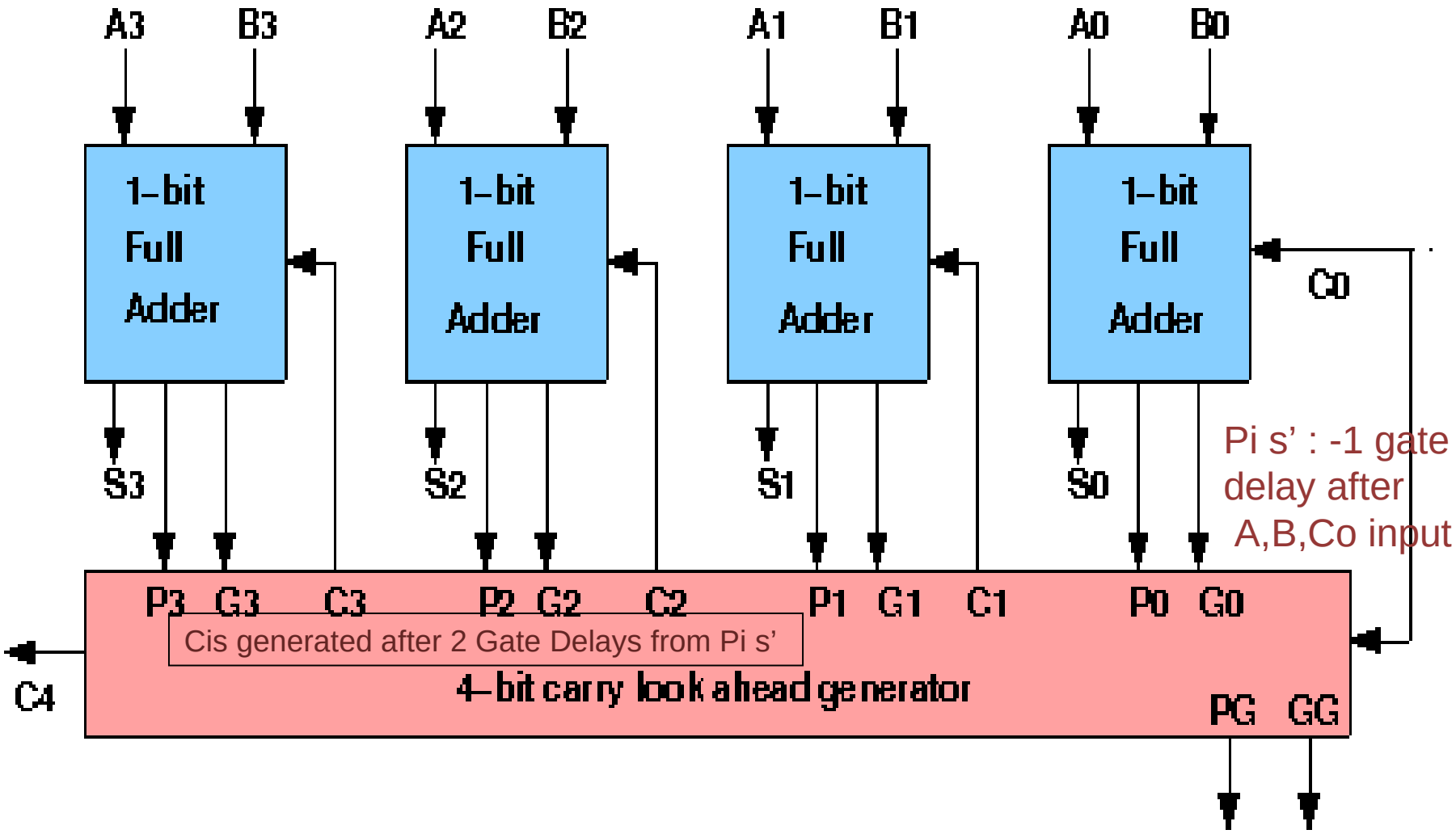
Design of Carry Lookahead Adders

- To reduce the computation time
  - faster ways to add two binary numbers by using carry lookahead adders.
  - creating two signals P (**Carry Propagator)** and G (**Carry Generator)**. The block diagram of a 4-bit Carry Lookahead Adder is shown in next slide


