Dadda Tree (H1)

20170422

References

Some Figures from the following sites

[1] http://pages.hmc.edu/harris/cmosvlsi/4e/index.html
Weste & Harris Book Site

[2] en.wikipedia.org

The $\boxed{\textbf{Dadda}}$ **multiplier** is a hardware multiplier design invented by computer scientist Luigi Dadda in 1965. It is similar to the Wallace multiplier, but it is slightly faster (for all operand sizes) and requires fewer gates (for all but the smallest operand sizes).[1]

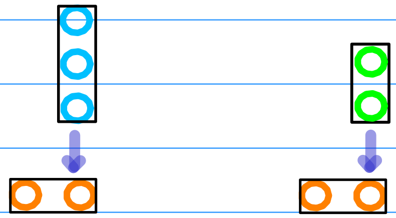
In fact, Dadda and Wallace multipliers have the same 3 steps:

1. Multiply (logical AND) each bit of one of the arguments, by each bit of the other, yielding n^2 results. Depending on position of the multiplied bits, the wires carry different weights, for example wire of bit carrying result of $a_2 b_3$ is 32.
2. Reduce the number of partial products to two by layers of full and half adders.
3. Group the wires in two numbers, and add them with a conventional adder.

https://en.wikipedia.org/wiki/Dadda_multiplier

However, unlike Wallace multipliers that reduce as much as possible on each layer, Dadda multipliers do as few reductions as possible. Because of this, Dadda multipliers have a less expensive reduction phase, but the numbers may be a few bits longer, thus requiring slightly bigger adders.

To achieve this, the structure of the second step is governed by slightly more complex rules than in the Wallace tree. As in the Wallace tree, a new layer is added if any weight is carried by three or more wires. The reduction rules for the Dadda tree, however, are as follows:



https://en.wikipedia.org/wiki/Dadda_multiplier

To achieve this, the structure of the second step is governed by slightly more complex rules than in the Wallace tree. As in the Wallace tree, a new layer is added if any weight is carried by three or more wires. The reduction rules for the Dadda tree, however, are as follows:

3 - Take any three wires with the same weights and input them into a full adder. The result will be an output wire of the same weight and an output wire with a higher weight for each three input wires.
2 - If there are two wires of the same weight left, and the current number of output wires with that weight is equal to 2 (modulo 3), input them into a half adder. Otherwise, pass them through to the next layer.
1 - If there is just one wire left, connect it to the next layer.

This step does only as many adds as necessary, so that the number of output weights stays close to a multiple of 3, which is the ideal number of weights when using full adders as 3:2 compressors.

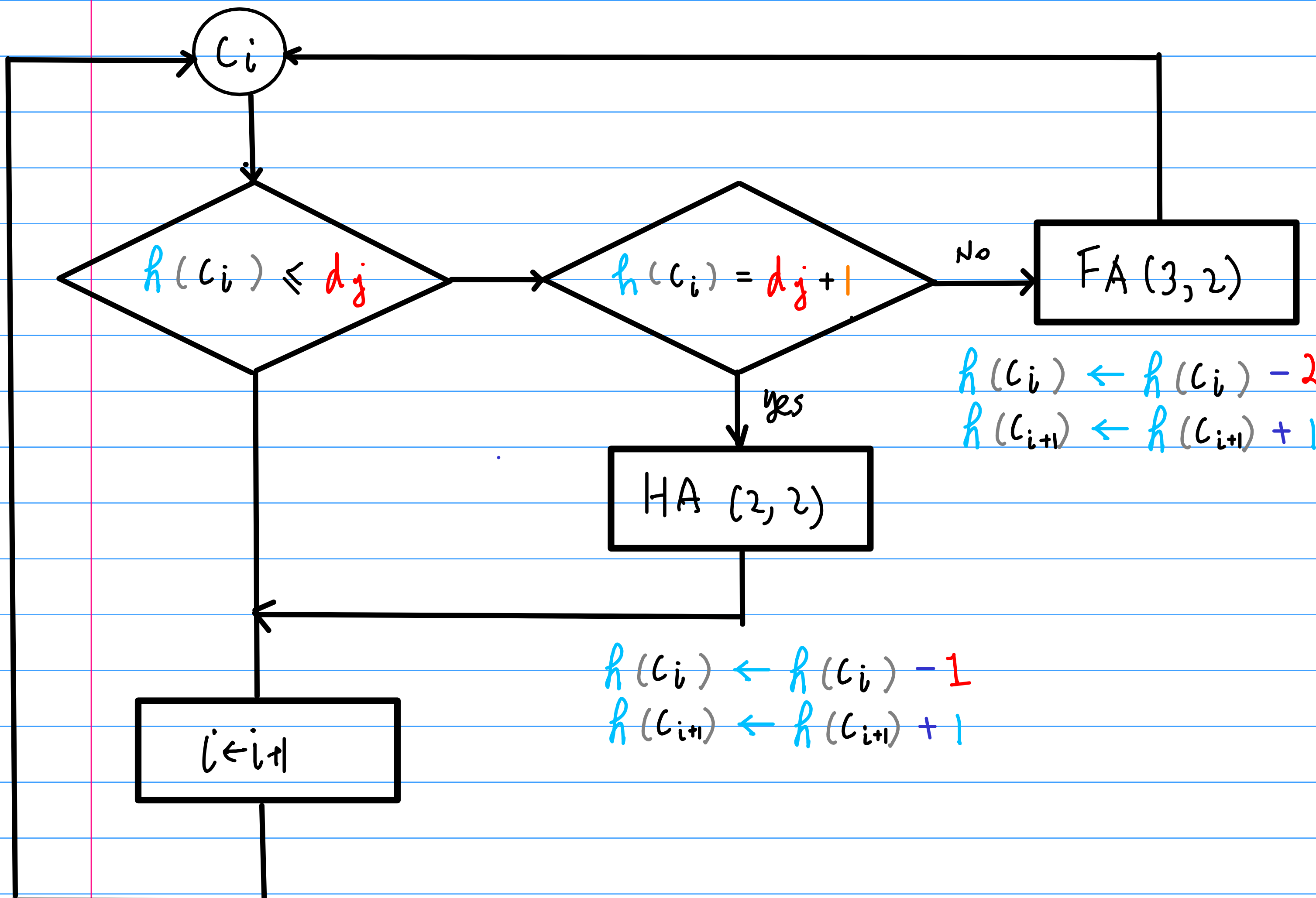
https://en.wikipedia.org/wiki/Dadda_multiplier

However, when a layer carries at most three input wires for any weight, that layer will be the last one. In this case, the Dadda tree will use half adder more aggressively (but still not as much as in a Wallace multiplier), to ensure that there are only two outputs for any weight. Then, the second rule above changes as follows:

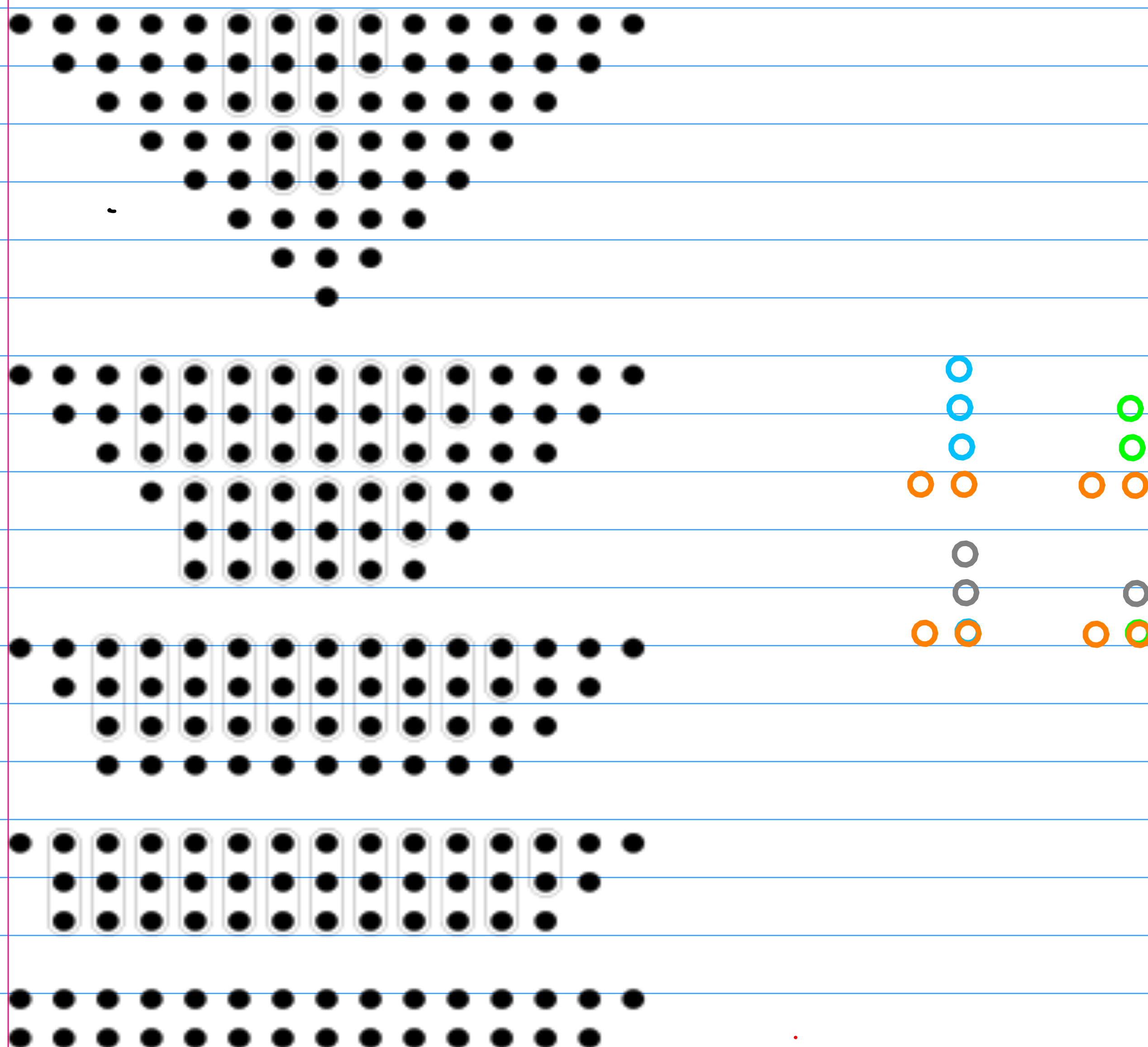
- If there are two wires of the same weight left, and the current number of output wires with that weight is equal to 1 or 2 (modulo 3), input them into a half adder. Otherwise, pass them through to the next layer.

