دانشکده مهندسی کامپیوتر

استاد : خانم دکتر پریا دربانی

سید مهدی رضوی

آبان ۱۴۰۱

فهرست مطالب

۱ مقدمه

همانطور که از آزمایش‌های گذشته به یاد داریم ، برای ساختن قطعات سخت‌افزاری از قطعات کوچکتر استفاده خواهیم کرد و با به هم متصل کردن این مدارها ، مدار مدنظر خود را شکل خواهیم‌داد.
در آزمایش گذشته مدار یک جمع‌کننده چهاربیتی را بررسی کردیم. حال قصدداریم مساله خود را تعمیم داده و مدارجمع کننده برای اعداد با بیش از ۴ بیت را تشکیل دهیم.
برای دستیابی به پاسخ این مساله در ابتدا باید با مفهوم کدهای BCD آشنا شویم .
در مرحله بعد باید به ازای هر ۴ بیت هر دو عدد به مدار جمع‌کننده BCD بدهیم.
سپس رویکردمان مبنی بر این خواهد بود که از همان مدار جمع‌کننده ۴ بیتی استفاده کنیم.

۲ کد BCD

برای تبدیل یک عدد در مبنای ۱۰ به کد BCD می‌بایستی به ازای هر رقم معادل دودویی آن را با ۴ بیت بنویسیم.
به عنوان مثال خواهیم داشت :

$$number0 = 7 \implies BCD(number0) = (0111)_{BCD}$$

$$number1 = 8 \implies BCD(number1) = (1010)_{BCD}$$

$$number2 = 12 \implies BCD(number2) = (0001, 0010)_{BCD}$$

$$number3 = 22 \implies BCD(number3) = (0010, 0010)_{BCD}$$

دو عدد BCD دلخواه را در نظر بگیرید :
۱)نحوه جمع آن دو عدد را به صورت دستی بنویسید :

$$7 + 2 = (0111)_{BCD} + (0010)_{BCD} = (1001)_{Binary} = (1001)_{BCD}$$

همانطور که در بالا نیز مشاهده کردید ، اگر حاصل‌جمع دو عدد کمتر از ۱۰ باشد ، آنگاه خروجی مدار BCD Adder نفاوتی با حالت جمع دو دویی نخواهد داشت.

$$5 + 7 = (0101)_{BCD} + (0111)_{BCD} = (1100)_{Binary} = (0001, 0010)_{BCD}$$

$$(1100)_{Binary} + (0110)_{Binary} = (1, 0010)_{Binary}$$

تفاوت مدار ما با حالت جمع‌کننده دودویی زمانی رخ خواهدداد که حاصل‌جمع دو عدد در مبنای ۱۰ تک‌رقمی نباشد .
زیرا کد کننده BCD به ازای هر رقم در مبنای ۱۰ یک عدد ۴ بیتی تولید خواهد کرد.
در صورتی که مبنای ۲ همچین رویکردی را در پیش نخواهد گرفت.

$$8 + 9 = (1000)_{BCD} + (1001)_{BCD} = (10001)_{Binary} = (0001, 0111)_{BCD}$$

$$(10001)_{Binary} + (0110)_{Binary} = (1, 0111)_{Binary}$$

باز هم مثالی از این حالت را در بالا مشاهده می‌کنیم. تفاوت
۲)نحوه تفریق آن دو عدد را به صورت دستی بنویسید :

