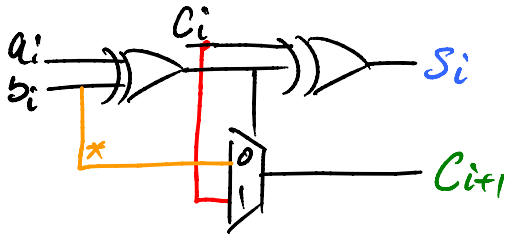
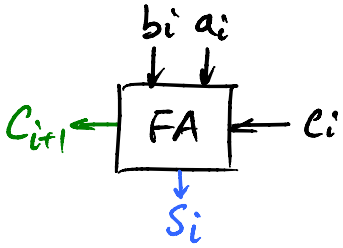
Arithmetic Circuits (Adder, Subtractor)

Recall full adder (FA)





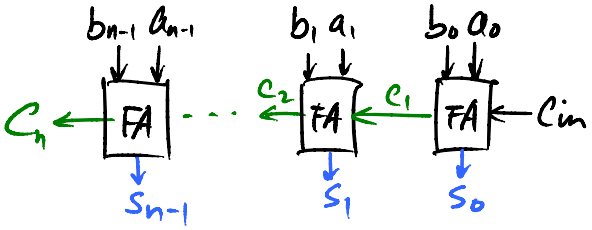
C_i b_i a_i	C_{i+1}	S_i
0 0 0	0	0
0 0 1	0	1
0 1 0	0	1
0 1 1	1	0
1 0 0	0	1
1 0 1	1	0
1 1 0	1	0
1 1 1	1	1

* using a_i would have the same effect. because this value goes through only when both a_i and b_i have the same value

Verilog for a FA

```
module FA (input Cin, a, b, output Cout, S);
    assign S = Cin^a^b;
    assign Cout = (a==b)? b : Cin;
endmodule
```

conditional assignment



Verilog code for n bit adder using a "for" loop

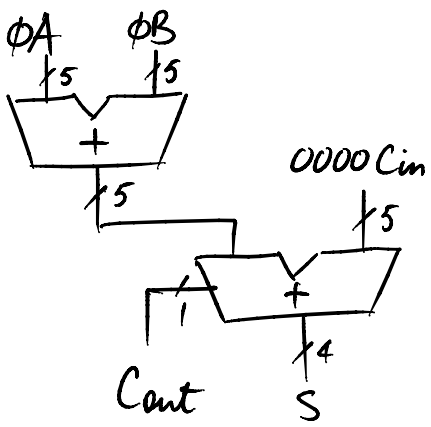
```
Module addn (Cin, A, B, Cout, S);
 parameter n=4;
 input Cin;
 input [n-1:0] A, B;
 output reg Cout;
 output reg [n-1:0] S;
 reg [0:n] C;
 always@ (*);
 begin
    C[0] = Cin; // initialization
    for (k=0; k<n; k=k+1)
    begin
      S[k] = A[k]^B[k]^C[k];
      C[k+1] = (A[k]==B[k])? A[k] : C[k];
    end
```

Using the "+" operator

```
    .
    .
    .
for (k=0; k<n; k=k+1)
  {C[k+1], S[k]} = A[k]+B[k]+C[k];
    .
    .
    .
```

```
  A(k)
  B(k)
+ C(k)
-------
C[k+1] S[k]
```

```
  Cout = C[n];
end
endmodule
```




```
module Addn ( ... );
  parameter n=4;
  :
always@(*)
  {Cout, S} = A+B+Cin;
endmodule
```

Signed Numbers (2's complement)

to add $5+1=6$

$5+(-1)=4$

$5+(1111)=4$ } $-1 \leftarrow 1111$

$5+(-2)=3$

$5+(1110)=3$ } $-2 \leftarrow 1110$

4-bit binary

0	0 0 0 0	
+1	0 0 0 1	
+2	0 0 1 0	+14
+3	0 0 1 1	+15
+4	0 1 0 0	
+5	0 1 0 1	
+6	0 1 1 0	
+7	0 1 1 1	
−8	1 0 0 0	
−7	1 0 0 1	
−6	1 0 1 0	
−5	1 0 1 1	
−4	1 1 0 0	
−3	1 1 0 1	
−2	1 1 1 0	
−1	1 1 1 1	

for 4 bits.
to represent $-k$ we can do

$2^4 - k$

$2^4 = (16)_{10}$
$\quad - 1$
$\quad 15$

eg. $-k = -1$

$2^4 = (10000)_2$
$\quad - 1$
$\quad 1111$

for n bit. $-k \Leftarrow 2^n - k$

another way $\Leftarrow [(2^n - 1) - k] + 1$

$(2^n - 1)$: $1\ 1\ \cdots\ 1$
$-\ k_{n-1}\ k_{n-2}\ \cdots\ k_0$
$\bar{k}_{n-1}\ \cdots\ \bar{k}_0$ $(\bar{k})$
$+\ 1$
2's comp. of $-k$

