Design Problem

Design a 4-bit incremerter; a circuit that increments a 4-bit # (\(A_3, A_2, A_1, A_0\)) by one using 4 1-bit adders.

Carry Propagation 1

Addition of two numbers in parallel implies that all bits are available for computation.

Total propagation delay = propagation delay of a gate x # gate levels.

![4-bit Adder Diagram](image)
Q: Find the total Carry propagation delay in the 4-bit full adder circuit. Delay (XOR) = 20 ns, Delay (AND) = Delay (OR) = 10 ns.

The carry propagation time is a limiting factor on the speed with which two numbers are added.

The most widely technique for reducing the carry propagation time in a parallel adder uses the principle of *carry lookahead*.

**Algorithm:**

- **Carry Propagate**  \( P_i = A_i \oplus B_i \)
- **Carry Generate**  \( G_i = A_i B_i \)
- \( S_i = P_i \oplus C_i \)
- \( C_{i+1} = G_i + P_i C_i \)
Carry Lookahead Generator

\[ C_0 = \text{input carry} \]
\[ C_1 = G_0 + P_0 C_0 \]
\[ C_2 = G_1 + P_1 C_1 = G_1 + P_1 (G_0 + P_0 C_0) = G_1 + P_1 G_0 + P_1 P_0 C_0 \]
\[ C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0 \]

4-bit Adder with Carry Lookahead

Question:
What is the propagation delay?
Binary Subtractor

The subtraction of unsigned binary numbers can be done by complements.

Review complements:

1’s complement of \( N = (2^n - 1) - N \) (\( N \) is a binary #)
1’s complement can be formed by changing 1’s to 0’s and 0’s to 1’s

2’s complement of a number is obtained by leaving all least significant 0’s and the first 1 unchanged, and replacing 1’s with 0’s and 0’s with 1 in all higher significant digits.

Taking the 1’s complement and adding 1 to the least significant bit in the number.

‘Programmable’ Binary Adder/Subtractor

Line \( M \) controls the the operation (addition or subtraction)
\( M = 0 \rightarrow \text{Adder: } A + B, C_0 = 0 \)
\( B \oplus 0 = (B \cdot 1) + (B’ \cdot 0) = B \)
\( M = 1 \rightarrow \text{Adder: } A + 2’s \text{ complement and a } C_0 = 1 \rightarrow A - B \)
\( M = 1: B \oplus l = (B \cdot 0) + (B’ \cdot 1) = B’ \)
Overflow

Overflow occurs when two numbers of n digits are added and the sum occupies n + 1 digits.

If $V = 0 \rightarrow$ no overflow: n-bit results is correct.
If $V = 1 \rightarrow$ overflow: The result contains n + 1 bits, and the (n+1)th bit is the actual sign.

$$V = C_3 \oplus C_4$$

Overflow (continued)

Fig. 4-13 4-Bit Adder Subtractor
**Binary Multiplier**

Multiplication of binary numbers is done in the same way as decimal numbers. Multiplicand \( B \) is multiplied by the multiplier \( A \) starting from the LSB. Successive partial products are shifted one position from the left and the final product is obtained from the sum of partial products.

HA is used because there are more bits in the partial product. The LSB is formed by the output of the first AND and doesn’t need to go through the HA.

![2-Bit by 2-Bit Binary Multiplier](image)

---

**\( J \times K \) Binary Multiplier**

\( J \times K = (3 \times 4) \)

\( J \times K \) AND gates

\( (J-1) \) \( K \)-bit adders

Result: \( J + K \) bits

![4-Bit by 3-Bit Binary Multiplier](image)