EECS 151/251A Homework 9

Due Friday, Nov 20$^{\text{th}}$, 2020

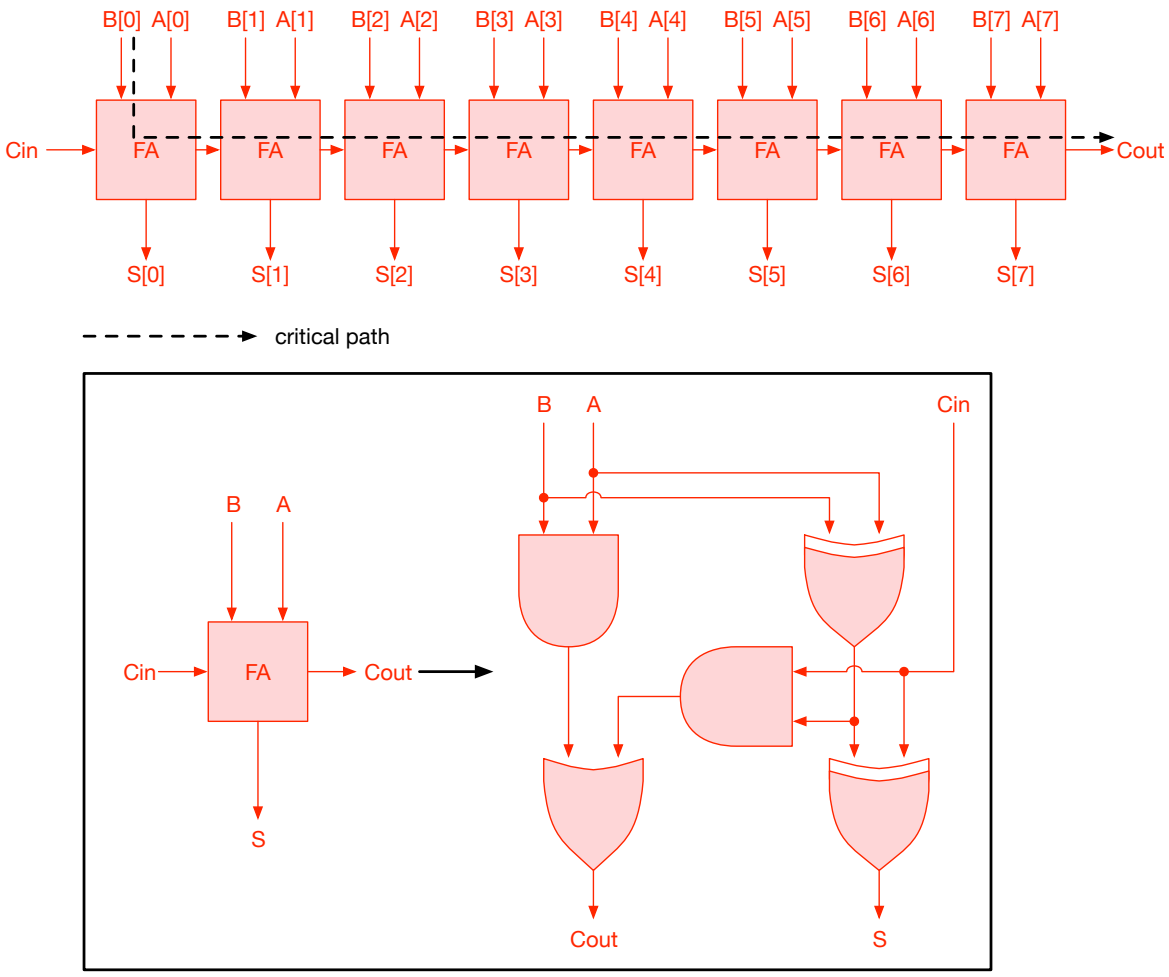
Problem 1: Adders

a) Draw the block diagram of an 8-bit ripple carry adder and an 8-bit carry-lookahead adder. For each, derive the critical path in terms of the following constants. Use only 2-input gates. Include the final carry-out bit.

