CSIE 2344: Homework 3

Due April 29 (Monday) at Noon

There are 90 points in total. Points will be deducted if no appropriate intermediate step is provided.

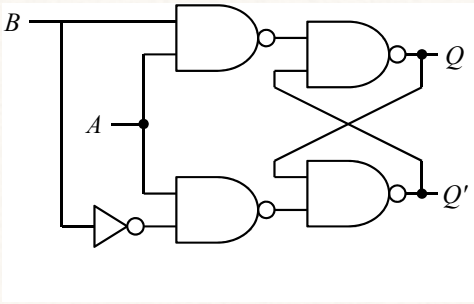
When you submit your homework on Gradescope, please select the corresponding page(s) of each question.

1 MUXes and Three-State Buffers (8pts)

Show how to make an 8-to-1 MUX using two 4-to-1 MUXes, two three-state buffers, and one inverter.

2 Latch I (8pts)

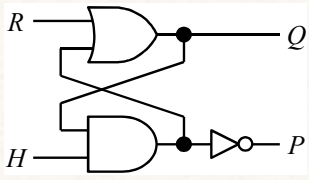
Given the latch as below, complete the following truth table.



A	B	Q	Q^+
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

3 Latch II (16pts)

A latch can be constructed from an OR gate, an AND gate, and an inverter connected as below.



1. What restriction must be placed on R and H so that P will always equal Q' (under steady-state conditions)?
2. Construct a next-state table and derive the minimum characteristic (next-state) equation for the latch.

4 Lab 1: Source Code (28pts)

1. (16pts) Print out your `cla_gl` source code and attach it with your Homework 3.
2. (12pts) Print out your `rca_gl` source code and attach it with your Homework 3.

We may ask you to demo in the future. Points will be deducted if your demo fails. **You can only demo with the source code same as that you submit**.

5 Lab 1: Waveform (12pts)

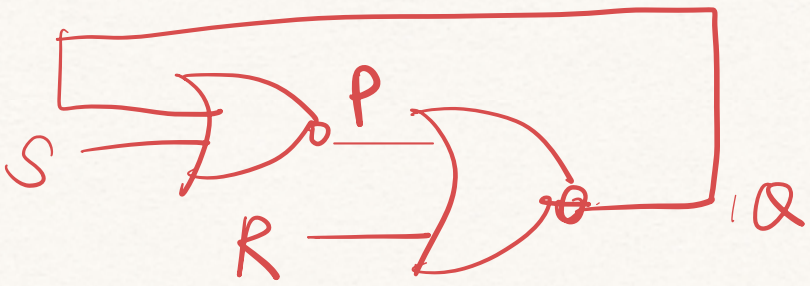
Show the waveform of `cla_gl` on input transition from $000 + 000 + 0$ to $000 + 111 + 1$. You should select all the input and output signals of `cla_gl` module.

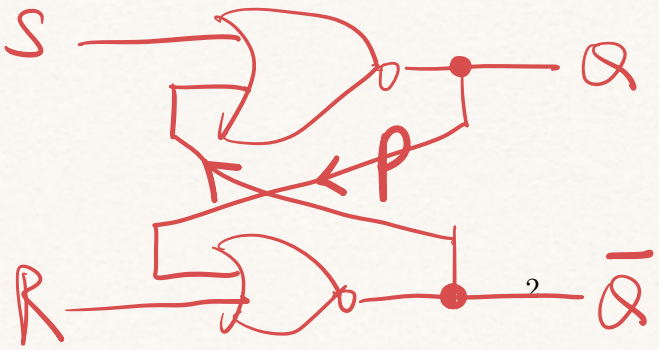
6 Lab 1: Propagation Delays (8pts)

1. (4pts) Find the maximum propagation delay of `rca_gl` and one of the corresponding input transitions.
2. (4pts) Find the maximum propagation delay of `cla_gl` and one of the corresponding input transitions.

7 Lab 1: Some Derivation (10pts)

Assume that only 2-input gates are used. Derive the number of levels needed in an n-bit carry-lookahead adder as a function of n.