https://en.wikipedia.org/wiki/Dadda_multiplier

i-th column

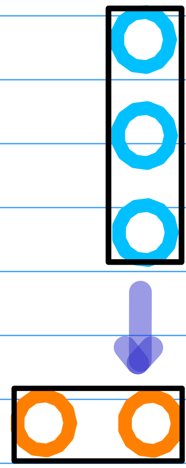


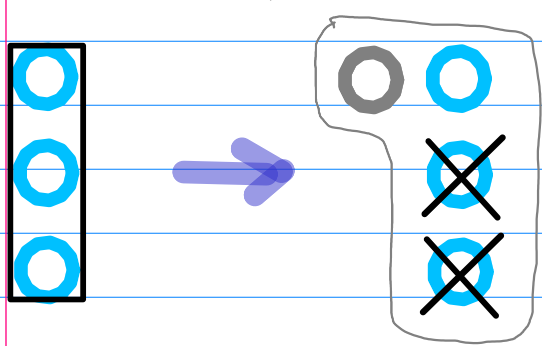
c_i
$h(c_i) \leq d_j$
$h(c_i) = d_j + 1$
No
FA (3, 2)
$h(c_i) \leftarrow h(c_i) - 2$
$h(c_{i+1}) \leftarrow h(c_{i+1}) + 1$
yes
HA (2, 2)
$h(c_i) \leftarrow h(c_i) - 1$
$h(c_{i+1}) \leftarrow h(c_{i+1}) + 1$
$i \leftarrow i+1$



https://en.wikipedia.org/wiki/Dadda_multiplier

(3, 2) FA

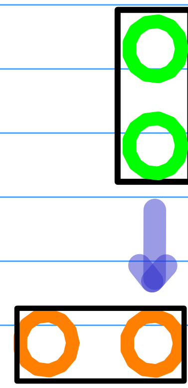


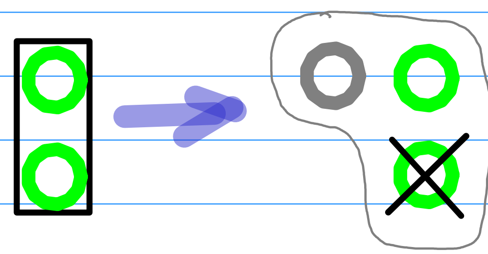


$h(c_i) \leftarrow h(c_i) - 2$

$h(c_{i+1}) \leftarrow h(c_{i+1}) + 1$

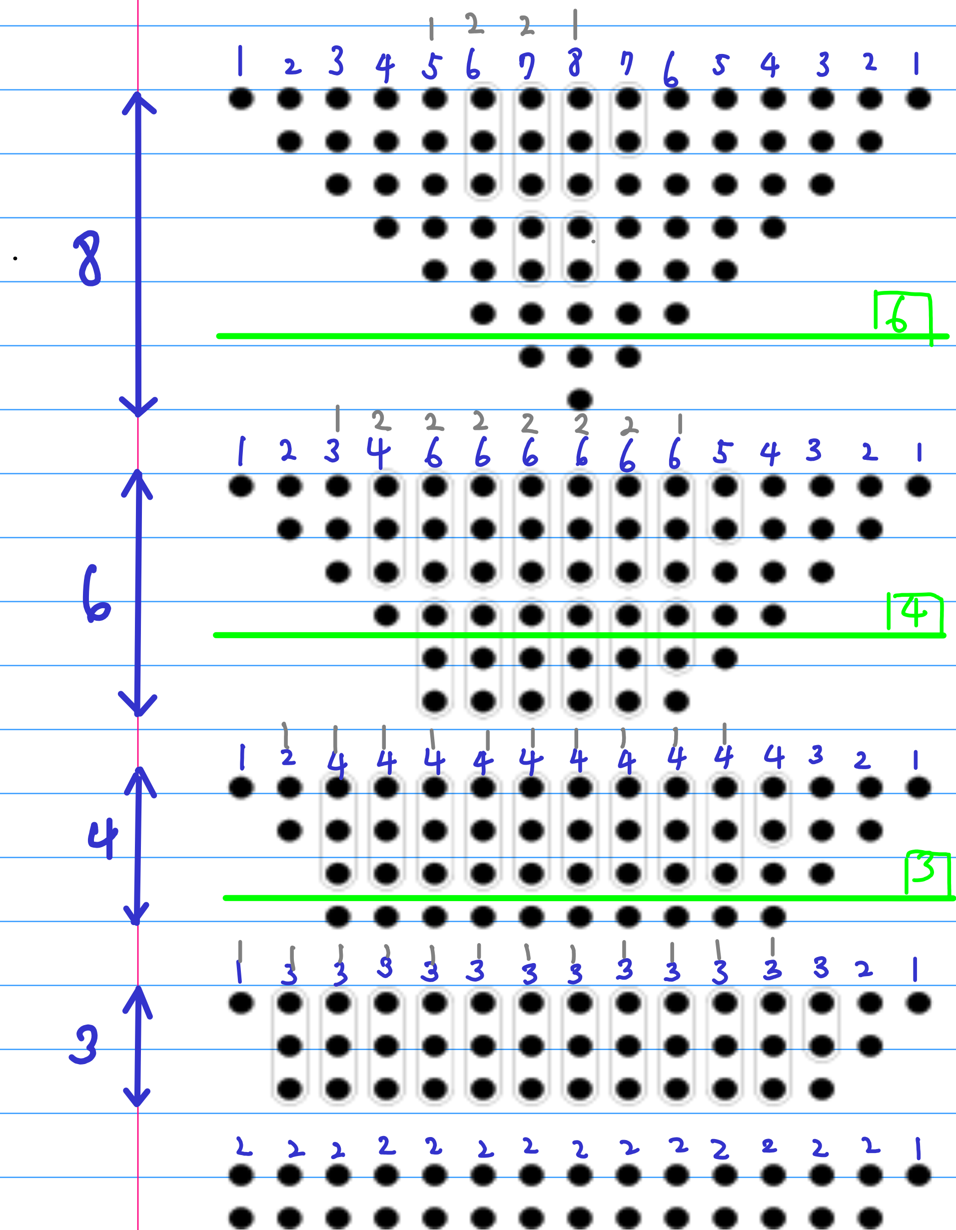
(2, 2) HA





$h(c_i) \leftarrow h(c_i) - 1$

$h(c_{i+1}) \leftarrow h(c_{i+1}) + 1$



7 → 6 (-1)
9 → 7 → 6 (-2) (-1)
8 → 6 (-2)