$t_{MUX} = 6ps$
$t_{OR} = 5ps$
$t_{AND} = 4ps$
$t_{XOR} = 6ps$

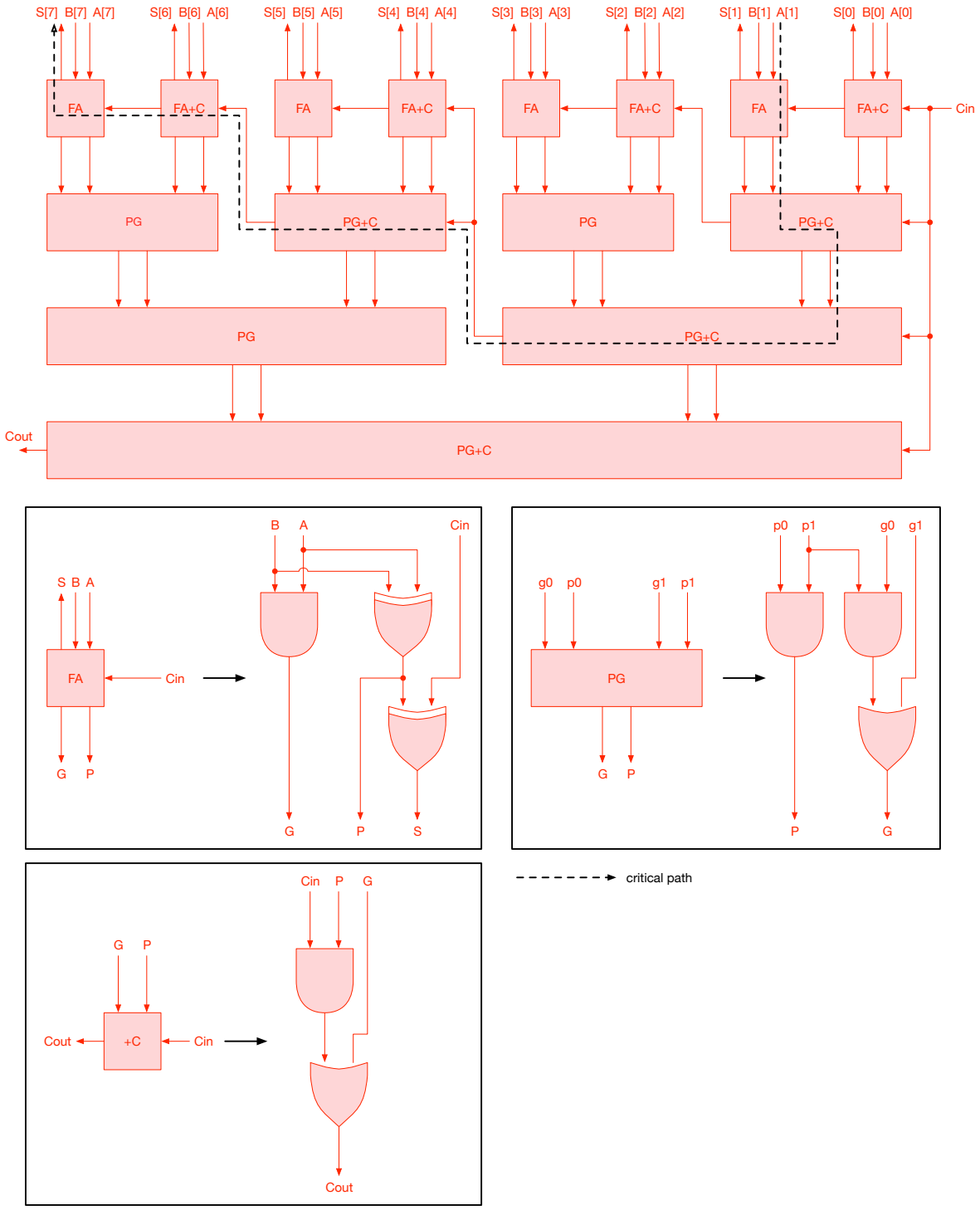
Solution:

Note: Your answers may be slightly different if you chose a different full adder topology.
Ripple Carry:



$$t_{ripple-carry} = t_{XOR} + t_{AND} + t_{OR} + (8-1)\cdot(t_{AND} + t_{OR}) = 6ps + 4ps + 5ps + 7\cdot(4ps + 5ps) = 78ps$$

Carry-Lookahead:



$$t_{carry-lookahead} = 6ps + 2 \cdot (4ps + 5ps) + 5ps + 2 \cdot (4ps + 5ps) + 6ps = 53ps$$

b) Now derive, generally, the critical path for an N-bit ripple-carry adder and an N-bit carry-lookahead adder. It is okay to make approximations for cases where $\sqrt{N}$ or $log_2(N)$ are not integers. Use only 2-input gates and include the final carry-out bit.

