

Computation 1

Reading

Petzold Pp. 131-134

Handouts

Binary Addition

Mexican Folk Arithmetic Algorithm Makes Perfect Sense

Multicultural Mathematics

Resources

Addition by partial sums (Mexican Folk algorithm)

<https://everydaymath.uchicago.edu/teaching-topics/computation/add-partial-sums.html>

Binary addition <https://ryanstutorials.net/binary-tutorial/binary-arithmetic.php#addition>

1) Addition in decimal: the “Standard” algorithm.

Here is a typical problem, showing how to add two 3-digit numbers in decimal (show place values)

$$\begin{array}{r} 893 \\ +745 \\ \hline \end{array}$$

The algorithm taught universally in the US (but not everywhere, as we shall see), is to start from the right, by adding the 3 and the 5, and proceed towards the left, “carrying” or “regrouping” a 1 above the 893 every time the sum exceeds 9. To show the carried 1’s clearly, spaces are added between the digits and the carried 1’s are shown in **bold**.

1 1	carry or regroup
8 9 3	addend
+ <u>7 4 5</u>	addend
1 6 3 8	sum

Why is there a “carry” or “regrouping” and what does it mean?

In the 1’s column, we have $3 + 5 = 8$, which fits completely in the 1’s column, so there is no carry or regrouping.

However, in the 10’s column we must add $9 + 4$. Because the place value in this column is 10, this sum is $90 + 40 = 130$, which contains 3 in the 10’s column and 1 in the 100’s column.

The three represents 30, so it stays in the 10’s column. However, the 1 is really 100, so we “carry” or “regroup” it to the 100’s column to the left and write it above the 8 and the 7 already occupying that column. The 16, which has 4 in the 100’s column, but also generates a “carry”, pushing the 1 into the 1000’s column.

The 1’s column contains the least significant digits (LSD), and that is where we start in the standard algorithm, working our way towards the most significant digits (MSD) on the left. Is that the only way it could be done?

2) Addition in Decimal: The Mexican folk algorithm (Partial Sum Method)

This algorithm is common in various parts of the world. Unfortunately, children who have learned it in other countries may migrate to the US, use this algorithm, get the right answer efficiently, but be told they are “wrong”, because they did not follow the only method the teacher knows.

Unlike the standard algorithm taught in US schools, this algorithm starts from the 100’s column and creates the partial sums directly, without showing the carried 1’s explicitly. The MSD column is $800 + 700 = 1500$, which is recorded on left in the table below. Next, we add the 90 and 40 in the 10’s column, which gives 130, which goes below the 1500 in the second box from the left. Then we add the 3 and 5 in the 1’s column, giving 8, shown on the last line in the second box from the right. The last box adds all the partial sums, giving 1638.

<u>100’s column:</u> $800 + 700 = 1500$	<u>10’s column:</u> $90 + 40 = 130$	<u>1’s column</u> $3 + 5 = 8$	total
$\begin{array}{r} 893 \\ + 745 \\ \hline 1500 \end{array}$	$\begin{array}{r} 893 \\ + 745 \\ \hline 1500 \\ 130 \end{array}$	$\begin{array}{r} 893 \\ + 745 \\ \hline 1500 \\ 130 \\ \hline 8 \end{array}$	$\begin{array}{r} 893 \\ + 745 \\ \hline 1500 \\ 130 \\ \hline 8 \\ \hline 1638 \end{array}$

This might seem cumbersome, because each step is spelled out completely, but with a little practice students collapse the partial sums into a running total, like this:

You can

<u>100’s column:</u> $800 + 700 = 1500$	<u>10’s column:</u> $90 + 40 = 130$	<u>1’s column</u> $3 + 5 = 8$
$\begin{array}{r} 893 \\ + 745 \\ \hline 1500 \end{array}$	$\begin{array}{r} 893 \\ + 745 \\ \hline 1630 \end{array}$	$\begin{array}{r} 893 \\ + 745 \\ \hline 1638 \end{array}$

Individual Activity:

a) Use both algorithms to do the following problems:

$89 + 23 = ?$ $719 + 305 = ?$ $61 + 39 = ?$ $77 + 55 = ?$

b) How do you think the two algorithms compare? Which is faster? Which is easier to remember? Which is less prone to error?