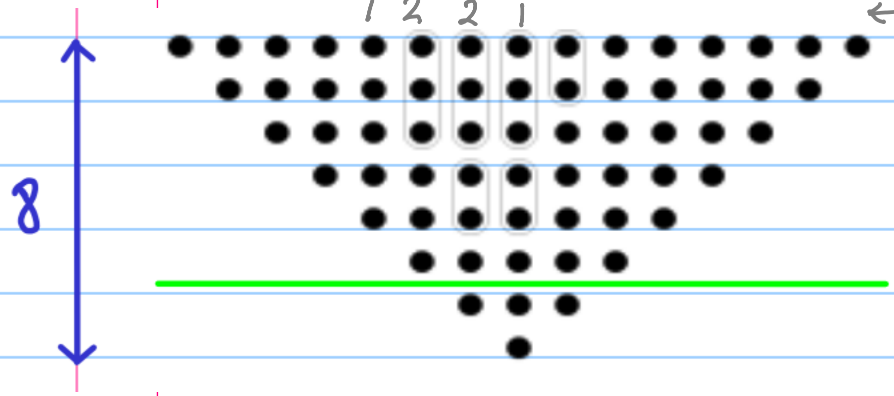
5 → 4 (-1)
7 → 5 → 4 (-2) (-1)
8 → 6 → 4 (-2) (-2)
6 → 4 (-2)

4 → 3 (-1)
5 → 3 (-2)

3 → 2 (-1)
4 → 2 (-2)

5 6 7 8 7 6

1 2 2 1



← carry = # of FA's HA's of the previous stage

$h(c_{i+1}) \leftarrow h(c_{i+1}) + 1$

$h(c_i) \leftarrow h(c_i) - 1$ **HA**

$h(c_i) \leftarrow h(c_i) - 2$ **FA**

$$
\begin{array}{cccc}
6 & 7 & 8 & 7 \\
2 & 2 & 1 & \\
\hline
8 & 9 & 9 & 7
\end{array}
$$

← orignal

← carry out

* 7 → 6   2 → 1 −1

* 9 → 7   3 → 1 −2

→ 6   2 → 1 −1

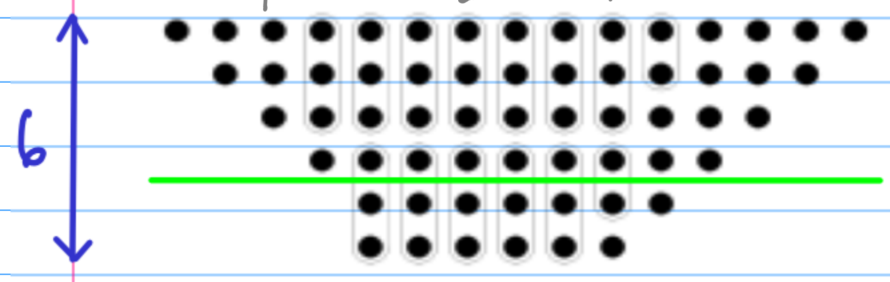
* 8 → 6   3 → 1 −2

3 4 6 6 6 6 6 6 5

1 2 2 2 2 2 2 1 ← carry = # of FA's HA's of the previous stage

$h(c_{i+1}) \leftarrow h(c_{i+1}) + 1$

6



$h(c_i) \leftarrow h(c_i) - 1$ **HA**

$h(c_i) \leftarrow h(c_i) - 2$ **FA**

4 6 6 6 6 6 6 5

2 2 2 2 2 2 1

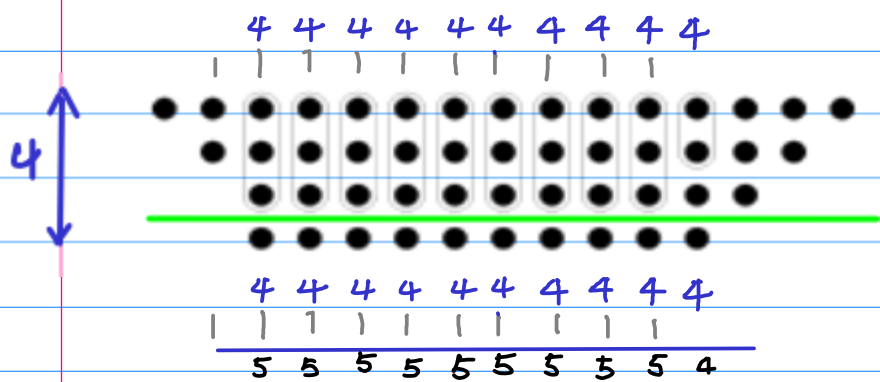
. 6 8 8 8 8 8 7 5

* 5 → 4   2 → 1   -1

* 7 → 5   3 → 1   -2
  → 4   2 → 1   -1

* 8 → 6   3 → 1   -2
  6 → 4   3 → 1   -2

* 6 → 4   3 → 1   -2



← Carry = # of FA's HA's of the previous stage

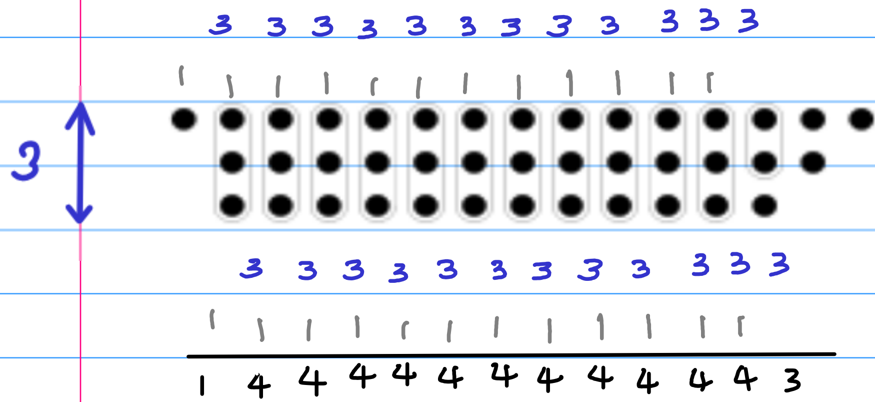
$h(c_{i+1}) \leftarrow h(c_{i+1}) + 1$

$h(c_i) \leftarrow h(c_i) - 1$ **HA**

$h(c_i) \leftarrow h(c_i) - 2$ **FA**

* $4 \rightarrow 3$ $\quad$ $2 \rightarrow 1$ $\quad$ -1

* $5 \rightarrow 3$ $\quad$ $3 \rightarrow 1$ $\quad$ -2



← carry = # of FA's HA's of the previous stage

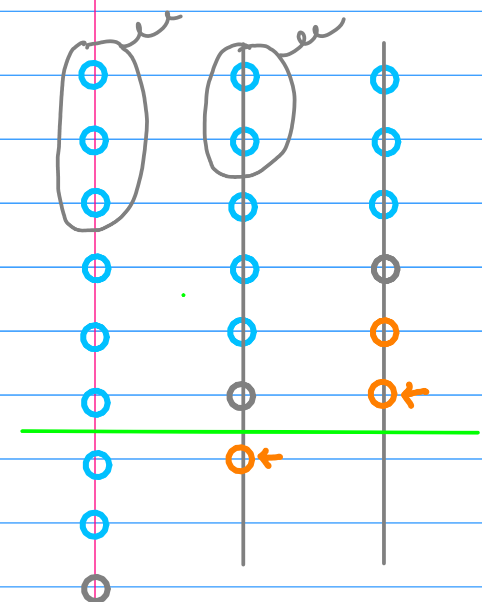
$h(c_{i+1}) \leftarrow h(c_{i+1}) + 1$

$h(c_i) \leftarrow h(c_i) - 1$ HA

$h(c_i) \leftarrow h(c_i) - 2$ FA

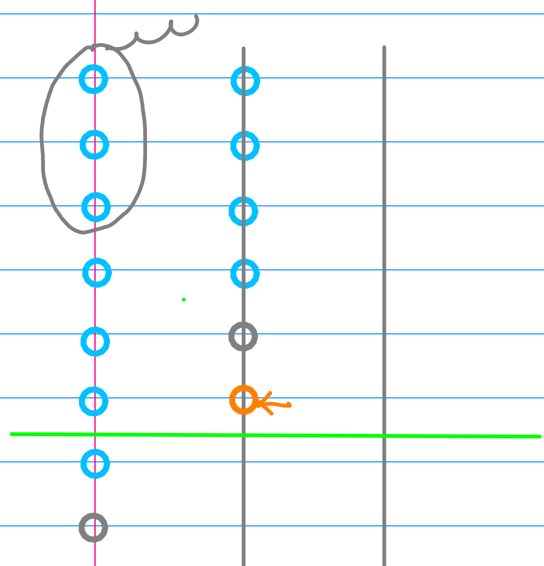
* $3 \rightarrow 2$ $\quad$ $2 \rightarrow 1$ (-1)

* $4 \rightarrow 2$ $\quad$ $3 \rightarrow 1$ (-2)



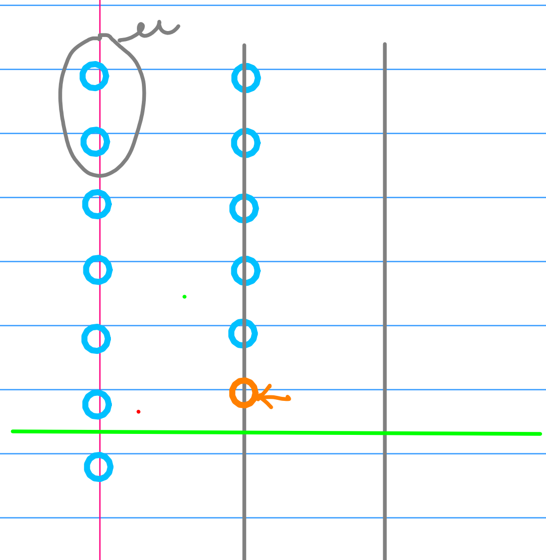
$9 \to 7 \to 6$

-3	-2
$+1$	$+1$
-2	-1
FA	HA



$8 \to 6$

-3
$+1$
-2



$7 \to 6$

-2
$+1$
-1

Maximum Height Sequence d_j

The progression of the reduction is controlled by a maximum-height sequence d_j, defined by:

$d_1 = 2$ and $d_{j+1} = floor(1.5 * d_j)$.