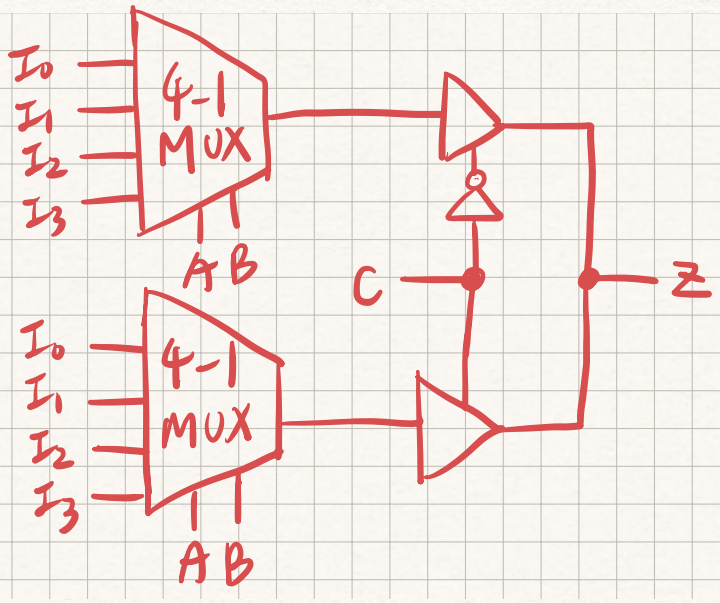




1 MUXes and Three-State Buffers (8pts)

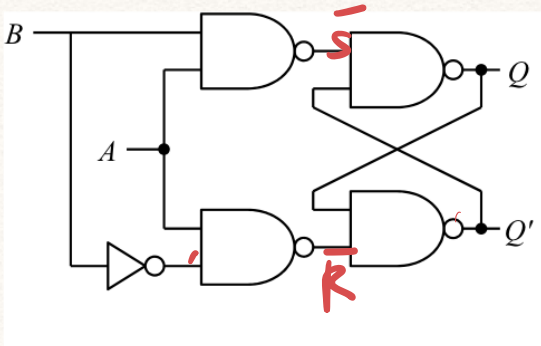
Show how to make an 8-to-1 MUX using two 4-to-1 MUXes, two three-state buffers, and one inverter.

Ans:



2 Latch I (8pts)

Given the latch as below, complete the following truth table.



A	B	Q	Q^+
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Ans:

A	B	$\bar{S}$	$\bar{R}$	Q	Q^+
0	0	1	1	0	0
0	0	1	1	1	1
0	1	1	1	0	0
0	1	1	1	1	1
1	0	1	0	0	0
1	0	1	0	1	0
1	1	0	0	0 1	X 1
1	1	0	0	1 1	X 1

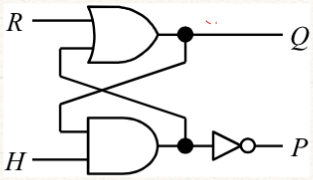
$\bar{S} = (AB)' = A' + B'$

$\bar{R} = (AB')' = A' + B$

$\bar{S}$	$\bar{R}$	Q	Q^+	
1	1	0	0	Unchanged
1	1	1	1	
1	0	0	0	Reset to 0
1	0	1	0	
0	1	0	1	Set to 1
0	1	1	1	
0	0	0	X	Inputs Not Allowed
0	0	1	X	

3 Latch II (16pts)

A latch can be constructed from an OR gate, an AND gate, and an inverter connected as below.



1. What restriction must be placed on R and H so that P will always equal Q' (under steady-state conditions)?
2. Construct a next-state table and derive the minimum characteristic (next-state) equation for the latch.

Ans:

1. According all the cases listed in the table, only when R=1 and H=0, P and Q would be different. Therefore, the restriction is that R=1 and H=0.

Q	R	H	Q^t (P)	
0	0	0	0	1
0	0	1	0	1
0	1	0	1	1
0	1	1	X 1	1
1	0	0	1	1
1	0	1	1	0
1	1	0	0 1	1
1	1	1	X 1	0

2. By the K-map,

Q \ RH	00	01	11	10
0	0	0	1	X
1	0	1	1	X

$Q^t = R + HQ$

4 Lab 1: Source Code (28pts)

1. (16pts) Print out your `cla_gl` source code and attach it with your Homework 3.
2. (12pts) Print out your `rca_gl` source code and attach it with your Homework 3.

We may ask you to demo in the future. Points will be deducted if your demo fails. **You can only demo with the source code same as that you submit**.

