# Tutorial Arithmetic

Shreyas Dodamani
19114079

1) $(-5) \times (-7)$

Iterative: Convert $-5$ and $-7$ to positive numbers, multiply them, the negative if inital signs contrast.

$5 \times 7$:

(initially)

| C | A | Q (5) | M (7) | |
|---|---|---|---|---|
| 0 | 0000 | 0101 | 0111 | Initial values |
| 0 | 0111 | 0101 | 0111 | A=A+M } (1) |
| 0 | 0011 | 1010 | 0111 | Shift |
| 0 | 0001 | 1101 | 0111 | shift } (2) |
| 0 | 1000 | 1101 | 0111 | A=A+M } (3) |
| 0 | 0100 | 0110 | 0111 | Shift |
| 0 | 0010 | 0011 | 0111 | Shift } (4) |

AQ : 00100011 = 35 = 5x7

Since signs of $-5$ & $-7$ match, don't negative

$-5 \times -7 = 35$

BOOTH'S :

| A | Q ($-5$) | $Q_{-1}$ | M ($-7$) | | |
|---|---|---|---|---|---|
| 0000 | 1011 | 0 | 1001 | Initial values | |
| 0111 | 1011 | 0 | 1001 | A=A+(-M) | $Q_0 Q_{-1} = 10$ |
| 0011 | 1101 | 1 | 1001 | Shift | |
| 0001 | 1110 | 1 | 1001 | shift | $Q_0 Q_{-1} = 11$ |
| 1010 | 1110 | 1 | 1001 | A=A+M | $Q_0 Q_{-1} = 01$ |
| 1101 | 0111 | 0 | 1001 | Shift | |
| 0100 | 0111 | 0 | 1001 | A=A−M=A+(−M) | |
| 0010 | 0011 | 1 | 1001 | Shift | $Q_0 Q_{-1} = 10$ |

$\Rightarrow$ ANSWER = AQ = 00100011 = 35

$(-5) \times (-7) = 35$

2) Multiplicand: 0101 1010 1110 1110
Multiplier: 0111 0111 1011 1101

Multiplier

$Q_{-1}$

0 1 1 1 0 1 1 1 1 0 1 1 1 1 0 1 0

\+ - + - + - + -

Every time we encounter 01: ADD
10: SUBTRACT

⇒ 4 ADD operations
4 SUBTRACT operations

8 TOTAL

3) (A) 101010...1010 ⇒ we must perform and add or subtract operation on every step. b/c every step has either 10 or 01

calculations done while ignoring initial 0 value of $Q_{-1}$ b/c it adds to each option by same amount

(B) 1000...001 ⇒ one add and one subtract operation at ~~first~~ initial cycle & final cycle.

\- +

(C) 111....111 ⇒ NO ADD/SUBTRACT operations b/c always 11

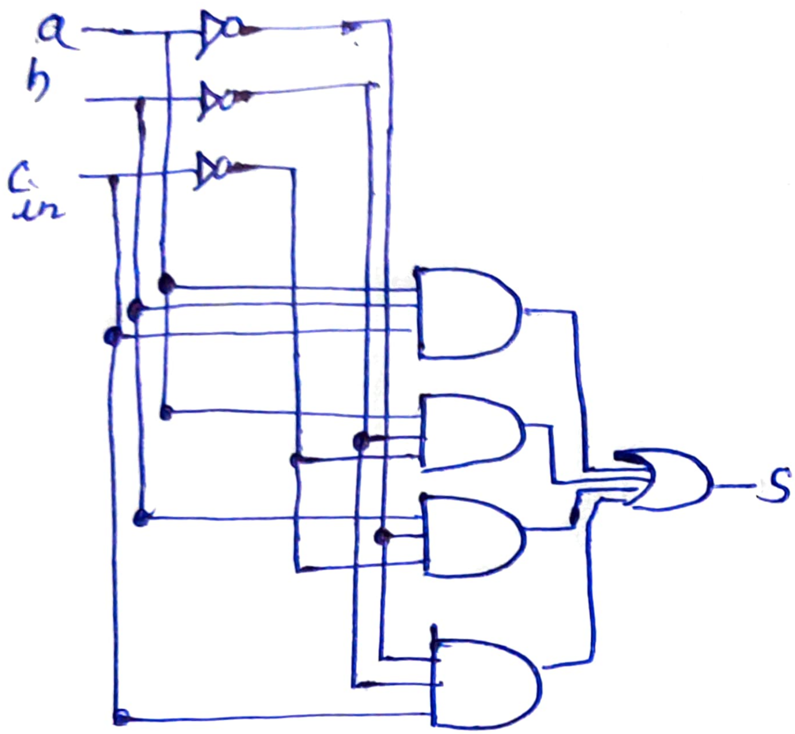
(D)

