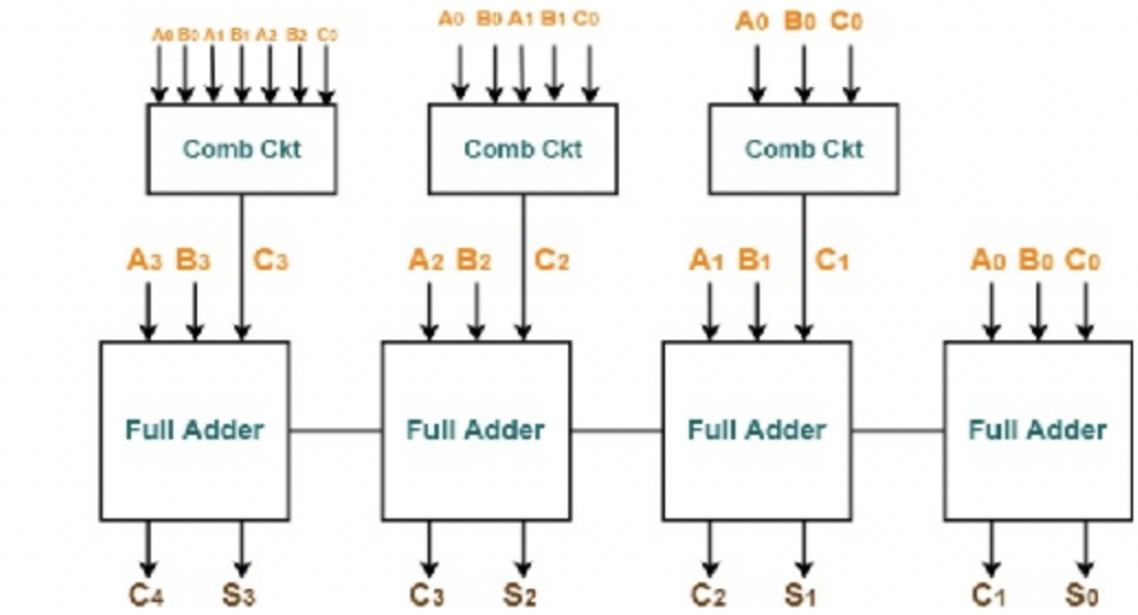
Carry Look-Ahead Adder

Motivation:
We implement Carry Look-ahead adder because it is a faster version of the ripple carry adder.
In ripple carry adders, for each one-bit adder block, the two bits that are to be added are available instantly. However, the ripple carry adder is slow to give the final output because each adder block in it waits for the carry to arrive from its previous block. So the carry-in to the last adder block comes after a long time delay accounting for propagation time through each of the earlier adders.
In essence, it is not possible to generate the sum and carry of a given block until the input carry is known- hence leading to a carry propagation delay.
Concept:
Carry Look-ahead adder reduces the propagation delay which occurs during addition by using more complex hardware circuitry. In this adder, the carry output at any stage of the adder is made dependent only on the bits which are added in the previous stages and the carry input (C_{in}), and hence each one-bit adder can produce output (both sum and carry) irrespective of when the previous carry out is produced.
In other words, the circuit does not have to wait for the generation of carry-bit from the previous adder and each carry bit can be evaluated at any instant of time.`



Truth Table:

A_i	B_i	C_i	C_{i+1}	Condition
0	0	0	0	No carry
0	0	1	0	
0	1	0	0	
0	1	1	1	Carry propagate
1	0	0	0	
1	0	1	1	Carry propagate
1	1	0	1	Carry generate
1	1	1	1	

$C_{in} = C_0$

1)From the truth table, we see that if both A_i and B_i are 1 (i.e $A_i.B_i = 1$), then $C_{i+1} = 1$ irrespective of C_i. This is called “Carry Generate” because the carry out is generated by A_i and B_i, and not dependent on C_i.

2)We also see that if $A_i \wedge B_i = 1$, then $C_{i+1} = 1$ if $C_i = 1$. This is called “Carry Propagate” because the carry out is propagated by $A_i \ XOR \ B_i$, and is dependent on C_i.From the above 2 points, we can write $C_{i+1} = A_i.B_i + (A_i \wedge B_i).C_i$

Let $G_i = A_i.B_i$ (stands for generation), and letl $P_i = A_i \wedge B_i$ (stands for propagation).

The formula becomes $C_{i+1} = G_i + P_i.C_i$

We use this equation to get each consequent carry in terms of C_{in}.

- $C_1 = C_0.P_0 + G_0.$
- $C_2 = C_1.P_1 + G_1 = (C_0.P_0 + G_0).P_1 + G_1.$
- $C_3 = C_2.P_2 + G_2 = (C_1.P_1 + G_1).P_2 + G_2.$
- $C_4 = C_0.P_0.P_1.P_2.P_3 + P_3.P_2.P_1.G_0 + P_3.P_2.G_1 + G_2.P_3 + G_3$
- The above logical values can be implemented as shown in the circuit diagram on the first page.