گزارش آزمایش 14 آزمایشگاه معماری

ترم بهار 1400

گروه 3

مهلا شریفی 9831035

رها احمدی 9831108

نویسندگان گزارش :

مهلا شریفی

تاریخ تحویل : 1400/3/18

نکته : در جمع و تفریق اعداد ممیز شناور overflow رخ نخواهد داد به همین دلیل بیت carry تعریف نشده است.

توضیح : ممکن است فرضا توان حاصل جمع که چهار بیتی است و نهایتا می تواند +7 باشد ، در 4 بیت جا نشود. مثلا اگر حاصل جمع 1.1111*2^7 با خودش محاسبه شود حاصل بیش از بزگترین عدد قابل نمایش می شود. اما گفته می شود خطا است و نه overflow.

نکته : اگر sel یک عملیات تفریق و در غیراین صورت عملیات جمع صورت می پذیرد.

عملیات تفریق از همان عملیات جمع استفاده میکند با این تفاوت که بیت علامت دومین عدد ورودی را نقیض میکند. زیرا A – B = A + (-B)

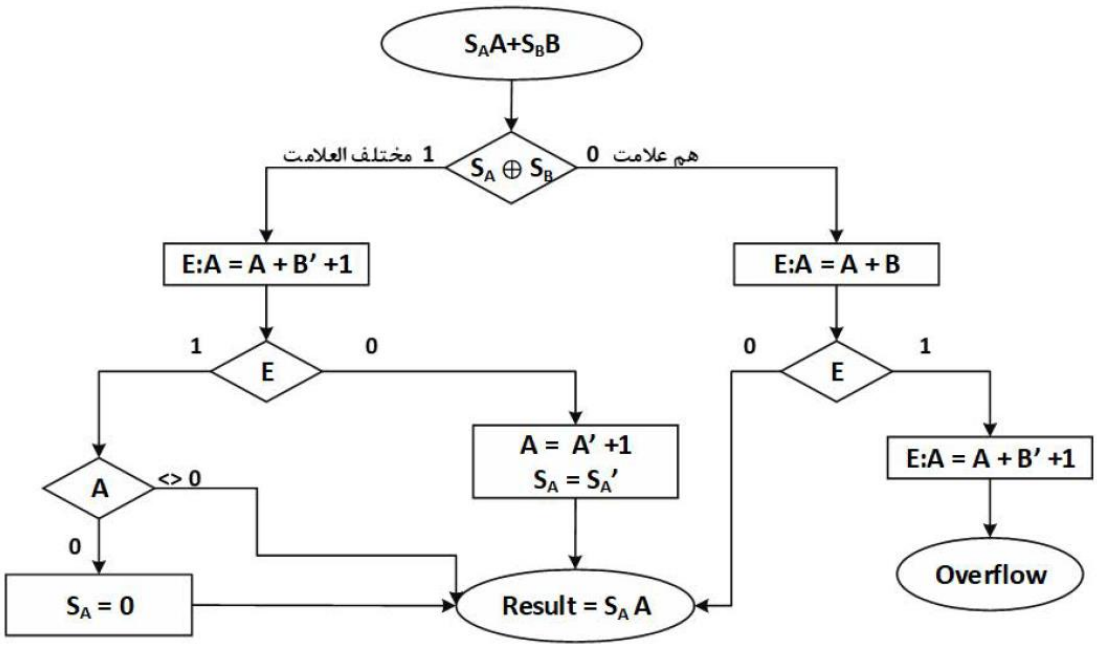
الگوریتم مورد استفاده :

- ۱- چک کردن صفر
  - اگر **A** صفر بود، جواب **B** است، اگر **B** صفر بود، جواب **A** است.
- ۲- ردیف کردن نماها
  - عدد با نمای کوچک به عدد با نمای بزرگ رسانده شود و مانتیس عدد کوچک به اندازه اختلاف نماها شیف به راست پیدا کند
- ۳- مانتیس‌ها با جمع‌کننده اندازه-علامت جمع شوند.
- ۴- چنانچه نتیجه ناهنجار بود، هنجار شود.

نکته : نتیجه ناهنجار یکی از دو حالت overflow و underflow می تواند باشد.

نکته : کد دقیقا بر مبنای الگوریتم بالا و فلوچارت جمع اندازه علامت زیر پیاده سازی شده است.

