CSE 331 Computer Organization

Project 2 – ALU with Structural Verilog

Bu projedeki amaç 32 bit ALU tasarlamaktı. 32 bit A ve B sayıları input olarak geliyor ve bunları işlemlere tabi tutarak 32 bir bir Result değeri output veriyoruz.

ALU tasarımında 8 farklı işlemden sadece 1 tanesinin sonucunu output olarak vermemiz gerekiyor. Bunun içinde ALU da 8x1 multiplexer kullanmamız gerekiyor.

Burada yapılacak işlemler şöyledir:

ALU select (S)	Operation
000	R = A AND B
001	R = A OR B
010	R = A + B
011	R = A XOR B
100	R = A – B
101	R = A >> B (arithmetic shift right)
110	R = A << B (shift left)
111	R = A NOR B

3 bit veri seçicilerimiz var. Bunlar da ALU input olarak geliyor. Bunlardaki değere göre yukarıdaki işlemlerden yalnızca bir tanesinin sonucunu output olarak veriyoruz.

000 seçiminde 32 bit A ve B binary sayıları bitwise olarak and işlemini uyguladık.

001 seçiminde 32 bit A ve B binary sayıları bitwise olarak or işlemini uyguladık.

010 seçiminde 32bit A ve B binary sayılarını input olarak veriyoruz. Output olarak 32 bit result verip yanında 1 bit taşma biti çıktısı veriyoruz. Burada full adder kullandım. Full adder tasarlarkende half adder kullandım. Half adder da input olarak 1 bit elde ekleyemediğimiz için full adder kullandım. Bunda 1 bit de input olarak Cin veriyoruz.

011 seçiminde 32 bit A ve B sayılarını xor işlemini uyguluyoruz.

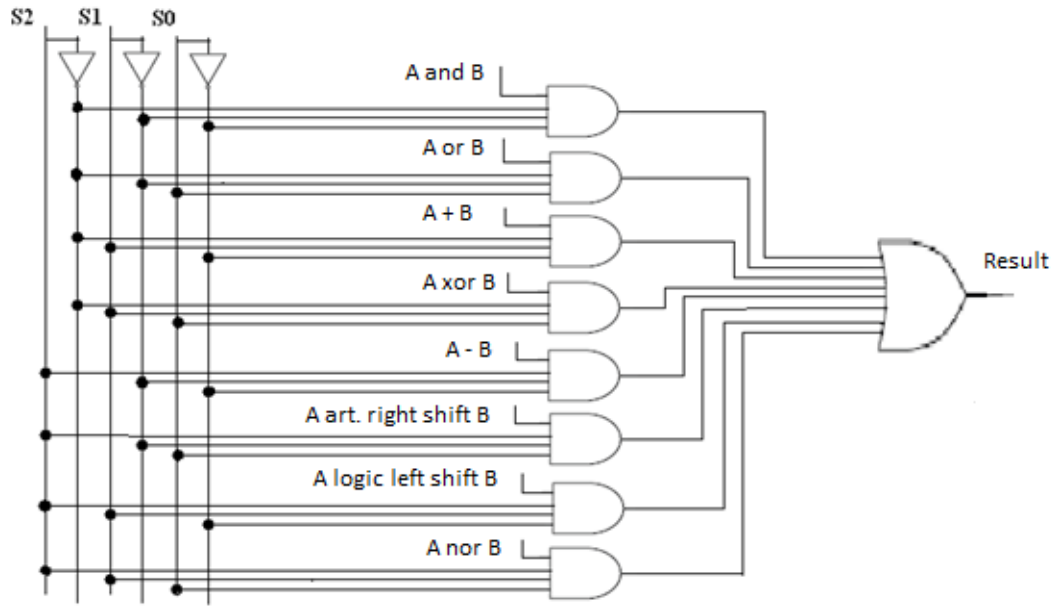
100 seçiminde A ve B sayılarında çıkarma işlemini uyguluyoruz. Burada önemli bir kısım var. Two's complement metoduna göre işlem yapıyorum. B sayısının 32 bit not işlemi uygulayıp. Full adder'a 1 bit elde ile ekliyorum. Full adder burada 1 biti ile twos complement'e çevirmiş oluyor.

101 seçimi arithmetic right shift işlemi. Dersin slatylarında ki gibi bir yol izledim. 2x1 mux kullanarak tasarladım. Aritmetic olduğu için sağa kaydırdıktan sonra sol tarafa gelen sayının en soldaki most sign bitini boşalan kısımlara ekliyoruz.

110 seçimi logic left shift işlemi. Bu işlem içinde right shift işleminin benzeri bir yol izledim. Logic olduğu için sola kaydırdıktan sonra sağ tarafa 0 yerleştiriyoruz.

111 işleminde A ve B ye nor işlemini uyguluyoruz.

ALU da A ve B nin tüm işlemlerinin sonuçlarını öncelikle buluyoruz. Sonra bu bulduğumuz sonuçları gelen veri seçiciler ile AND işlemi uyguluyoruz. Bundaki amaç bize ihtiyacımız olan sonuç elimizde kalsın. Diğerleri sıfırlansın. Aşağıdaki şemadaki gibi tüm işlemlerin sonuçlarını and'ledikten sonra bulduğumuz sonuçlardan sadece 1 tanesi bizim ihtiyacımız olan sonuç. Ondan dolayı da tüm sonuçları OR layıp direk Result'a output olarak veriyoruz. Doğru sonucu çıktı olarak vermiş oluyoruz.



