is
        port ( x, y, b: in std_logic;
                        d: out std_logic
                        );
end my_subtractor;

architecture out_bit of my_subtractor is
begin
        d <= ( (not x) and (not y) and b ) or ( (not x) and y and (not b) ) or ( x and (not y) and (not b) ) or ( x and y and b );
end out_bit;

architecture borrow_bit of my_subtractor is
begin
        d <= ( (not x) and y ) or ( (not x) and b ) or ( y and b);
end borrow_bit;
```