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 avaliable 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 the previous block.
So the carry-in to the last adder block comes after a long time delay accounting for the propogation 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 propogation delay.

Concept: Carry Look ahead adder reduces the propogation 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 initial carry input (C_{in}). hence each one-bit adder can produce output (both sum and carry) irrespective of when the previous carry out is produced.

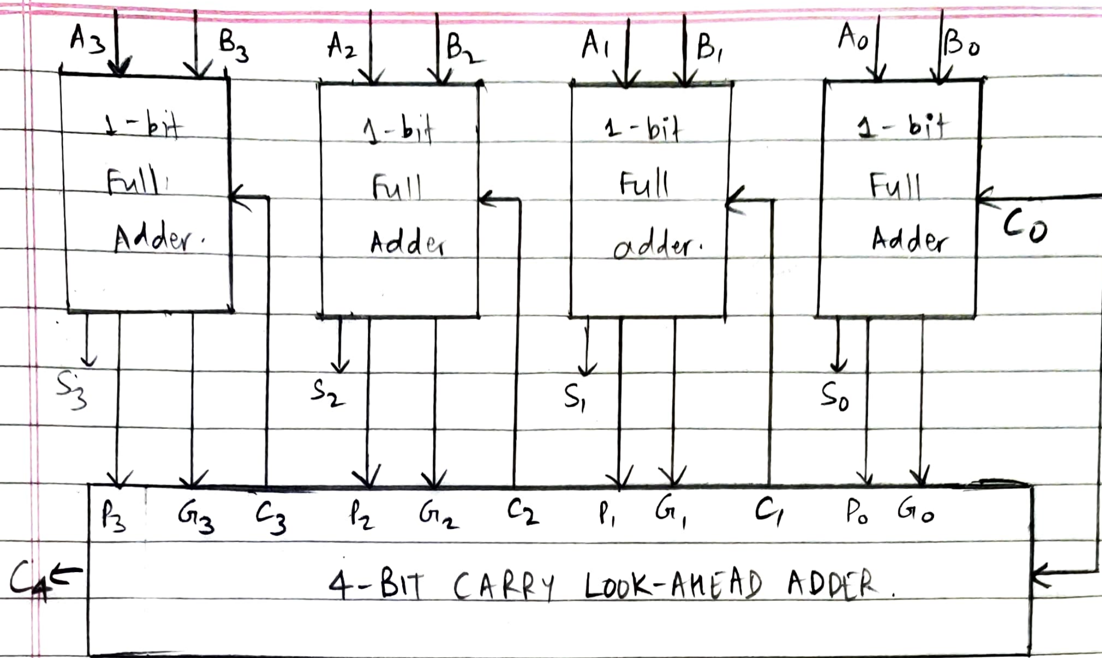

Diagram will be similar for 8 bit.

A_i	B_i	C_i	C_{i+1}	Condition.
0	0	0	0	
0	0	1	0	No Carry.
0	1	0	0	
0	1	1	1	Propagate
1	0	0	0	No Carry.
1	0	1	1	Propagate
1	1	0	1	Generate
1	1	1	1	Generate.

→ TRUTH TABLE

1) From it, we see if $A_i \cdot B_i = 1$, then $C_{i+1} = 1$ irrespective of C_i. This is called "Carry Generate", as it is generated" by A_i & B_i.

2) if $A_i \oplus B_i = 1$, then $C_{i+1} = 1$ if $C_i = 1$.

This is called Carry propogate;

Combining ① and ② $C_{i+1} = A_i \cdot B_i + (A_i \oplus B_i) \cdot C_i$

Let $A_i \cdot B_i = G_i$ and $A_i \oplus B_i = P_i \Rightarrow C_{i+1} = G_i + P_i \cdot C_i$

We know that $C_0 = C_{in}$ = Carry input. Use it to calculate $C_1 \ldots C_n$.

1) $C_1 = G_0 + P_0 C_0$

2) $C_2 = G_1 + P_1 G_0 + P_1 P_0 C_0$

3) $C_3 = G_2 + P_2(G_1 + C_1 P_1) = G_2 + P_2(G_1 + P_1 G_0 + P_1 P_0 C_0)$

$C_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$

4) $C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0$

if 8 bit ⇒ 8) $C_8 = G_7 + P_7 G_6 + P_7 P_6 G_5 + P_7 P_6 P_5 G_4 + \ldots\ldots + P_7 P_6 P_5 P_4 P_3 P_2 P_1 P_0 C_0$