Note Title 22-08-2012

Aug 22nd

Multipliers.

Where were we:

$n \times n$ bit multiplication

naive brute force algo. $\rightarrow O(n \log n)$

smarter method (tree adder) $\rightarrow O(\log(n)^2)$

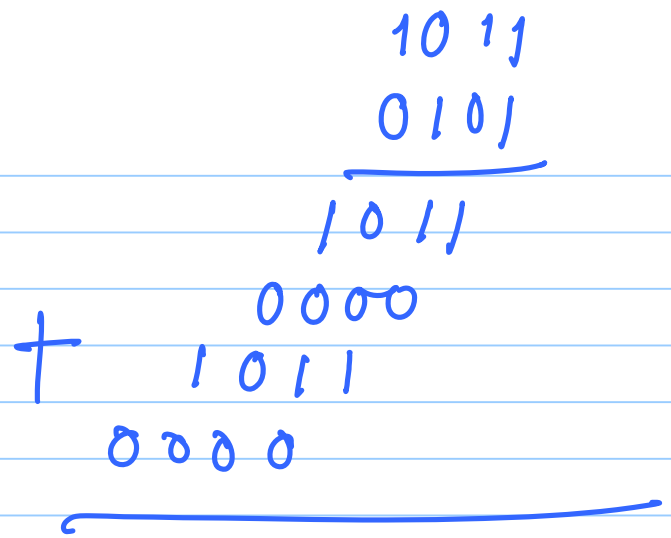
Carry-Save Adder O(1)

