# Carry Look Ahead Adders

- In order to reduce the propagation delay and overflow in ripple adders, the concept of **Carry Look Ahead Adder** (a binary parallel adder) is generated.

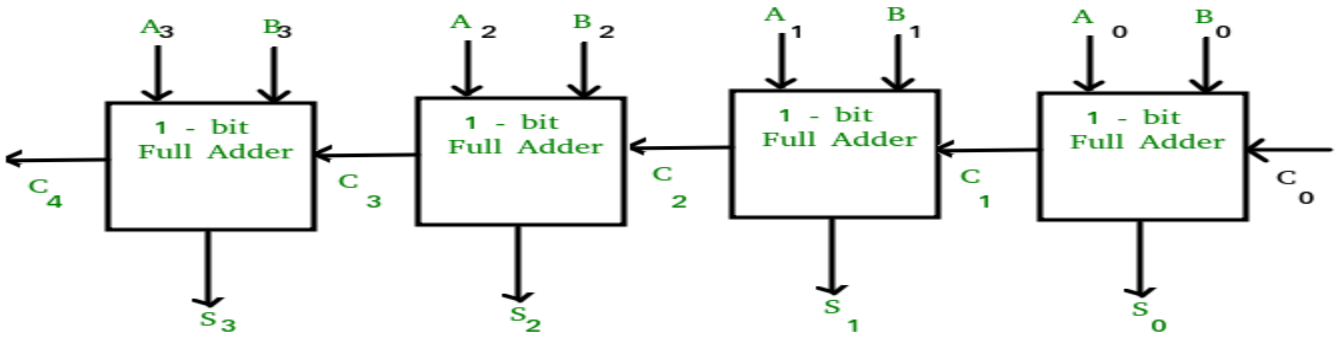


Figure: A 4 bit ripple carry adders

- In ripple carry adders, each adder depends (waits for the carry to arrive in ) on its previous adder to produce output.
- Thus each adder experiences a certain amount of increasing delay. For long numbers, which become unacceptably high.

# Carry Look Ahead Adders(contd.)

- Here, the carry input signals are generated directly from the inputs instead of the ripple arrangement. So, each single bit adder can function independently.
  - To generate a carry input for each adder, the following expressions are carried out:

    $C_0=A_0B_0$

    $C_1= A_1B_1+A_1A_0B_0+B_1A_0B_0$

    $C_2=A_2B_2+A_2A_1B_1+A_2A_1A_0B_0+A_2B_1A_0B_0+B_2A_1B_1+B_2A_1A_0B_0+B_2B_1A_0B_0$
- Thus, each carry can be expressed as a SOP form from the inputs directly with no previous carry.
- Only two levels of delay occur here regardless of the length of the adder.
- The advantage is that the length of time it requires to produce the correct SUM does not depend on the number of data bits used in the operation.
- The speed of this parallel adder is greatly improved with this carry look ahead logic.
- Typically, carry look ahead is done for 4 to 8 bits at a time, because for long numbers this approach becomes excessively complicated.

Stallings (Chapter: Digital Logic; Adders)

# Carry Save Adder

- It is a parallel full adder that adds three (or more) numbers without any horizontal connection.
- If there are three numbers A, B, and C, then it produces two outputs sum S and carry C, and the summation is carried out by the following expression:
  - C+S=A+B+C.
- S and C are calculated as follows:
  - $S_i = A_i \oplus B_i \oplus C_i$ (exclusive OR operation)
  - $C_{i+1} = A_iB_i + A_iC_i + B_iC_i$ (AND and OR operations)
- For example, if A=101 ($A_2$=1, $A_1$=0, $A_0$=1), B=110 ($B_2$=1, $B_1$=1, $B_0$=0) and C=111 ($C_2$=1, $C_1$=1, $C_0$=1), Then the S and C are calculated as follows:

# Carry Save Adder (contd.)

$$A = 1\ 0\ 1$$
$$B = 1\ 1\ 0$$
$$C = 1\ 1\ 1$$
$$S = 1\ 0\ 0$$
$$C = 1\ 1\ 1$$

So the final output is =10 0 1 0

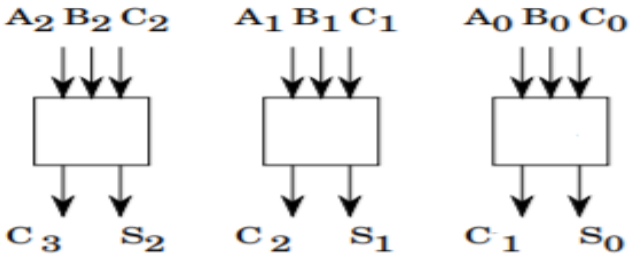


Figure: Carry Save Adder

# Carry Save Adder(contd.)

- The CSA is typically used in a binary multiplier because the addition of more than two numbers are involved after multiplication.
- The advantage is that it reduces the addition of three numbers to the addition of two numbers.
- It reduces propagation delay as the C and S are calculated in parallel.
- The propagation delay does not depend on the number of bits.