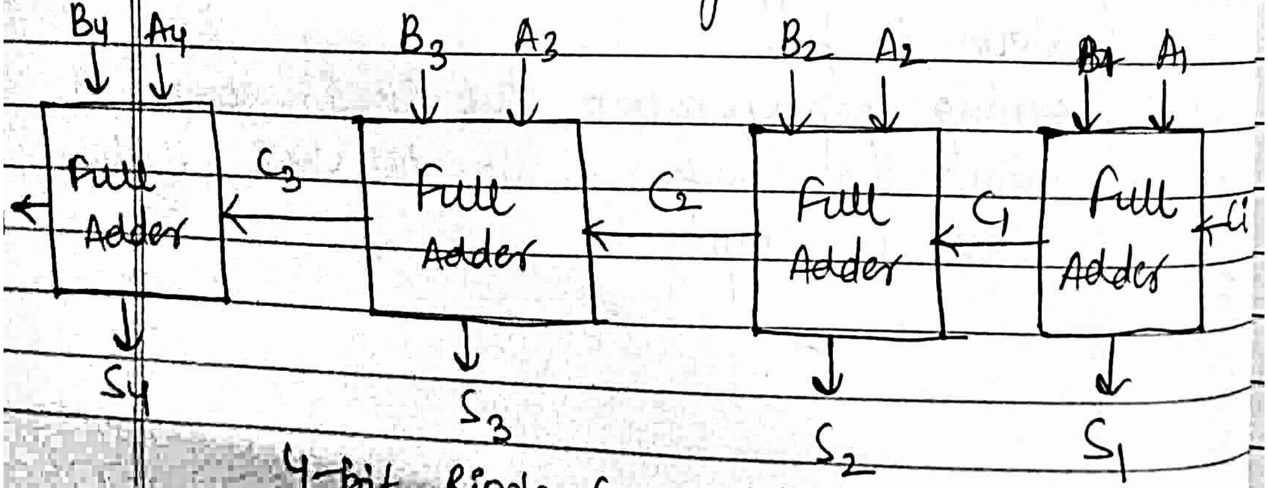
Date: / /

Carry Look Ahead Adder

* A digital computer must contain circuits to perform arithmetic operations like addition, subtraction, multiplication, and division.
* Among these, addition and subtraction are the basic operations whereas multiplication and division are the repeated addition and subtraction respectively.

* To perform these operations "Adder circuits" are implemented using basic logic gates.
* Adder circuits are of various types like Half-adder, Full adder, Ripple carry Adder and Carry Look-Ahead Adder.

* Carry Look-Ahead Adder is the fastest adder circuit.
* It reduces the propagation delay, which occurs during addition.
* It uses more complex hardware circuitry.
* It is designed by transforming the ripple-carry Adder circuit such that the carry logic of the adder is changed into two-level logic.



4-bit Ripple Carry Adder

Page No:
Date: / /

* In this circuit the sum S_3 can be produced as soon as the inputs A_3 and B_3 are given but carry C_3 cannot be computed until the carry bit C_2 is applied. whereas C_2 depends on C_1

* Therefore to produce final steady-state results, carry must propagate through all the states.

* This increases the carry propagation delay of the circuit.

* The propagation delay of the adder is calculated as "the propagation delay of each gate times the no of stages in the circuit".

* For the computation of a large no of bits, more stages have to be added, which makes the delay much worse.

* Therefore, to solve this situation Carry Look Ahead Adder was introduced.

A	B	Ci	Carry
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

<u>Truth Table of Look Ahead Adder</u>

$A \cdot B \rightarrow$ Generation carry

$[A \oplus B] \cdot C \rightarrow$ Carry propagation

$C_o = A \cdot B + [A \oplus B] \cdot C$

$\boxed{C_{i+1} = A_iB_i + C_i[A_i \oplus B_i]}$ —— (1)

From here, we can write :-

$C_1 = C_0(A_0 \oplus B_0) + A_0B_0$
$C_2 = C_1(A_1 \oplus B_1) + A_1B_1$
$C_3 = C_2(A_2 \oplus B_2) + A_2B_2$
$C_4 = C_3(A_3 \oplus B_3) + A_3B_3$

$G_i = A_iB_i$ $\qquad P_i = A_i \oplus B_i$

where,

* <u>G_i</u> :- Carry Generator (which produces the carry when both A_i, B_i ~~and~~ are one regardless of the input carry).