$$A + B + C = D + E$$

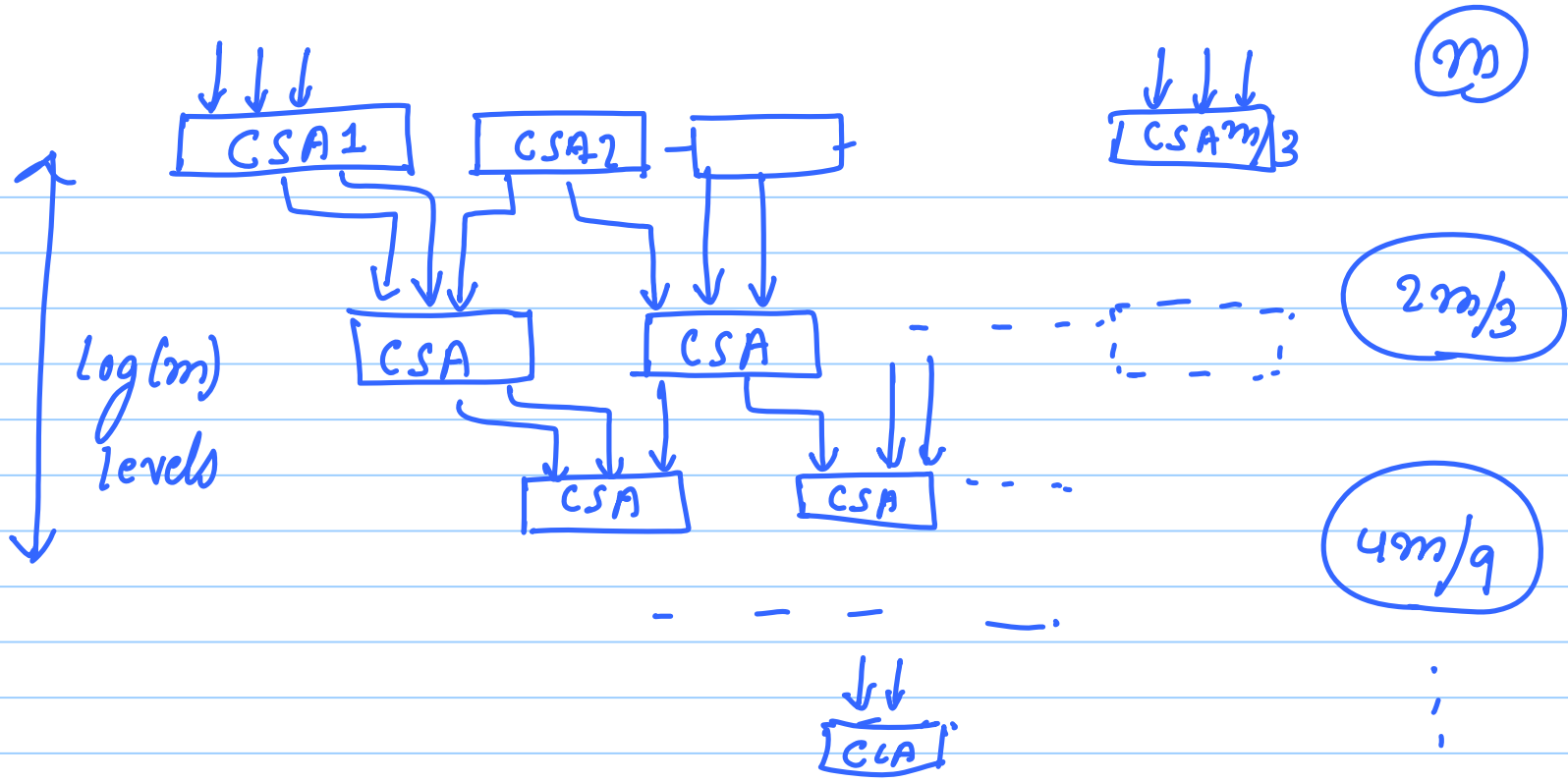
A 1011
B 0001
C 0101

1111 ← D
0010 ← E

1) Write Sum bit at that bit position
2) Write Carry bit one position to the left



How long does it take to add m n-bit numbers?



How many stages? $\log_{3/2} m$

Last Stage:

Input Size: $O(n + \log(m))$

Time: $O(\log(n + \log(m)))$

Total Time: $O[\log(n + \log(m)) + \log(m)]$ ---- (1)

$m = n$

$O(\log(n))$

Multiplication:

Multiplication is nothing but:
adding n partial sums.

Max. Size of each partial sum: $2n-1$
↳ add n $(2n-1)$ bit numbers
in parallel.

Substitute in Eqn. 1

Time for multiplication ($n \times n$ bit)
🚩 $O(\log(n))$

Tree of CSA adders: Wallace Tree

(Wallace Tree Multiplier)

Division. [Super Slow]

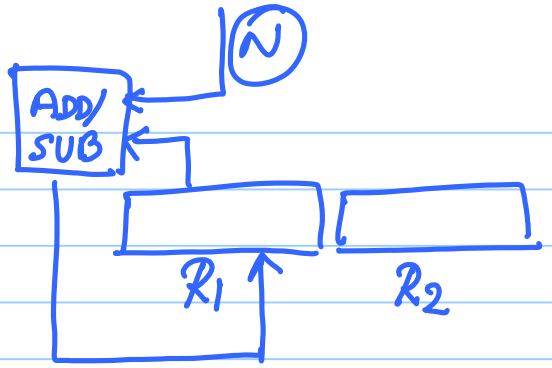
Restoring Division.

$$D = q \cdot N + R \qquad (R < N)$$

$D \rightarrow$ Dividend $\qquad R \rightarrow$ Remainder.

$q \rightarrow$ Quotient

$N \rightarrow$ Divisor



Start:

Load Dividend in R_2

Load Divisor

Steps:

Repeat n times.

i) Left Shift $R_1 R_2$

2) $R_1 = R_1 - N$

3) If ($R_1 < 0$) {

$R_1 += N$

$LSB(R_2) = 0$

}

else {

$LSB(R_2) = 1$

}

End: $R_2 \rightarrow Q$, $R_1 \rightarrow Rem$

E.g. 11/3 1011/0011

Start

①

②

③

- 011

④

- 0011

② ③

R_1 R_2

11/3

Time Complexity : $O(n \log(n))$

Non-Restoring Algo

Some Setup.

Repeat n times.

1) Left Shift R_1, R_2
2) if ($R_1 < 0$)
   $R_1 += N$
   else
   $R_1 -= N$
3) If ($R_1 < 0$)

LSB(R_2) ← 0

else

LSB (R_2) ← 1

4) If ($R_1 < 0$) $R_1 += N$

5) Results:

R_1 → Remainder R_2 → Quotient