Solution:

$$t_{ripple-carry} = t_{XOR} + t_{AND} + t_{OR} + (N-1) \cdot (t_{AND} + t_{OR})$$

$$t_{carry-lookahead} = t_{XOR} + (\lceil log_2(N) \rceil - 1) \cdot (t_{AND} + t_{OR}) + t_{OR} + (\lceil log_2(N) \rceil - 1) \cdot (t_{AND} + t_{OR}) + t_{XOR}$$

Problem 2: Tree Adders

The goal of this problem is to design a 10-bit Kogge-Stone adder optimized for delay:

a) Design the following logic blocks at a gate level. You may draw the gates or write the Boolean functions. Use the given inputs and outputs as hints.

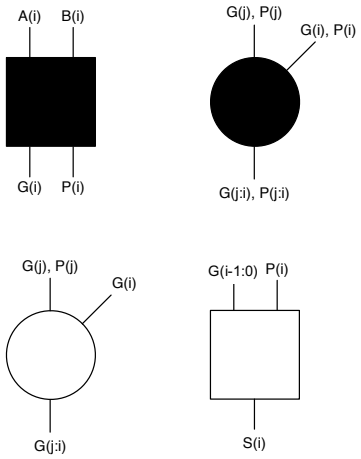


Figure 1: Building Blocks for Kogge-Stone Adder

Solution:

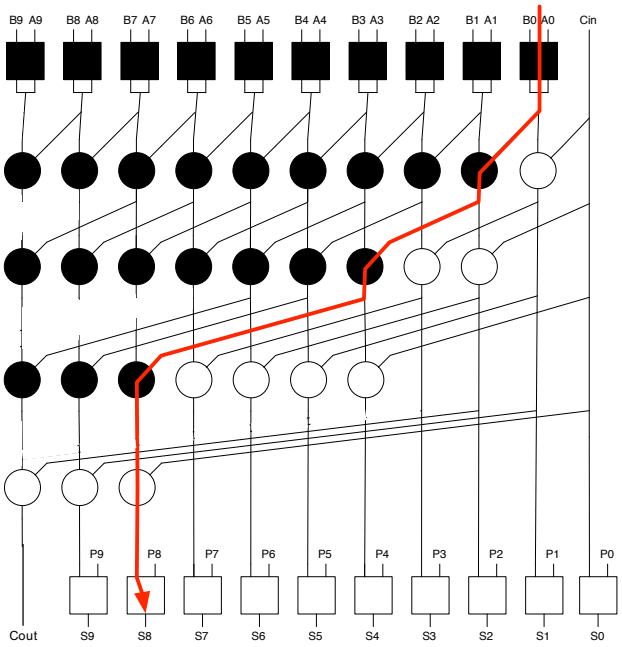
Black square (Modified FA): $G_i = A_i B_i$, $P_i = A_i \oplus B_i$
Black circle (Processing): $G_{j:i} = G_j + P_j G_i$, $P_{j:i} = P_j P_i$
White circle (Buffer): $G_{j:i} = G_j + P_j G_i$
White square (Simplified FA): $S_i = P_i \oplus G_{i-1:0}$

b) Using the logic blocks you designed in part (a), design a 10-bit logarithmic adder with a carry input and a carry output. Use a radix-2 Kogge-Stone implementation. Highlight the critical path of your design? Give a block-level estimate, assuming that more complex blocks have more delay.

Solution:

The first stage is the black boxes: here we generate the bit propagate (P_i) and generate (G_i) signals that will be used by the tree. For the actual tree, the Kogge-Stone implementation first groups the (P_i, G_i) in groups of 2, therefore generating $(P_{1:0}, G_{1:0})$, $(P_{2:1}, G_{2:1})$ etc. Then those signals are grouped again in groups of 2 to form $(P_{3:0}, G_{3:0})$, $(P_{4:1}, G_{4:1})$ etc.

The key here is that you need to incorporate the C_{in} signal into the tree. Remember that in order to get a sum bit you need $S_i = P_i \oplus C_i = P_i \oplus G_{i-1:0}$. Therefore for S_0 you need P_0 and C_{in}, for S_1 you need P_1 and $G_0 + P_0 C_{in}$ etc. So we add the white circles to generate the carries needed for the final sum, including the C_{in}.



