Addition and Subtraction

Lecture Objectives:

1) Explain the relationship between addition and subtraction with twos compliment numbering systems
2) Explain the concept of numeric overflow when dealing with twos compliment numbers.
3) Explain the concept of an exception.
4) Define multiplier and multiplicand.
5) Explain the concept of left and right shifting.
6) Explain how a computer may use left and right shifting to perform arithmetic multiplication sequentially.
7) Explain the advantage of using left or right shifts when multiplying or dividing by a power of 2.
8) Explain how hardware can be added to improve multiplication times.
9) Compare and contrast the time complexity of multiplication with addition and subtraction.
10) Explain how a computer deals with signed multiplication.
11) Explain the size relationship between the multiplier, multiplicand, and the product.

Intro

Course

- You will be given a sheet of paper with arithmetic problems on it.
- You will have exactly 2 minutes to work as many problems as is possible

Survey

- How many did you get done?

12 All 20

3 8 2

- What type of problem did you solve?

⇒ Simple Add/Subtract Multiplication/Division Decimal calculations...

- 4 basic types of problems
  - Simple addition
  - Simple multiplication
  - Complex addition (Floating Point)
  - Complex multiplication (Floating Point)

The quiz sheets

Not really complex.

Addition

- Binary Addition
  - Add 7 + 6 as 5 bit numbers

Sign bit

11

7=> 0 0111

6 => 0 0110

01101 => 0x0D

8+4+1

11

13

Binary Addition

- Add 15 + 14 as 5 bit numbers

– 15 =>

– 14=>

```
  1  1 1
  0 1111
+ 0 1110
-------
  1 1101
```

Problem!

Overflow: Two positives resulted in a negative #

Overflow

- Overflow if result out of range
  - Adding +ve and –ve operands, no overflow
  - Adding two +ve operands
    - Overflow if result sign is 1
  - Adding two –ve operands
    - Overflow if result sign is 0

- Overflow can generate an exception
  - An unscheduled event (interrupt) that disrupts program execution, used to detect overflow.
- Interrupt
  - An exception that comes from outside of the processor

Subtraction

- Take the number that is to be added and calculate the twos compliment
  - Simply add the two numbers together

Addition w/ a 2's compliment

- Example

15 - 4

11

0 0 1 0 0

1 1 0 1 1

1 1 1 0 0

5 bit #'s

0 1 1 1 1

1 1 1 0 0

1 0 1 0 1

8 + 2 + 1 = 11

Definitions (Pg 230)

