

# **CS51: Models of Computation**

## **Lecture 9**

### **Integer Multiplication**

# Review

- Topics covered last time:
  - integer addition
  - carry lookahead addition
  - standard integer multiplication
  - carry-save addition of three vectors:  
sums are saved in one vector, carries are saved in another
  - carry-save multiplication

# Today's Topics

- Review of **carry-save** multiplication
- **Divide-and-conquer** integer multiplication
- Integer multiplication via **exponent addition**

# Standard Integer Multiplication

- Integers represented in binary:

$$\left( |u| = \sum_{i=0}^{n-1} u_i 2^i \right), \left( |v| = \sum_{j=0}^{n-1} v_j 2^j \right)$$

- Integer product – standard algorithm:

$$|u||v| = \left( \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} u_i v_j 2^{i+j} \right)$$

# Integer Product

- Integer product – standard algorithm:

$$|u||v| = \left( \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} u_i v_j 2^{i+j} \right)$$

$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
			$v_3 u_0$	$v_2 u_0$	$v_1 u_0$	$v_0 u_0$
		$v_3 u_1$	$v_2 u_1$	$v_1 u_1$	$v_0 u_1$	0
	$v_3 u_2$	$v_2 u_2$	$v_1 u_2$	$v_0 u_2$	0	0
$v_3 u_3$	$v_2 u_3$	$v_1 u_3$	$v_0 u_3$	0	0	0

- Standard algorithm adds first row to second, adds result to third, etc.
- Even with **fast carry lookahead** addition requires size  $O(n^2)$  and depth  $O(n \log_2 n)$ .

# Carry-Save Addition

- Given three binary numbers, **u**, **v**, and **w**, add them, placing the **sum bits** in **s** and the **carry bits** (shifted by one place) in **c**. Thus,

$$|\mathbf{c}| + |\mathbf{s}| = |\mathbf{u}| + |\mathbf{v}| + |\mathbf{w}|$$

E.g.

**u** = 101

**v** = 110

**w** = 111

---

**s** = 100

**c** = 1110

#1's in col

---

2 1 0

3 2 2

- The vectors **s** and **c** can be computed by a circuit of **Full Adders** with size  $O(n)$  and depth  $O(1)$ .