A3
B3
A2
B2
A1
B1
A0
B0
1-bit Full Adder
1-bit Full Adder
1-bit Full Adder
1-bit Full Adder
C0
S3
S2
S1
S0
Pi s' : -1 gate delay after A,B,Co input
P3
G3
C3
P2
G2
C2
P1
G1
C1
P0
G0
Cis generated after 2 Gate Delays from Pi s'
C4
4-bit carry look ahead generator
PG
GG

- $\mathbf{Sum_i = A_i \oplus B_i \oplus C_i}$
- $\mathbf{C_{i+1} = A_i \cdot B_i + (A_i \oplus B_i)\, C_i}$

Replace by

- $\mathbf{P_i = A_i \oplus B_i}$
- $\mathbf{G_i = A_i \cdot B_i}$
- The output sum and carry can be expressed as
- $\mathbf{Sum_i = P_i \oplus C_i}$
- $\mathbf{C_{i+1} = G_i + (P_i \cdot C_i)}$

Boolean function for the carry output of each stage and substitute for each Ci its value from the previous equations:

- $\mathbf{C_1 = G_0 + P_0 \cdot C_0}$
- $\mathbf{C_2 = G_1 + P_1 \cdot C_1 = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0}$
- $\mathbf{C_3 = G_2 + P_2 \cdot C_2 = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_0}$
- $\mathbf{C_4 = G_3 + P_3 \cdot C_3 = G_3 + P_3 \cdot G_2 + P_3 \cdot P_2 \cdot G_1 + P_3 \cdot P_2 \cdot P_1 \cdot G_0 + P_3 \cdot P_2 \cdot P_1 \cdot P_0 \cdot C_0}$

(Note:Five fan in gate is practical limit)

carry look-ahead generator

- C_3 is propagated at the same time as C_2 and C_1.