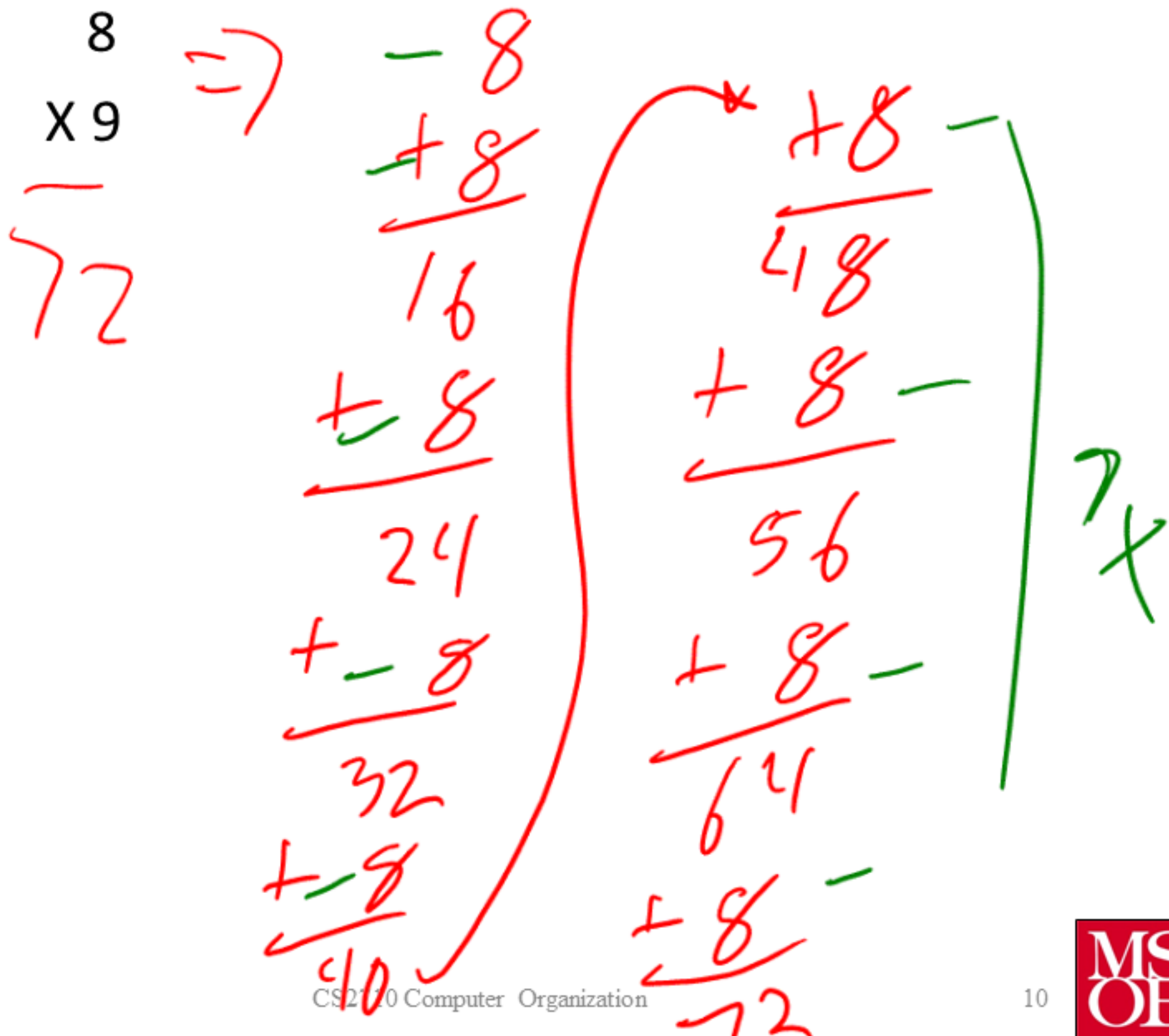
- Multiplicand
  - The first operand of a multiplication operation
- Multiplier
  - The second operand of a multiplication operation
- Product
  - The final result of a multiplication operation

Basic definitions

Multiplication

- First approach (Long Multiplication)

8
X 9
—
72



$$\begin{array}{r} 101 \\ \times\quad 5 \\ \hline 505 \end{array} \Rightarrow \begin{array}{r} 101 \\ \times\quad 15 \\ \hline 505 \\ 1010 \\ \hline 1515 \end{array}$$

