Digital Design and Computer Architecture (252-0028-00L), Spring 2021
Optional HW 1: DRAM Refresh and Combinational Logic
SOLUTIONS

Instructor: Prof. Onur Mutlu
TAs: Juan Gomez-Luna, Jisung Park, Hasan Hassan, Mohammed Alser, Lois Orosa, Minesh Patel, Jawad Haj-Yahya, Haiyu Mao, Behzad Salami, Jeremie Kim, Giray Yaglikci, Can Firtina, Geraldo De Oliveira Junior, Rahul Bera, Konstantinos Kanellopoulos, Nika Mansouri, Gagandeep Singh

Released: Thursday, March 5, 2021

1 DRAM Refresh

A new supercomputer has a DRAM-based memory system with the following configuration:

- The total capacity is 1 ExaByte (EB).
- The DRAM row size is 8 KiloByte (KB).
- The minimum retention time among all DRAM rows in the system is 64 ms. In order to ensure that no data is lost, every DRAM row is refreshed once every 64 ms. (Note: For each calculation in this question, you may leave your answer in simplified form in terms of powers of 2 and powers of 10.)

(a) How many DRAM rows does the memory system have?

Capacity = 2^{60} and row size = 2^{13}, so there are 2^{47} rows.

(b) How many DRAM refreshes happen in 64ms?

Because each row needs to be refreshed every 64ms, all rows need to be refreshed within 64ms.
Thus, the number of refreshes = number of rows = 2^{47}.

(c) What is the power consumption of DRAM refresh? (Hint: you will need to figure out how much current the DRAM device draws during refresh operations. You can find useful information in the technical note by Micron `https://safari.ethz.ch/digitaltechnik/spring2021/lib/exe/fetch.php?media=tn4704.pdf`. Use the current (IDD) numbers specified in the datasheet posted on the website `https://safari.ethz.ch/digitaltechnik/spring2021/lib/exe/fetch.php?media=1gb_ddr3_sdram.pdf`. Clearly state all the assumptions and show how you derive the power numbers. You are welcome to use other data sheets as well. Make sure you specify how you obtain the power numbers and show your calculations and thought process.)

Let $P_{refresh}$ be the power consumption of the DRAM device while performing only refresh operations. To find that number, we can use $P_{refresh} = IDD5B * V_{DD}$, where $IDD5B$ is the average current consumption while the DRAM device is being continuously refreshed. Both $IDD5B$ and V_{DD} can be found in the datasheet. There are many different $IDD5B$ parameters to pick from in the datasheet depending on the types of DRAM we are assuming.
If we assume DDR3-1066 with $IDD5B = 160mA$ (as can be seen in Table 19 in the datasheet), $P_{refresh} = 160mA * 1.5V = 240mW$.

(d) What is the total energy consumption of DRAM refresh during a refresh cycle? And during a day?

$Energy = Power * Time$. $Power$ is $P_{refresh}$ from part (c).
If $Time$ is 64ms (i.e., a refresh cycle), $Energy$ is $E = 240mW * 64ms = 1.54 * 10^{-2}J$.
If $Time$ is a day (i.e., $T = 24hrs * 60mins * 60secs$), $Energy$ is $E = 2.07 * 10^4 J$.

This question is an extended version of the question on Slide 17 in Lecture 3a:
https://safari.ethz.ch/digitaltechnik/spring2021/lib/exe/fetch.php?media=onur-digitaldesign_comparch-2021-lecture3a-mysteries-ii-beforelecture.pdf.

2 Main Memory Potpourri

A machine has a 4 GB DRAM main memory system. Each row is refreshed every 64 ms.

Note: This question is open ended. We provide one possible set of solutions. There could be other possible right solutions.

(a) The machine's designer runs two applications A and B (each run alone) on the machine. Although applications A and B have a similar number of memory requests, application A spends a surprisingly larger fraction of cycles stalling for memory than application B does? What might be the reasons for this?

A large number of application A's memory requests are row-buffer conflicts, whereas a large number of application B's memory requests are row-buffer hits.
Hence, application A's requests take longer to service and it spends more time stalling for memory.

(b) Application A also consumes a much larger amount of memory energy than application B does. What might be the reasons for this?

A row-buffer conflict consumes more energy than a a row-buffer hit. A row-buffer conflict requires a precharge, an activate and a read/write, whereas a row-buffer hit only requires a read/write.
Hence, application A consumes more memory energy.

(c) When applications A and B are run together on the machine, application A's performance degrades significantly, while application B's performance does not degrade as much. Why might this happen?

When the applications are run together, they interfere with each other. Hence, both applications' performance degrades when they run together. However, if a memory scheduler that favors row-buffer hits over row-buffer conflicts (like FR-FCFS) is used, it would favor application B's requests over application A's requests.
Therefore, application A's performance degrades more.