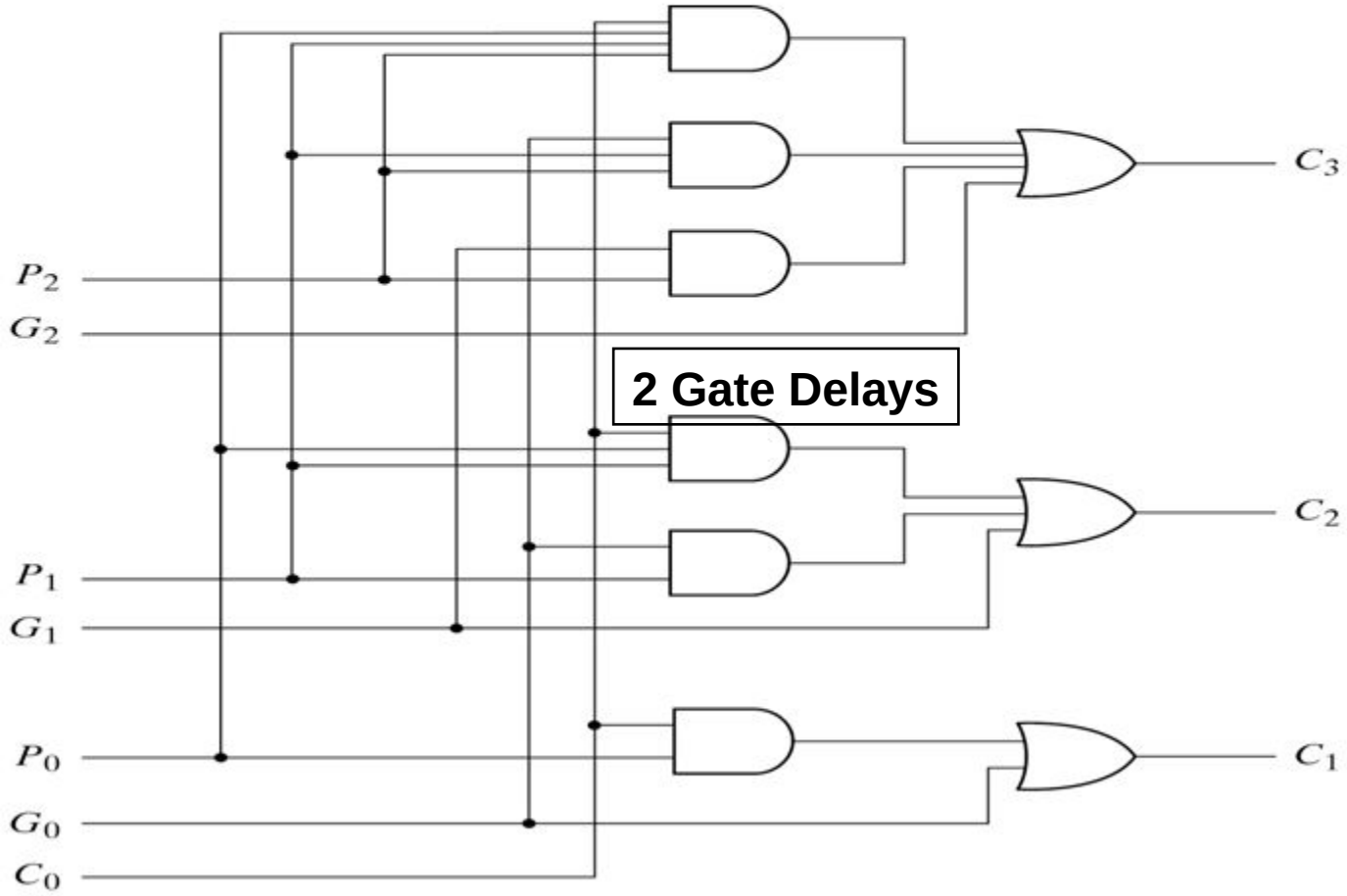