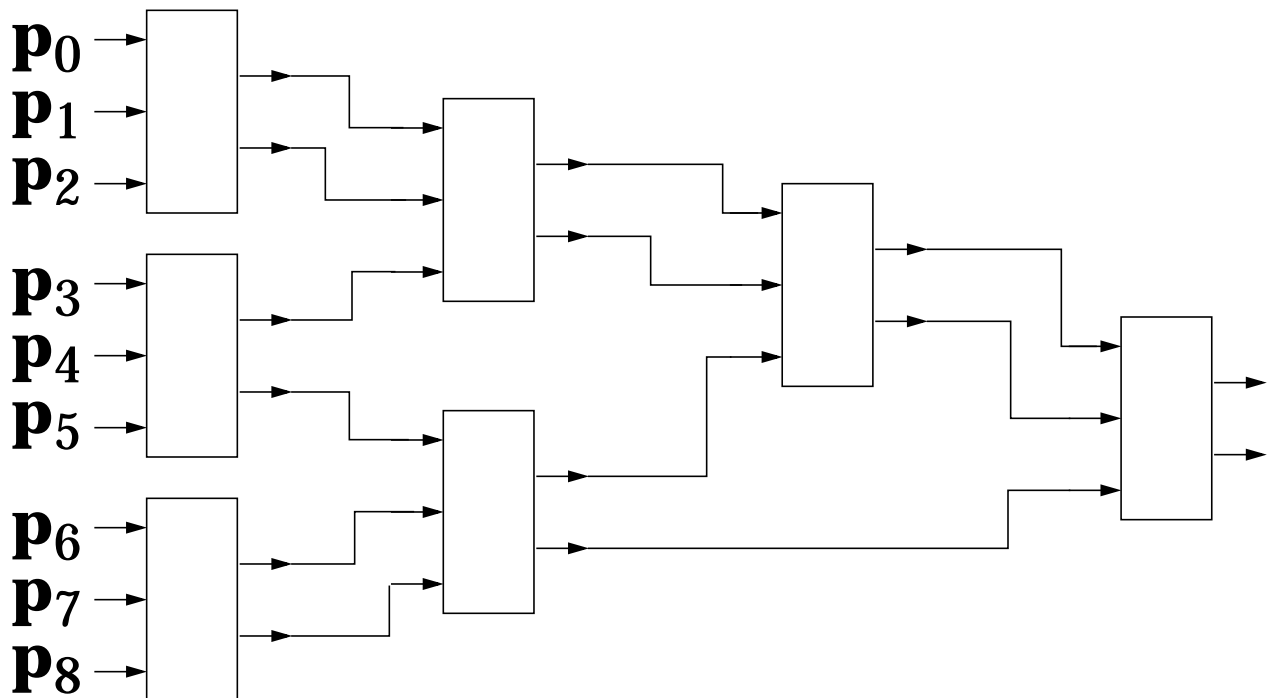
# Standard Integer Multiplication

$$|u||v| = \left( \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} u_i v_j 2^{i+j} \right)$$

$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$	
			$v_3 u_0$	$v_2 u_0$	$v_1 u_0$	$v_0 u_0$	<b>= <math>\mathbf{p}_0</math></b>
{	<b>n</b>		$v_3 u_1$	$v_2 u_1$	$v_1 u_1$	$v_0 u_1$	<b>= <math>\mathbf{p}_1</math></b>
			$v_3 u_2$	$v_2 u_2$	$v_1 u_2$	$v_0 u_2$	<b>= <math>\mathbf{p}_2</math></b>
			$v_3 u_3$	$v_2 u_3$	$v_1 u_3$	$v_0 u_3$	<b>= <math>\mathbf{p}_3</math></b>
$\underbrace{\hspace{15em}}_{2n-1}$							<b>n = 4 here</b>

- Let  $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_{n-1}$  be these n numbers.
- Add  $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_{n-1}$  in groups of three **in parallel** using carry-save addition. Each addition provides an **s** and a **c**.

# Carry-Save Multiplication



- Let  $m_j$  be the number of tuples at level  $j$ .