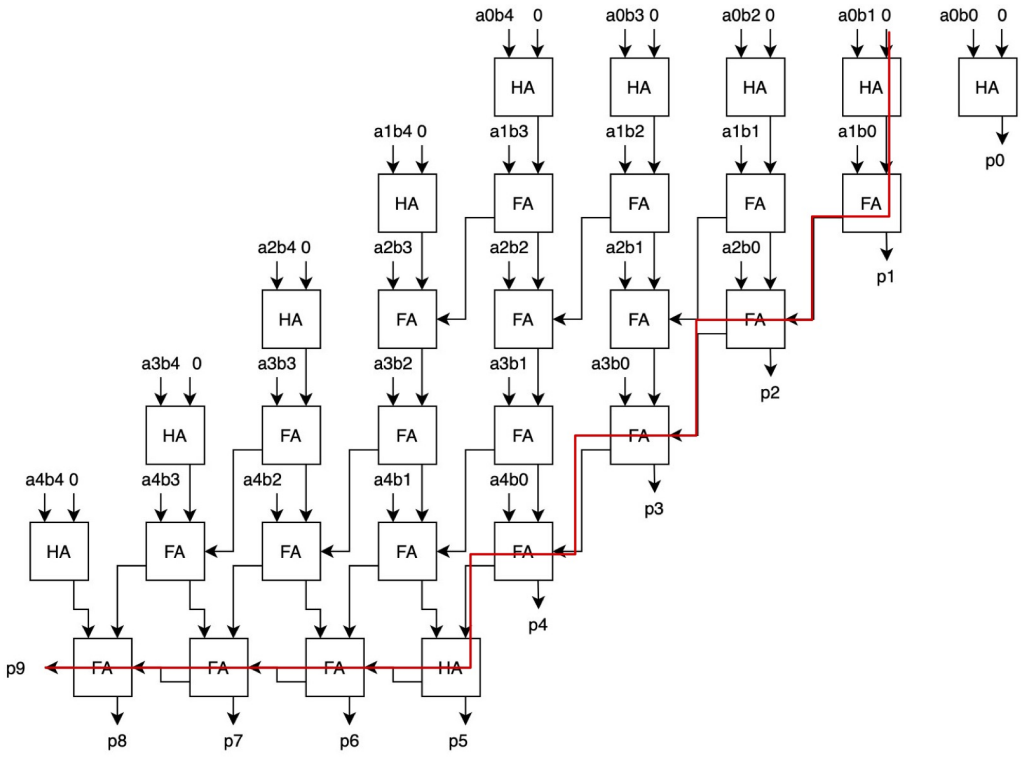
The critical path is shown on the tree (one example - there are multiple critical paths). In this case, a block-level estimate of the critical path is: $t_d = t_{blacksquare} + 3 * t_{blackcircle} + t_{whitecircle} + t_{whitesquare}$.

Problem 3: Multipliers

a) Draw a 5 x 5 Carry-Save multiplier and compute the critical path using t_{carry}, t_{sum} and t_{and}.

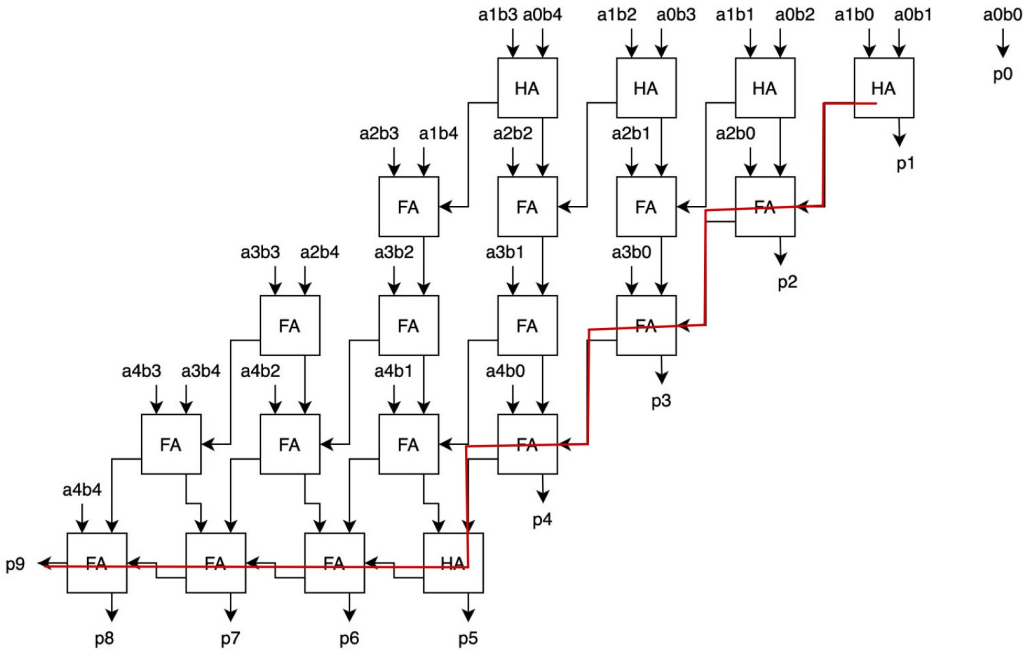
Solution:

Following the multiplier structure on lecture 20 slide 23, we have the carry-save multiplier below, with the critical path marked in red.
$t_{critical} = t_{and} + t_{HA,sum} + 7t_{FA,carry} + t_{HA,carry}$



We can make the carry-save multiplier more compact by removing the HAs that take 0 as an input.

$t_{critical} = t_{and} + 2t_{HA,carry} + 6t_{FA,carry}$



b) Draw a wallace tree for a 5 x 5 multiplier using Full Adder and Half Adder cells. What is the critical path?

Solution:

Showing the steps for the dot diagram:

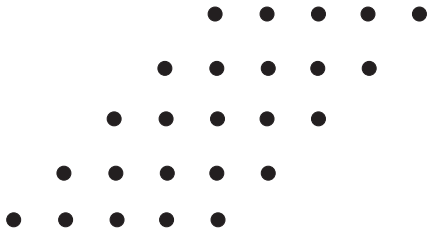


Figure 2: Original organization of partial products. Note that each dot represents a partial product.

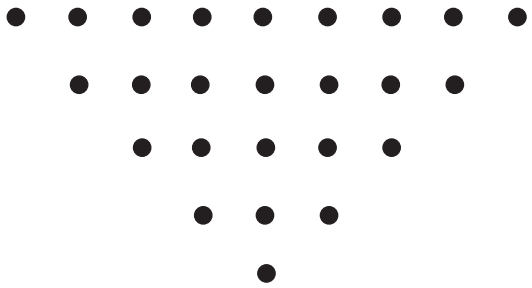


Figure 3: We rearrange the partial products in order to make grouping easier.

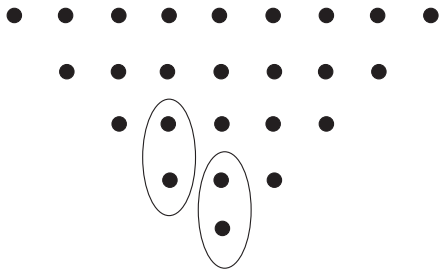


Figure 4: The circles show the first two groups, i.e. the first stage of the tree. We have two groups of two dots each and no carries so far, so we need two half adders for the first stage. After the first stage evaluates, the generated carries will pass to the left, and appear as dots on the next dot diagram.

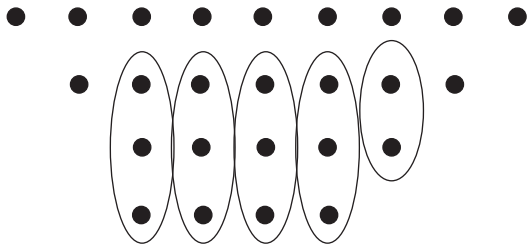


Figure 5: Now we can group the dots for the second stage, including the carries generated before. For every group of 3 we will use a FA, and we will use a HA for the group of 2. Again, the generated carries from each column will appear as a dot on the left column in the next diagram.

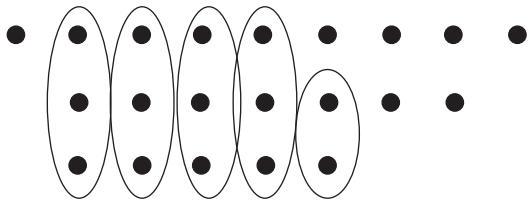


Figure 6: We keep grouping in a similar manner.

