EE466

VLSI Design

Final Exam

Dec. 12, 2018. (3:10pm – 5:10pm)

Instructor: Dae Hyun Kim (daehyun@eecs.wsu.edu)

Name:

WSU ID:

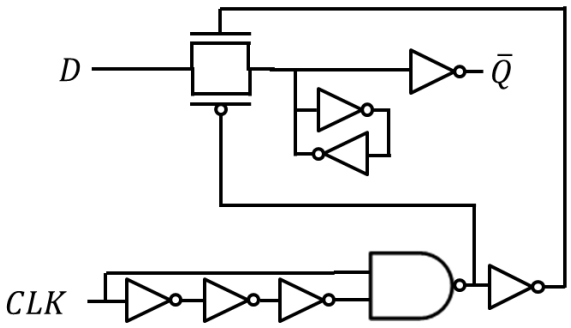
Problem	Points	
1	20	
2	10	
3	10	
4	20	
5	10	
6	10	
7	40	
8	50	
Total	170	

* Allowed: Textbooks, cheat sheets, class notes, notebooks, calculators, watches, electronic devices.

* Not allowed: Chat apps.

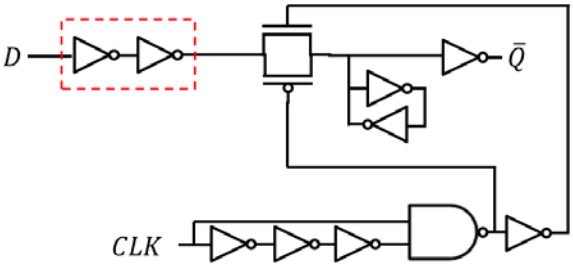
Problem #1 (Sequential Logic, 20 points)

Schematic A below shows an explicit-pulsed D flip-flop. The signal strength of input D is unknown (e.g., someone designed a circuit and its output is connected to input D).



<Schematic A>

Since we have no clue on the strength of input D, we suggest the following D-F/F design. We can properly size the two inverters in the dotted rectangle.



<Schematic B>

Question: Compare Schematic A and Schematic B (quantitatively and/or qualitatively) in terms of 1) setup time constraint, 2) hold time constraint, 3) clock-to-Q delay, and 4) output slew ($\Delta V_{\bar{Q}}/\Delta t$) where $V_{\bar{Q}}$ is the voltage at the output node $\bar{Q}$.

1) Setup time constraint: $t_{setup,A}$ (<) > = $t_{setup,B}$
   The input of the transmission gate should be stable for t_{setup} before a clock rising edge. Thus, $t_{setup,A} = t_{setup}$. For Schematic B, the input D should be stable for $t_{setup} + 2t_{inv}$ before a clock rising edge (t_{inv} is the delay of an inverter).
2) Hold time constraint: $t_{hold,A}$ < (>) = $t_{hold,B}$
   The input of the transmission gate should be stage for t_{hold} after a clock rising edge. Thus, $t_{hold,A} = t_{hold}$. For Schematic B, a change at input D does not immediately lead to a change at the input value of the transmission gate. Thus, $t_{hold,B} = t_{hold} - 2t_{inv}$.
3) Clock-to-Q delay: $t_{C-Q,A}$ < > = $t_{C-Q,B}$

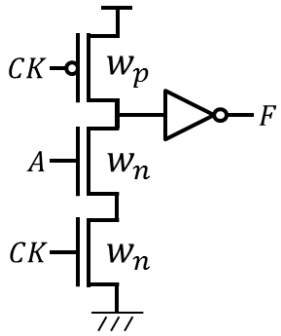
It depends on the strength of input D. If the driver of input D in Schematic A is weaker than that in Schematic B, $t_{C-Q,A} > t_{C-Q,B}$ will hold. If the driver of input D in Schematic A is stronger than that in Schematic B, $t_{C-Q,A} < t_{C-Q,B}$ will hold.

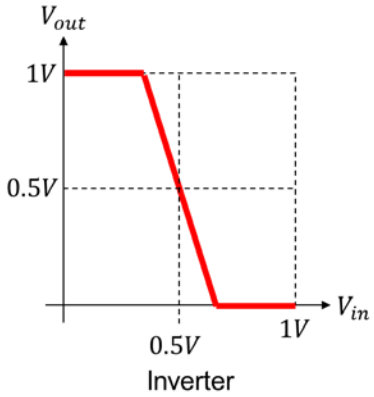
4) Output slew: s_A < > = s_B

For the same reason as the answer for 3), the output slew is dependent on the strength of the driver of input D.

