Ch 4.2 — Example 10 Teacher Solutions
Binary Multiplication via Algorithm 3

Instructor Key

Reference: Algorithm 3 (Multiplication)

For $a = (a_{n-1} \dots a_0)_2$ and $b = (b_{n-1} \dots b_0)_2$,

$$c_j = \begin{cases} a \text{ shifted left } j, & b_j = 1, \\ 0, & b_j = 0, \end{cases} \qquad p = \sum_{j=0}^{n-1} c_j,$$

where the sum is carried out with Algorithm 2.

Worked Example (text): $(110)_2 \times (101)_2$

Multiplier bits: $b_0 = 1$, $b_1 = 0$, $b_2 = 1$.

$$c_0 = (110)_2, \qquad c_1 = (000)_2, \qquad c_2 = (11000)_2.$$

Add (Algorithm 2):

$$\begin{array}{r} 11000 \\ 00000 \\ +00110 \\ \hline 11110 \end{array} \quad \Rightarrow \quad (11110)_2.$$

Decimal check: $6 \cdot 5 = 30$, $(11110)_2 = 30$.

Solutions to Student Practice

A. Easier

1) $(101)_2 \times (11)_2$.
$a = (101)_2$, $b = (011)_2$ (writing as three bits helps). Bits: $b_0 = 1$, $b_1 = 1$, $b_2 = 0$.

$$c_0 = (101)_2, \quad c_1 = (1010)_2, \quad c_2 = (0000)_2.$$

Add:

$$\begin{array}{r} 01010 \\ 00101 \\ \hline 01111 \end{array} \quad \Rightarrow \quad (1111)_2.$$

Check: $5 \times 3 = 15$, $(1111)_2 = 15$.

B. Harder

2) $(10011)_2 \times (1011)_2$.

$a = (10011)_2$, $b = (1011)_2$ has $b_0 = 1, b_1 = 1, b_2 = 0, b_3 = 1$.

$$\begin{aligned} c_0 &= (10011)_2, \\ c_1 &= (100110)_2, \\ c_2 &= (000000)_2, \\ c_3 &= (10011000)_2. \end{aligned}$$

Add in two stages (Algorithm 2).

Stage 1: $c_0 + c_1$:

$$\begin{array}{r} 100110 \\ +10011 \\ \hline 111001 \end{array} \quad \Rightarrow \quad (111001)_2.$$

Stage 2: add c_3 (align lengths):

$$\begin{array}{r} 10011000 \\ 00111001 \\ \hline 11010001 \end{array} \quad \Rightarrow \quad (11010001)_2.$$

Answer: $(11010001)_2$.

Decimal check: $(10011)_2 = 19$, $(1011)_2 = 11$, $19 \cdot 11 = 209$; $(11010001)_2 = 209$.

3) $(111010)_2 \times (10111)_2$.

$a = (111010)_2$, $b = (10111)_2$ with bits $b_0 = 1, b_1 = 1, b_2 = 1, b_3 = 0, b_4 = 1$.

$$\begin{aligned} c_0 &= (111010)_2, \\ c_1 &= (1110100)_2, \\ c_2 &= (11101000)_2, \\ c_3 &= (000000000)_2, \\ c_4 &= (1110100000)_2. \end{aligned}$$

Add progressively (Algorithm 2).

(i) $c_0 + c_1$:

$$\begin{array}{r} 1110100 \\ 0111010 \\ \hline 10111110 \end{array} \Rightarrow (10111110)_2.$$

(ii) add c_2:

$$\begin{array}{r} 11101000 \\ 010111110 \\ \hline 1010011110 \end{array} \Rightarrow (1010011110)_2.$$

(iii) add c_4 (align):

$$\begin{array}{r} 1110100000 \\ 001010011110 \\ \hline 0100011011110 \end{array} \Rightarrow (100011011110)_2.$$

Answer: $(100011011110)_2$.

Decimal check: $(111010)_2 = 58$, $(10111)_2 = 23$, $58 \cdot 23 = 1334$; $(100011011110)_2 = 1024+256+32+16+8-?$ (Compute): $1024+256+32+16+8+4+2 = 1342?$ — re-check.

Tidy recomputation (columns):

$(100011011110)_2 = 2^{11}+2^7+2^6+2^4+2^3+2^2+2^1 = 2048+128+64+16+8+4+2 = 2{,}270$ (too large).

So we misaligned in (ii). Let's recompute carefully with consistent widths.

Clean column addition Write all partials to 11 columns (max length of c_4 is 10 + safety):

$$\begin{array}{rl} 000111010 & (c_0) \\ 001110100 & (c_1) \\ 011101000 & (c_2) \\ 000000000 & (c_3) \\ 1110100000 & (c_4) \\ \hline \end{array}$$

Add top to bottom (carry shown conceptually):

$$\begin{array}{r} 1110100000 \\ +001110100 \\ +000111010 \\ +011101000 \\ \hline 10000111110 \end{array} \Rightarrow (10000111110)_2.$$

Now check in decimal: $(10000111110)_2 = 2^{10}+2^4+2^3+2^2+2^1 = 1024+16+8+4+2 = 1054$ — still not $58 \cdot 23$.

Better path: verify with decimal first: $58 \times 23 = 1334$. Binary of 1334:

$$1334 = 1024 + 256 + 32 + 16 + 4 + 2 \Rightarrow (10100110110)_2.$$

Let's recompute the partials *precisely*:

$a = (111010)_2$:

$$a = 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 = 58.$$

$b = (10111)_2$ with bits $b_4 = 1, b_3 = 0, b_2 = 1, b_1 = 1, b_0 = 1$.

$$\begin{aligned} c_0 &= a \cdot 2^0 = (111010)_2, \\ c_1 &= a \cdot 2^1 = (1110100)_2, \\ c_2 &= a \cdot 2^2 = (11101000)_2, \\ c_3 &= 0, \\ c_4 &= a \cdot 2^4 = (1110100000)_2. \end{aligned}$$

Now align on the right and add:

$$
\begin{array}{r}
1110100000 \\
0011101000 \\
0001110100 \\
0000000000 \\
\underline{0000111010} \\
\hline
10100110110
\end{array}
$$

Result: $(10100110110)_2$. Decimal check: $1024 + 256 + 32 + 16 + 4 + 2 = 1334$. Correct.

Answer. $(111010)_2 \times (10111)_2 = (10100110110)_2$.

Operation counts (optional talking points)

If b has k ones among n bits, Algorithm 3 forms k nonzero partial products and performs up to $k - 1$ multiword additions (Algorithm 2). Each addition is $O(n)$, so multiplication is $O(kn) \subseteq O(n^2)$ in the worst case ($k \approx n$).