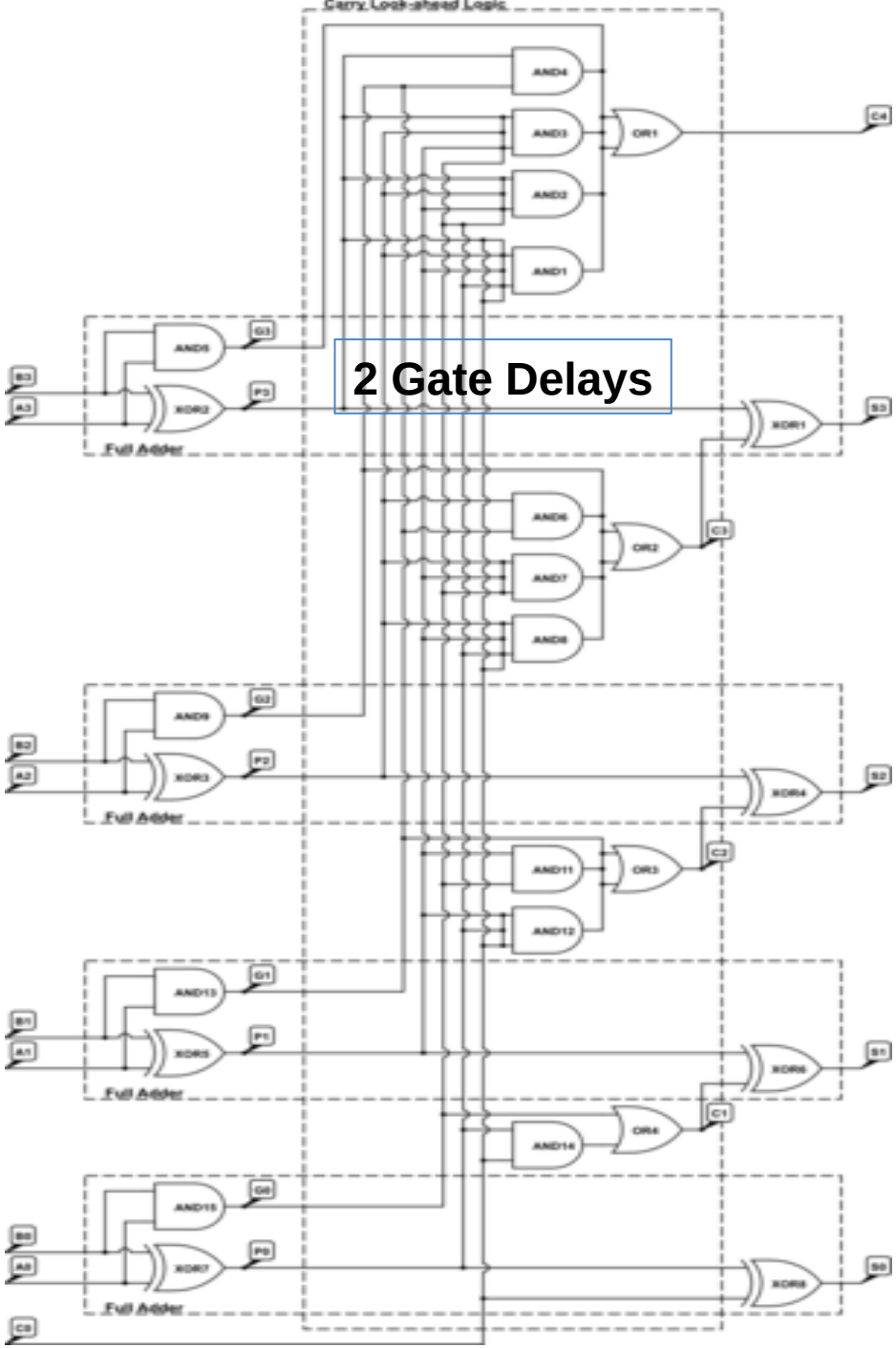