1.

```
// carry-lookahead adder, gate level modeling
module cla_gl(
    output C3,       // carry output
    output[2:0] S,   // sum
    input[2:0] A, B, // operands
    input C0         // carry input
    );

    // Implement your code here.
    wire [3:0] C;
    wire [2:0] G, P, M;

    AND g0(G[0],  A[0],  B[0]);
    AND g1(G[1],  A[1],  B[1]);
    AND g2(G[2],  A[2],  B[2]);
    OR p0(P[0], A[0], B[0]);
    OR p1(P[1], A[1], B[1]);
    OR p2(P[2], A[2], B[2]);

    assign C[0]=C0;
    assign C3=C[3];

    //assign C[1] = G[0] | (P[0] & C[0]);
    AND p0c0(M[0], P[0], C[0]);
    OR c1(C[1], G[0], M[0]);

    //assign C[2] = G[1] | (P[1] & C[1]);
    AND p1c1(M[1], P[1], C[1]);
    OR c2(C[2], G[1], M[1]);

    //assign C[3] = G[2] | (P[2] & C[2]);
    AND p2c2(M[2], P[2], C[2]);
    OR c3(C[3], G[2], M[2]);

    FA fa0(, S[0], A[0], B[0], C[0]);
    FA fa1(, S[1], A[1], B[1], C[1]);
    FA fa2(, S[2], A[2], B[2], C[2]);

endmodule
```

2.

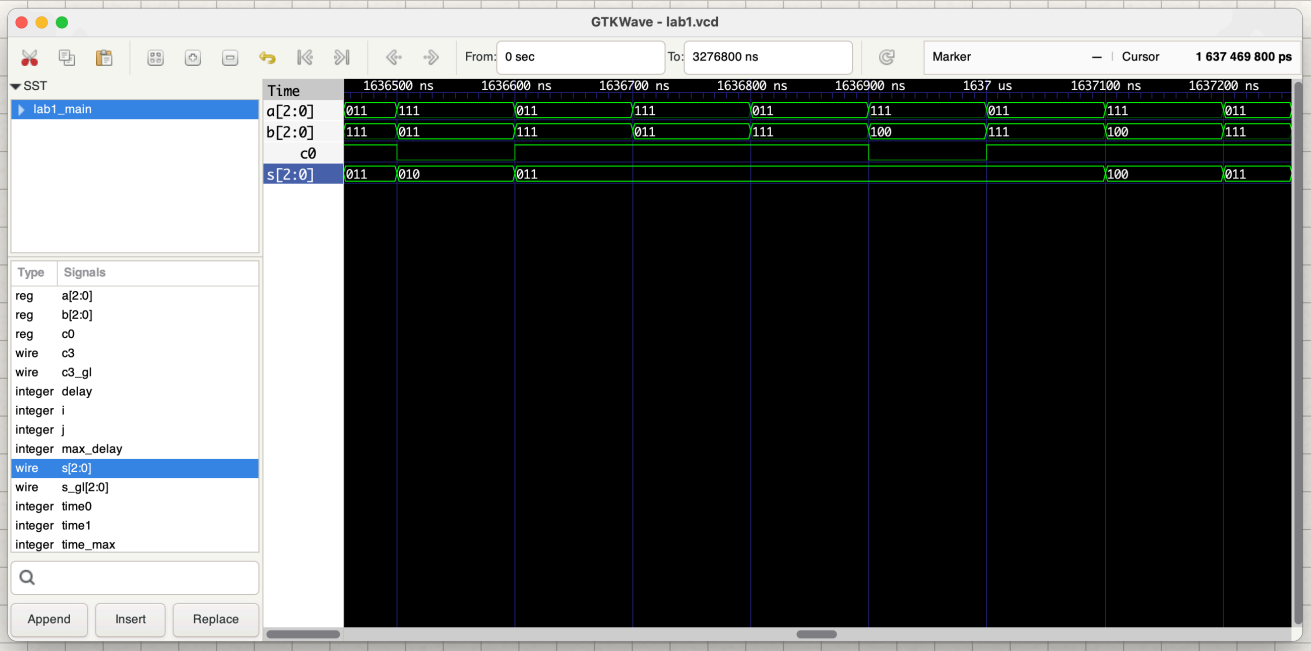
```
//  ripple-carry adder, gate level modeling
module rca_gl(
    output C3,       // carry output
    output[2:0] S,   // sum
    input[2:0] A, B, // operands
    input C0         // carry input
    );
    wire[3:0] c;

    assign c[0] = C0;
    assign C3 = c[3];

    FA fa0(c[1], S[0], A[0], B[0], c[0]);
    FA fa1(c[2], S[1], A[1], B[1], c[1]);
    FA fa2(c[3], S[2], A[2], B[2], c[2]);
endmodule
```

Lab 1: Waveform (12pts)

v the waveform of `cla_gl` on input transition from $000 + 000 + 0$ to $000 + 111 + 1$. You should select ne input and output signals of `cla_gl` module.