* <u>P_i</u> :- Carry Propagator (it is associated with the propagation of carry from C_i ~~and~~ to C_{i+1})

* The sum output and carry output can be expressed as :-

$$\boxed{\begin{array}{l} S_i = A_i \oplus B_i \oplus C_i = P_i \oplus C_i \\ C_{i+1} = G_i + P_i C_i \end{array}} \text{ — } ②$$

Therefore, the carry bits C_1, C_2, C_3 and C_4 can be calculated as:-

$$C_1 = C_0 \cdot P_0 + G_0$$
$$C_2 = C_1 \cdot P_1 + G_1 = (C_0 \cdot P_0 + G_0) \cdot P_1 + G_1 \text{ — } ③$$
$$C_3 = C_2 \cdot P_2 + G_2 = [(C_0 \cdot P_0 + G_0) \cdot P_1 + G_1) \cdot P_2 + G_2 \text{ — } ④$$
$$C_4 = C_3 \cdot P_3 + G_3 = C_0 \cdot P_0 \cdot P_1 \cdot P_2 \cdot P_3 + P_3 \cdot P_2 \cdot P_1 \cdot G_0 \text{ ⑤}$$
$$+ G_3 \cdot P_3 \cdot P_2 \cdot G_1 + G_2 \cdot P_3 + G_3 \text{ — } ⑥$$

* It can be observed from the equations that carry C_{i+1} only depends on the carry C_0, not on the intermediate carry bit.

Equations show that the carry-in of any stage full adder depends only on:-

* Bits being added in the previous stage.
* Carry bit which was provided in the beginning.

* In carry Look Ahead Adder, the carry input at any stage of the adder is independent of the carry bits generated at the intermediate stages.

* Output of any stage is dependent only on the bits which are added in the previous stages and the carry input provided at the beginning stage.

* Therefore, the circuit at any stage does not have to wait for the generation of carry bit from the previous stage.

* Carry bit can be evaluated at any instant of time.