8 ⇒ 1000

x9 ⇒ Multiplier 1001

1000 ← 1 clk

0000 ← 1 clk

000000 ← 1 clk

1000000 ← 1 clk

1001000

$2^6 \Rightarrow 64$

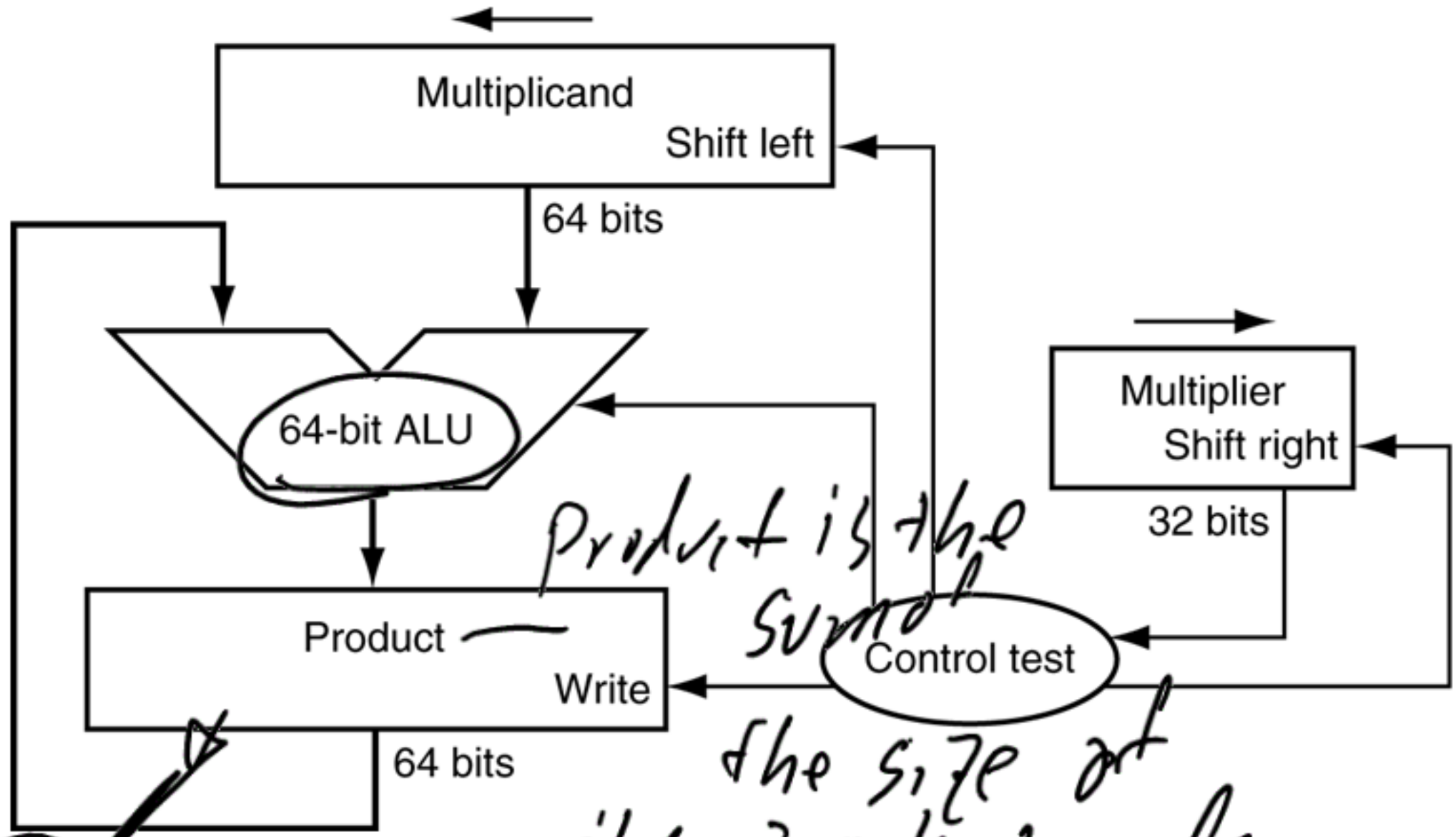
8

72

$2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

At least 4 cycles

Multiplication Hardware

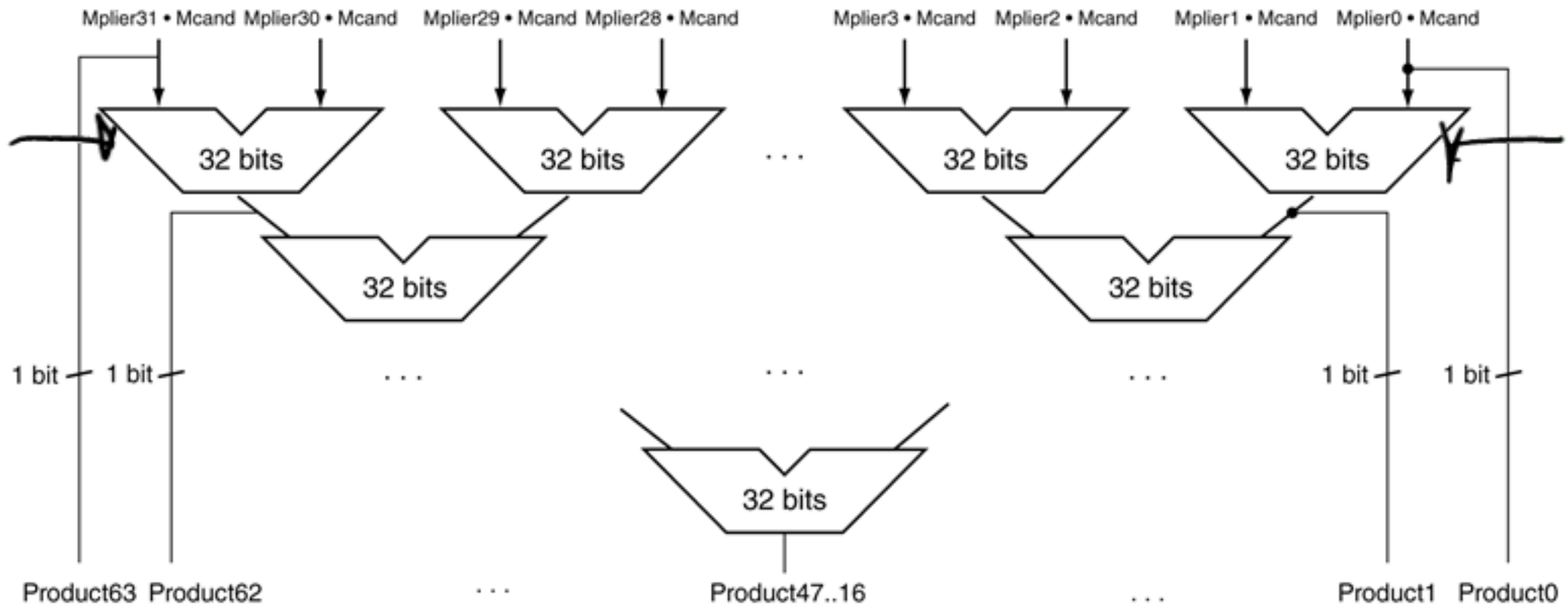


- How long would 8 x 9 take to calculate as 4 bit numbers?

High Low time is related to the size of the multiplicands

Faster Multiplier

- Uses multiple adders
  - Cost/performance tradeoff



N adders to perform the multiplication

MIPS Multiplication

- Two 32-bit registers for product
  - HI: most-significant 32 bits
  - LO: least-significant 32-bits
- Instructions
  - `mult rs, rt  /  multu rs, rt`
    - 64-bit product in HI/LO
  - `mfhi rd  /  mflo rd`
    - Move from HI/LO to rd
    - Can test HI value to see if product overflows 32 bits
  - `mul rd, rs, rt`
    - Least-significant 32 bits of product –> rd

Poor mans multiplication

- What happens if a number is shifted to the left one bit?

→ unsigned

Doubles the ⇒ multiplies by 2.

- What happens if a number is shifted to the right one bit?

⇒ half the number.

Divide by 2

Multiplying signed numbers

1. Convert the numbers to positive numbers, remembering the signs
2. Multiply them as positive numbers
3. Convert back to the appropriate sign based on the initial input.

$$
\begin{array}{r}
-2 \\
\times\ 5 \\
\hline
-10
\end{array}
\Rightarrow
$$

0010 ⇒ 1101

1 1110

0 0101

Binary

⇒

$$
\begin{array}{r}
00010 \\
00101 \\
\hline
10 \\
1000 \\
\hline
1010
\end{array}
$$

⇒ 0101

1 0 1 1 0