6 Lab 1: Propagation Delays (8pts)

1. (4pts) Find the maximum propagation delay of `rca_gl` and one of the corresponding input transitions.

2. (4pts) Find the maximum propagation delay of `cla_gl` and one of the corresponding input transitions.

1. 23 ticks , 000+000+0 → 000+111+1

```
The maximum delay is 23 ticks on transition 000+000+0 --> 000+111+1
lab1.v:70: $finish called at 3276800000 (1ps)
3276800 / 000 / 111 / 1 / 1000 / 1111
```

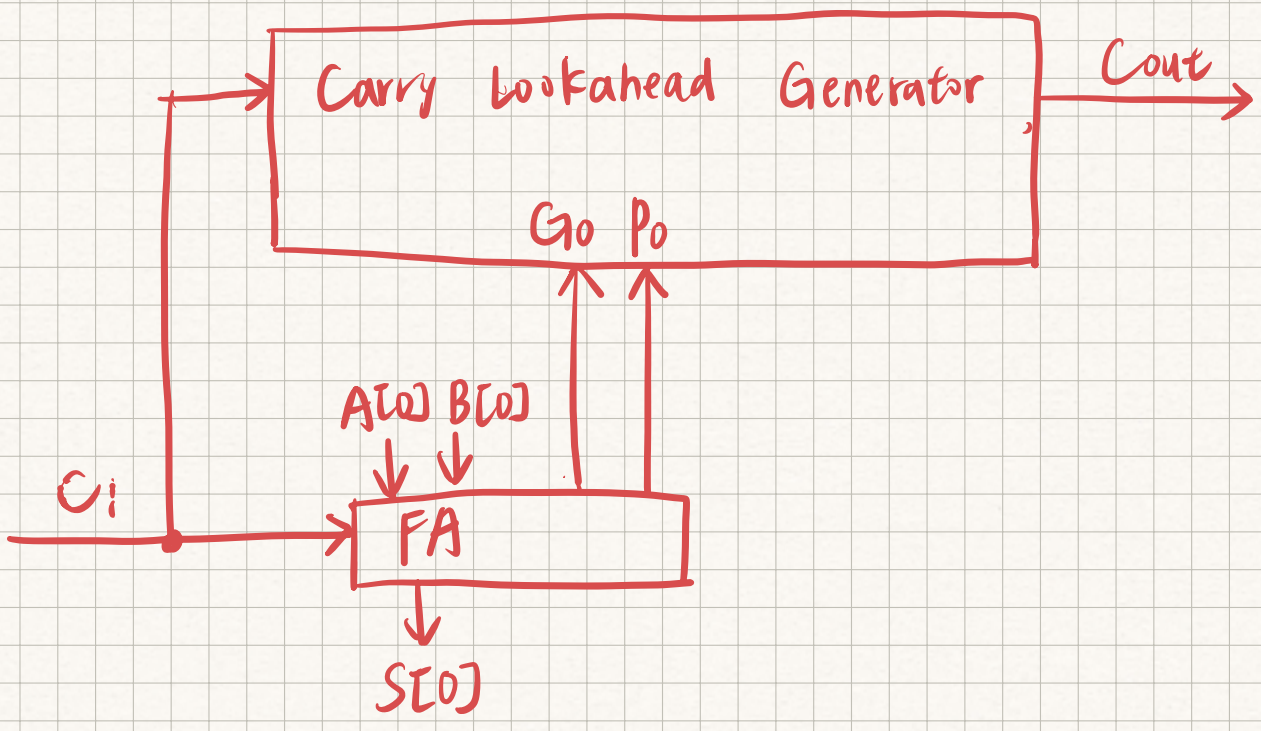
2. 21 ticks , 000+000+0 → 000+111+1

```
The maximum delay is 21 ticks on transition 000+000+0 --> 000+111+1
lab1.v:70: $finish called at 3276800000 (1ps)
3276800 / 000 / 111 / 1 / 1000 / 1111
```

7 Lab 1: Some Derivation (10pts)

Assume that only 2-input gates are used. Derive the number of levels needed in an n-bit carry-lookahead adder as a function of n.