Problem #2 (DC Analysis of a Domino Logic, 10 points)

The left figure shows a domino-logic-based buffer design and the right figure shows a DC curve of the inverter in the schematic. $\mu_n = 2\mu_p$. w: Transistor minimum width.



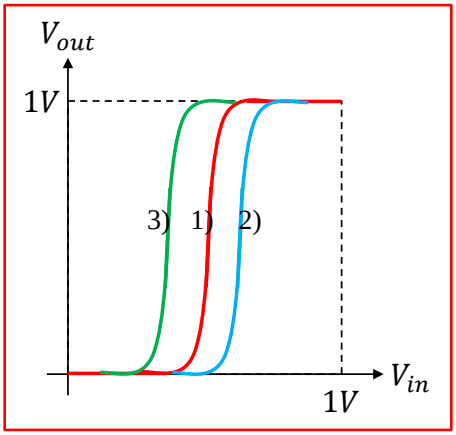


Draw (rough sketches will be accepted) three DC curves (x-axis: V_A, y-axis: V_F) for the buffer for

1) $w_p = w_n = w$

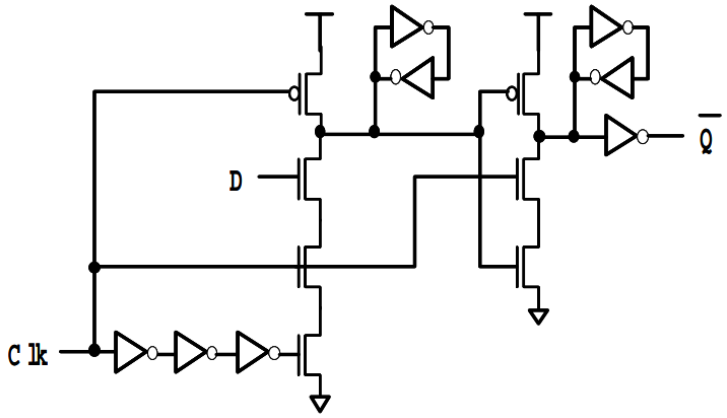
2) $w_p = 2w, w_n = w$

3) $w_p = w, w_n = 2w$



Problem #3 (Sequential Logic, 10 points)

The following shows a schematic of a positive-edge triggered D-F/F.

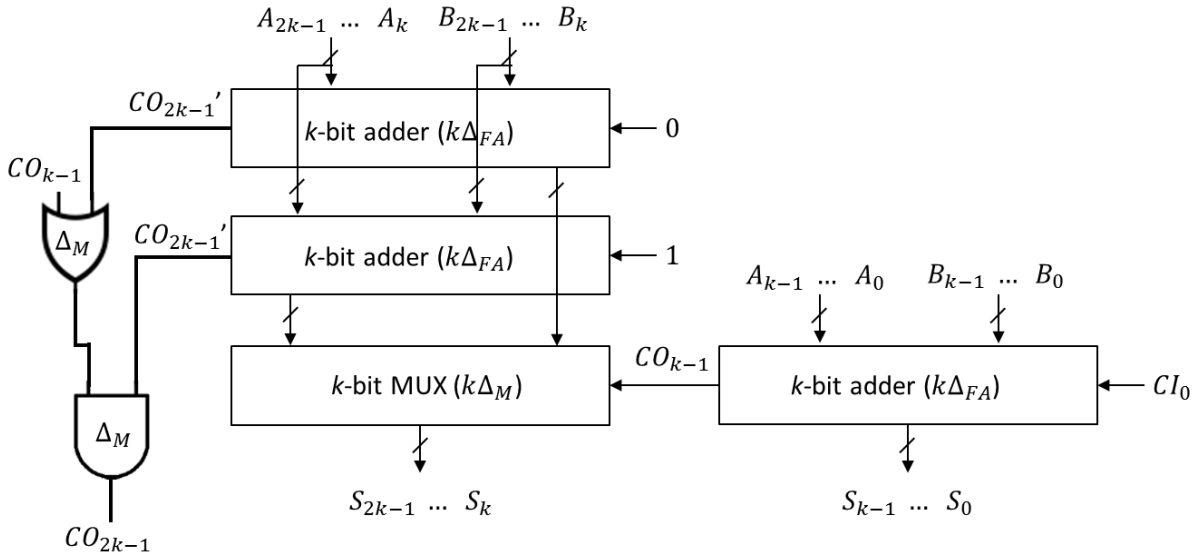


Describe how you can estimate the hold time constraint of the F/F above. (Hold time constraint: input D should be stable (should not change) for some time after a clock rising edge.)

Input D should be stable until the output of the third inverter in the inverter chain becomes 0 after each clock rising edge. Thus, the hold time constraint is approximately $3\Delta_{inv}$ where Δ_{inv} is the delay of an inverter.

Problem #4 (Carry Select Adder, 20 points)

The following shows a schematic of a *2k*-bit adder designed using *k*-bit carry select adders. The delay of a *k*-bit adder is $k\Delta_{FA}$, the delay of a *k*-bit MUX is $k\Delta_M$, and the delay of a two-input AND (or OR) gate is Δ_M.



1) We are supposed to design an *N*-bit adder using carry select adders (# groups: $\frac{N}{k}$). Find k minimizing the delay of the *N*-bit adder (express the optimal *k* as a function of *N*, Δ_M, and Δ_{FA}). Notice that the worst-case delay occurs at C_N (the final carry out) or $S_{N-1:0}$ (the final sum).

$$\tau = k\Delta_{FA} + \left(\frac{N}{k} - 2\right) \cdot 2\Delta_M + k\Delta_M$$

$$\frac{d\tau}{dk} = \Delta_{FA} - \frac{2N\Delta_M}{k^2} + \Delta_M = 0$$

$$\therefore k = \sqrt{\frac{2N\Delta_M}{\Delta_{FA} + \Delta_M}}$$

2) Now, the *k*-bit adders are designed using conditional sum adders, so the delay of a *k*-bit adder is $\Delta_M \cdot \ln k$ instead of $k\Delta_{FA}$. Find k minimizing the delay of the new *N*-bit adder (express the optimal *k* as a function of *N* and Δ_M).

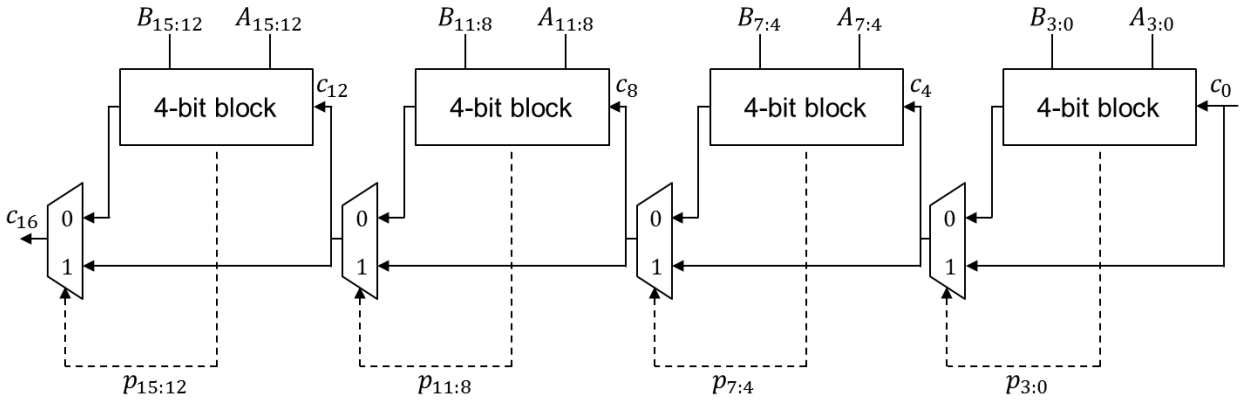
$$\tau = \Delta_M \cdot \ln k + \left(\frac{N}{k} - 2\right) \cdot 2\Delta_M + k\Delta_M$$

$$\frac{d\tau}{dk} = \frac{\Delta_M}{k} - \frac{2N\Delta_M}{k^2} + \Delta_M = 0$$

$$\therefore k = \frac{-1 + \sqrt{1 + 8N}}{2}$$

Problem #5 (Carry Skip Adder, 10 points)

The following diagram shows a 16-bit carry-skip adder designed using 4-bit adders.



To improve the speed of the carry-skip adder, we replace the 4-bit adders (4-bit blocks designed using 4-bit ripple-carry adders) by 4-bit conditional sum adders. The delay of each multiplexer step in the conditional sum adders is Δ_M (so, if all the operands are available at time 0, the delay of a 4-bit conditional sum adder is $3\Delta_M$.) The delay of each 2:1 MUX in the schematic above is Δ_M. The delay of each $p_{i:i-3}$ is $2\Delta_M$. Calculate the delay of the new 16-bit carry-skip adder.

Delay of the carry-out of any 4-bit adder block = $3\Delta_M$

Delay of any $p_{i+3:i}$ = $2\Delta_M$

Delay of $c_4 = MAX(3\Delta_M, 2\Delta_M) + \Delta_M = 4\Delta_M$

Delay of c_8 = $4\Delta_M + \Delta_M = 5\Delta_M$

Delay of c_{12} = $5\Delta_M + \Delta_M = 6\Delta_M$

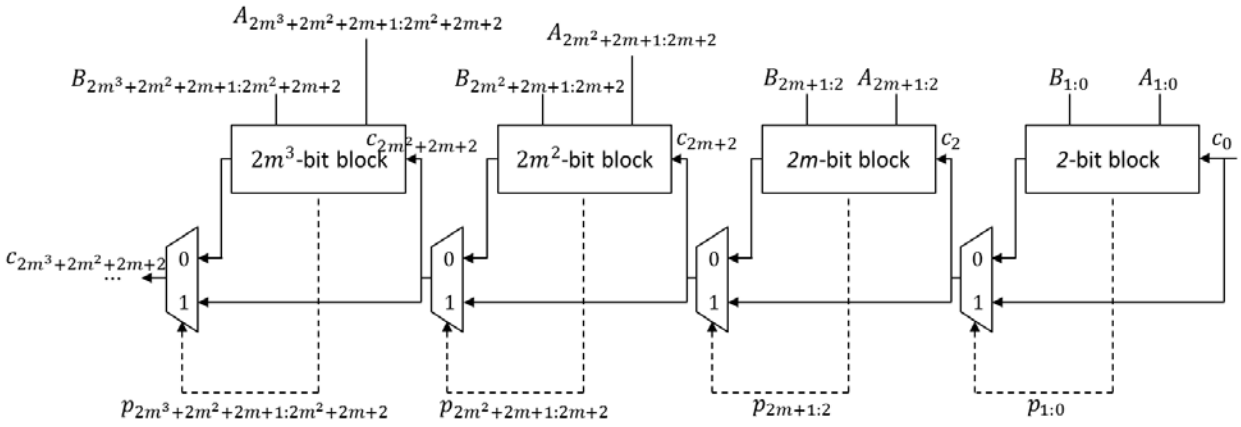
Delay of c_{16} = $6\Delta_M + \Delta_M = 7\Delta_M$

Delay of $s_{15:12}$ = $6\Delta_M + \Delta_M = 7\Delta_M$

Thus, the total delay is $7\Delta_M$.

Problem #6 (Carry Skip Adder, 10 points)

To radically improve the delay of a carry-skip adder, we design an *N*-bit carry-skip adder as follows:



We design the *k*-bit adder blocks using *k*-bit conditional sum adders where k is $2m^i$ (i is an integer greater than or equal to 0). We also design the group-propagation signal $p_{i:j}$ using OR gates hierarchically. The following shows the delays of the components:

- *k*-bit adder: $(1 + \ln k) \cdot \Delta_M$
- *k*-bit group-propagation signal $p_{i:i-k+1}$: $(2 + \ln \frac{k}{4}) \cdot \Delta_M$
- 2-bit MUX: Δ_M

Find ***m*** ($m \geq 2$) minimizing the total delay of an *N*-bit carry-skip adder designed using the new architecture shown above.

Delay of the carry-out of the first *2*-bit adder block = $(1 + \ln 2)\Delta_M$

Delay of $p_{1:0}$ = $(2 + \ln 0.5)\Delta_M = (2 - \ln 2)\Delta_M$

Delay of $c_2 = MAX\big((1 + \ln 2)\Delta_M, (2 + \ln 0.5)\Delta_M\big) + \Delta_M = (2 + \ln 2)\Delta_M$

Delay of the carry-out of the second *2m*-bit adder block = $(1 + \ln 2m)\Delta_M = (1 + \ln 2 + \ln m)\Delta_M$

Delay of $p_{2m+1:2}$ = $\left(2 + \ln \frac{2m}{4}\right) \cdot \Delta_M = (2 - \ln 2 + \ln m) \cdot \Delta_M$

Delay of $c_{2m+2} = MAX\big((1 + \ln 2 + \ln m)\Delta_M, (2 - \ln 2 + \ln m) \cdot \Delta_M\big) + \Delta_M = (2 + \ln 2 + \ln m)\Delta_M$

Delay of the carry-out of the third $2m^2$-bit adder block = $(1 + \ln 2m^2)\Delta_M = (1 + \ln 2 + 2\ln m)\Delta_M$

Delay of $p_{2m^2+2m+1:2m+2}$ = $\left(2 + \ln\frac{2m^2}{4}\right) \cdot \Delta_M = (2 - \ln 2 + 2\ln m) \cdot \Delta_M$

Delay of $c_{2m^2+2m+2} = MAX\left((1 + \ln 2 + 2\ln m)\Delta_M, (2 - \ln 2 + 2\ln m) \cdot \Delta_M\right) + \Delta_M = (2 + \ln 2 + 2\ln m)\Delta_M$

...

Delay of $c_N = (2 + \ln 2 + x\ln m)\Delta_M$ where x satisfies

$$2 + 2m + \cdots + 2m^x = N$$

$$2 + 2m + \cdots + 2m^x = \frac{2(m^{x+1} - 1)}{m - 1} = N$$

$$x = \frac{\ln\left(1 + \frac{N(m-1)}{2}\right)}{\ln m} - 1$$

Thus, the delay of
$c_N = \left(2 + \ln 2 + \ln\left(1 + \frac{N(m-1)}{2}\right) - \ln m\right)\Delta_M = \left(2 + \ln 2 + \ln\left(\frac{Nm-N+2}{2m}\right)\right)\Delta_M$

(The MSB sum has the same delay.)

To minimize the delay of c_N, m=**2**.

Problem #7 (Prefix Adder, 40 points)

Use the following delay values:

- AND, OR, XOR: Δ
- Two-level (sum-of-product) logic: 2Δ
- $g_i = a_i \cdot b_i$, $p_i = a_i \oplus b_i$

We are designing a 1024-bit Kogge-Stone adder.

1) Represent s_{999} hierarchically using group-generated and group-propagated carries ($g_{i:k}$, $p_{i:k}$) and c_0 (primary carry-in), then compute the delay to compute s_{999} assuming all the primary input signals are available at time 0 (10 points).

$$s_{999} = p_{999} \oplus c_{999}$$

$$c_{999} = g_{998:0} + p_{998:0} \cdot c_0$$

$$g_{998:0} = g_{998:487} + p_{998:487} \cdot g_{486:0}$$

$$g_{998:487} = g_{998:743} + p_{998:743} \cdot g_{742:487}$$

$$g_{998:743} = g_{998:871} + p_{998:871} \cdot g_{870:743}$$

$$g_{998:871} = g_{998:935} + p_{998:935} \cdot g_{934:871}$$

$$g_{998:935} = g_{998:967} + p_{998:967} \cdot g_{966:935}$$

$$g_{998:967} = g_{998:983} + p_{998:983} \cdot g_{982:967}$$

$$g_{998:983} = g_{998:991} + p_{998:991} \cdot g_{990:983}$$

$$g_{998:991} = g_{998:995} + p_{998:995} \cdot g_{994:991}$$

$$g_{998:995} = g_{998:997} + p_{998:997} \cdot g_{996:995}$$

$$g_{998:997} = g_{998} + p_{998} \cdot g_{997}$$

Delay = $\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + \Delta = 24\Delta$

2) Represent s_{768} hierarchically using group-generated and group-propagated carries ($g_{i:k}$, $p_{i:k}$) and c_0 (primary carry-in), then compute the delay to compute s_{768} assuming all the primary input signals are available at time 0 (10 points).