This yields a sequence like so:

$d_1 = 2, d_2 = 3, d_3 = 4, d_4 = 6, d_5 = 9, d_6 = 13, \ldots$

maximum height sequence

$$
\begin{aligned}
d_1 &= 2 \\
d_2 &= \left\lfloor \tfrac{3}{2} \cdot 2 \right\rfloor = 3 \\
d_3 &= \left\lfloor \tfrac{3}{2} \cdot 3 \right\rfloor = 4 \\
d_4 &= \left\lfloor \tfrac{3}{2} \cdot 4 \right\rfloor = 6 \\
d_5 &= \left\lfloor \tfrac{3}{2} \cdot 6 \right\rfloor = 9 \\
d_6 &= \left\lfloor \tfrac{3}{2} \cdot 9 \right\rfloor = 13
\end{aligned}
$$

$$2 \xrightarrow{\cdot \frac{3}{2}} 3 \xrightarrow{\cdot \frac{3}{2}} \frac{9}{2} \xrightarrow{\cdot \frac{3}{2}} \frac{27}{4} \xrightarrow{\cdot \frac{3}{2}} \frac{81}{8}$$

4.xx 6.xx 10.xx

$$2 \xrightarrow{\cdot \frac{3}{2}} \lfloor 3 \rfloor \xrightarrow{\cdot \frac{3}{2}} \left\lfloor \frac{9}{2} \right\rfloor \xrightarrow{\cdot \frac{3}{2}} \lfloor 6 \rfloor \xrightarrow{\cdot \frac{3}{2}} \lfloor 9 \rfloor \xrightarrow{\cdot \frac{3}{2}} \left\lfloor \frac{27}{2} \right\rfloor \xrightarrow{\cdot \frac{3}{2}} \left\lfloor \frac{39}{2} \right\rfloor$$

$$\left\lfloor \frac{9}{2} \right\rfloor = 4, \quad \left\lfloor \frac{27}{2} \right\rfloor = 13, \quad \left\lfloor \frac{39}{2} \right\rfloor = 19$$

$d_1 = 2$

$d_2 = \left\lfloor \frac{3}{2} \cdot 2 \right\rfloor = 3$

$d_3 = \left\lfloor \frac{3}{2} \cdot 3 \right\rfloor = 4$

$d_4 = \left\lfloor \frac{3}{2} \cdot 4 \right\rfloor = 6$

$d_5 = \left\lfloor \frac{3}{2} \cdot 6 \right\rfloor = 9$

$d_6 = \left\lfloor \frac{3}{2} \cdot 9 \right\rfloor = 13$

$$\left\lceil 13 \times \frac{2}{3} \right\rceil = \left\lceil \frac{26}{3} \right\rceil = 9$$

$$\left\lceil 9 \times \frac{2}{3} \right\rceil = 6$$

$$\left\lceil 6 \times \frac{2}{3} \right\rceil = 4$$

$$\left\lceil 4 \times \frac{2}{3} \right\rceil = \left\lceil \frac{8}{3} \right\rceil = 3$$

$$\left\lceil 3 \times \frac{2}{3} \right\rceil = 2$$

$d_1 = 2$

$d_2 = 3$

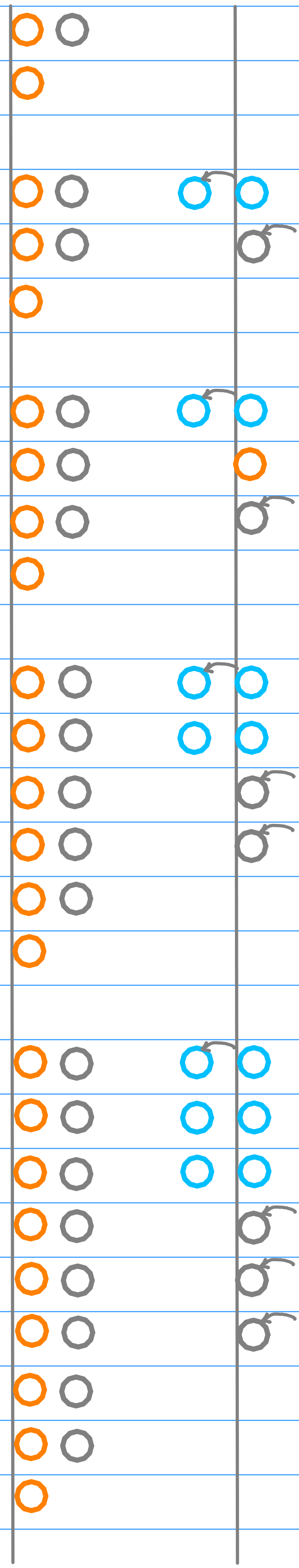
$d_3 = 4$

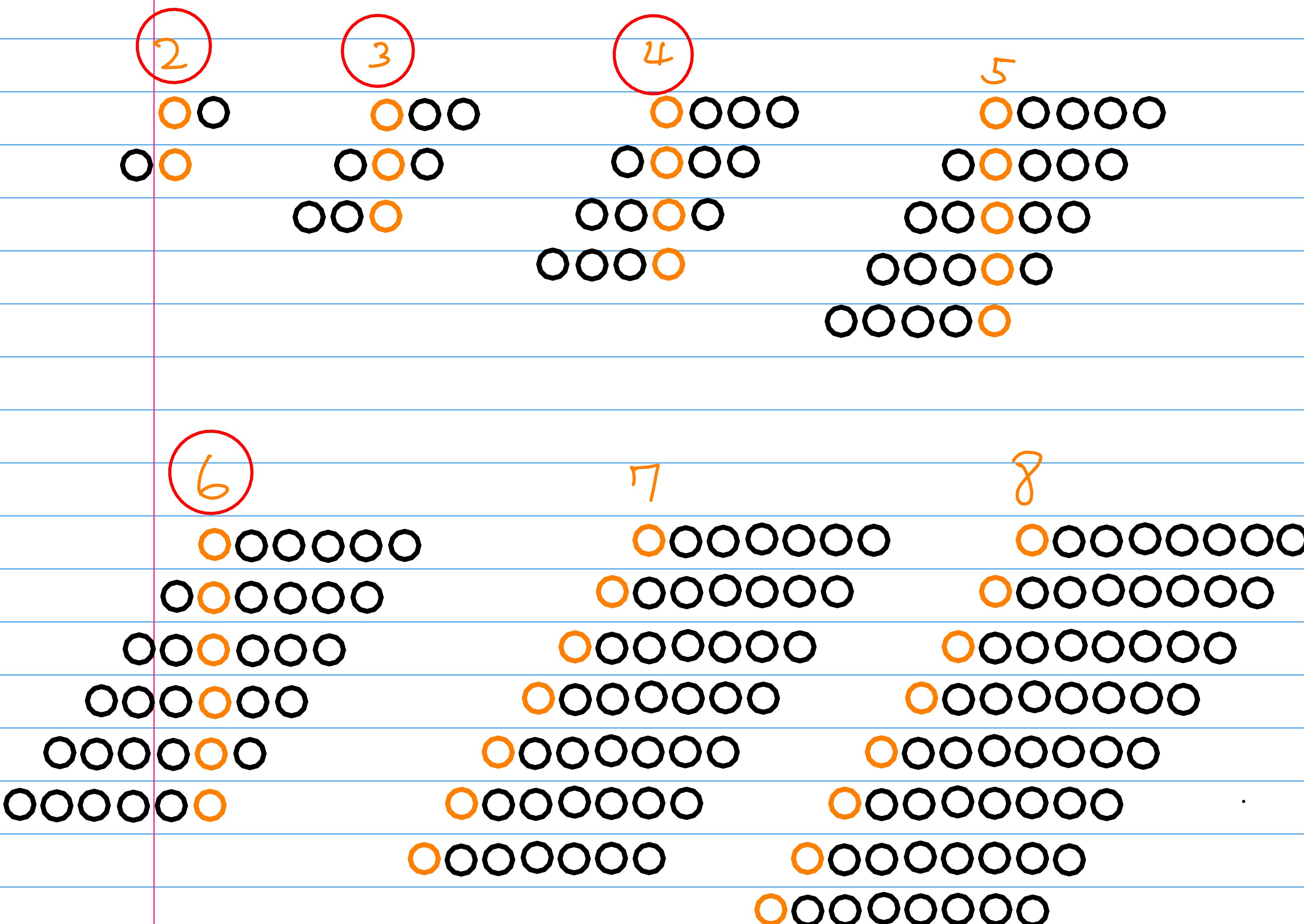
$d_4 = 6$

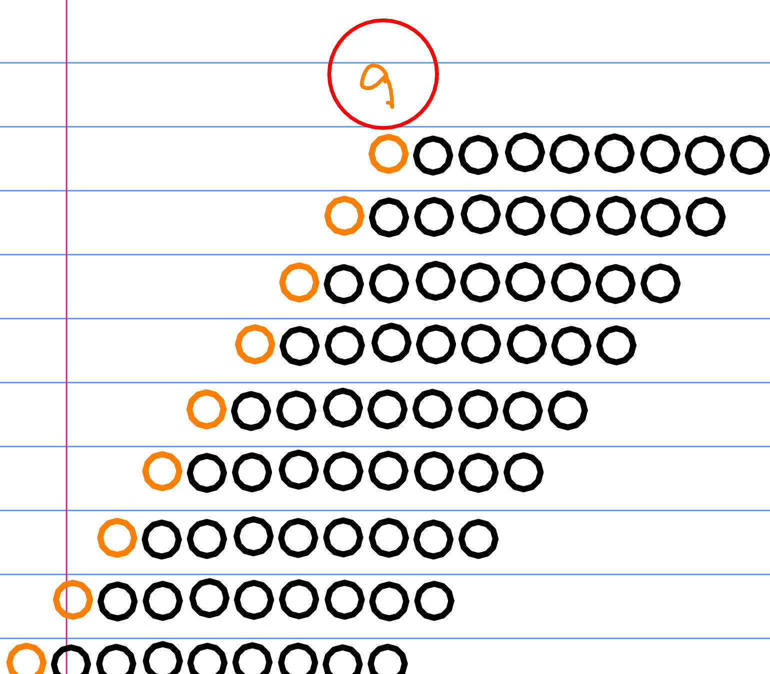
$5 \to 2$

$d_4 = 9$

$8 \to 3$







$d_1 = 2$

$d_2 = \lfloor \frac{3}{2} \cdot 2 \rfloor = 3$

$d_3 = \lfloor \frac{3}{2} \cdot 3 \rfloor = 4$

$d_4 = \lfloor \frac{3}{2} \cdot 4 \rfloor = 6$

$d_5 = \lfloor \frac{3}{2} \cdot 6 \rfloor = 9$

$d_6 = \lfloor \frac{3}{2} \cdot 9 \rfloor = 13$

$d_1 = 2$

$d_2 = \lfloor \frac{3}{2} \cdot 2 \rfloor = 3$

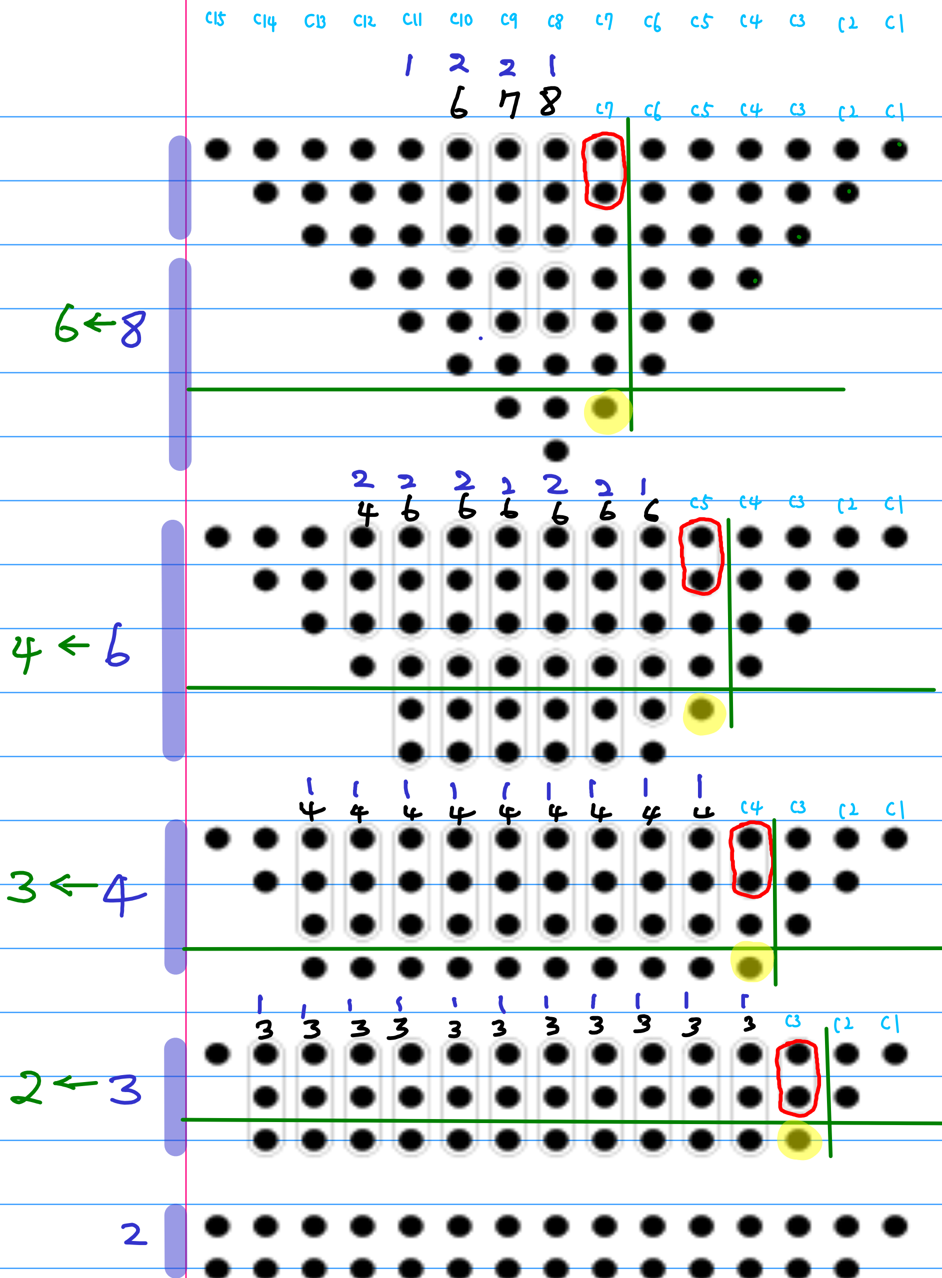
$d_3 = \lfloor \frac{3}{2} \cdot 3 \rfloor = 4$

$d_4 = \lfloor \frac{3}{2} \cdot 4 \rfloor = 6$

$d_5 = \lfloor \frac{3}{2} \cdot 6 \rfloor = 9$

$d_6 = \lfloor \frac{3}{2} \cdot 9 \rfloor = 13$

$d_6 = 13,\ d_5 = 9,\ d_4 = 6,\ d_3 = 4,\ d_2 = 3,\ d_1 = 2$



$C_7 = 6 + 1$
$= d_4 + 1$

$C_5 = 4 + 1$
$= d_3 + 1$

$C_4 = 3 + 1$
$= d_2 + 1$

$C_3 = 2 + 1$
$= d_1 + 1$

$d_1 = 2$

$d_2 = \lfloor \frac{3}{2} \cdot 2 \rfloor = 3$

$d_3 = \lfloor \frac{3}{2} \cdot 3 \rfloor = 4$

$d_4 = \lfloor \frac{3}{2} \cdot 4 \rfloor = 6$

$d_5 = \lfloor \frac{3}{2} \cdot 6 \rfloor = 9$

$d_6 = \lfloor \frac{3}{2} \cdot 9 \rfloor = 13$

Maximum Height Sequence d_j

the progression of reduction

— controlled by a maximum height sequence d_j

$$\boxed{\begin{aligned} d_1 &= 2 \\ d_{j+1} &= \lfloor 1.5 * d_j \rfloor \end{aligned}}$$

$$\begin{aligned}
d_1 &= 2 \\
d_2 &= \lfloor \tfrac{3}{2} \cdot 2 \rfloor = 3 \\
d_3 &= \lfloor \tfrac{3}{2} \cdot 3 \rfloor = 4 \\
d_4 &= \lfloor \tfrac{3}{2} \cdot 4 \rfloor = 6 \\
d_5 &= \lfloor \tfrac{3}{2} \cdot 6 \rfloor = 9 \\
d_6 &= \lfloor \tfrac{3}{2} \cdot 9 \rfloor = 13
\end{aligned}$$