3) Binary Addition

a) Adding two one-bit numbers

Group activity: Perform each of the additions in binary:

$0 + 0 = \underline{\quad}$

$0 + 1 = \underline{\quad}$

$$1 + 0 = \underline{\quad}$$

$$1 + 1 = \underline{\quad}$$

Hint: The answer to the last problem cannot be 2, because the only symbols in binary are 0 and 1. Recall what happens when we run out of symbols – we start a new column! In decimal this happens after 9 – the new column is the ten’s column, and the number after 9 is expressed as 10, which stands for 1 ten and 0 ones. What is the place value of the second column in binary?

b) Addition tables in decimal and binary:

Whole-class activity: Construct the first five or six rows and columns of the decimal addition table.

Individual Activity: Take the results of the four addition problems, and put them in the form of an addition table:

+	0	1
0		
1		

Notice that each answer has two bits, not one, because $1 + 1$ generates a “carry” into the two’s column. In other words, an adder must be able to have at least two inputs, the two one-bit addends, and two outputs the sum and the carry bits. You might wonder where the carry output will go. As we’ll see below, an efficient adder does not add it in immediately, so each adder needs only two inputs, the two addends.

c) Truth tables for binary addition

Whole-class Activity: Let’s call the two addends A and B. Separate the two outputs and call the LSB S for Sum, and the MSB C for Carry, and transfer the 0’s and 1’s from the addition table to the chart below. Note that the meaning of the word “Sum” is a little different from the way we used it in multidigit decimal addition earlier.

Inputs		Outputs	
A	B	C	S

Next create the a Truth Table by renaming 0 as F (for False) and 1 as T (for true). In the next course, we'll see how computers use these truth tables to add two numbers.

4) The "Standard" Addition Algorithm in Binary

In the "Standard" method, just like in decimal, we move from right to left, and write a carry of 1, when it exists, above the top addend. Now we are dealing with bits (binary digits), so the column on the right is called the Least Significant Bit (MSB), and we work our way from there to the Most Significant bit (MSB). Here are two examples:

a) $110 + 100$

ii) $1011 + 011$.

What numbers are these in decimal, and what should each result be? Showe the place value for each column in binary.

$$\begin{array}{r} 1 \\ 110 \\ + 100 \\ \hline 1010 \end{array}$$

$$\begin{array}{r} 11 \\ 1011 \\ + 011 \\ \hline 1110 \end{array}$$

Notice that in the second column of the second problem, we are forced to add three 1's to get 3, or 11 in binary, which both generates a 1 below the line and a carry of 1. Here are some practice problems:

#1	#2	#3	#4
Binary	Binary	Binary	Binary
Decimal	Decimal	Decimal	Decimal
$C_{in} \ 1$	C_{in}	C_{in}	C_{in}
A $11 = 3$	$111 =$	$110 =$	$100 =$
B $+ 10 = 2$	$+ 11 =$	$+ 100 =$	$+ 101 =$
S $101 = 5$	S $=$	S $=$	S $=$
$C_{out} \ 1$	C_{out}	C_{out}	C_{out}

To solve these problems, because there could be a carry input to each column except the first, you have to use a different addition table for those columns. It has three inputs: the addends A and B, as before, plus the possible Carry input from the column to the right.

Inputs			Outputs	
A	B	C _{in}	C _{out}	S
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

The addition tables and sample problems reveal why the standard method would not be used by a computer for implementing binary addition:

- i. it requires a three-input adder circuit, because the carry bit must be added to the two addend bits; and
- ii. because the operation of adding each column must wait for the output of the previous column, in order to include the carry bit.