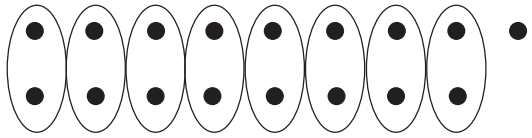


Figure 7: This is the final stage. Note that in this stage we will have a carry propagate-adder. This means that only the right-most adder will be a half adder, and all the others will be full adders (adding the two dots, plus the carry-out of the previous stage).

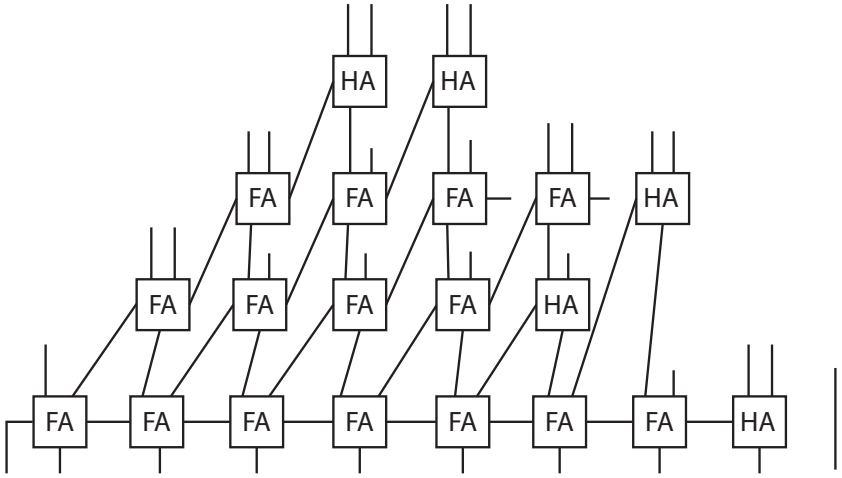


Figure 8: Showing this using Full Adders and Half Adders

Note that you can use a fast (carry-bypass, carry-select, etc.) type of adder for the last stage instead of the ripple-carry.

The critical path for the above diagram is shown below, and the delay is $t_d = t_{HA} + 7 * t_{FA}$. Note that there are different implementations for this problem, so if you used e.g. a fast adder in the final stage, your critical path may be shorter.

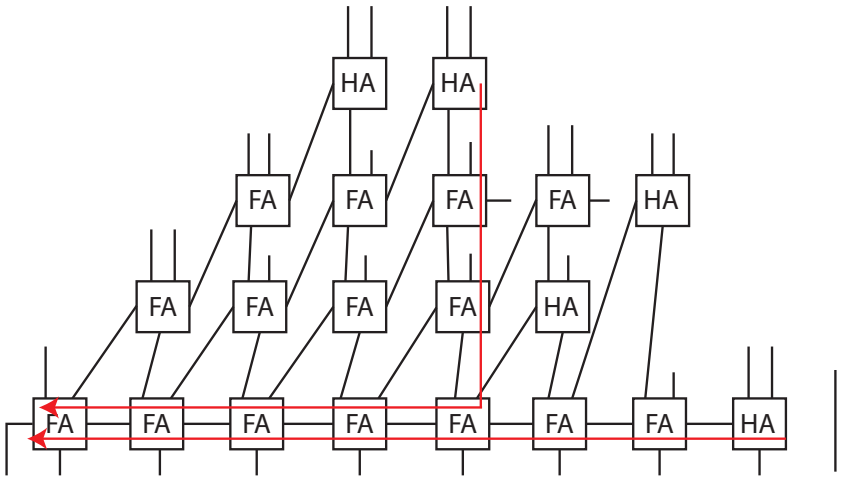


Figure 9: The critical path

Problem 4: Latches & Flip-flops

Consider the latch design shown below. You may assume that the inverters are symmetrical with input capacitance C, self-loading capacitance also C and equivalent driving resistance R. The transmission gate is sized to have an equivalent resistance R and parasitic capacitance C on each side.