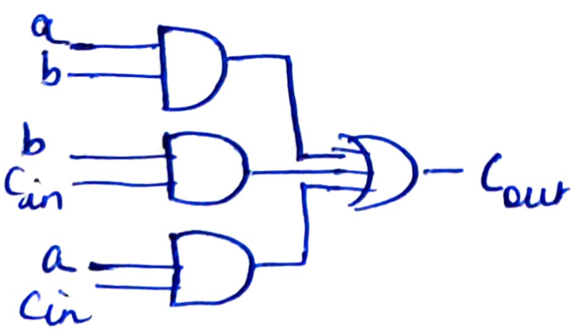
01111...1110 ⇒ one subtract operation in beginning and one at end.

⇒ 2 OPERATIONS

SINCE (A) is most computationally expensive, Booth's algorithm has Least performance

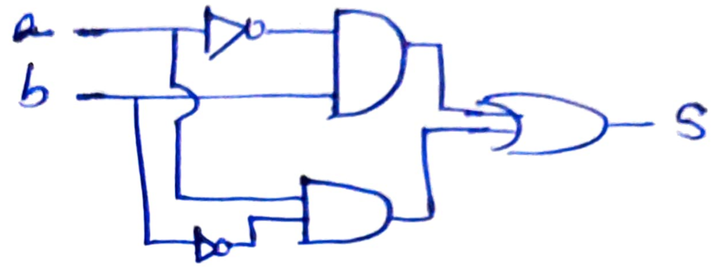
## 4) Full Adder




⇒ 7AND
2OR

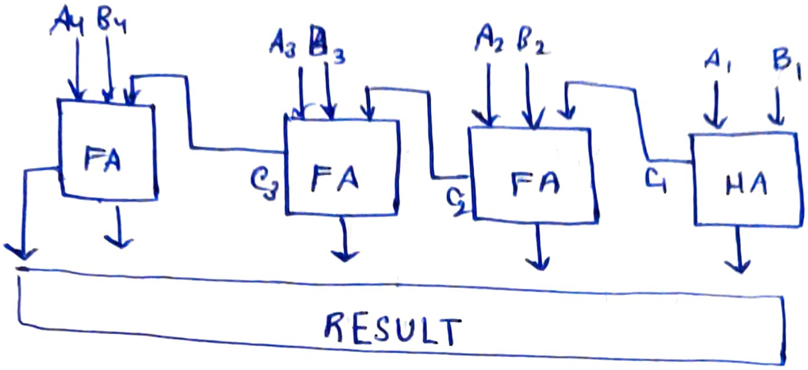
## Half Adder





⇒ 3AND
1OR

# Ripple carry Adder




We require 3 full adders and 1 Half adder

In 1 full adder, we have
7 AND gates
2 OR gates

$\Rightarrow$ 3(7) AND GATES + 3(2) OR GATES = 21 and gates & 6 OR gates

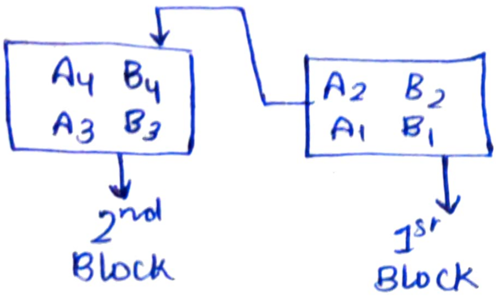
In 1 half adder we have

3 AND gates
1 OR gate

total: (3 + 21) = 24 AND gates
(6 + 1) = 7 OR gates

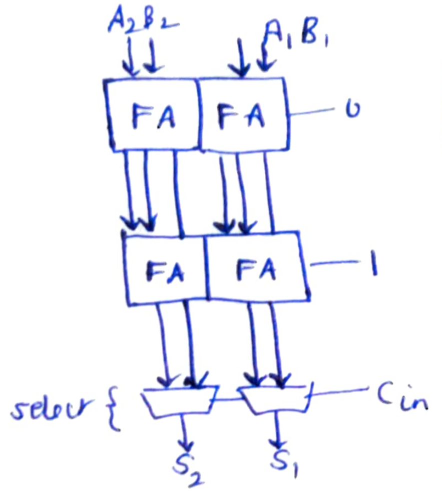
# Carry select adder.

