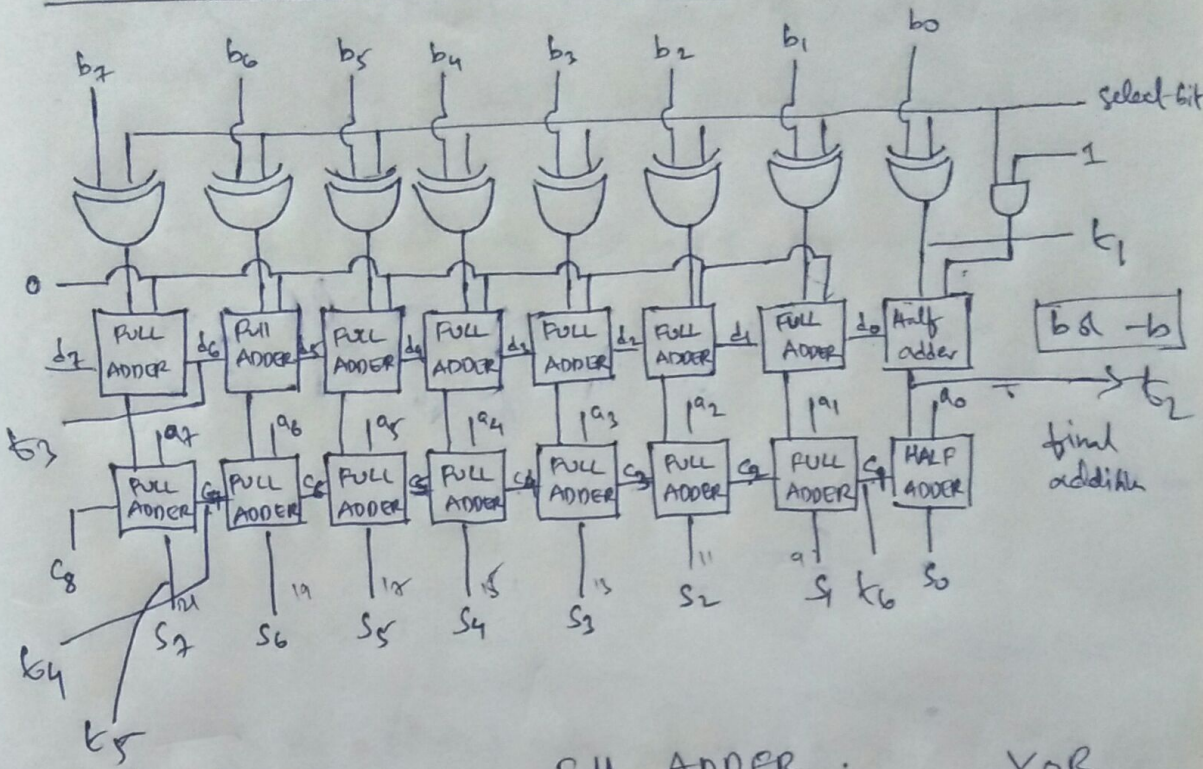
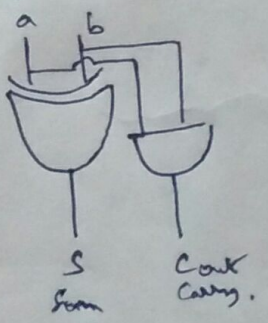
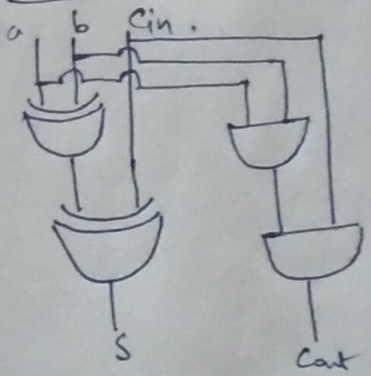
8-bit Adder-Subtractor Circuit



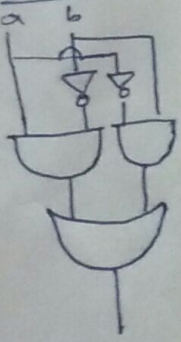
Half ADDER



Full ADDER.



XOR



Circuit Complexity:

38 - XOR gates, 31 - AND gates, treating XOR as 2-input gate.

Otherwise.

((38 x 2) + 31) - AND gates and 38 - OR gates

ie 107 - AND gates and 38 - OR gates.

Time Delay: (Assume unit time for AND gate and OR gate).

So an Half adder takes 2 units of time to output carry bit and 2 units of time to output sum bit.

while for a Full adder takes 2 units of time to output carry bit and 4 units of time to output sum bit.