Fig. 4-11 Logic Diagram of Carry Lookahead Generator


Carry Look-ahead Logic
AND4
AND3
AND2
AND1
OR1
C4
G3
P3
B3
A3
XOR2
AND5
Full Adder
XOR1
S3
2 Gate Delays
AND6
AND7
AND8
OR2
C3
G2
P2
B2
A2
XOR3
AND9
Full Adder
XOR4
S2
AND11
AND12
OR3
C2
G1
P1
B1
A1
XOR5
AND13
Full Adder
XOR6
S1
AND14
OR4
C1
G0
P0
B0
A0
XOR7
AND15
Full Adder
XOR8
S0
C0

Group Propagate and Group Generate Delay

- The Carry Look Ahead 4-bit adder can also be used in a higher-level circuit by having each CLA Logic circuit produce a propagate and generate signal to a higher-level CLA Logic circuit. The **group propagate (**PI_0**)** and **group generate (**GI_0**)** for a 4-bit CLA are:

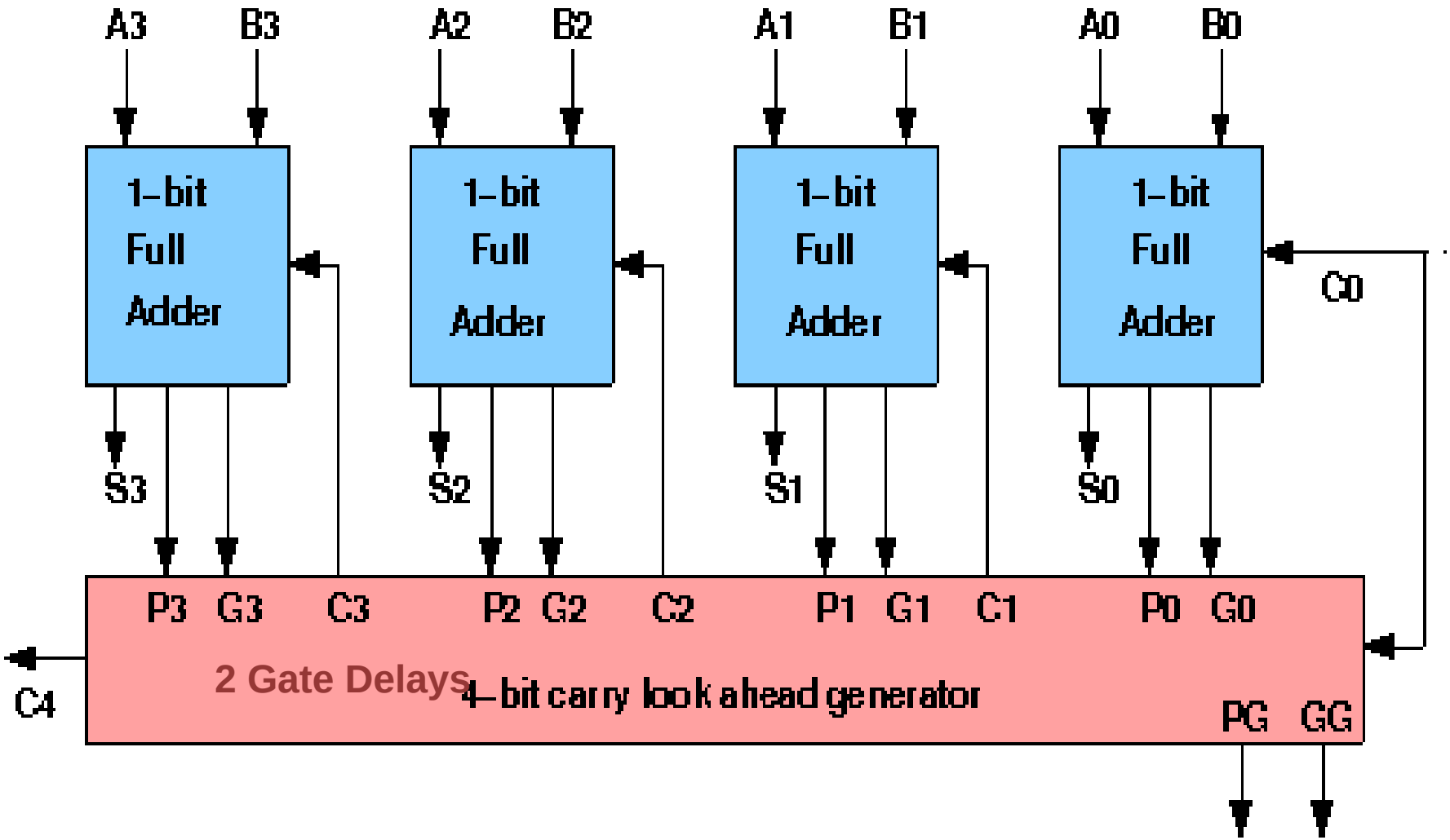
  $PI_0 = P_3 \cdot P_2 \cdot P_1 \cdot P_0$ **1 gate level (and) after Pi**

