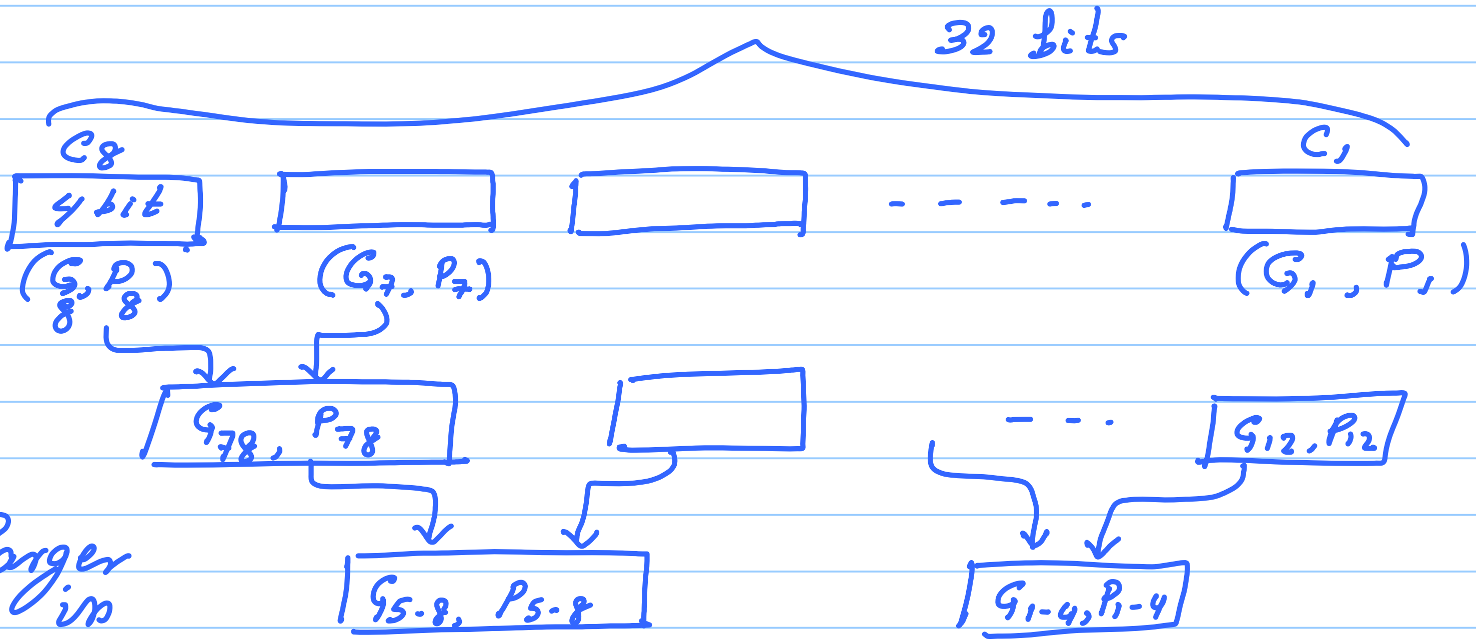
Note Title 24-08-2012

Aug-24

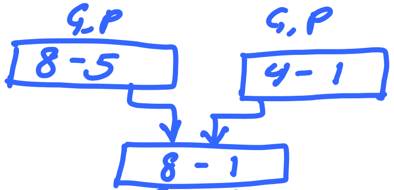
Carry Lookahead Adder

Stage 1:

c_1
⋮ → chunk
c_8 1 … 8



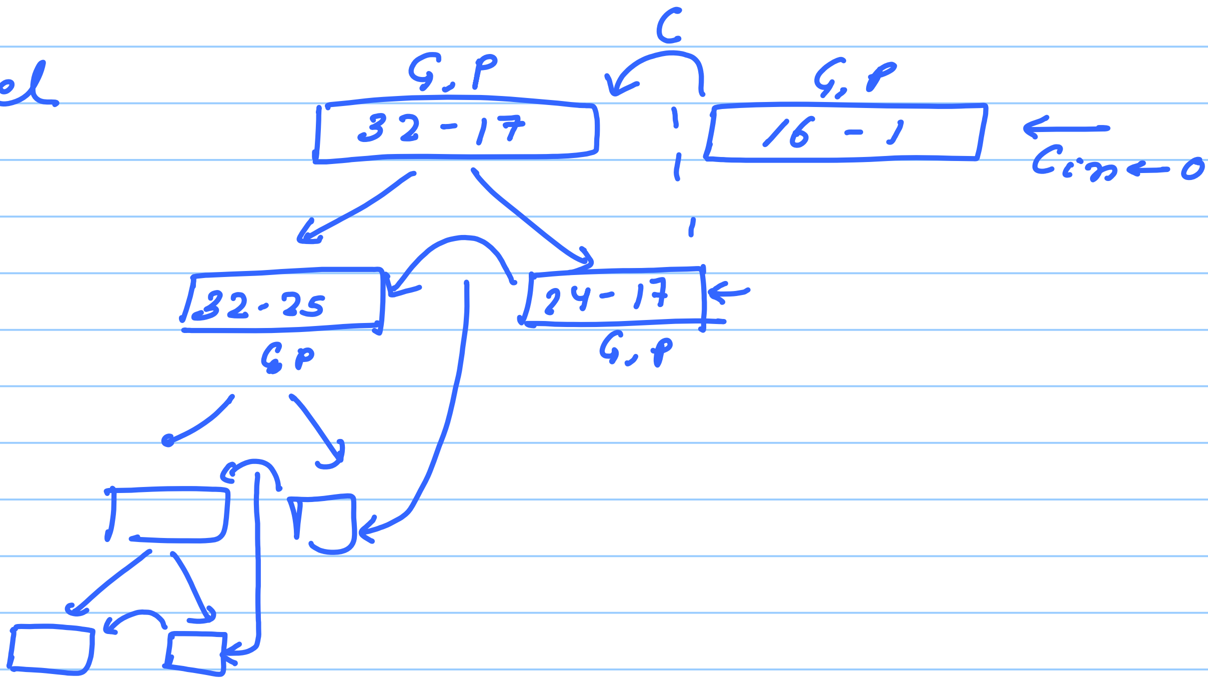
Compute the G, P functions for larger sequences of bits in a tree fashion.



(G,P) --- (Eqn. 1) [23rd]

Stage 2

last level



~~Stage 3~~ Generate final result.

Time

Stage 1: Successively compute G, P for
$O(\log(n))$ larger & larger chunks

Stage 2: Compute the C_{in} values to every
$O(\log(n))$ chunk by traversing the tree from
the root to the leaves

Stage 3: Do the final addition in each
$O(1)$ chunk & compute result

Total time taken : $O(\log(n))$
[fastest possible adder]

c

$$\begin{array}{lcl} & a_n \ldots \; a_{n/2} & a_{n/2-1} \ldots \; a_1 \\ + & b_n \qquad b_{n/2} & b_{n/2-1} \ldots \; b_1 \end{array}$$

Two independent sub problems

Multiplication.

(unsigned positive numbers)

$$
\begin{array}{r}
\overbrace{1001}^{n} \leftarrow \text{multiplicand} \\
\times\ 1011 \leftarrow \text{Multiplier.} \\
\hline
1001 \\
1001 \\
0000 \\
+\quad 1001 \\
\hline
\text{Product}
\end{array}
$$