$$8 - 5 = (1000)_{BCD} - (0101)_{BCD} = (0011)_{Binary} = (0011)_{BCD}$$

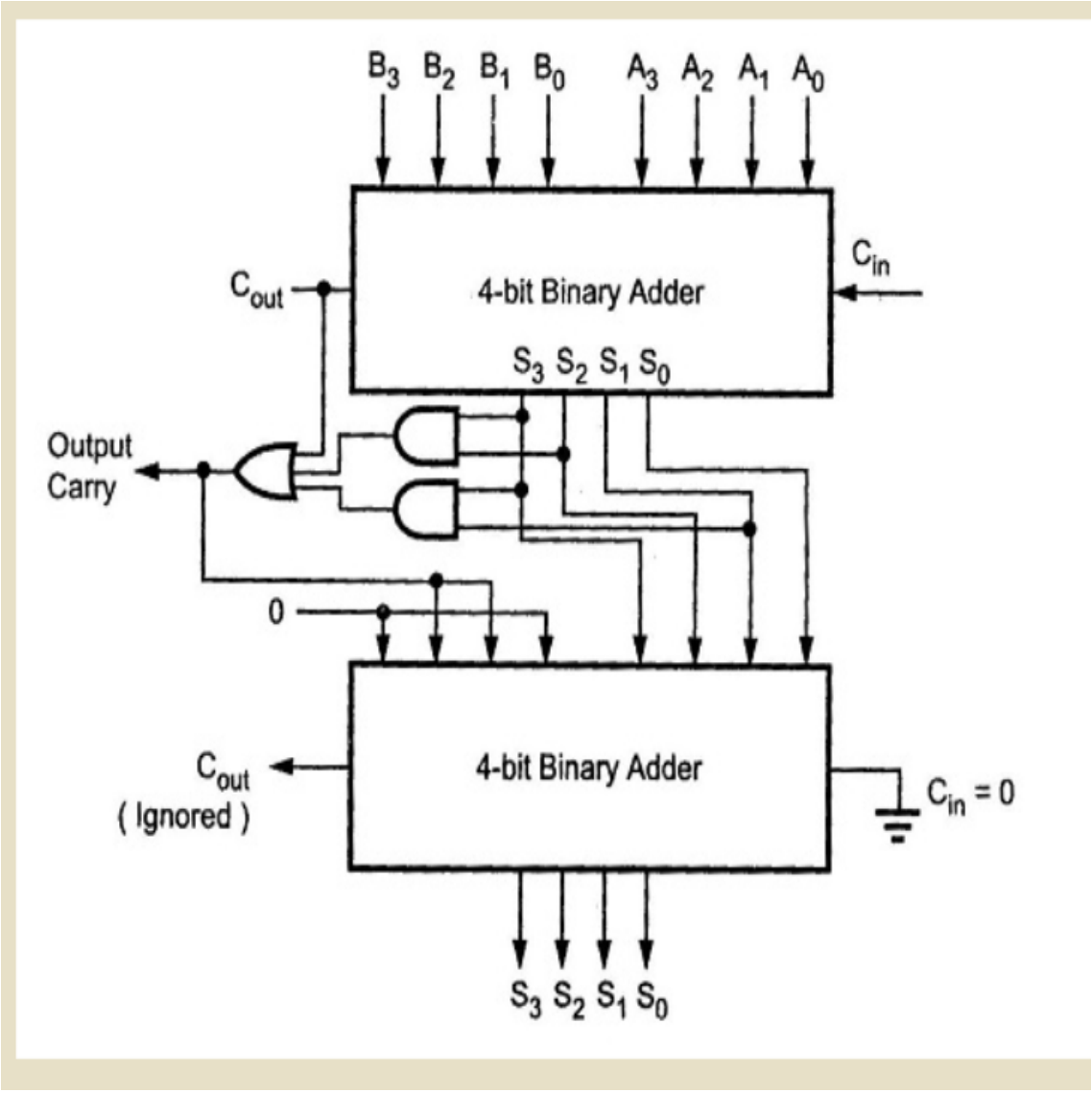
برای این منظور باید ابتدا مکمل ۹ عدد دوم را به دست‌آوریم ، سپس عملیات جمع را با عدد اول انجام‌دهیم.

$$a - b = a + (-b)$$

$$a - b = a + (b'9 + 1)$$

$$9 - 5 = (1001)_{BCD} - (0101)_{BCD}$$

$$(1001)_{BCD} + not\,((0101)_{BCD} + (0110)_{BCD} + 1) = (0100)_{BCD} = (0100)_{BCD}$$



شکل ۱: مدار جمع‌کننده با کد BCD

۳ پیاده‌سازی مدار جمع‌کننده BCD

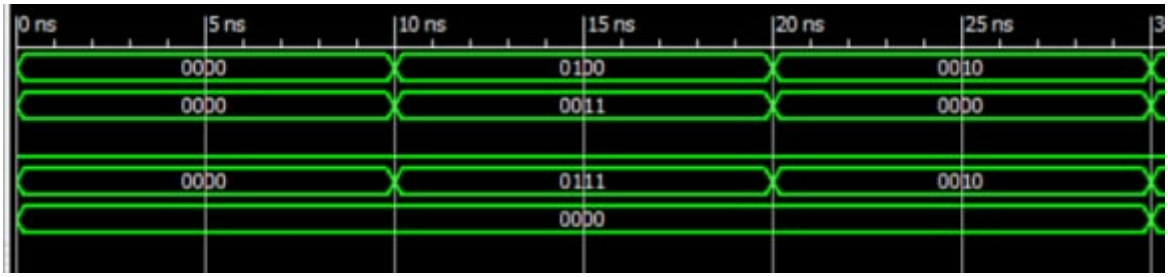
در ادامه آزمایش گذشته ، همانطور که از شکل ۱ مشخص است ، از دو قطعه تمام‌جمع‌کننده استفاده کرده‌ایم.
فقط برای تشخیص اینکه به جمع‌کننده دوم نیاز هست یا خیر ، می‌بایستی یک مدار تشخیص‌دهنده نیز داشته‌باشیم که تشخیص‌دهد که حاصل‌جمع دو ورودی ما بزرگتر از ۱۰ هست یا خیر.
در صورتی که بزرگتر از ۱۰ نباشد، با یک قطعه جمع‌کننده کار ما به پایان می‌رسد ، اما در غیر این صورت بایستی حاصل‌جمع باینری را با عدد (۰۱۱۰) جمع کنیم تا عدد باینری به فرم کد BCD تبدیل شود.