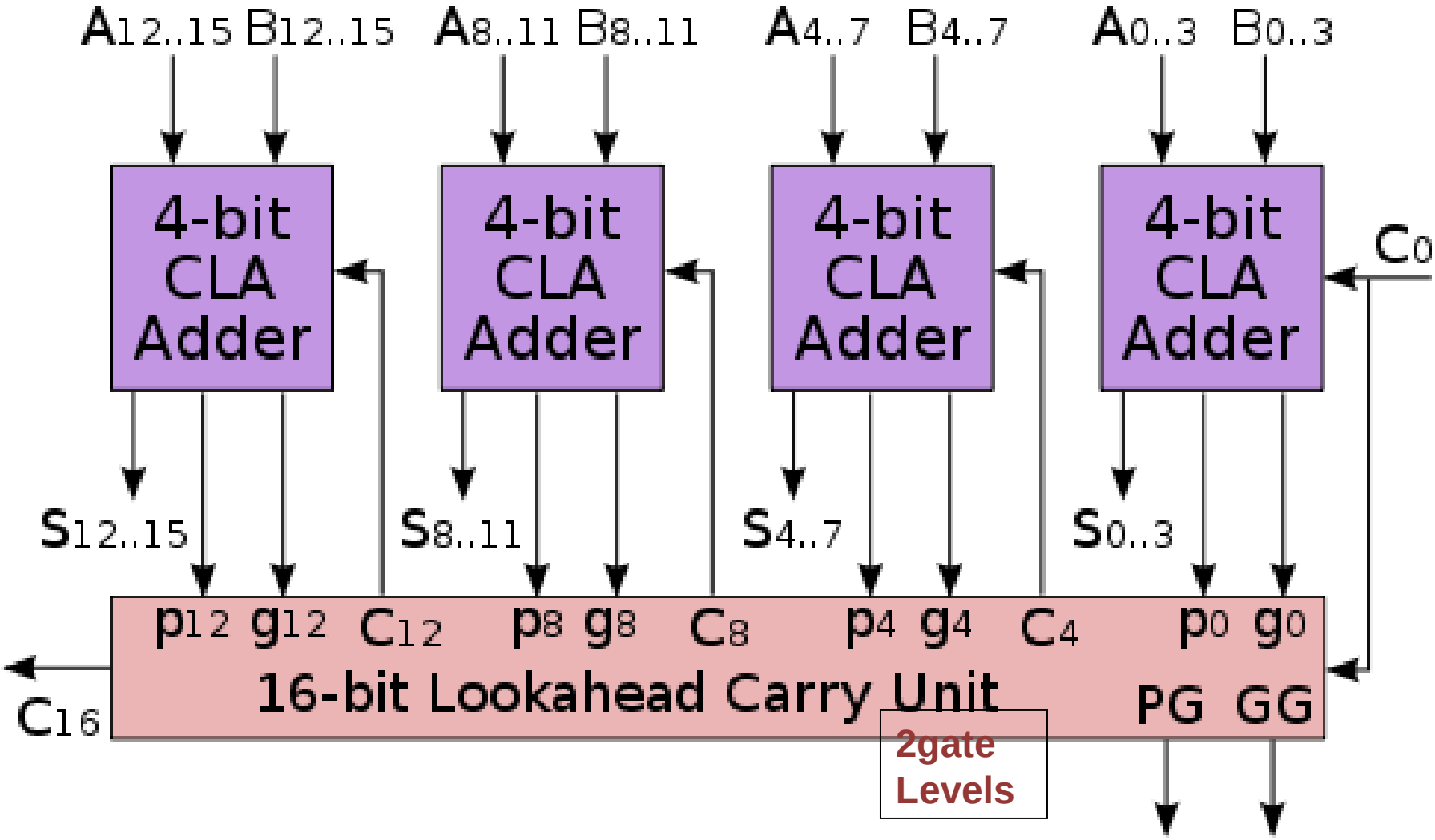
  $GI_0 = G_3 + P_3 \cdot G_2 + P_3 \cdot P_2 \cdot G_1 + P_3 \cdot P_2 \cdot P_1 \cdot G_0$

  (2gate levels (and,or) after Pi and Gi)

All 4bit adder Si - 4 gate delays, $PG=PI_0$

All 4 bit adder Ci – 3 gate delays $GG=GI_0$





C4 to C12 : 5 gate delays, *C15 : 2 more gate delays in 4bit CLA(5+2=7)*

S15 .. S 4 : 1 more gate delays= 1+7=8

Compare Carry Look Ahead to Ripple Carry

- The signal from input carry C_{in} to output carry C_{out} requires an AND gate and an OR gate
  - two gate levels.
- four full adders in the parallel adder, the output carry C_5 would have 2 X 4 = 8 gate levels from C_1 to C_5. For an n-bit parallel adder, there are 2n gate levels to propagate through for **ripple carry adder**
- 16 bit ripple carry 2 x 16 = 32gate delay