$$s_{768} = p_{768} \oplus c_{768}$$

$$c_{768} = g_{767:0} + p_{767:0} \cdot c_0$$

$$g_{767:0} = g_{767:256} + p_{767:256} \cdot g_{255:0}$$

$$g_{767:256} = g_{767:512} + p_{767:512} \cdot g_{511:256}$$

$$g_{767:512} = g_{767:640} + p_{767:640} \cdot g_{639:512}$$

$$g_{767:640} = g_{767:704} + p_{767:704} \cdot g_{703:640}$$

$$g_{767:704} = g_{767:736} + p_{767:736} \cdot g_{735:704}$$

$$g_{767:736} = g_{767:752} + p_{767:752} \cdot g_{751:736}$$

$$g_{767:752} = g_{767:760} + p_{767:760} \cdot g_{759:752}$$

$$g_{767:760} = g_{767:764} + p_{767:764} \cdot g_{763:760}$$

$$g_{767:764} = g_{767:766} + p_{767:766} \cdot g_{765:764}$$

$$g_{767:766} = g_{767} + p_{767} \cdot g_{766}$$

Delay = $\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + \Delta = 24\Delta$

3) Calculate the total gate area to build the 1024-bit Kogge-Stone adder. Use the following area values. Notice that you should generate all sum ($s_{1023:0}$) and carry-out (c_{1024}) signals (10 points).

- Two-input AND, OR gate: k
- Two-input XOR: $4k$
- $g_i = a_i \cdot b_i$, $p_i = a_i \oplus b_i$

For position j, $s_j = p_j \oplus c_j$, so we need an XOR for each s_j. (1024*4k)

For position j, we should generate g_j and p_j, so we need an AND and an XOR for each j. (1024*5k)

A unit merging g_i, p_i, g_j, p_j needs one OR and two AND gates.

Now, let's count how many those units we need.

Level 1: $g, p_{1023:1022}, \dots, g, p_{1:0}$ => 1023

Level 2: $g, p_{1023:1020}, \dots, g, p_{2:0}$ => 1022

Level 3: $g, p_{1023:1016}, \dots, g, p_{4:0}$ => 1020

Level 4: $g, p_{1023:1008}, \dots, g, p_{8:0}$ => 1016

Level 5: $g, p_{1023:992}, \dots, g, p_{16:0}$ => 1008

Level 6: $g, p_{1023:960}, \dots, g, p_{32:0}$ => 992

Level 7: $g, p_{1023:896}, \dots, g, p_{64:0}$ => 960

Level 8: $g, p_{1023:768}, \dots, g, p_{128:0}$ => 896

Level 9: $g, p_{1023:512}, \dots, g, p_{256:0}$ => 768

Level 10: $g, p_{1023:0}, \dots, g, p_{512:0}$ => 512

so we need 9217 merging units. (9217*3k=27651k)

$c_j = g_{j-1:0} + p_{j-1:0} \cdot c_0$, so we need an AND and an OR to generate c_j. (1024*2k)

Thus, the total area = 1024*4k + 1024*5k + 27651k + 1024*2k = **38,915k**

4) When we designed a Kogge-Stone adder, we used an XOR gate to calculate $p_i = a_i \oplus b_i$. Prove that we can also use $a_i + b_i$ (+ is an OR operation) for p_i, i.e., prove that replacing $p_i = a_i \oplus b_i$ by $p_i = a_i + b_i$ does not change the final sum and carry-out values. (10 points).

The only difference between $a_i + b_i$ and $a_i \oplus b_i$ is when both a_i and b_i are 1. For position i, the carry-out is $c_i = g_i + p_i c_{i-1}$. If the OR is used for p_i, $c_i = 1 + c_{i-1} = 1$ if both a_i and b_i are 1. If the XOR is used for p_i, $c_i = 1 + 0 = 1$. Thus, they have the same carry out value. Thus, the replacement does not change the final sum and carry-out values.

Problem #8 (Carry Look-Ahead Adder, 50 points)

The max. fanout is 4. Use the following delay values:

- AND, OR, XOR: Δ
- Two-level (sum-of-product) logic: 2Δ
- $g_i = a_i \cdot b_i$, $p_i = a_i \oplus b_i$

We are designing a 1024-bit carry look-ahead adder.