مدار تشخیص‌دهنده به صورت زیر عمل خواهدکرد.
در صورتی عددی ۴ بیتی بزرگتر از ۱۰ است که یکی از ۳ حالت زیر را شامل‌شود :
عدد ۱۰ باشد یا بالاتر (روشن بودن پایه‌های S۳ یا S۱)
عدد ۱۲ باشد یا بالاتر (روشن بودن پایه‌های S۳ یا S۲)
روشن بودن پایه‌ی Carry Out

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


entity GreaterThan10 is
    Port ( S : in STD_LOGIC_VECTOR (3 downto 0);
           cin : in STD_LOGIC;
           output : out STD_LOGIC);
end GreaterThan10;

architecture Behavioral of GreaterThan10 is

begin
   output <= (cin) or (S(3) and S(2)) or (S(3) and S(1));
end Behavioral;
```

همانطورکه درکد نیز واضح است ، خروجی این مدارکه تشخیص دو رقمی بودن این مدار است حاصل ترکیب فصلی سه گزاره مطرح شده بالا خواهد بود.

```
entity BCDAddeer is
Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
b : in STD_LOGIC_VECTOR (3 downto 0);
cin : in STD_LOGIC;
sum_low : out STD_LOGIC_VECTOR (3 downto 0);
sum_high : out STD_LOGIC_VECTOR (3 downto 0));
end BCDAddeer;


architecture Behavioral of BCDAddeer is
component FourBitAdder port(

a : in STD_LOGIC_VECTOR (3 downto 0);

b : in STD_LOGIC_VECTOR (3 downto 0);

carry_in : in STD_LOGIC;

sum : out STD_LOGIC_VECTOR (3 downto 0);

carry_out : out STD_LOGIC);

end component;
```



شکل ۲: شکل موج مدار جمع‌کننده

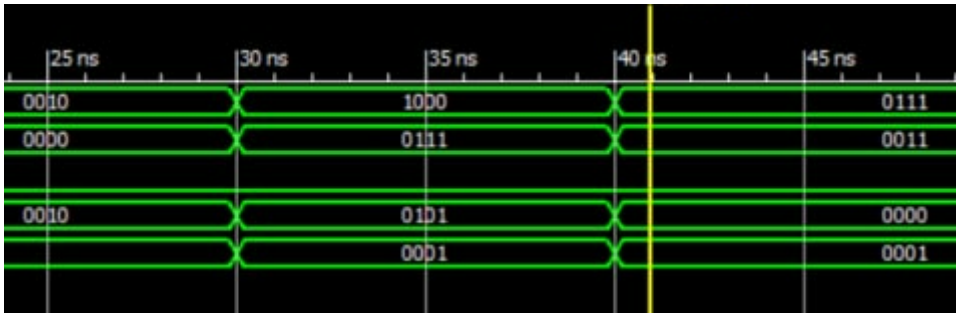
```
begin

BinarySum : FourBitAdder port map(a , b , cin , binary_sum ,binary_sum_carry_out );

Detector : GreaterThan10 port map(binary_sum , binary_sum_carry_out ,
    is_greater_than10);

Concatenation <= '0' & is_greater_than10 & is_greater_than10 & '0';

sum_high(3 downto 0) <= "000" & is_greater_than10;

BCDConvertor : FourBitAdder port map(binary_sum , Concatenation , '0' , sum_low);

end Behavioral;
```

همانطور که از کد شفاف است ، پس از حاصل‌جمع دو عدد ۴ بیتی ، مدار تشخیص دو رقمی‌بودن را استفاده خواهیم کرد. سپس با به هم چسباندن بیت‌های صفر و حاصل مدار تشخیص دو رقمی‌بودن بار دیگر از مدار جمع‌کننده استفاده خواهیم کرد.

’0’ & isGreaterThan10 & isGreaterThan10 & ’0’

اگر بیت حاصل مدار تشخیص دو رقمی بودن ۰ بود حاصل ما تفاوتی با حاصل‌جمع قبل نخواهد داشت. اما اگر ۱ بود ، عدد حاصل جمع دورقمی است و برای تبدیل باینری به کد BCD می‌بایستی عدد باینری را با عدد (۰۱۱۰) جمع کنیم.

همانطور که در شکل موج خروجی ها نیز مشخص است ، اعدادی که حاصل‌جمع آن‌ها بزرگتر از ۱۰ می‌باشد ، دارای کد ۸ بیتی خواهندبود.



شکل ۳: شکل موج مدار جمع‌کننده

۴ نتیجه‌گیری

یکی از بزرگترین اهداف از پیاده‌سازی مدار جمع‌کننده BCD برای جمع کردن اعداد بزرگتر از ۴ بیت می‌باشد.
به عنوان مثال برای جمع کردن دو عدد بزرگ ۱۰۰ بیتی به حداقل ۲۵ مدار نیاز خواهیم داشت.
در واقع به ازای هر ۴ بیت به یک مدار نیاز خواهیم داشت.