The first problem is solved by using a circuit called a “full adder”, which is much more complicated than the “half adder” that has no carry input.

Because of the second problem, an adder that uses the standard method is called a Ripple Carry Adder, because the carry must ripple from right to left from LSB to MSB to complete the addition.


For both reasons, the addition algorithm used in some computers is much closer to the Mexican Folk Method than to the Standard method. In computers, methods based on this idea are called Carry Select and Carry Free Addition.

BREAK BETWEEN SESSIONS

- 5) Alternative Method in Binary: As with the Mexican Folk Algorithm in Decimal, there is no reason why addition must start with the LSB at the right. Using an alternative algorithm, addition can start from either the left or the right side, or even be performed on all the columns simultaneously. The alternative algorithm never adds more than two bits at a time, because the carried 1’s are added in later. It is therefore potentially much faster and simpler than the “Standard” method. Using the same examples as before, let’s start with the first problem. This time we add 110 and 100, in either direction, recording the carries on top, but without adding them in yet. Another difference is that if there is no carry of 1, we record a 0 in the carry row, because we will need it in the next step. Adding the 110 and 100 generates a new row, called the sum.

$$\begin{array}{r}
 \mathbf{1\ 0\ 0\ 0\ \text{carry}} \\
 1\ 1\ 0 \\
 + \underline{1\ 0\ 0} \\
 \hline
 0\ 1\ 0\ \text{sum}
 \end{array}$$

Next, we move the carry row below the sum row:

$$\begin{array}{r}
 1\ 1\ 0 \\
 \underline{1\ 0\ 0} \\
 0\ 1\ 0 \\
 1\ 0\ 0\ 0
 \end{array}$$



Finally, we add the carry row to the sum row to get the answer:

$$\begin{array}{r}
 1\ 1\ 0 \\
 \underline{1\ 0\ 0} \\
 0\ 1\ 0\ \text{sum} \\
 + \underline{1\ 0\ 0\ 0\ \text{carry}} \\
 \hline
 1\ 0\ 1\ 0\ \text{result}
 \end{array}$$

The result is 1010, just like before. The other example is similar. Again, we fill 0's into the carry row (shown in bold), but don't add it in yet. We do add the two original numbers, 1011 and 011, but ignore the carry row in this addition

$$\begin{array}{r}
 \mathbf{0\ 1\ 1\ 0\ \text{carry}} \\
 1\ 0\ 1\ 1 \\
 + \underline{0\ 1\ 1} \\
 \hline
 1\ 0\ 0\ 0\ \text{sum}
 \end{array}$$

Now we bring the carry row down to the bottom, below the sum

$$\begin{array}{r}
 1\ 0\ 1\ 1 \\
 \underline{0\ 1\ 1} \\
 1\ 0\ 0\ 0 \\
 0\ 1\ 1\ 0
 \end{array}$$


Finally, we add the sum and the carry rows to get the result:

$$\begin{array}{r}
 1\ 0\ 1\ 1 \\
 \underline{0\ 1\ 1} \\
 1\ 0\ 0\ 0\ \text{sum} \\
 + \underline{0\ 1\ 1\ 0\ \text{carry}} \\
 \hline
 1\ 1\ 1\ 0\ \text{result}
 \end{array}$$

Group activity:

Do the following problems, using both methods

$1001 + 0101 = ?$

$1100 + 100 = ?$

$1101 + 111 = ?$

In each case, check your result by converting the addends and the result to decimal.

- 6) Left to right or right to left? In decimal, is it better to start from the Least Significant Digit (LSD) on the right or the Most Significant Digit (MSD) on the left? Here are some examples to discuss:
- a) When you receive change from a purchase, should you receive the big bills, the small bills, or the change first?
 - b) When you leave a tip, which should you put down first: the big bills, the small bills or the change?
 - c) If you must estimate how much money you have left in your bank account, purse, or wallet, are the digits on the left or the right most important?
- 7) Introduction to the final project: