Create a combinational RTL model for a 4-bit expandable carry lookahead adder (CLA).
You do not need to use modules for full or half adders.
Recall that a behavioral dataflow model uses only continuous assignment (assign) statements with/without gate delays.

Use below Gate delays:

- AND gate 2ns
- OR gate 2ns
- XOR gate 3ns

Using the 4-bit CLA module you created and verified using the test bench, create an expandable 8-bit adder and simulate it with the testbench provided.

Your design modules MUST be declared as follows:

```
module CLA4Bit(ain, bin, cin, sum, cout);
     input [3:0]    ain, bin;
     input          cin;
     output logic [3:0] sum;
     output logic cout;

module CLA8Top(ain, bin, cin, sum, cout);
     input [7:0]    ain, bin;
     input          cin;
     output logic [7:0] sum;
     output logic cout;
```

Refer the schematic below for a 4-bit CLA carry out logic:

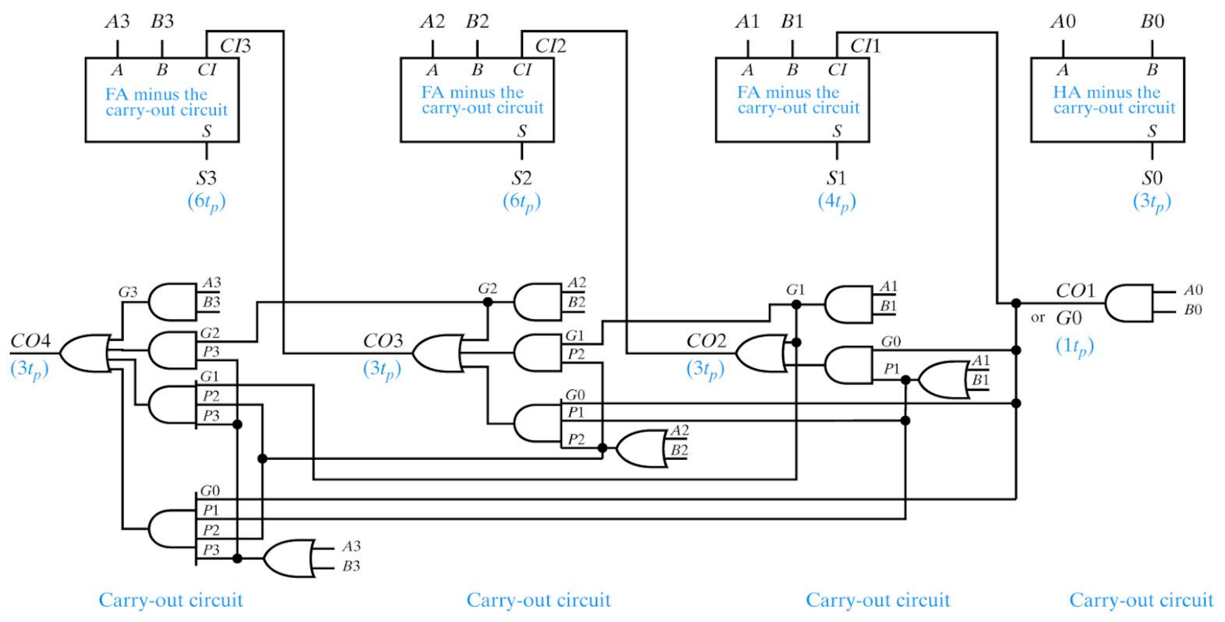

Figure 4.5.9 A 4-bit Carry Look-Ahead Adder showing the carry-out circuitry.

Refer below equations for carry out logic:

Cout = A · B + A · Cin + B · Cin
Cout = A · B + Cin · (A+ B)
Cout = G + Cin · P

where,
Gn = An.Bn and Pn = An + Bn

G (carry generate) is dependent upon current stage's ability to generate a carry.

P (carry propagate) is dependent upon current (and prior) stages' ability to propagate a carry.

A carry is generated at a stage i if it is generated by stage i's inputs (Ai,Bi) or by any prior stage and propagated by every succeeding stage.

Hence,

Cout n+1 = Gn + Cin · Pn

where,
Gn = An.Bn and Pn = An + Bn

And the equation for Sum output will be as follows:

S = A ^ B ^ CIN ;

(Hint: For the sum output, you need to figure out the logic for CIN using the schematic, it is not straight forward, unlike inputs A and B which can be used as it is!)

You are encouraged to do further readings on the working of a CLA and ask questions if things are unclear.

Submit your System Verilog code for both design modules with the transcript via D2L in a zipped file named as **HW1.zip** with names as follows:
CLA4Top.sv your 4-bit CLA design module
CLA8Top.sv your 8-bit CLA design module
transcripts for both simulations

NOTE: If you do not follow these instructions for declaring, naming and submitting your modules to D2L, you will receive partial credit even if your answers are correct.