Result = [((A and B).(~s2).(~s1).(~s0)) + ((A or B).(~s2).(~s1).s0) + ((A + B).(~s2).s1.(~s0)) + ((A xor B).(~s2).s1.s0) + ((A - B).s2.(~s1).(~s0)) + ((A ars B).s2.(~s1).s0) + ((A lls B).s2.s1.(~s0)) + ((A nor B).s2.s1.s0)]

3 farklı 32 bit A ve B sayılarını tüm işlemler için test sonuçlarını aşağıdaki ekran görüntüsünde görebilirsiniz.

```
VSIM 5> step -current
#  a = 10101010101010101010101010101010,  b = 11001100110011001100110011001100,  select = 000,  result = 10001000100010001000100010001000,  zeroBit = 0,  overflow = 0
#  a = 11111111111111110000000000000000,  b = 00000000000000001111111111111111,  select = 000,  result = 00000000000000000000000000000000,  zeroBit = 1,  overflow = 0
#  a = 10101011110011011110111100010010,  b = 00000000000000001111111111111111,  select = 000,  result = 00000000000000001110111100010010,  zeroBit = 0,  overflow = 0
#  a = 10101010101010101010101010101010,  b = 11001100110011001100110011001100,  select = 001,  result = 11101110111011101110111011101110,  zeroBit = 0,  overflow = 0
#  a = 11111111111111110000000000000000,  b = 00000000000000001111111111111111,  select = 001,  result = 11111111111111111111111111111111,  zeroBit = 0,  overflow = 0
#  a = 10101011110011011110111100010010,  b = 00000000000000001111111111111111,  select = 001,  result = 10101011110011011111111111111111,  zeroBit = 0,  overflow = 0
#  a = 11111111111111110000000000000000,  b = 11001100110011001100110011001100,  select = 010,  result = 11001100110011001100110011001100,  zeroBit = 0,  overflow = 0
#  a = 11111111111111110000000000000000,  b = 00000000000000001111111111111111,  select = 010,  result = 11111111111111111111111111111111,  zeroBit = 0,  overflow = 0
#  a = 10101011110011011110111100010010,  b = 00000000000000001111111111111111,  select = 010,  result = 10101011110011011110111100010001,  zeroBit = 0,  overflow = 0
#  a = 10101010101010101010101010101010,  b = 11001100110011001100110011001100,  select = 011,  result = 01100110011001100110011001100110,  zeroBit = 0,  overflow = 1
#  a = 11111111111111110000000000000000,  b = 00000000000000001111111111111111,  select = 011,  result = 11111111111111111111111111111111,  zeroBit = 0,  overflow = 0
#  a = 10101011110011011110111100010010,  b = 00000000000000001111111111111111,  select = 011,  result = 10101011110011010001000011101101,  zeroBit = 0,  overflow = 0
#  a = 10101010101010101010101010101010,  b = 11001100110011001100110011001100,  select = 100,  result = 11111111111111111111111111111111,  zeroBit = 0,  overflow = 0
#  a = 11111111111111110000000000000000,  b = 00000000000000001111111111111111,  select = 100,  result = 00000000000000000000000000000000,  zeroBit = 1,  overflow = 0
#  a = 10101011110011011110111100010010,  b = 00000000000000001111111111111111,  select = 100,  result = 00000000000000000000000000000000,  zeroBit = 1,  overflow = 0
#  a = 10101010101010101010101010101010,  b = 11001100110011001100110011001100,  select = 101,  result = 11111111111110101010101010101010,  zeroBit = 0,  overflow = 0
#  a = 11111111111111110000000000000000,  b = 00000000000000001111111111111111,  select = 101,  result = 11111111111111111111111111111111,  zeroBit = 0,  overflow = 0
#  a = 10101011110011011110111100010010,  b = 00000000000000001111111111111111,  select = 101,  result = 11111111111111111111111111111111,  zeroBit = 0,  overflow = 0
#  a = 10101010101010101010101010101010,  b = 11001100110011001100110011001100,  select = 110,  result = 10101010101010101000000000000000,  zeroBit = 0,  overflow = 0
#  a = 11111111111111110000000000000000,  b = 00000000000000001111111111111111,  select = 110,  result = 00000000000000000000000000000000,  zeroBit = 1,  overflow = 0
#  a = 10101011110011011110111100010010,  b = 00000000000000001111111111111111,  select = 110,  result = 00000000000000000000000000000000,  zeroBit = 1,  overflow = 0
#  a = 10101010101010101010101010101010,  b = 11001100110011001100110011001100,  select = 111,  result = 00010001000100010001000100010001,  zeroBit = 0,  overflow = 1
#  a = 11111111111111110000000000000000,  b = 00000000000000001111111111111111,  select = 111,  result = 00000000000000000000000000000000,  zeroBit = 1,  overflow = 0
#  a = 10101011110011011110111100010010,  b = 00000000000000001111111111111111,  select = 111,  result = 01010100001100100000000000000000,  zeroBit = 0,  overflow = 0

VSIM 6>
```


Ahmet Yuşa Telli

151044092