eg. n=4, we try to find -1

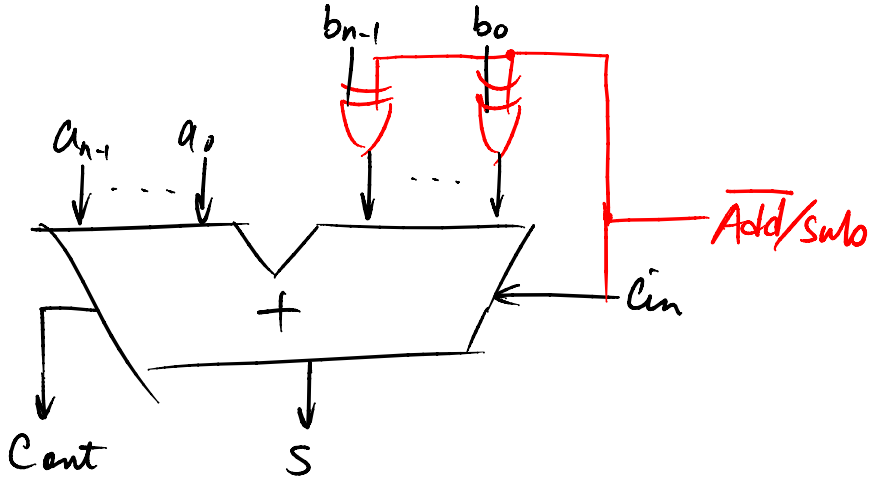
$2^n - 1 \rightarrow$ 1 1 1 1 $(2^4 - 1 = 15)$

− 0 0 0 1 (1)

1 1 1 0 ← complement of 1

\+ 1

1 1 1 1 (2's complement of -1)



when $\overline{Add}/sub = 0$
it is A+B
when $\overline{Add}/sub = 1$
it is A−B

To find 2's complement of k, ① complement $k \rightarrow \bar{k}$
② then do $\bar{k} + 1$

eg. 4 bit system, if we want -6,

① 0110 (+6)
1001 (comp)

② \+ 1

1010 (−6 in 2's comp)

Shortest cut to find the 2's complement

eg. 4 bit signed number

0110 (4 bit +6)

(2's comp. of −6) B: 1010

complement the remaining bits

copy down all "0s" and the 1st "1" you see from the right hand side (i.e. starting from the lowest bit)

Multiplication

(for +ve #s)

$(22)_{10} \times (13)_{10}$

66

+22

$(286)_{10}$

12 × 11

12

+12

132

1100 × 1011

1100

1100

0000

+) 1100

10000100

$A = a_3 a_2 a_1 a_0$

$\times B = b_3 b_2 b_1 b_0$

$P = P_7 P_6 P_5 P_4 P_3 P_2 P_1 P_0$

$a_3\ a_2\ a_1\ a_0$

$\times\ b_3\ b_2\ b_1\ b_0$

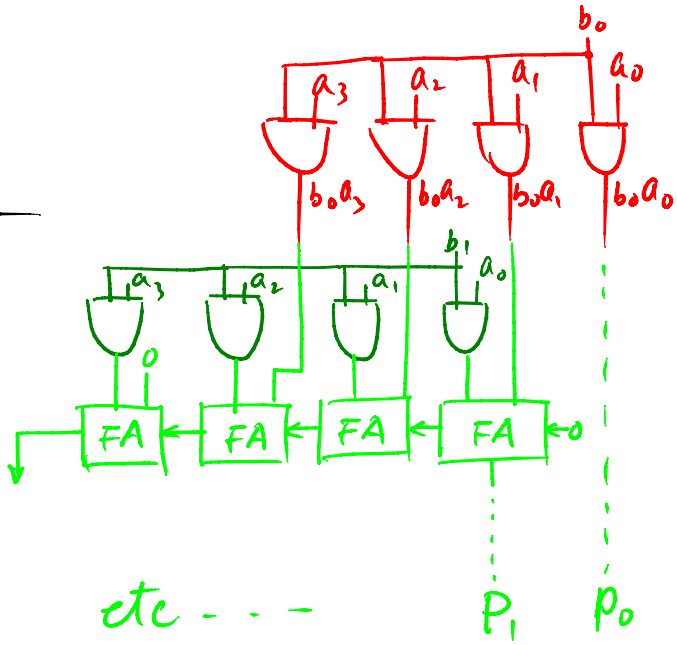
$b_0a_3\ b_0a_2\ b_0a_1\ b_0a_0$

$b_1a_3\ b_1a_2\ b_1a_1\ b_1a_0$

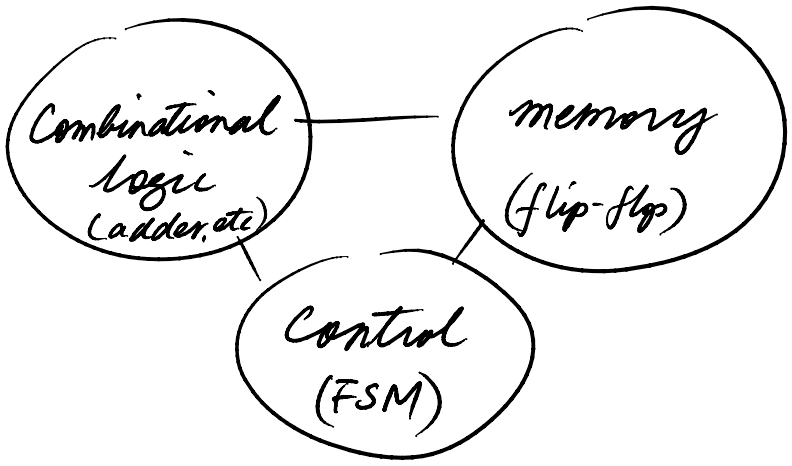
$b_2a_3\ b_2a_2\ b_2a_1\ b_2a_0$

+) $b_3a_3\ b_3a_2\ b_3a_1\ b_3a_0$

$P_7\ P_6\ P_5\ P_4\ P_3\ P_2\ P_1\ P_0$



Finite State Machine (FSM)



Digital Systems