The initial value of j is chosen as the largest value such that $d_j < max(n_1, n_2)$, where n_1 and n_2 are the number of bits in the input multiplicand and multiplier. The larger of the two bit lengths will be the maximum height of each column of weights after the first stage of multiplication. For each stage j of the reduction, the goal of the algorithm is the reduce the height of each column so that it is less than or equal to the value of d_j .

the height of each column $\leq d_j$

8×8 multiplication

$n_1 = n_2 = 8$

$j = 6 < max(8, 8) = 8$

$d_1 = 2$

$d_2 = \lfloor \frac{3}{2} \cdot 2 \rfloor = 3$

$d_3 = \lfloor \frac{3}{2} \cdot 3 \rfloor = 4$

$d_4 = \lfloor \frac{3}{2} \cdot 4 \rfloor = 6$

$d_5 = \lfloor \frac{3}{2} \cdot 6 \rfloor = 9$

$d_6 = \lfloor \frac{3}{2} \cdot 9 \rfloor = 13$

$d_6 = 13$

$d_5 = 9$

$d_4 = 6$

$d_3 = 4$

$d_2 = 3$

$d_1 = 2$

$d_j < \max(n_1, n_2)$

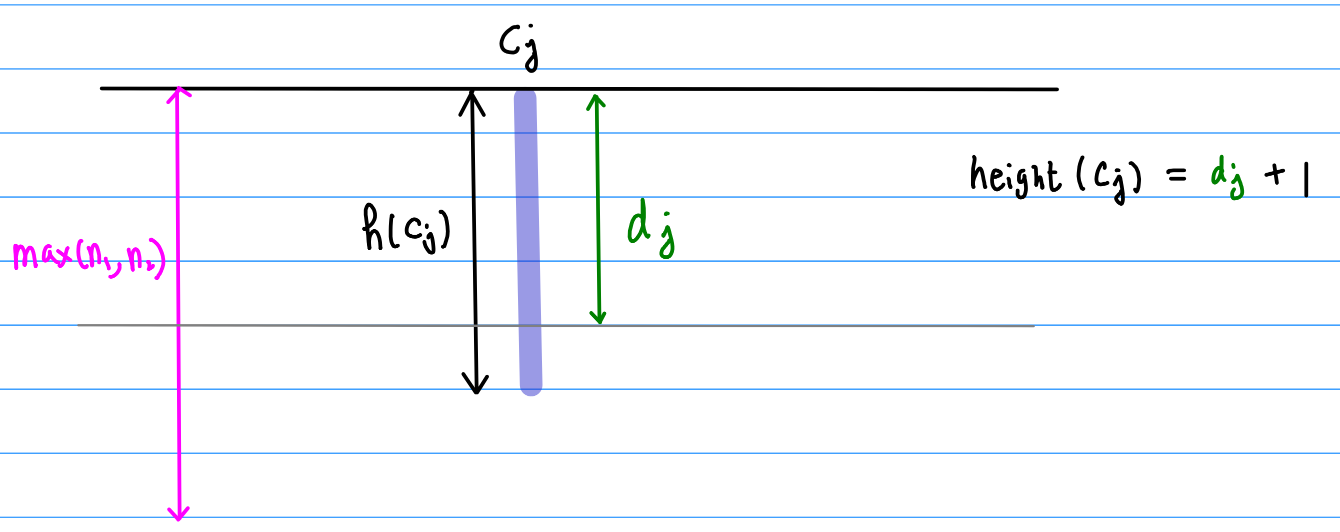
Stage $j \to 1$ decreasing order d_j : target height

For each stage from $j..1$, reduce each column starting at the lowest-weight column, c_0 according to these rules:

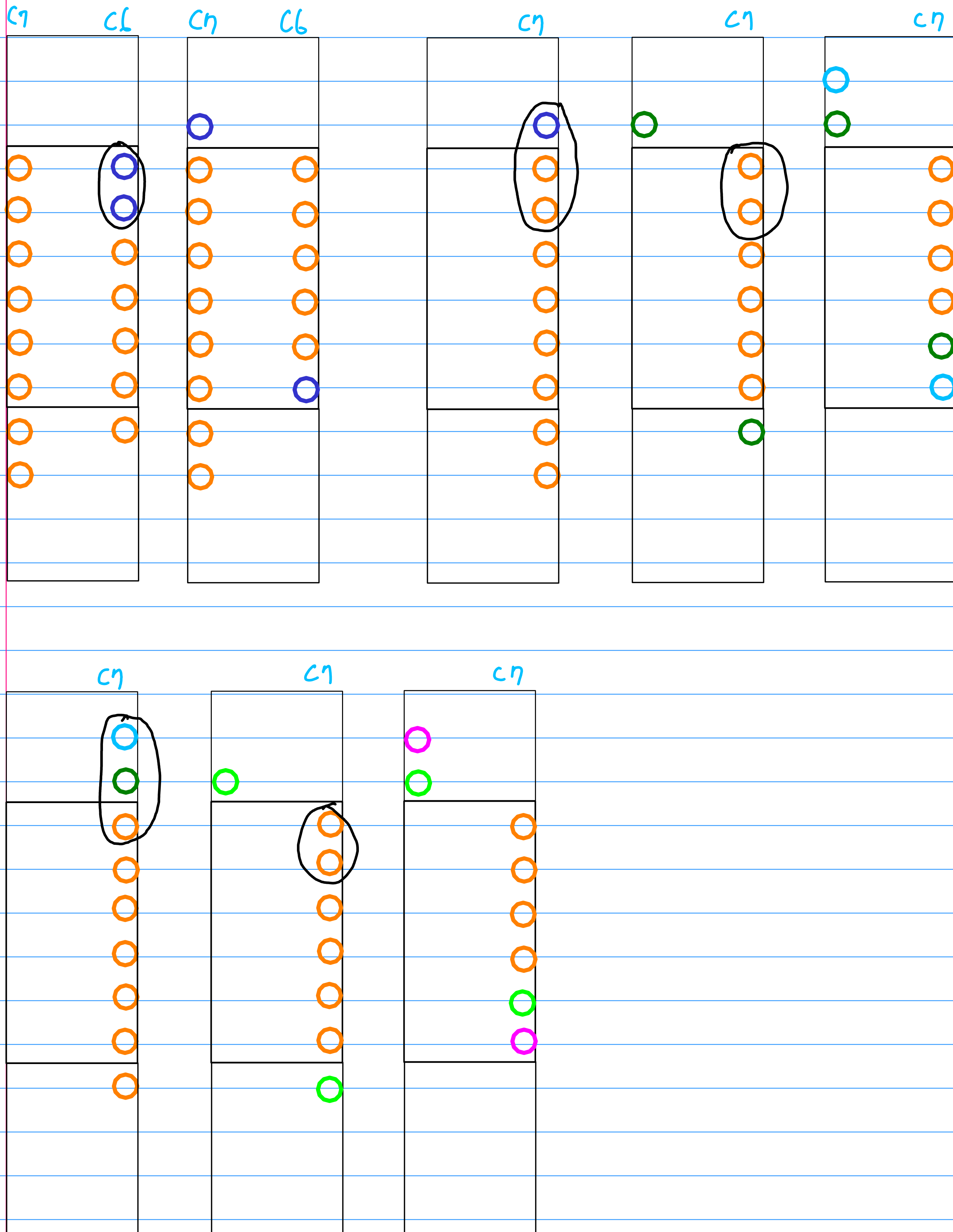
Column i increasing order

Start of reduction

1. If $height(c_i) \leqslant d_j$ the column does not require reduction, move to column c_{i+1}
2. If $\boxed{height(c_i) = d_j + 1}$ add the top two elements in a $\boxed{\text{half-adder}}$, placing the result at the bottom of the column and the carry at the top of column c_{i+1} , then move to column c_{i+1}
3. Else, add the top three elements in a full-adder, placing the result at the bottom of the column and the carry at the top of column c_{i+1} , restart c_i at step 1