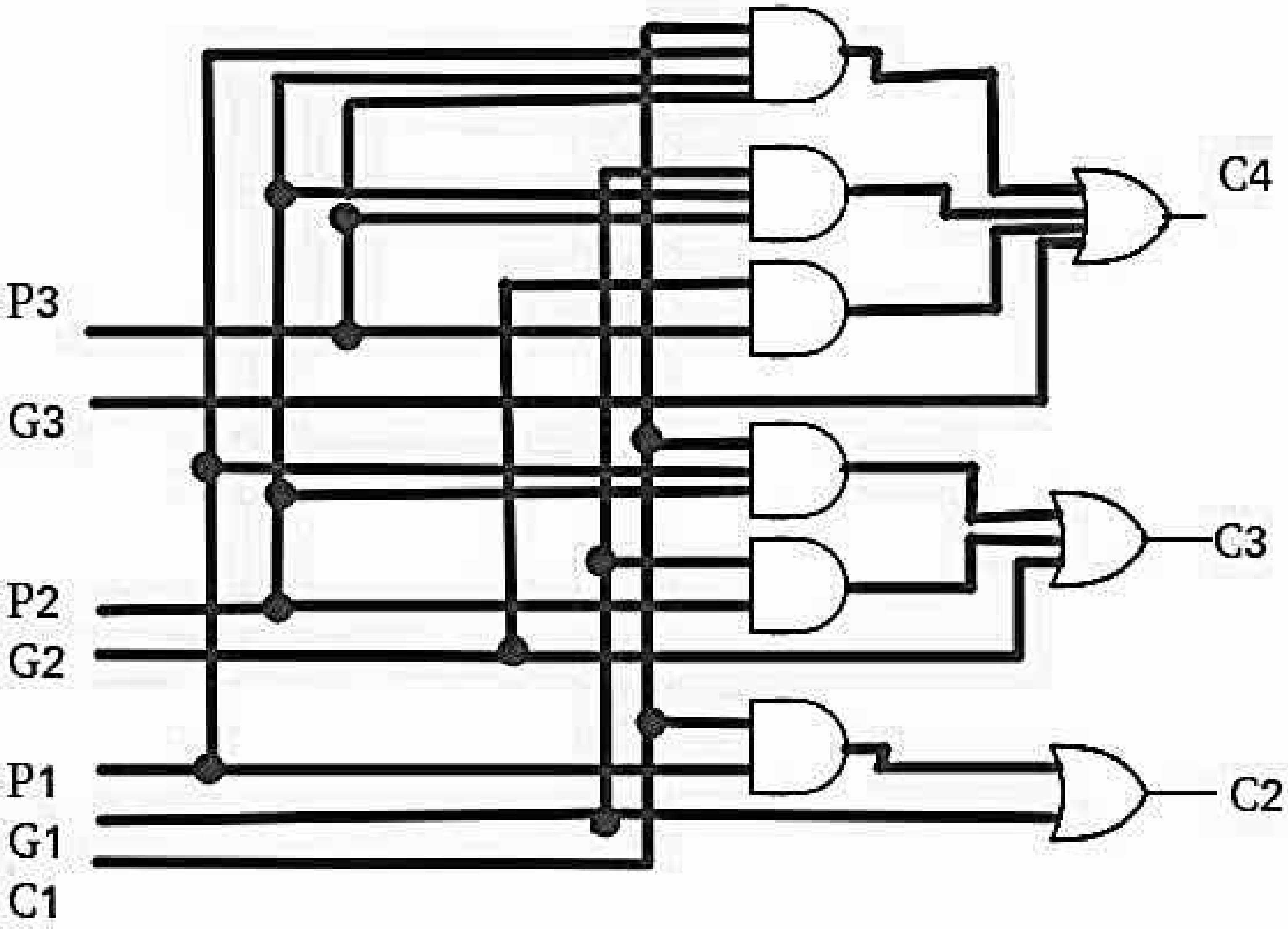
Advantages

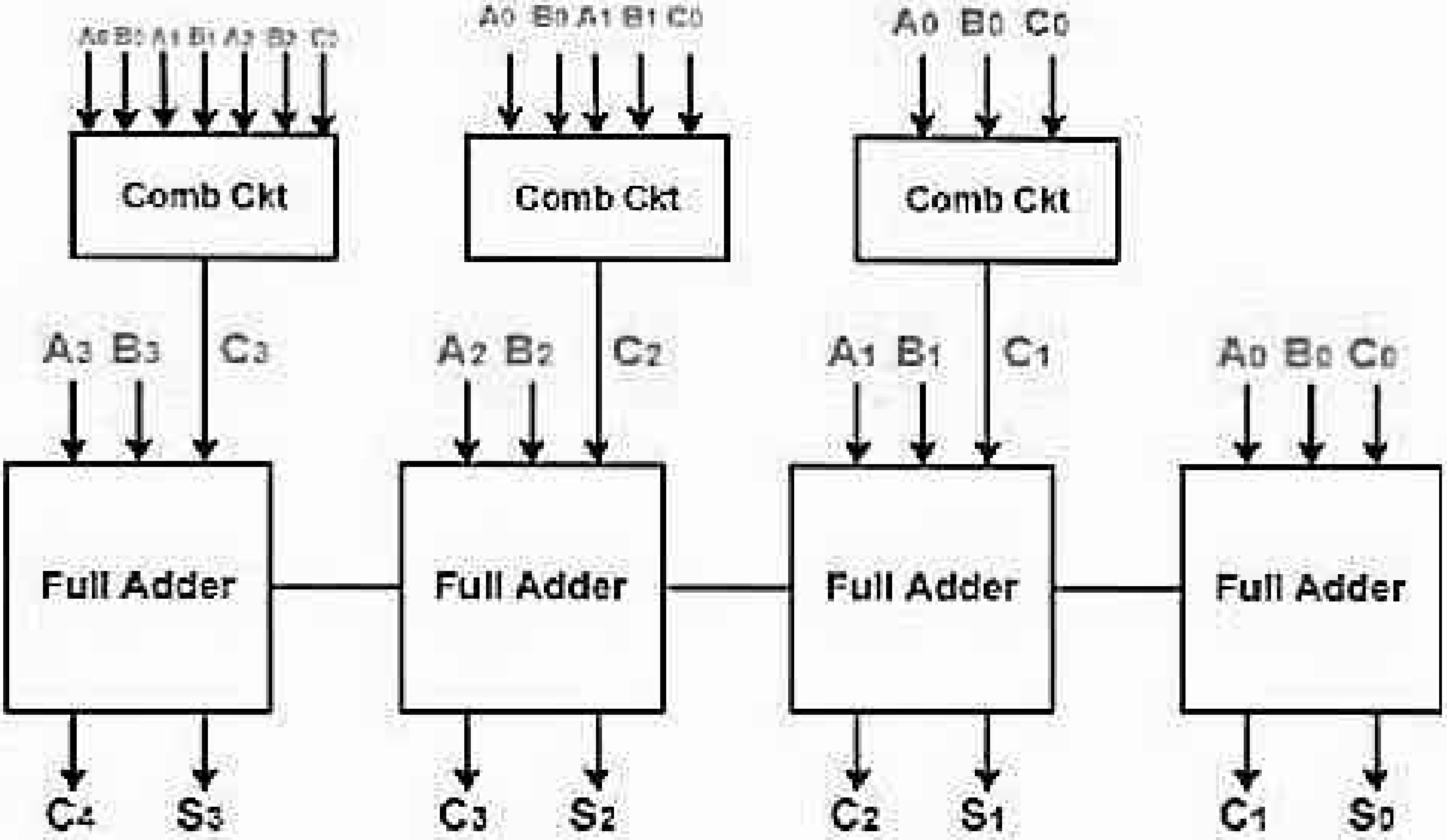
* It generates the carry-in for each full adder simultaneously.
* Reduces the propagation delay.
* Fastest adder.

Page No:
Date: / /

Disadvantages

* Complex hardware.
* It is costlier since it involves complex hardware.
* It gets more complicated as the no of bits increases.
* With increase in number of variables, circuit implements more hardware and hence implemented as an IC.


C4
P3
G3
C3
P2
G2
P1
C2
G1
C1



A0 B0 A1 B1 A2 B2 C0
Comb Ckt
A0 B0 A1 B1 C0
Comb Ckt
A0 B0 C0
Comb Ckt
A3 B3 C3
Full Adder
C4 S3
A2 B2 C2
Full Adder
C3 S2
A1 B1 C1
Full Adder
C2 S1
A0 B0 C0
Full Adder
C1 S0