1) Represent s_{999} hierarchically using group-generated and group-propagated carries ($g_{i:k}$, $p_{i:k}$) and c_0 (primary carry-in), then compute the delay to compute s_{999} assuming all the primary input signals are available at time 0 (10 points).

$$s_{999} = p_{999} \oplus c_{999}$$

$$c_{999} = g_{998} + p_{998} \cdot g_{997} + p_{998} \cdot p_{997} \cdot g_{996} + p_{998} \cdot p_{997} \cdot p_{996} \cdot c_{996}$$

$$c_{996} = g_{995:992} + p_{995:992} \cdot c_{992}$$

$$c_{992} = g_{991:976} + p_{991:976} \cdot g_{975:960} + p_{991:976} \cdot p_{975:960} \cdot c_{960}$$

$$c_{960} = g_{959:896} + p_{959:896} \cdot g_{895:832} + p_{959:896} \cdot p_{895:832} \cdot g_{831:768} + p_{959:896} \cdot p_{895:832} \cdot p_{831:768} \cdot c_{768}$$

$$c_{768} = g_{767:512} + p_{767:512} \cdot g_{511:256} + p_{767:512} \cdot p_{511:256} \cdot g_{255:0} + p_{767:512} \cdot p_{511:256} \cdot p_{255:0} \cdot c_0$$

$$g_{255:0} = g_{255:192} + p_{255:192} \cdot g_{191:128} + p_{255:192} \cdot p_{191:128} \cdot g_{127:64} + p_{255:192} \cdot p_{191:128} \cdot p_{127:64} \cdot g_{63:0}$$

$$g_{63:0} = g_{63:48} + p_{63:48} \cdot g_{47:32} + p_{63:48} \cdot p_{47:32} \cdot g_{31:16} + p_{63:48} \cdot p_{47:32} \cdot p_{31:16} \cdot g_{15:0}$$

$$g_{15:0} = g_{15:12} + p_{15:12} \cdot g_{11:8} + p_{15:12} \cdot p_{11:8} \cdot g_{7:4} + p_{15:12} \cdot p_{11:8} \cdot p_{7:4} \cdot g_{3:0}$$

$$g_{3:0} = g_3 + p_3 \cdot g_2 + p_3 \cdot p_2 \cdot g_1 + p_3 \cdot p_2 \cdot p_1 \cdot g_0$$

Delay = $\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + 2\Delta + \Delta = 20\Delta$

2) Represent s_{768} hierarchically using group-generated and group-propagated carries ($g_{i:k}$, $p_{i:k}$) and c_0 (primary carry-in), then compute the delay to compute s_{768} assuming all the primary input signals are available at time 0 (10 points).

$$s_{768} = p_{768} \oplus c_{768}$$

$$c_{768} = g_{767:512} + p_{767:512} \cdot g_{511:256} + p_{767:512} \cdot p_{511:256} \cdot g_{255:0} + p_{767:512} \cdot p_{511:256} \cdot p_{255:0} \cdot c_0$$

$$g_{255:0} = g_{255:192} + p_{255:192} \cdot g_{191:128} + p_{255:192} \cdot p_{191:128} \cdot g_{127:64} + p_{255:192} \cdot p_{191:128} \cdot p_{127:64} \cdot g_{63:0}$$

$$g_{63:0} = g_{63:48} + p_{63:48} \cdot g_{47:32} + p_{63:48} \cdot p_{47:32} \cdot g_{31:16} + p_{63:48} \cdot p_{47:32} \cdot p_{31:16} \cdot g_{15:0}$$

$$g_{15:0} = g_{15:12} + p_{15:12} \cdot g_{11:8} + p_{15:12} \cdot p_{11:8} \cdot g_{7:4} + p_{15:12} \cdot p_{11:8} \cdot p_{7:4} \cdot g_{3:0}$$

$$g_{3:0} = g_3 + p_3 \cdot g_2 + p_3 \cdot p_2 \cdot g_1 + p_3 \cdot p_2 \cdot p_1 \cdot g_0$$

Delay = $\Delta + 5 * (2\Delta) + \Delta = 12\Delta$

3) Calculate the total gate area to build the 1024-bit carry look-ahead adder. Use the following area values. Notice that you should generate all sum ($s_{1023:0}$) and carry-out (c_{1024}) signals (20 points).