Partial Sum

n

$2n-1$

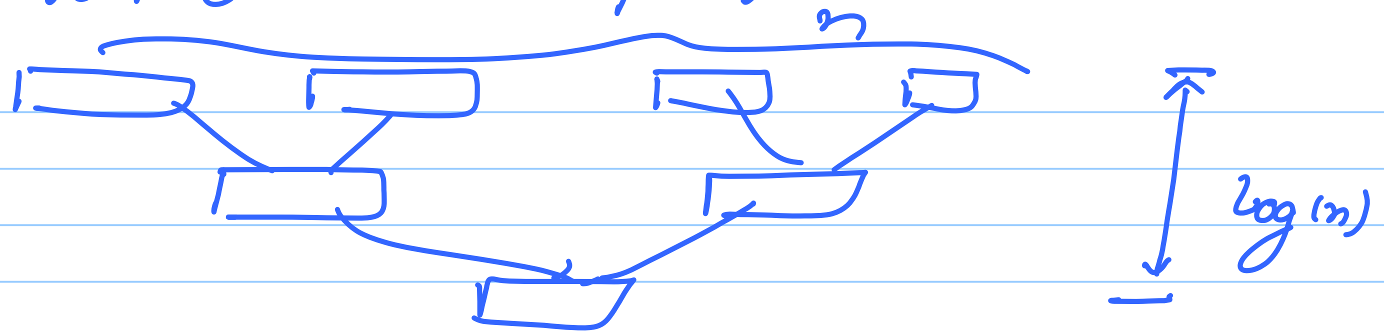
$2n$

Product = Σ Partial Sum

Simple algorithm : Add the n partial sum

Sequential addition $O(n \log(n))$ time.

Slightly smarter (tree multiplier)



Total time: # levels × (Time per level)

$= O(\log(n) \times \log(n))$

$= O((\log(n))^2)$

Aim: Multiply in $O(\log(n))$ time.

New kind of adder: Carry Save Adder (CSA)

CSA:

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

$$
\begin{array}{lrl}
A & 1000 & \cdots (8) \\
B & 0101 & \cdots (5) \\
C & 1011 & \cdots (11) \\
\hline
 & 0110 & \longleftarrow D \\
 & 1001 & \longleftarrow E
\end{array}
\quad \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} + \; 24
$$

$$
\left\{\begin{array}{l} D = 0110 \\ E = 10010 \end{array}\right\}
\begin{array}{l} \cdots (6) \\ \cdots (18) \end{array}
\; + \; 24
$$