فلوچارت دو عدد اندازه علامت :



Variable و دلیل استفاده آن در کد :

مزیت استفاده از متغیر به استفاده از سیگنال عدم اتصال تا انتهاستو مقدار پذیری در لحظه است..

فرضا وقتی سیگنال A به سیگنال B اساین می شود. علاوه بر اینکه در لحظه اساین شدن مقدار B در A هم ریخته می شود (بهتر است گفته شود که هردو سیگنال یکی هستند) با تغییر B ، A نیز تغییر خواهد کرد.

مقدار variable به طور آنی تغییر میکند اما سیگنال بسته به اینکه در محیط combinational یا sequential اساین شود به طور متفاوتی مقدار میگیرد.

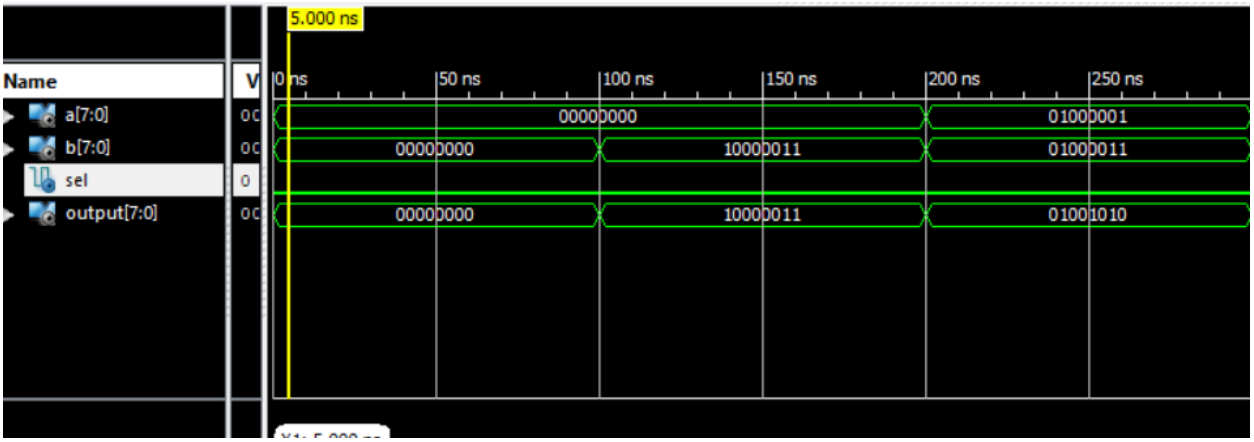
در محیط process و به شکل متوالی سیگنال پس از طی شدن کل یک پراسز مقدار میگیرد. یعنی آخرین مقداری که به آن اساین شده را دریافت میکند ولی اگر در محیط موازی اجرا شود همان لحظه مقدار مربوطه به آن اساین می شود.

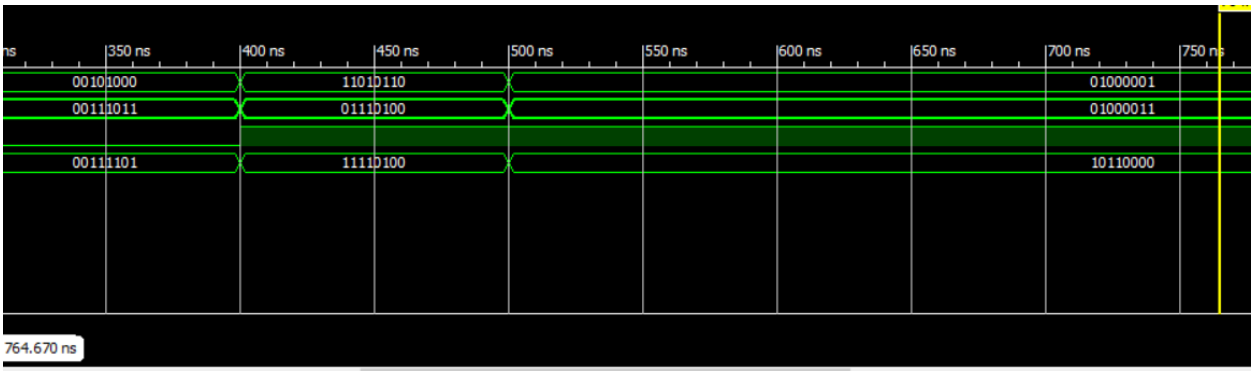
اما متغیر بدین گونه نیست و وقتی مقدار می گیرد یا مقدار جدیدی در آن قرار می گیرد عملا اتصالی به وجود نمی آید و مستقلا می تواند تغییر کند.

