Unsigned integer multiplication done by shifting and adding

Start with the operands at the top of two columns. Repeatedly divide the left column's value by 2 (integer division) and multiply the right column's value by 2, until the left column's value reaches one. Omit all rows in which the left column's value is even. Then add the remaining rows of the right column. For example (36 * 26):

Here are the columns formed by halving left value and doubling right value until left column reaches 1:

```
             26               36
             13               72
              6              144
              3              288
              1              576
```

Omit all rows where left column is even then add the right column's values:

```
~~           26               36~~
             13               72
~~            6              144~~
              3              288
              1              576
                            ----
                             936      (36 * 26 = 936)
```

Why does this work? Here's the same example shown in binary:

```
~~        11010           100100~~
           1101          1001000       $72_{10}$
~~          110         10010000~~
             11        100100000      $288_{10}$
              1       1001000000      $576_{10}$
                     -----------
                      1110101000      $936_{10}$
```

Showing the binary multiplication (it's the same algorithm as for decimal multiplication):

```
                          100100
                           11010
                         -------
                          000000
                         1001000
                        00000000
                       100100000
                      1001000000
                     -----------
                      1110101000
```

Notice when the appropriate bit of the lower operand is 0, the row in the resulting addition is all zero bits, so it can be omitted. The sum of the non-zero rows is the product.