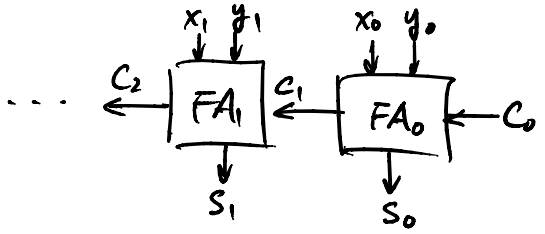
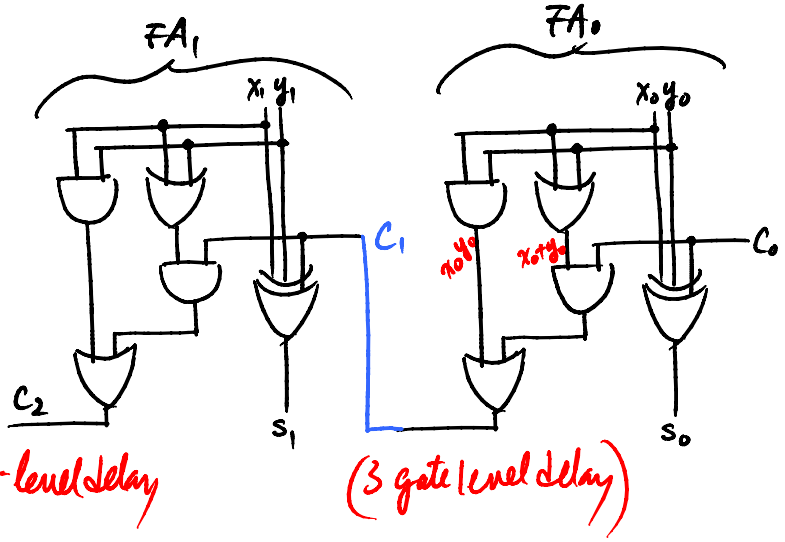
3.4 Carry-ripple adder

$$S_0 = x_0 \oplus y_0 \oplus C_0$$
$$C_1 = x_0 y_0 + x_0 C_0 + y_0 C_0 = x_0 y_0 + (x_0 + y_0) C_0$$
$$S_1 = x_1 \oplus y_1 \oplus C_1$$
$$C_2 = x_1 y_1 + (x_1 + y_1) C_1$$



For a 16-bit carry-ripple adder:

for C_{16}: ($16 \times 2 + 1 = 33$ gate-level delay)



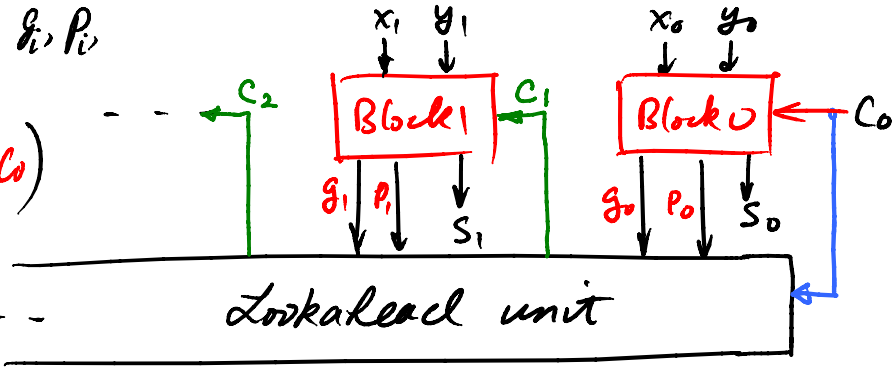
Lookahead adder (aka fast adder)

$$S_0 = x_0 \oplus y_0 \oplus C_0$$
$$C_1 = x_0 y_0 + x_0 C_0 + y_0 C_0 = x_0 y_0 + (x_0 + y_0) C_0 \rightarrow C_1 = g_0 + p_0 C_0$$
$$S_1 = x_1 \oplus y_1 \oplus C_1$$
$$C_2 = x_1 y_1 + (x_1 + y_1) C_1 = \underbrace{x_1 y_1}_{g_1} + \underbrace{(x_1 + y_1)}_{p_1}\left[\underbrace{x_0 y_0}_{g_0} + \underbrace{(x_0 + y_0)}_{p_0} C_0\right] = g_1 + p_1 g_0 + p_1 p_0 C_0$$

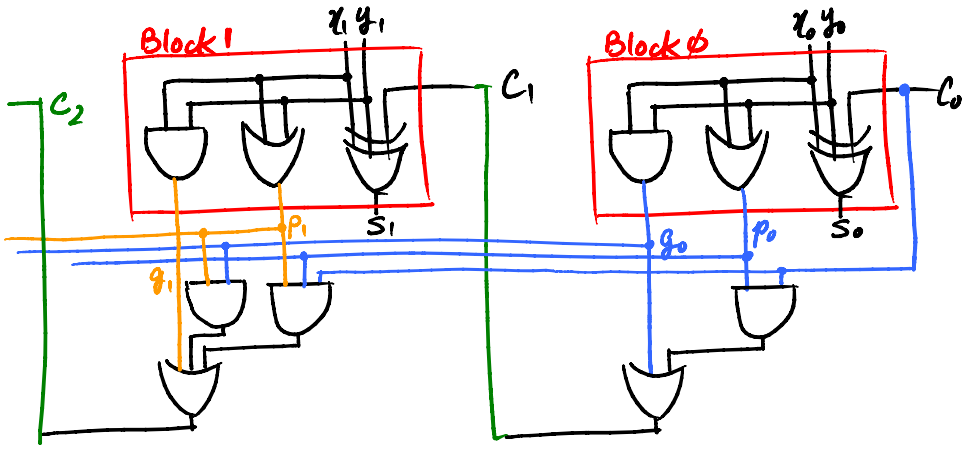
→ 1 gate-level delay to produce g_i, p_i

→ 1 gate-level delay to produce all product terms (i.e. $p_1 g_0 C_0$)

→ 1 gate-level delay to produce the sum term.



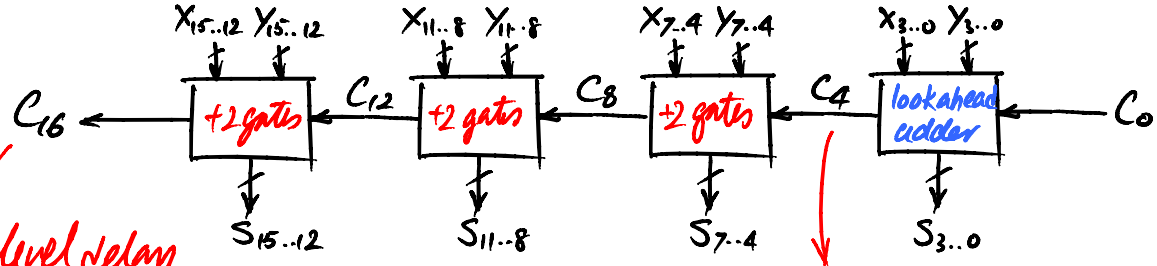
$\Rightarrow$ in total 3 gate-level delay for any C_i (carry bit)



Pros: faster

Cons: large number of inputs to the gates. ← fan-in

Option1 Combination of lookahead and carry-ripple



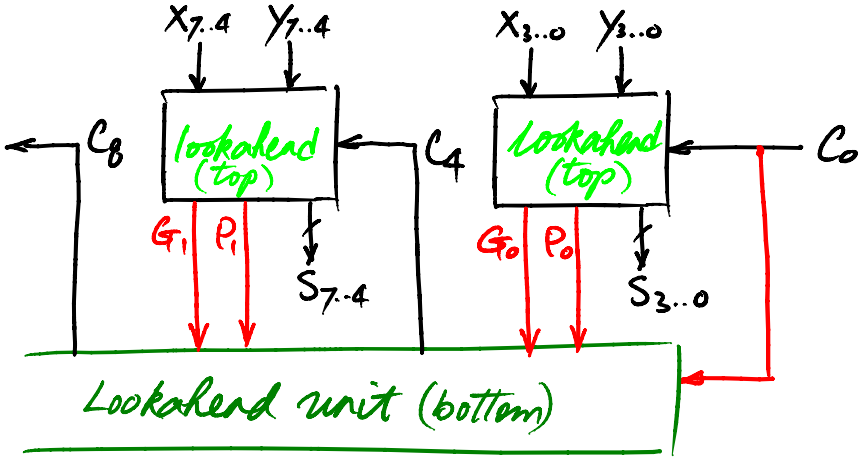
9 gate-level delay

3 gate-level delay

→ reduced fan-in

→ increased delay (trade-off)

Option2 Hierachical lookahead



carry-ripple Lookahead

$$C_4 = x_3y_3 + (x_3+y_3)C_3 = g_3 + p_3C_3 = \underbrace{g_3 + p_3g_2 + p_3p_2g_1 + p_3p_2p_1g_0}_{G_0} + \underbrace{p_3p_2p_1p_0}_{P_0}C_0$$

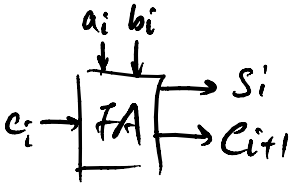
lookahead (Top) → produces G_0, P_0 in 3 gate-level delay
1 for g_i, p_i, 1 for product terms (i.e. $p_3p_2p_1g_0$), 1 for $G_0 + P_0C_0$

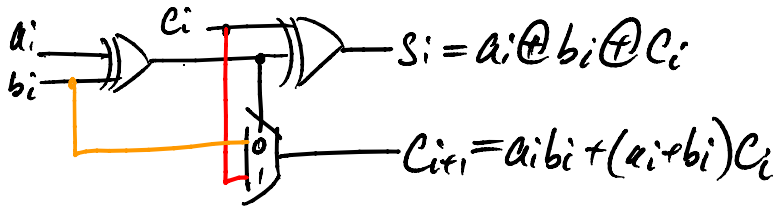
lookahead (bottom) → produces $C_4 = G_0 + P_0C_0$ in 2 gate-level delay

in total 5 gate-level delay to generate C_{16}

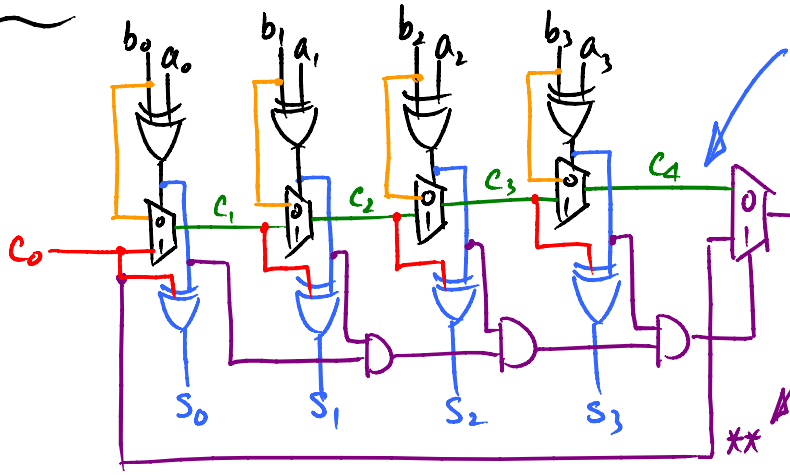
Contrast: 16 bit adding, to generate C_{16}, gate-level delay
- carry-ripple → 33
- lookahead → 3 (large fan-in)
- ripple-lookahead → 9 (fan-in ≤ 5)
- hierachical lookahead → 5 (fan-in ≤ 5)

→ FA





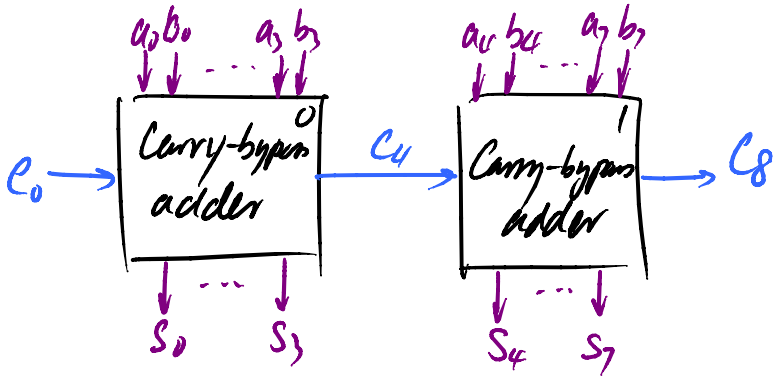
4 bit adder



carry-ripple path for C_4 any $a_i = b_i$

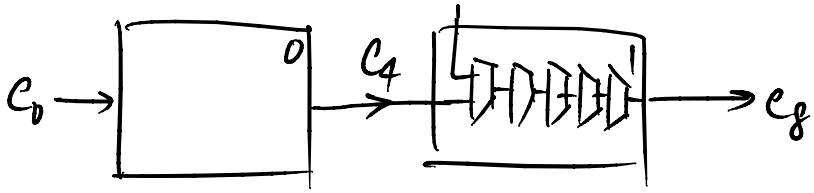
if all $a_i \neq b_i$, ie 1,0 or 0,1 combination bypass is faster than the carry-ripple. $C_4 = C_0$

build a 8-bit adder.



Q: What is the worst case delay through the 8-bit adder. (c_8)?

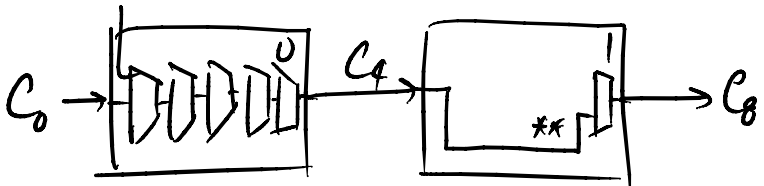
A: Case1 for any one of $a_i = b_i$ in block①



delays in c_4 doesn't get through to c_8.

5  delays. in total

Case2 for all $a_i \neq b_i$, (1,0) or (0,1) in block①



Worst-case delay in block0

bypass path in block1

6  delays. in total