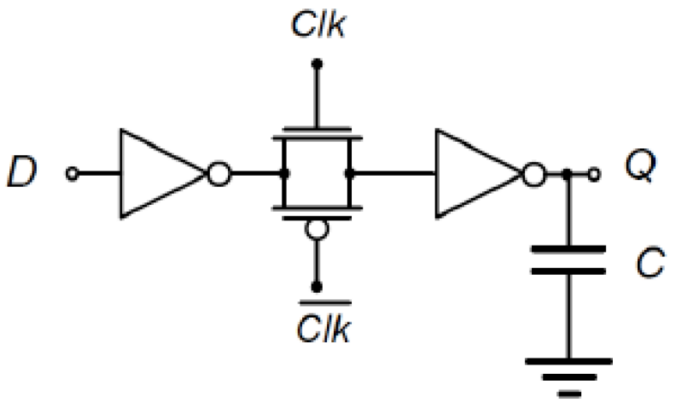


Figure 10: Dynamic Latch

a) Calculate the Clk-Q and D-Q delays as a function of R and C of this latch. In each case, show the equivalent RC circuit and explain.

Solution:

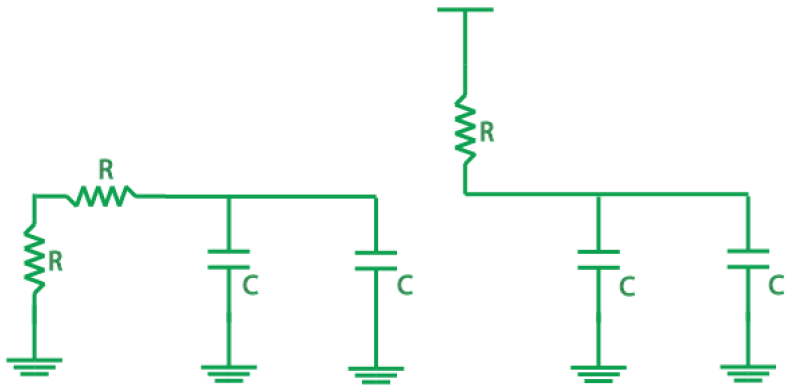


Figure 11: Clk-Q RC model

$$t_{clk-q} = \ln(2)(2R \cdot 2C + R \cdot 2C) = 6\ln(2)RC$$

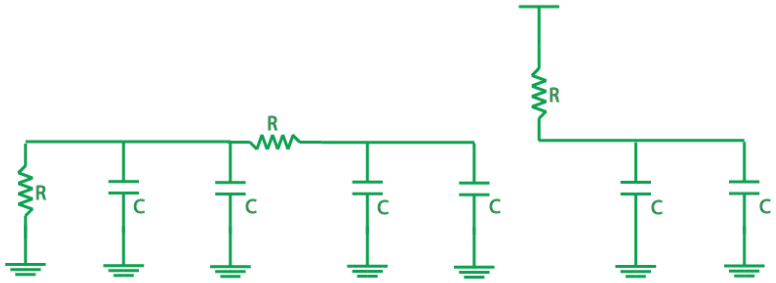


Figure 12: D-Q RC model

$$t_{d-q} = \ln(2)(R \cdot 4C + R \cdot 2C + R \cdot 2C) = 8\ln(2)RC$$

b) What is the approximate setup time for this latch? Explain your answer.

Solution:

Before the clock goes low, we want the data to have settled at the input of the second inverter.

$$t_{setup} \approx \ln(2)(R \cdot 4C + R \cdot C) = 6\ln(2)RC$$

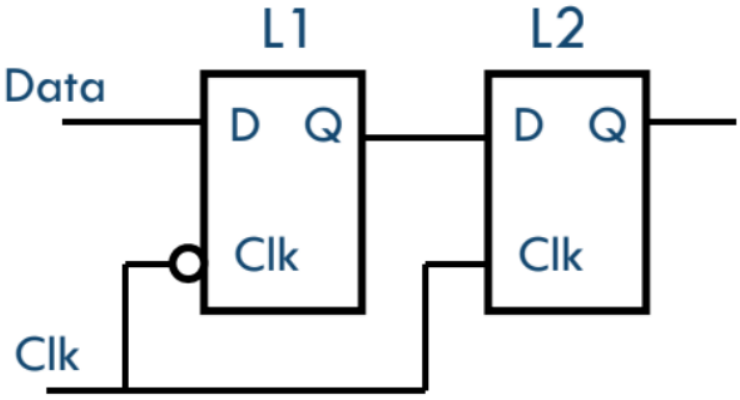


Figure 13: flip-flop

c) We build a flip-flop(figure 13) with two dynamic latches. Is the flip-flop positive-edge triggered or negative-edge triggered?

Solution:

The flip-flop is positive-edge triggered.