So total time delay

$t_1 = 2$ units.

$t_2 = (2+2) = 4$ units.

$t_3 = 15$ units.

~~$t_4 = 17$ units.~~

$t_6 = 5$ unit.

$t_4 = 17$ unit

$t_5 = 21$ units.

So total time will be equal to $t_5 = 21$ units.

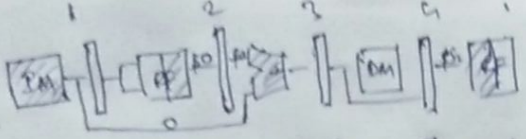
3. Give the circuit idea for adding all the elements of an array.

```
/* sum of elements of an array */
  int i;
int A[3] = {a1, a2, a3};
  int sum = 0;
  for (i=0; i<3; ++i) {
      sum = sum + A[i];
    }
```

MIPS code:

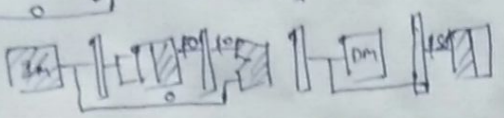
```
# $s0 = i, $s1 = base address of A, $s2 = sum
    addi  $s2, $0, 0      # sum = 0.
    addi  $s0, $0, 0      # i = 0.
    addi  $t0, $0, 3      # $t0 = 3
loop: slt   $t1, $s0, $t0 # if (i<3) $t1 = 1, else $t1 = 0.
    beq   $t1, $0, done   # if $t1 = 0, branch done.
    lw    $t2, ($s1)
    add   $s2, $s2, $t2
    addi  $s1, $s1, 4
    addi  $s0, $s0, 1
    j     loop
done:
```

addi $s2, $0, 0



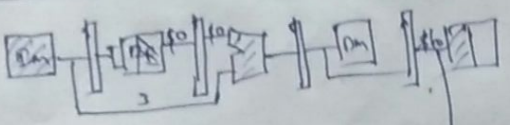
$s2 = 0.

addi $s0, $0, 0



i = 0.

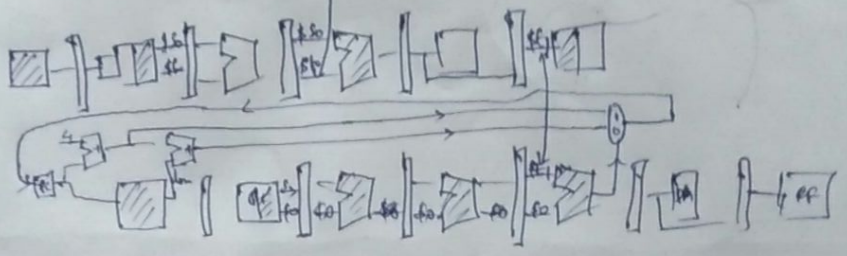
addi $t0, $0, 3



$t0 = 3

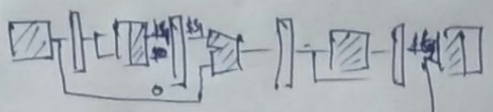
Loop:

slt $t1, $s0, $t0

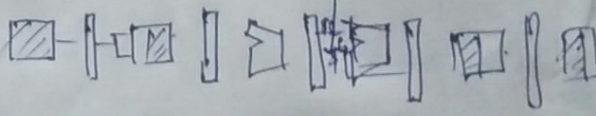


beq $t1, $0, done

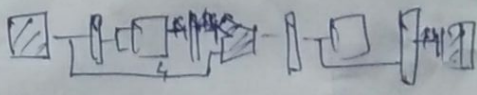
lw $t2, ($s1)



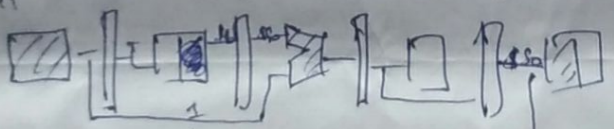
add $s2, $s2, $t2



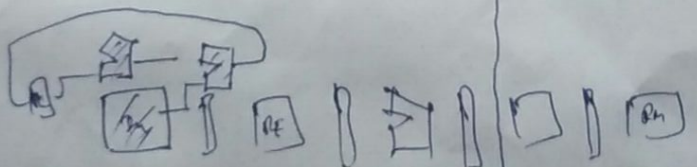
addi $s1, $s1, 4



addi $s0, $s0, 1



j loop



Now again repeat loop. 2 new things. until beq when $t1 = 0, that jumps to the address done.

slt $t1, $s0, $t0