Unsigned Multiply

A picture containing text

Description automatically generated

Diagram

Description automatically generated

Unsigned shift-add multiplier (version 1)

64-bit multiplicand reg, 64-bit ALU, 64-bit product reg, 32-bit multiplier reg

Diagram

Description automatically generated

Multiply Algorith Version 1

Diagram

Description automatically generated

Test LSB of multiplier

If it is 1 add multiplicand to product and put result to product register

If it is 0 performs no addition just shift multiplicand left 1 bit at each iteration

Multiply Hardware Version 2

32-bit multiplicand reg, 32-bit ALU, 64-bit product reg, 32-bit multiplier reg

Diagram

Description automatically generated with low confidence

Instead of shifting multiplicand to left, we can shift product to right.

We will only add most significant 32 bits. So 32-bit ALU will be enough.

Initially product is all 0s. Addition is always done with the most significant bits. So at the beginning we can put multiplier to least significant bits of product. While shifting the product right by 1 bit, we both shift the product and also the multiplier. We don’t need another right shift multiplier register.

Multiply Algorithm Version 2

Diagram

Description automatically generated

Multiply Hardware Version 3

32-bit multiplicand reg, 32-bit ALU, 64-bit product reg, (0-bit multiplier reg)

Diagram

Description automatically generated

Multiply Algorithm Version 3

Diagram

Description automatically generated

Shifters

Diagram, text

Description automatically generated

Note: these are single bit shifts. A given instruction might request 0 to 32 bits to be shifted.

Shifting left has only logical shift. There is no meaning for arithmetic. We have to apply these 3:

* srl
* sra
* sll

Combinational Shifter from MUXes

Diagram, shape

Description automatically generated

Boxes are multiplexers.

srl

* From left hand side, inputs are 0.
* First level is responsible for shifting by 1, second level by 2, third level by 4.
* For example if we want to shift number by 3, we put s2s1s0 to 011 so first level shift by 1 and second level will shift that shifted number by 2.
* Example shift right logical by 5:

Diagram

Description automatically generated with medium confidence

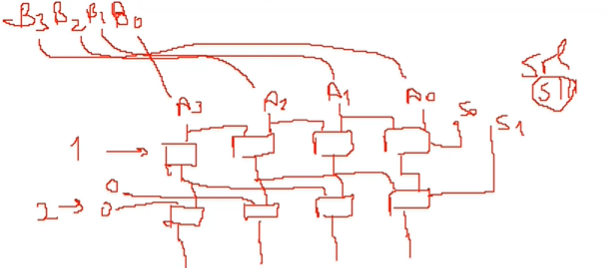


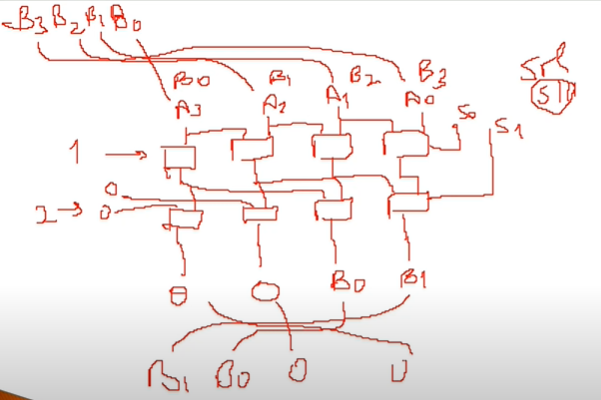
How many levels for 32-bit shifter?

2x = 32 🡪 x = 5 so we need 5 levels.  
 by 1 – by 2 – by 4 – by 8 – by 16

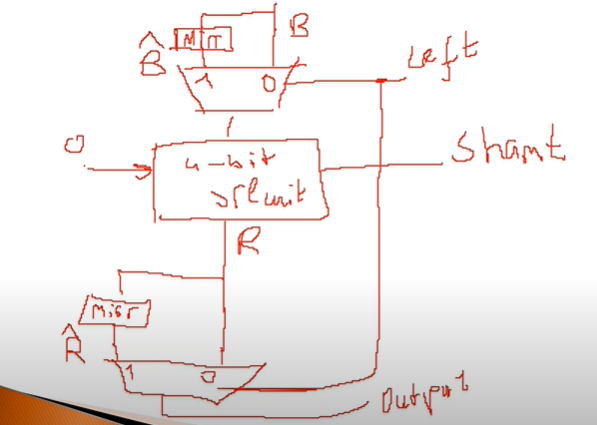
We have 32 multiplexers for each level. 🡪 5x32 = 160 mux

You can shift a number left by using right shifter.

If I take reverse of the number and apply shift right, we actually shift left.



We shift B left logical by 2.

mirr:mirror

MUXlara doğru sinyalleri yollamak önemli. Örneğin burada left sinyalini control unit ayarlar.