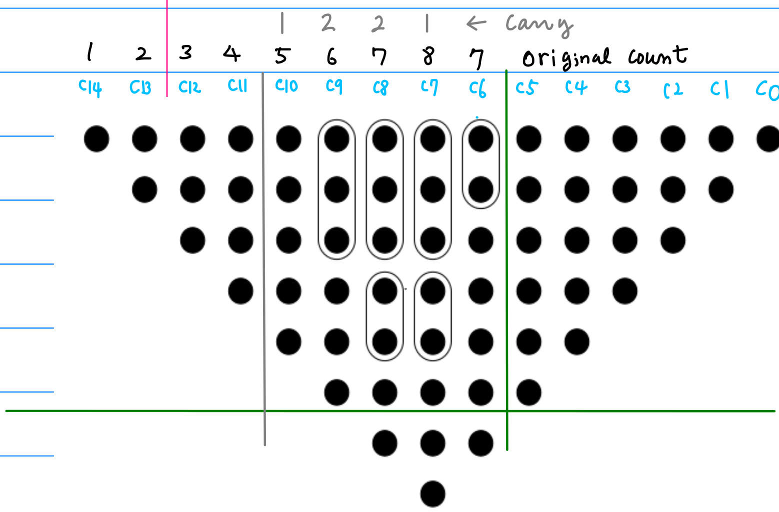
HA higher priority than FA

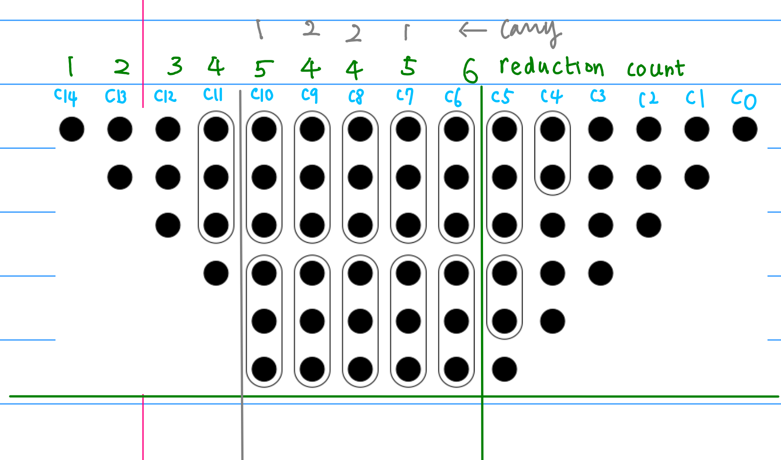

C7
C6
C7
C6
C7
C7
C7
C7
C7
C7

max height = 8 $\quad d_6 = 13,\ d_5 = 9,\ \boxed{d_4 = 6},\ d_3 = 4,\ d_2 = 3,\ d_1 = 2$

8

$\boxed{d_4 = 6} \quad j = 4 \quad h(c_6) = 6 + 1$

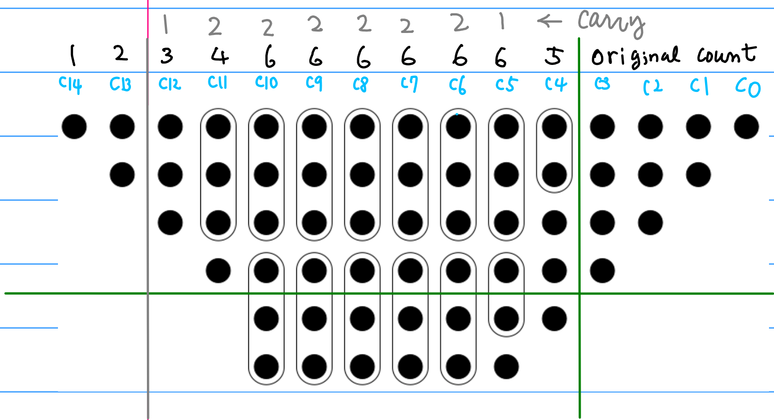


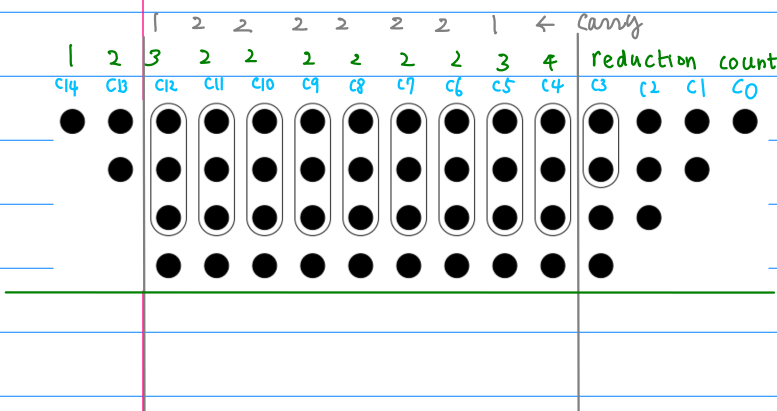


- $height(c_0..c_5)$ are all less than or equal to six bits in height, so no changes are made
- $height(c_6) = d_4 + 1 = 7$, so a half-adder is applied, reducing it to six bits and adding its carry bit to c_7
- $height(c_7) = 9$ including the carry bit from c_6 , so we apply ~~two full-adders~~ 1 FA & 1 HA to reduce it to six bits
- $height(c_8) = 9$ including two carry bits from c_7 , so we again apply ~~two full-adders~~ 1 FA & 1 HA to reduce it to six bits
- $height(c_9) = 8$ including two carry bits from c_8 , so we apply a single full-adder and reduce it to six bits
- $height(c_{10}..c_{14})$ are all less than or equal to six bits in height including carry bits, so no changes are made

max height = 6

$d_6 = 13$, $d_5 = 9$, $d_4 = 6$, $\boxed{d_3 = 4}$, $d_2 = 3$, $d_1 = 2$





$\boxed{d_3 = 4}$ $j = 3$ $h(c_4) = 4 + 1$

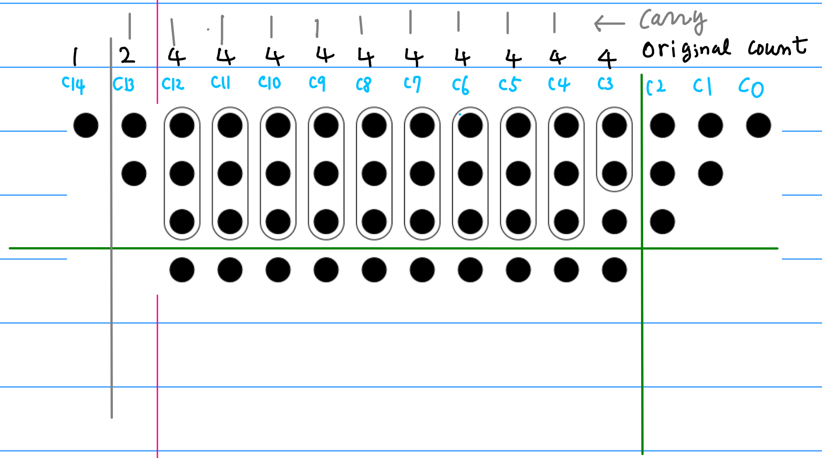
Stage $j = 3$, $d_3 = 4$

- $height(c_0..c_3)$ are all less than or equal to four bits in height, so no changes are made
- $height(c_4) = d_3 + 1 = 5$, so a half-adder is applied, reducing it to four bits and adding its carry bit to c_5
- $height(c_5) = 7$ including the carry bit from c_4 , so we apply a full-adder and a half-adder to reduce it to four bits
- $height(c_6..c_{10}) = 8$ including previous carry bits, so we apply two full-adders to reduce them to four bits
- $height(c_{11}) = 6$ including previous carry bits, so we apply a full-adder to reduce it to four bits
- $height(c_{12}..c_{14})$ are all less than or equal to four bits in height including carry bits, so no changes are made

max height = 4

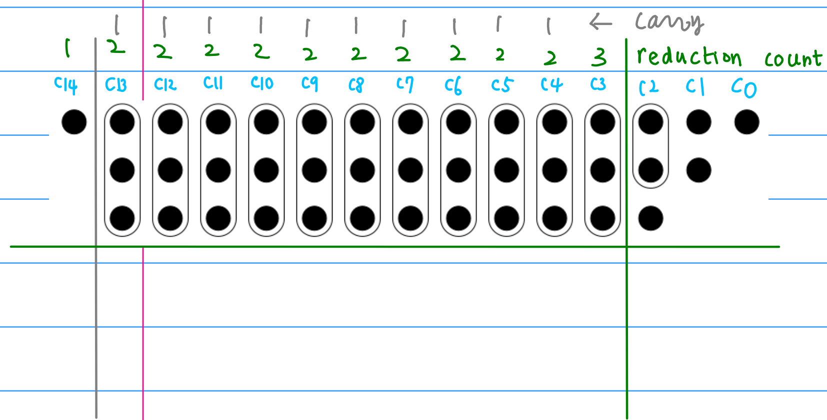
$d_6 = 13,\ d_5 = 9,\ d_4 = 6,\ d_3 = 4,\ \boxed{d_2 = 3},\ d_1 = 2$