$$m_0 = n$$

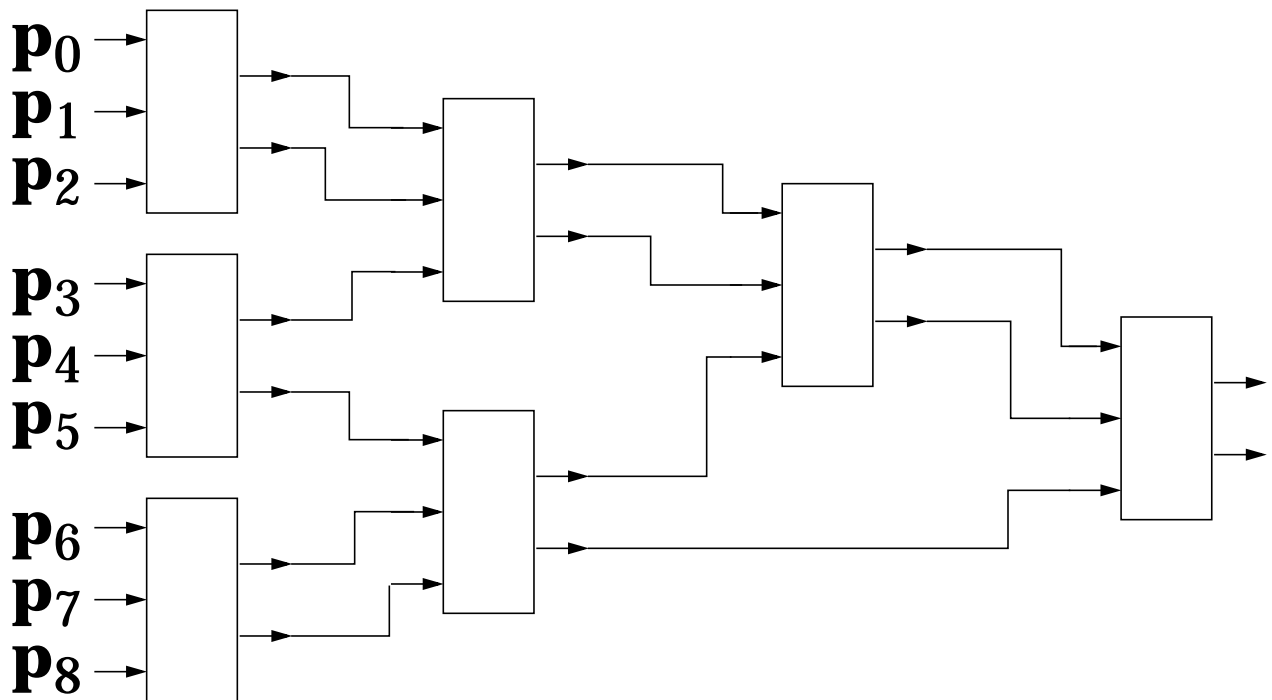
and

$$m_j = 2 \lfloor m_{j-1}/3 \rfloor + m_{j-1} - 3 \lfloor m_{j-1}/3 \rfloor = m_{j-1} - \lfloor m_{j-1}/3 \rfloor$$

- The  $k$  for which  $m_k = 2$  is  $k = O(\log n)$ .

If  $m_0 = 9$ ,  $m_1 = 6$ ,  $m_2 = 4$ ,  $m_3 = 3$ ,  $m_4 = 2$  and  $k=2$ .

# Carry-Save Multiplication



- Depth  $O(\log n)$  circuit produces two tuples. Tuple length is  $O(n)$ .
- Add the two tuples with an  $O(\log n)$  depth carry lookahead adder.  $O(n^2)$  gates used.

# Carry-Save Multiplication

- Number of stages  $s$  satisfies  $m_s = 2$ . Since  $(x-2)/3 \leq \lfloor x/3 \rfloor \leq x/3$ , it follows that

$$(2/3)^j n \leq m_j \leq (2/3)^j n + 2$$

- Thus,  $s$  satisfies

$$\frac{\log\left(\frac{n}{2}\right)}{\log\left(\frac{3}{2}\right)} \leq s \leq \frac{\log n}{\log\left(\frac{3}{2}\right)} + 1$$

- The length of the resultant pair of binary numbers is  $2n-1+s$ . (The length is increased by one bit per level.)

# Carry-Save Multiplication

**Theorem:** Integer multiplication  $f_{\text{mult}}$  can be realized by a circuit of size

$$5n(n-2) + C_{\Omega}(f_{\text{add}})$$

and depth

$$3s + D_{\Omega}(f_{\text{add}}).$$

- Thus, multiplication can be done in time  $O(\log_2 n)$ .
- Later we show the size bound can be reduced.

# Divide-and-Conquer Multiplication

- Let  $n$  be even.
- Let  $\mathbf{u}$  and  $\mathbf{v}$  be  $n$ -bit integers.
- Let  $\mathbf{u} = (\mathbf{u}_h, \mathbf{u}_l)$  and  $\mathbf{v} = (\mathbf{v}_h, \mathbf{v}_l)$  are divided into the **lower** and **upper** half of the bits.