Variable تنها در process می تواند تعریف شود . مکان تعریف آن بعد از تعریف process و قبل از کلمه "begin" است.

با استفاده از ":=" می توان مقدار جدیدی به variable اختصاص داد.

نتیجه شبیه سازی :





نتیجه ی مورد انتظار به صورت report در تست بنچ قرار داده شده است.

در زیر تصویری از آن موجود است.

```
52
53      A <= "00000000";     B <= "10000011";     sel <= '0';
54      report "A -> 00000000(0)  -  B -> 10000011(-1.011)  = 10000011 (-1.011)";
55      wait for 100 ns;
56
57      A <= "01000001";     B <= "01000011";     sel <= '0';
58      report "A -> 01000001(1.001 * 2^0)  +  B -> 01000011(1.011 * 2^0)  = 01001010(+1.010 * 2^1)";
59      wait for 100 ns;
60
61      A <= "00101000";     B <= "00111011";     sel <= '0';
62      report "A -> 00101000(1.000 * 2^-3)  +  B -> 00111011(1.011 * 2^-1)  = 00111101 (1.111 * 2^0)";
63      wait for 100 ns;
64
65      A <= "11010110";     B <= "01110100";     sel <= '1';
66      report "A -> 11010110(-1.11 * 2^2)  -  B -> 01110100(1.100 * 2^6)  = 11110100(-1.100 * 2^6)";
67      wait for 100 ns;
68
69
70      A <= "01000001";     B <= "01000011";     sel <= '1';
71      report "A -> 01000001(1.001 * 2^0)  -  B -> 01000011(1.011 * 2^0)  = 1110000(-1.000 * 2^-2)";
72      wait for 100 ns;
73
74      -- insert stimulus here
```

مطالب زیر که به فایل گزارش ضمیمه شده اند صرفا برای مستندسازی از مراحل ، نکات و نتایج خوبی است که در طی انجام گزارش به دست آمده اند. (جزو گزارش نیستند)

پکیج unsigned

```
package STD_LOGIC_UNSIGNED is

    function "+"(L: STD_LOGIC_VECTOR; R: STD_LOGIC_VECTOR) return STD_LOGIC_VECTOR;
```

پکیج numeric

```
-- Id: A.4
function "+" (L, R: SIGNED) return SIGNED;
-- Result subtype: SIGNED(MAX(L'LENGTH, R'LENGTH)-1 downto 0).
-- Result: Adds two SIGNED vectors that may be of different lengths.
```

```
-- Id: S.2
function SHIFT_RIGHT (ARG: UNSIGNED; COUNT: NATURAL) return UNSIGNED;
-- Result subtype: UNSIGNED(ARG'LENGTH-1 downto 0)
-- Result: Performs a shift-right on an UNSIGNED vector COUNT times.
--         The vacated positions are filled with '0'.
--         The COUNT rightmost elements are lost.
```

تبدیل std_logic_vector به unsigned در کتابخانه numeric است.

سینتکس :

unsigned (name of vector)

طبق جمع در هر یک از کتابخانه ها ،هر دوی جمع های زیر درست کار میکنند (تست شده است)

```
variable e : std_logic_vector (7 downto 0);
begin

    e := std_logic_vector (unsigned(A) + unsigned(B));
    e := A + B;
```

عبارت پایین نیز کار میکند (هر سه variable هستند)

```
var := var1 + var2;
var := var1 - var2;
```

Variable نمیتواند مقدار unsigned یا signed را دریافت کند و درصورت نیاز باید signed و unsigned به std_logic cast _vector شود. مثلا کد زیر ارور میگیرد.

```
--A , B = std_logic_vector
    var := unsigned(A);
```

آرگومان دوم شیفت به راست و چپ موجود در کتابخانه numeric (تعداد بیتی که باید شیفت داده شود) می تواند صفر باشد . و عملا خروجی همان ورودی خواهد بود. یعنی :

```
shift_right(unsigned_A, 0)) = ansinged_A;
```