- Two-input AND, OR gate: k
- Three-input AND, OR gate: $2k$
- Four-input AND, OR gate: $3k$
- Two-input XOR: $4k$
- $g_i = a_i \cdot b_i$, $p_i = a_i \oplus b_i$
- For a carry look-ahead unit for $a_{i+3:i}, b_{i+3:i}, c_i$, use the following formulae:
  - $c_{i+1} = g_i + p_i \cdot c_i$ (for this, you need a two-input OR and a two-input AND)
  - $c_{i+2} = g_{i+1} + p_{i+1} \cdot g_i + p_{i+1} \cdot p_i \cdot c_i$ (for this, you need a three-input AND, a two-input AND, a three-input OR)
  - $c_{i+3} = g_{i+2} + p_{i+2} \cdot g_{i+1} + p_{i+2} \cdot p_{i+1} \cdot g_i + p_{i+2} \cdot p_{i+1} \cdot p_i \cdot c_i$ (for this, you need a four-input AND, three-input AND, a two-input AND, a four-input OR)
- For a group carry look-ahead unit for $g_{i+3:i}, p_{i+3:i}, c_i$, use the following formulae:
  - g' (group generation) $= g_{i+3} + p_{i+3} \cdot g_{i+2} + p_{i+3} \cdot p_{i+2} \cdot g_{i+1} + p_{i+3} \cdot p_{i+2} \cdot p_{i+1} \cdot g_i$ (for this, you need a two-input AND, a three-input AND, a four-input AND, a four-input OR)
  - p' (group propagation) $= p_{i+3} \cdot p_{i+2} \cdot p_{i+1} \cdot p_i$ (for this, you need a four-input AND)
  - c' (group carry) $= g' + p' \cdot c_i$ (for this, you need a two-input AND and a two-input OR)

g, p_i for each i: 1024*(k+4k)

s_i for each i: 1024*(4k)

Level-1 carry look-ahead unit: 2k for c_{i+1}, 5k for c_{i+2}, 9k for c_{i+3}, 9k for group g, 3k for group p, total 28k

Level-1 carry look-ahead units: 1024/4 = 256

For all the other carry look-ahead unit: 9k for group g, 3k for group p, and 2k for c', total 14k.

Level-2 carry look-ahead units: 1024/16 = 64

Level-3 carry look-ahead units: 1024/64 = 16

Level-4 carry look-ahead units: 1024/256 = 4

Level-5 carry look-ahead units: 1024/1024 = 1

The total area = 1024*5k + 1024*4k + 256*28k + 85*14k = **17,574k**

The max. fanout is 2. Use the following delay values:

- AND, OR, XOR: Δ
- Two-level (sum-of-product) logic: 2Δ
- $g_i = a_i \cdot b_i$, $p_i = a_i \oplus b_i$

We are designing a 1024-bit carry look-ahead adder.

4) Represent s_{999} hierarchically using group-generated and group-propagated carries ($g_{i:k}$, $p_{i:k}$) and c_0 (primary carry-in), then compute the delay to compute s_{999} assuming all the primary input signals are available at time 0 (10 points).

$$s_{999} = p_{999} \oplus c_{999}$$

$$c_{999} = g_{998} + p_{998} \cdot c_{998}\ (2X)$$

$$c_{998} = g_{997:996} + p_{997:996} \cdot c_{996}\ (4X)$$

$$c_{996} = g_{995:992} + p_{995:992} \cdot c_{992}\ (8X)$$

$$c_{992} = g_{991:960} + p_{991:960} \cdot c_{960}\ (64X)$$

$$c_{960} = g_{959:896} + p_{959:896} \cdot c_{896}\ (128X)$$

$$c_{896} = g_{895:768} + p_{895:768} \cdot c_{768}\ (256X)$$

$$c_{768} = g_{767:512} + p_{767:512} \cdot c_{512}\ (512X)$$

$$c_{512} = g_{511:0} + p_{511:0} \cdot c_0\ (1024X)$$

$$g_{511:0} = g_{511:256} + p_{511:256} \cdot g_{255:0}$$

$$g_{255:0} = g_{255:128} + p_{255:128} \cdot g_{127:0}$$

$$g_{127:0} = g_{127:64} + p_{127:64} \cdot g_{63:0}$$

$$g_{63:0} = g_{63:32} + p_{63:32} \cdot g_{31:0}$$

$$g_{31:0} = g_{31:16} + p_{31:16} \cdot g_{15:0}$$

$$g_{15:0} = g_{15:8} + p_{15:8} \cdot g_{7:0}$$

$$g_{7:0} = g_{7:4} + p_{7:4} \cdot g_{3:0}$$

$$g_{3:0} = g_{3:2} + p_{3:2} \cdot g_{1:0}$$

$$g_{1:0} = g_1 + p_1 \cdot g_0$$

$$\text{Delay} = \Delta + 17 * (2\Delta) + \Delta = 36\Delta$$