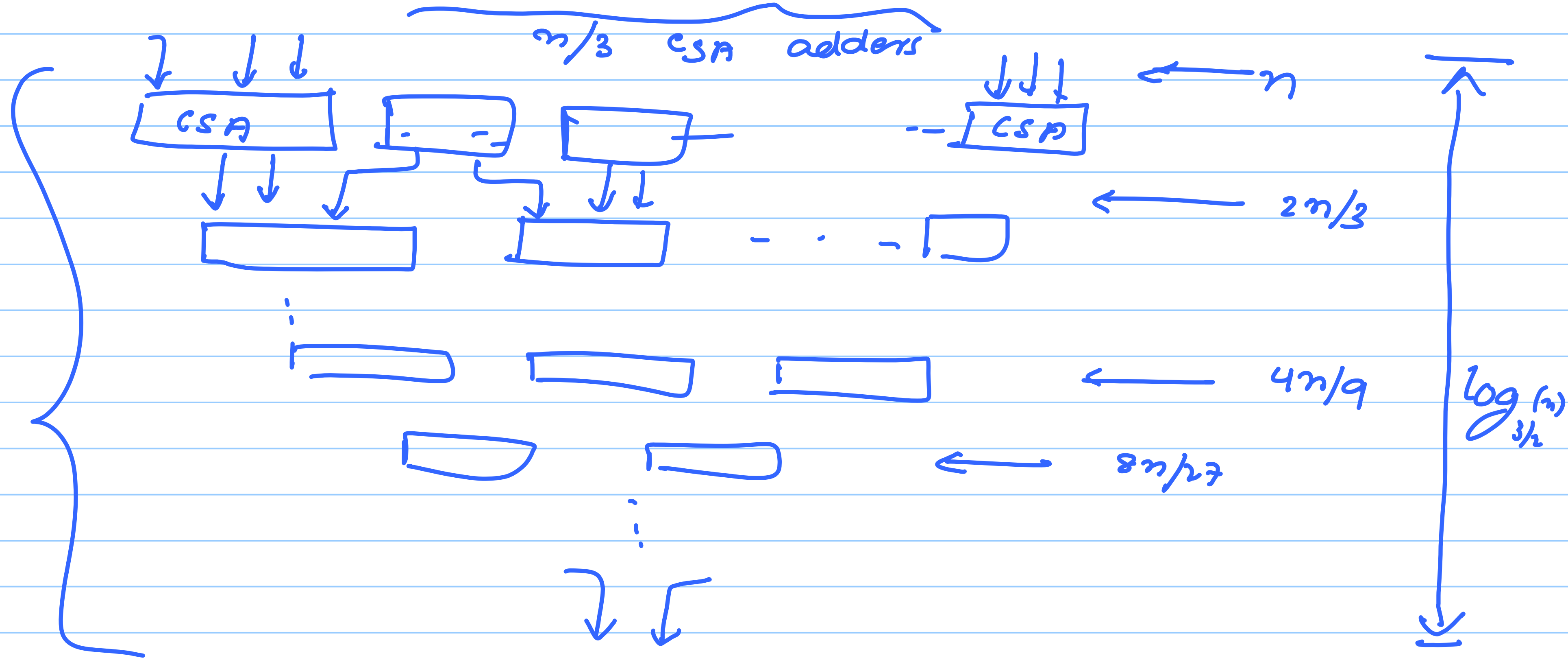
How long does a CSA take to compute

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

It takes $O(1)$ time

(No dependence across bits)

Problem of Multiplication: n independent partial sums need to be added



Levels : $O(\log_{3/2} n) \longleftrightarrow O(\log(n))$

Time per level: $O(1)$

All CSAs in one level operate parallely

Stage 1

$$\underbrace{P_1 \quad - \; - \; - \quad + P_n}_{\text{Partial Sums}} = \underbrace{P_1' + P_2'}_{2 \text{ sums}}$$

Total Time (Stage 1) = $O(\log(n))$

Stage 2

$$\text{Product} = \underbrace{\overset{(2n)}{P_1'} + \overset{(2n)}{P_2'}}_{\text{CLA}} \Rightarrow$$

$O(\log(n))$

Total time: $O(\log(n)) + O(\log(n)) = \boxed{O(\log(n))}$

Technology that we used:

Wallace Tree multiplier.

$O(\log(n))$ TREE OF CSAs

$+$

$O(\log(n))$ CLA

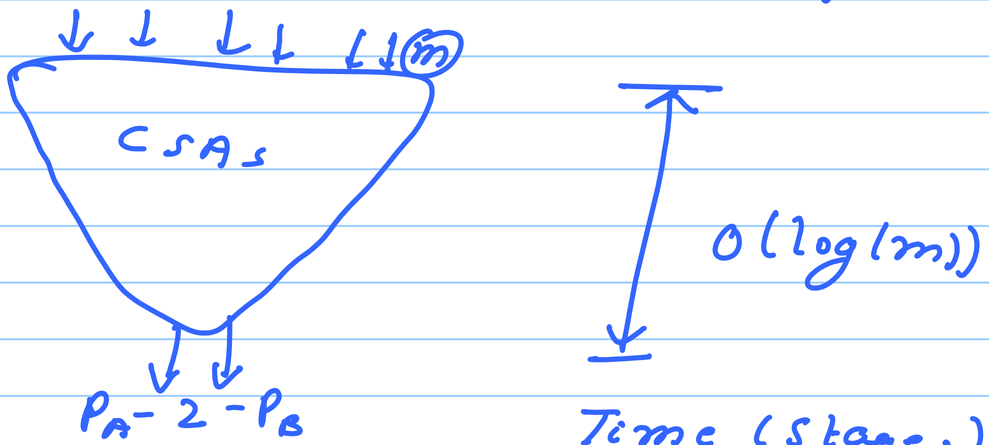
$= O(\log(n))$

CSA → Carry Save Adder

CLA → Carry Lookahead Adder.

Extension

Problem: Add m (n-bit) numbers.



Time (stage$_1$) $= O(\log m)$

Every level the size of the output is at max. 1 bit more than the size of the input.

$$\log(P_A) \leq n + \log(m)$$
$$\log(P_B) \leq n + \log(m)$$

Time (stage$_2$) $P_A + P_B$ = result
(CLA)

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

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

Next Class

Division (slow)
(unsigned)

Division when the sign of dividend & divisor is not the same.

$-7/3$

1. $-7 = 3 \times (-2) + (-1)$
2. $-7 = 3 \times (-3) + 2$

Homework

1) Q. 1

a) Write the algo. in C
→ read input files

b) arm-elf-gcc -S file.c

c) file.s
↙
1) open
2) reduce instructions

Do ot touch the line that contain directives

.align .word
.glob .lco

2) Process is same.

2x 160 bit nums