میشد همه مقادیر مورد نیاز متغیر تعریف شوند . به شکل زیر :

```
variable sign_A : STD_LOGIC;
variable sign_B : STD_LOGIC;
variable mantissa_A : STD_LOGIC_VECTOR(3 downto 0);
variable mantissa_B : STD_LOGIC_VECTOR(3 downto 0);
variable exp_A : STD_LOGIC_VECTOR(3 downto 0);
variable exp_B : STD_LOGIC_VECTOR(3 downto 0);
variable exp_out : STD_LOGIC_VECTOR(3 downto 0);
variable mantissa_out : STD_LOGIC_VECTOR(3 downto 0);
variable sign_out : STD_LOGIC;
variable exp_diff : STD_LOGIC_VECTOR(3 downto 0);
variable temp: std_logic_vector(4 downto 0);
```

کد زیر برای درک بهتر مزیت های variable نسبت به سیگنال است.

```
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22  use ieee.std_logic_unsigned.all;
23  use IEEE.NUMERIC_STD.ALL;
24
25
26  entity example_entity is
27     Port ( A : in std_logic_Vector ( 7 downto 0);
28            B : in std_logic_Vector ( 7 downto 0);
29            output : out std_logic_Vector ( 7 downto 0));
30
31  end example_entity;
32  architecture Behavioral of example_entity is
33
34  signal sig : std_logic_vector ( 7 downto 0) := "00000010";
35
36  begin
37
38        process (A, B)
39        begin
40
41           sig <= A;
42           if ( sig(0)  = '1') then
43              sig <= B;
44           else
45              sig <= "11110000";
46           end if;
47
48            output <= sig;
49        end process;
50
51
52  end Behavioral;
```

تست بنچ :

```
42    -- Stimulus process
43    stim_proc: process
44    begin
45       -- hold reset state for 100 ns.
46       A <= "00000000";        B<= "00000000";
47       wait for 100 ns;
48
49      A <= "01110001";         B<= "00000001";
50       wait for 100 ns;
51       -- insert stimulus here
52
53       wait;
54    end process;
```

نتیجه :

توضیحات : با اینکه مقدار A به سیگنال اساین شده است (خط 41 کد) اما چون این اساین شدن انتهای پراسز صورت میگیرد.

Sig (0) = ‘1’ ، false شده است و درنتیجه sig مقدار 11110000 را به خود گرفته است. پس یعنی از فرض اساین شدن در لحظه sig به A نمی توان استفاده کرد و این فرض غلط است.

در صورتی که با استفاده از variable این مشکل حل میشد.

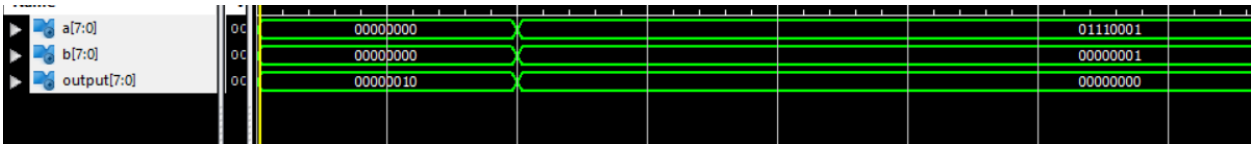
در نتیجه در جاهایی به فرم بالا که قرار است از مقدار اساین شده در ادامه استفاده گردد متغیر به جای سیگنال کارساز خواهد بود.

نکته:

با توضیحات بالا دلیل کد زیر واضح است . سیگنال output مقدار قبلی B را می گیرد. (مشابه استدلال بالا ...)

```
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22  use ieee.std_logic_unsigned.all;
23  use IEEE.NUMERIC_STD.ALL;
24
25
26  entity example_entity is
27     Port ( A : in std_logic_Vector ( 7 downto 0);
28            B : in std_logic_Vector ( 7 downto 0);
29            output : out std_logic_Vector ( 7 downto 0));
30
31  end example_entity;
32  architecture Behavioral of example_entity is
33
34  signal sig : std_logic_vector ( 7 downto 0) := "00000010";
35
36  begin
37
38        process (A, B)
39        begin
40
41           sig <= A;
42           if ( sig(1)  = '1') then
43              sig <= B;
44           else
45              sig <= "11110000";
46           end if;
47
48            output <= sig;
49        end process;
50
51
52  end Behavioral;
53
```

```
43      stim_proc: process
44      begin
45         -- hold reset state for 100 ns.
46         A <= "00000000";         B<= "00000000";
47         wait for 100 ns;
48
49        A <= "01110001";          B<= "00000001";
50         wait for 100 ns;
51         -- insert stimulus here
52
53         wait;
54      end process;
```



Useful links :

Unsigned package contents

Numeric package contents