if n=1

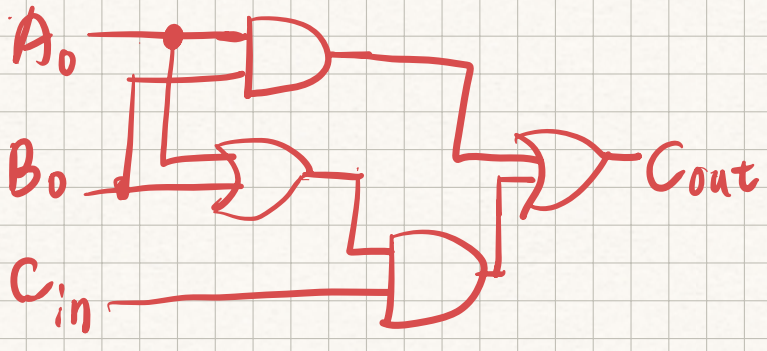


$C_0 = A_0B_0 + B_0C_{in} + A_0C_{in} = A_0B_0 + (A_0 + B_0)C_{in}$

$G_i = A_iB_i$, $P = A_i + B_i$

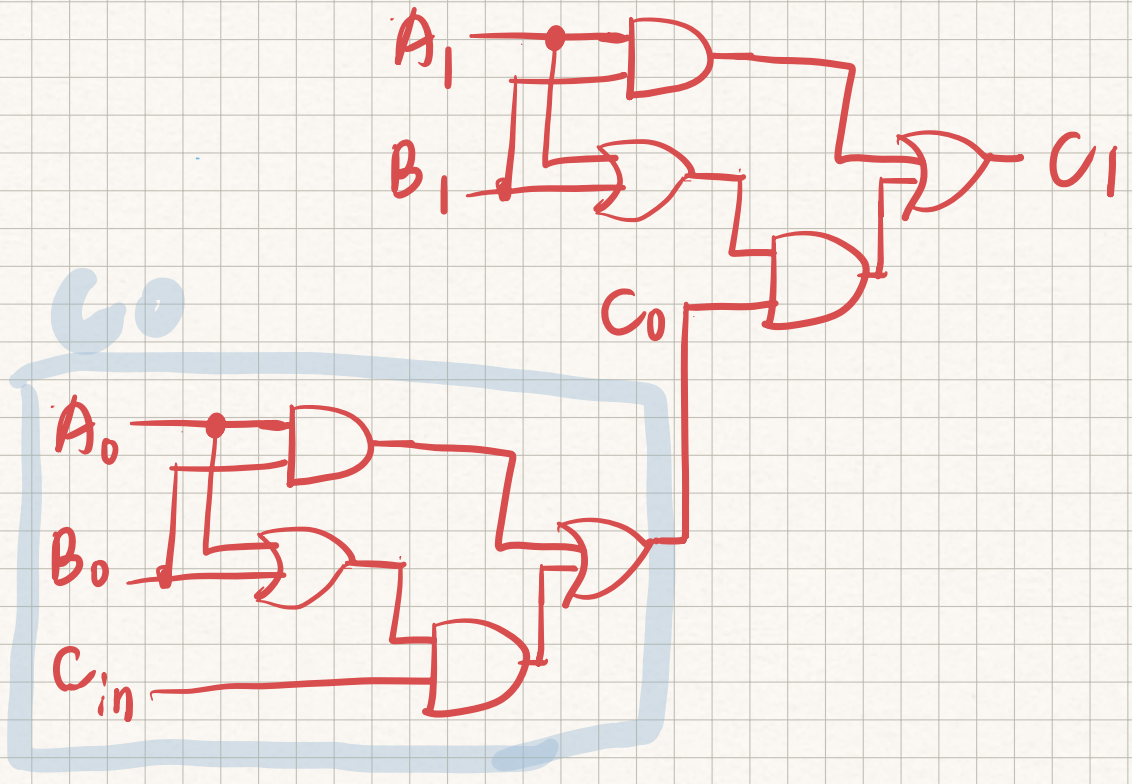
n=1, level=3

$C_0 = G_0 + P_0C_{in}$



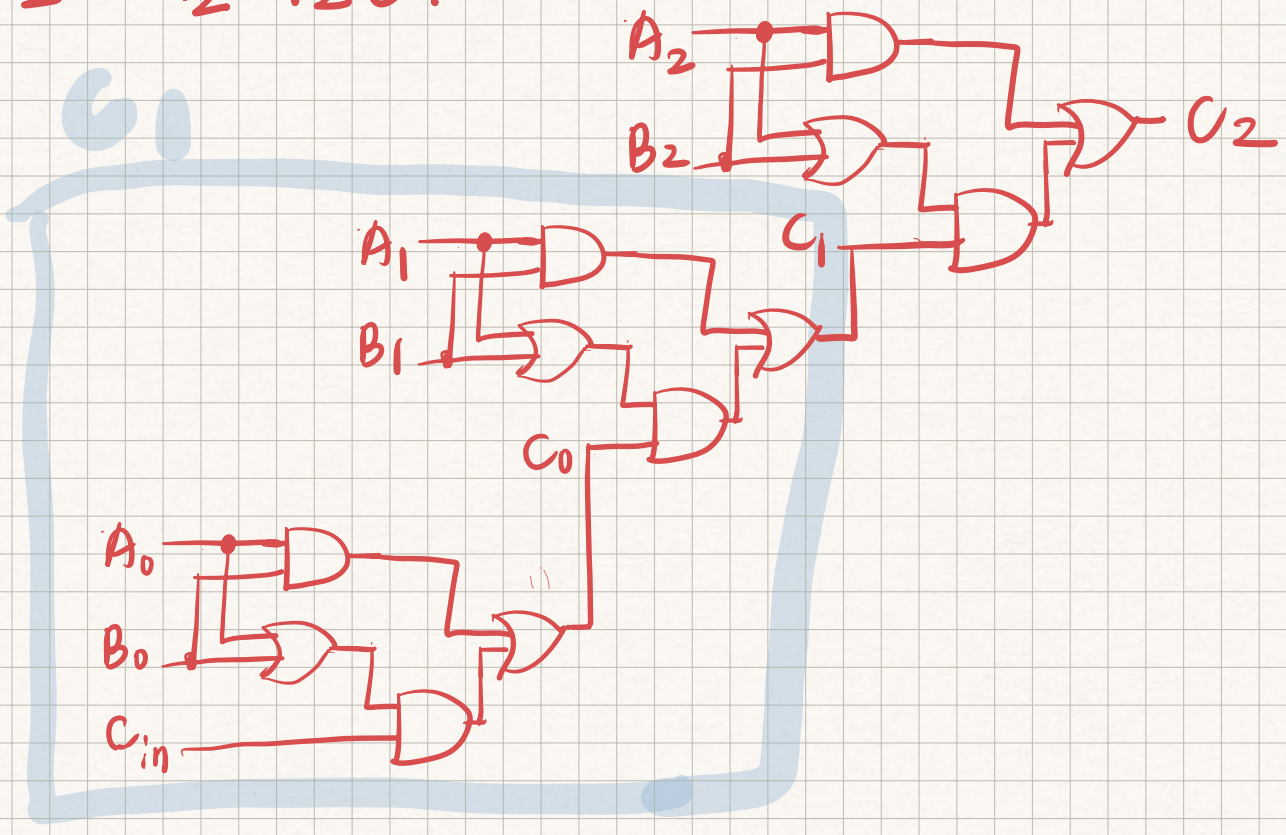
n=2, level=5

$C_1 = G_1 + P_1 C_0$



n=3, level=2

$C_2 = G_2 + P_2 C_1$



Assume that only 2-input gates are used. It needs $2n+1$ levels of gates in an n-bit carry lookahead adder.

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 C_1 = G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

...

$$C_n = \sum_{i=0}^{n-1} \left(G_i \prod_{j=i+1}^{n-1} P_j \right) + C_0 \prod_{j=0}^{n-1} P_j$$

The calculation involves the sum of $n+1$ terms and the product of at most $n+1$ terms, leading to $2\lceil \log_2(n+1) \rceil$ levels of logic. Adding an additional level for either the propagate (P_i) or generate (G_i) terms, the total number of levels becomes

$2\lceil \log_2(n+1) \rceil + 1$.

In another consideration, the calculation for S_{n-1} involves an exclusive OR (XOR) operation $C_{n-1} \oplus (A_{n-1} \oplus B_{n-1})$ which introduces an additional $2\lceil \log_2 n \rceil + 2$ levels. To sum up, it needs

$\max \{ 2\lceil \log_2(n+1) \rceil + 1, 2\lceil \log_2 n \rceil + 2 \}$

levels.