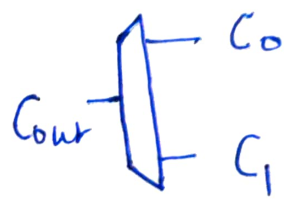
given $n=4 \Rightarrow k=\sqrt{4}=2$




$A_4 A_3 A_2 A_1$
$B_4 B_3 B_2 B_1$

Block:



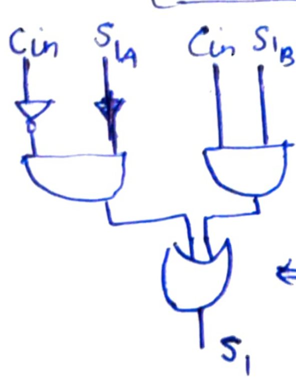




FOR one FA: 2 OR gates, 7 AND gates.

$\Rightarrow$ 4 FA = 4(2 OR + 7 AND)

= $\boxed{\text{8 OR + 28 AND}}$

FOR select:




⇐ 2 AND

⇐ 1 OR

1 selector $\Rightarrow$ 2AND + 1OR

$\Rightarrow$ 3(2AND + 1OR)

= 6 AND + 3 OR

3 selectors because

×2

×1

3

$\Rightarrow$ In one block: (6+28) AND + (3+8) OR

= (34) AND + (11) OR

since there are 2 blocks,
2 (34 AND + 11 OR)
= 68 AND and 22 OR gates
for carry select adder

## Carry Lookahead adder

To calculate $C_i$:

$C_0 = 0$

$C_1 = C_0 P_0 + G_0 = G_0$

$C_2 = G_1 + P_1 C_1 = G_1 + P_1 G_0$

$C_3 = G_2 + P_2 C_2 = G_2 + P_2 (G_1 + P_1 G_0)$

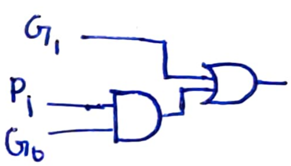
$= G_2 + P_2 G_1 + P_2 P_1 G_0$

$C_4 = G_3 + P_3 (C_3) = G_3 + P_3 (G_2 + P_2 G_1 + P_2 P_1 G_0) = G_3 + P_2 P_1 P_3 G_0 + P_3 G_2 + P_3 P_2 G_1$

FOR $C_0 \Rightarrow$ 0 Gates

$C_1 \Rightarrow G_0$ (0 Gates)

$C_2 \Rightarrow$




1 AND
1 OR

$$C_3 \Rightarrow \underbrace{G_2 + \underbrace{P_1 G_1}_{1\ AND} + \underbrace{P_2 P_1 G_0}_{2\ AND}}_{2\ OR\ GATES}$$

$$C_4 \Rightarrow \underbrace{G_3 + \underbrace{P_2 P_1 P_3 G_0}_{3\ AND} + \underbrace{P_3 P_2 G_1}_{2\ AND} + \underbrace{P_3 G_2}_{1\ AND}}_{3\ OR}$$

TOTAL = (1 + 3 + 6) AND & (1 + 2 + 3) OR GATES

= 10 and & 6 OR gates
To calculate values of C.

FOR $P_i$ & $G_i$ :

$$P_i = A_i \oplus B_i = \underbrace{A_i \bar{B}_i + \bar{A}_i B_i}_{2AND,\ 1OR}$$

$$G_i = \underbrace{A_i B_i}_{1AND}.$$

$$4 \text{ Bits} \Rightarrow 4(2AND + 1OR) + 4(1AND)$$
$$= 8AND + 4AND + 4OR$$
$$= 12AND + 4OR$$

$S_i$: $S_i = C_i(A_i \oplus B_i)$

FOR $S_0$: $C_0 = 0 \Rightarrow S_0 = A_0 \oplus B_1 \Rightarrow \underline{\underline{2\,AND}}, 1\,OR$

$S_{1,2,3}$: $S_i = C_i(A_i \oplus B_i) = 3\,AND,\ 1\,OR$

$\Rightarrow 3(3\,AND + 1\,OR) = 9\,AND + 3\,OR$

$\Rightarrow (9+2)\,AND + (1+3)\,OR = 11\,AND + 4\,OR.$

TOTAL $= (11+12+10)\,AND + (4+4+6)\,OR$

$= 33\,AND + 14\,OR$

5) complexity $= O(n) \times O(\text{complexity of addition})$

If we use carry look ahead adder,

Complexity of addition $= O(\log n)$

$\Rightarrow O(n \log n) =$ complexity of Booth's algorithm

If we use ripple carry,

complex of addition $= O(n)$

$\Rightarrow$ complexity of Booth's algorithm $= O(n^2)$

| Booth's Algorithm | Iterative Algorithm |
|---|---|
| → No need to negate values for negative operands | → Negative numbers need to be negated |
| → complexity same for | both algorithms is same |
| → for cases like 111111... Booth's is faster | → for cases like 101010... Iterative is faster. |