$\boxed{d_2 = 3}$ $j = 2$ $h(c_3) = 3 + 1$



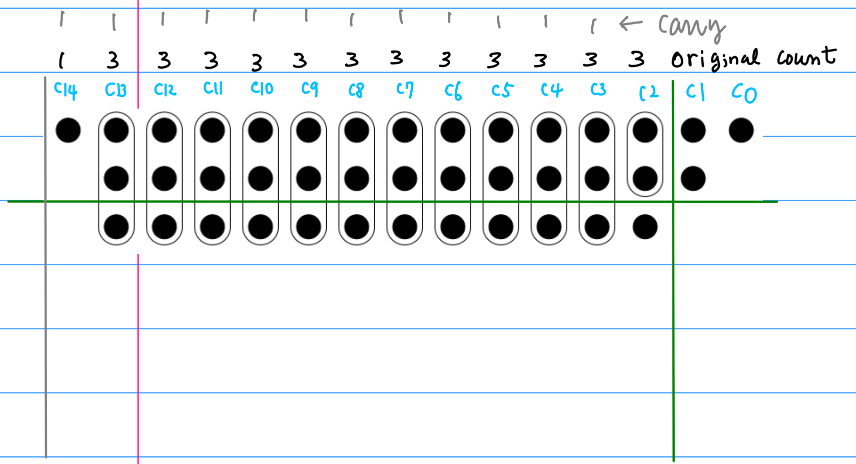
Stage $j = 2$, $d_2 = 3$

- $height(c_0..c_2)$ are all less than or equal to three bits in height, so no changes are made
- $height(c_3) = d_2 + 1 = 4$, so a half-adder is applied, reducing it to three bits and adding its carry bit to c_4
- $height(c_4..c_{12}) = 5$ including previous carry bits, so we apply one full-adder to reduce them to three bits
- $height(c_{13}..c_{14})$ are all less than or equal to three bits in height including carry bits, so no changes are made



max height = 3

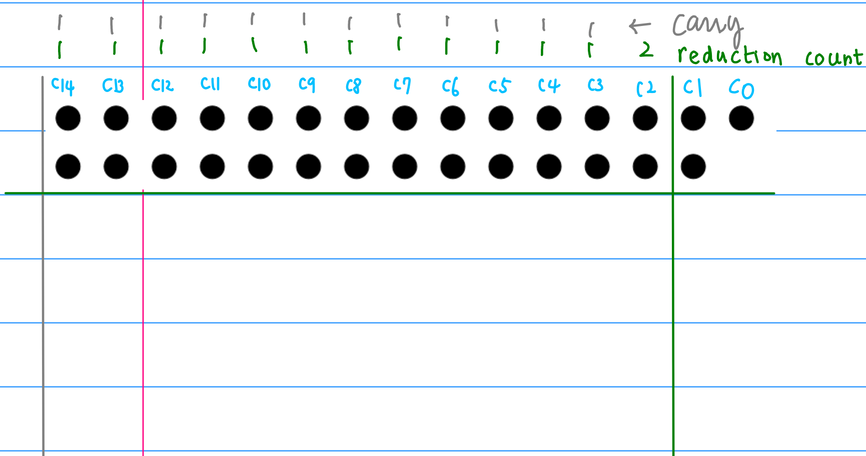
$d_6 = 13,\ d_5 = 9,\ d_4 = 6,\ d_3 = 4,\ d_2 = 3,\ \boxed{d_1 = 2}$

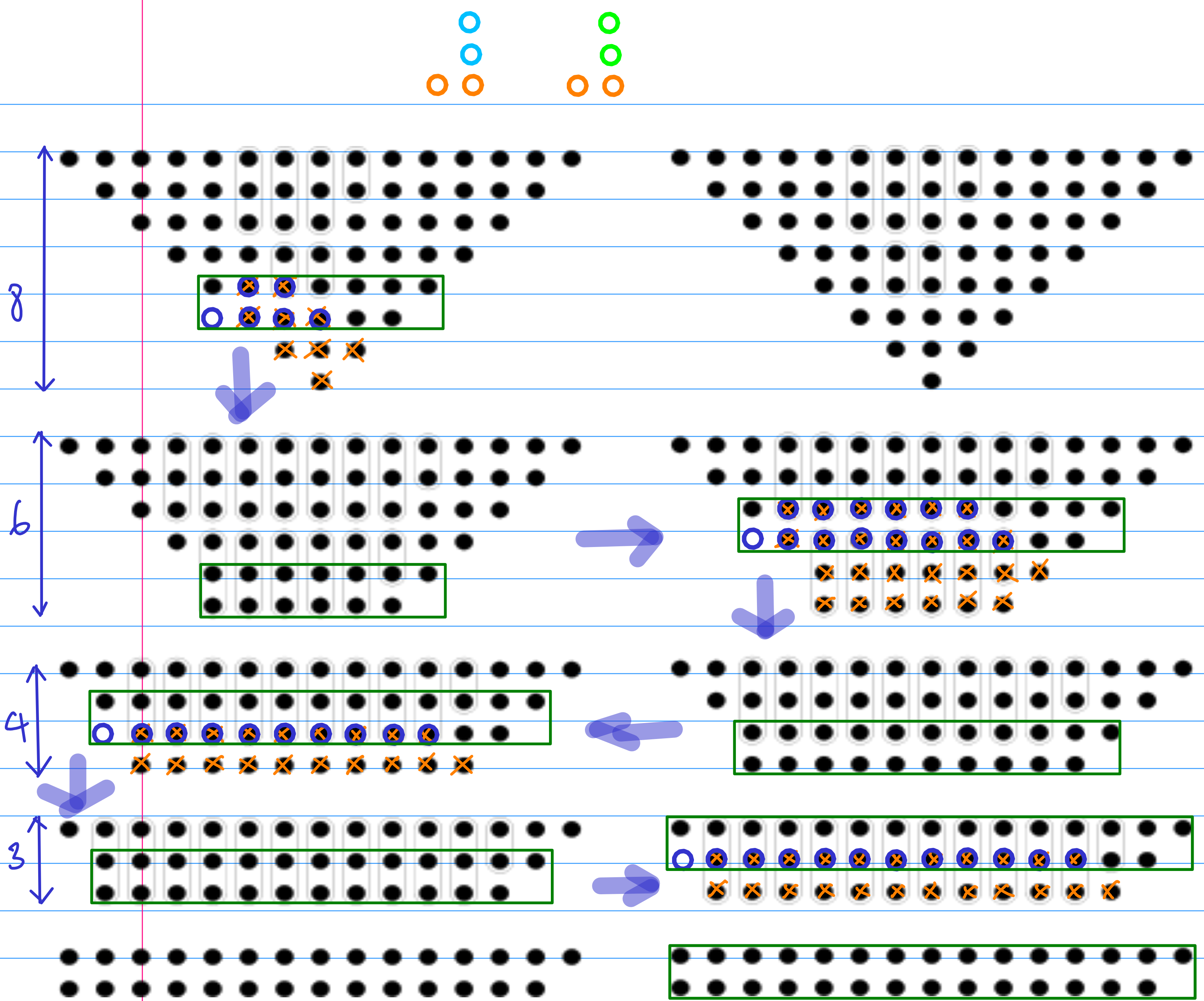


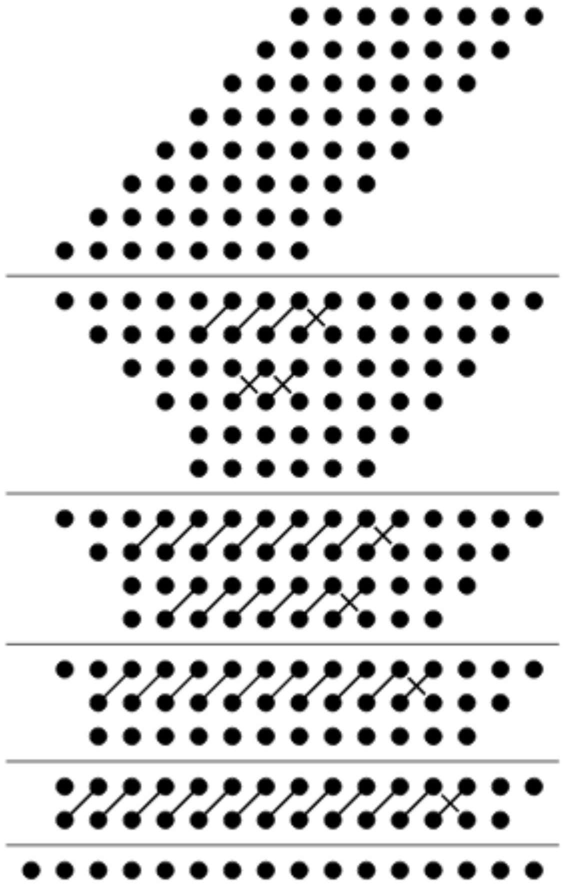
$\boxed{d_1 = 2}$ $j = 1$ $h(c_2) = 2 + 1$

Stage $j = 1\,,\ d_1 = 2$

- $height(c_0..c_1)$ are all less than or equal to two bits in height, so no changes are made
- $height(c_2) = d_1 + 1 = 3$, so a half-adder is applied, reducing it to two bits and adding its carry bit to c_3
- $height(c_3..c_{13}) = 4$ including previous carry bits, so we apply one full-adder to reduce them to two bits
- $height(c_{14}) = 2$ including the carry bit from c_{13} , so no changes are made




8
6
4
3



http://ieeemilestones.ethw.org/images/d/db/A_comparison_of_Dadda_and_Wallace_multiplier_delays.pdf