Now

$$|\mathbf{u}| = |\mathbf{u}_h| 2^{n/2} + |\mathbf{u}_l|$$

and

$$|\mathbf{v}| = |\mathbf{v}_h| 2^{n/2} + |\mathbf{v}_l|$$

# Divide-and-Conquer Multiplication

Thus,

$$\begin{aligned} |\mathbf{u}||\mathbf{v}| &= (|\mathbf{u}_h|2^{n/2} + |\mathbf{u}_l|)(|\mathbf{v}_h|2^{n/2} + |\mathbf{v}_l|) \\ &= |\mathbf{u}_h||\mathbf{v}_h|2^n + (|\mathbf{u}_l||\mathbf{v}_h| + |\mathbf{u}_h||\mathbf{v}_l|)2^{n/2} + |\mathbf{u}_l||\mathbf{v}_l| \\ &= |\mathbf{u}_h||\mathbf{v}_h|2^n + \\ &\quad (|\mathbf{u}_h||\mathbf{v}_h| + (|\mathbf{v}_h| - |\mathbf{v}_l|)(|\mathbf{u}_l| - |\mathbf{u}_h|) + \\ &\quad |\mathbf{u}_l||\mathbf{v}_l|)2^{n/2} + |\mathbf{u}_l||\mathbf{v}_l| \end{aligned}$$

**Note:** **three products** of  $(n/2)$ -bit numbers, **two subtractions**, and **three additions** of  $n$ -bit numbers.

Additions and subtractions can be done with circuit size  $O(n)$  and depth  $O(\log n)$ .

# Divide-and-Conquer Multiplication

- Let  $C(k)$  and  $D(k)$  be **size** and **depth** of a circuit for integer multiplication using this algorithm when  $n = 2^k$  ( $k = \log_2 n$ ).

- Then,

$$C(k) \leq 3 * C(k - 1) + c 2^k$$

$$D(k) \leq D(k - 1) + d k$$

for constants  $c, d > 0$ .

- Since  $C(1) = 2$  and  $D(1) = 1$ , we have the following bounds:

$$C(k) \leq (4c+2)3^{k-1} - c2^{k+1} = O(n^{\log 3})$$

$$D(k) \leq d(k + k-1 + \dots + 1) = O(\log^2 n)$$

# Very Fast Integer Multiplication

- **Prime number decomposition** (PND) of integers:

1	2	3	4	5	6	7	8	9	10	11
$1^1$	$2^1$	$3^1$	$2^2$	$5^1$	$2^1 3^1$	$7^1$	$2^3$	$3^2$	$2^1 5^1$	$11^1$

- Represent integers by **exponents of primes**.

E.g.  $6 \leftrightarrow 2^1 3^1 5^0 7^0 11^0 = (1, 1, 0, 0, 0)$

$$10 \leftrightarrow 2^1 3^0 5^1 7^0 11^0 = (1, 0, 1, 0, 0)$$

$$6 * 10 = 60 \leftrightarrow 2^2 3^1 5^1 7^0 11^0 = (2, 1, 1, 0, 0)$$

## Very Fast Integer Multiplication

- If  $n$  in  $\{0, 1, \dots, N-1\}$ , there are  $\lceil \log_2 N \rceil$  bits in binary representation. Multiplier depth is  $\lceil \log_2 \log_2 N \rceil$ .
- In PND **exponents are less than**  $\lceil \log_2 N \rceil$ . At most  $\lceil \log_2 \log_2 N \rceil$  bits to represent each exponent.
- Multiply integers by **adding their exponents**
  - Depth is:  $O(\log \log (\log N))$  **Wow!**
- What's wrong with this system?

## Summary

- Review of integer multiplication via **carry-save addition**. Circuit size is  $O(n^2)$  and depth is  $O(\log n)$ .
- **Divide-and-conquer** integer multiplication reduces size to  $O(n^{\log_2 3})$  but depth is now  $O(\log^2 n)$ . (Note:  $\log_2 3 = 1.585$ .)
- Multiplication using **prime number decomposition** is very fast but a poor way to add numbers.