(d) The designer decides to use a smarter policy to refresh the memory. A row is refreshed only if it has not been accessed in the past 64 ms. Do you think this is a good idea? Why or why not?

This can reduce refresh energy significantly if a large fraction of the rows in memory contain data and are accessed (within the 64 ms refresh window), as these rows do not have to be refreshed explicitly. However, if only a small number of rows contain data and only these rows are accessed, this policy will not provide much reduction in refresh energy as a large fraction of rows are still refreshed at the 64 ms rate.

(e) When this new refresh policy is applied, the refresh energy consumption drops significantly during a run of application B. In contrast, during a run of application A, the refresh energy consumption reduces only slightly. Is this possible? Why or why not?

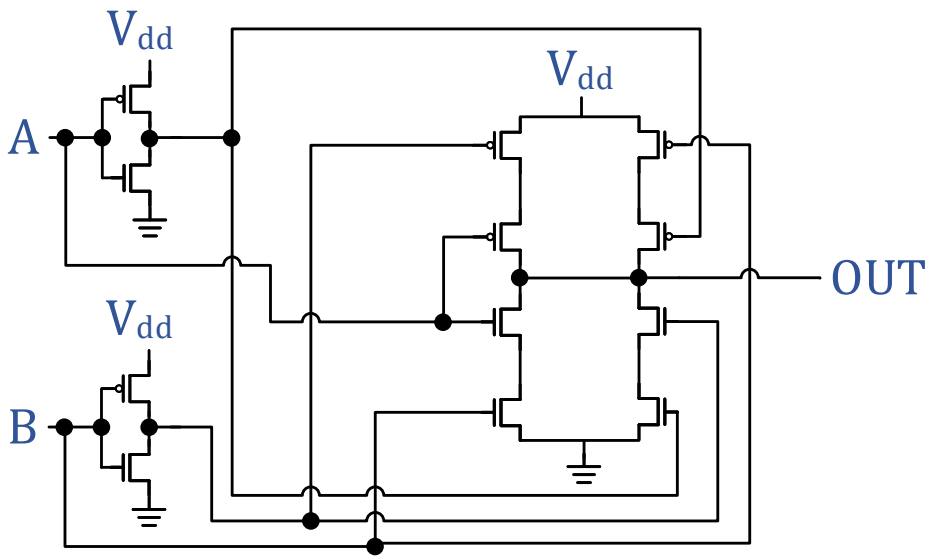
This is possible. If application B has a large working set, it could access a large fraction of the memory rows (within the 64 ms refresh window) and hence these rows do not have to be refreshed explicitly. On the other hand, application A could have a much smaller working set and hence a large fraction of rows still have to be refreshed at the 64 ms rate.

3 Transistor-Level Circuit Design

In Lecture 4, we learned how to implement digital circuits using the CMOS technology (i.e., p-type and n-type MOS transistors). In this assignment, we ask you to schematically design circuits using CMOS transistors for the following logic gates:

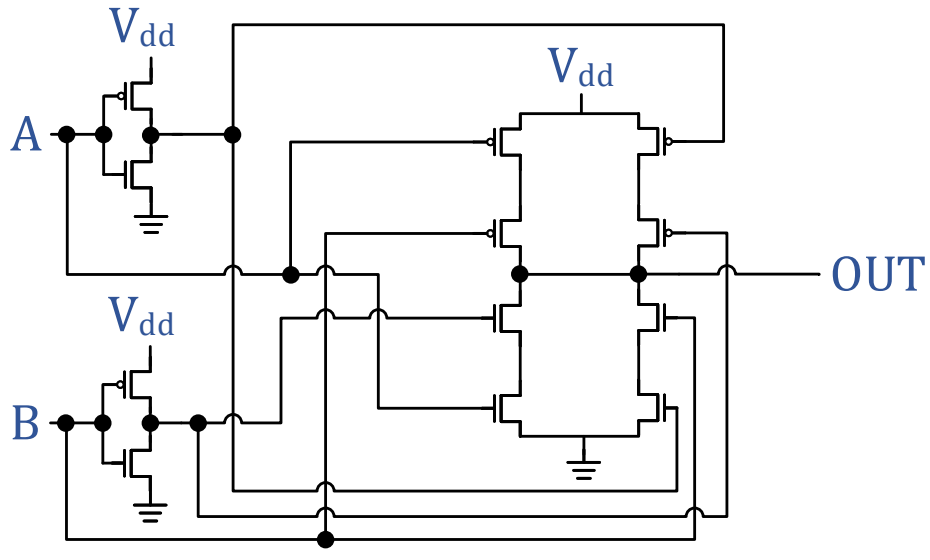
- Exclusive OR Gate (XOR)

Solution.



- Exclusive NOT OR Gate (XNOR)

Solution.

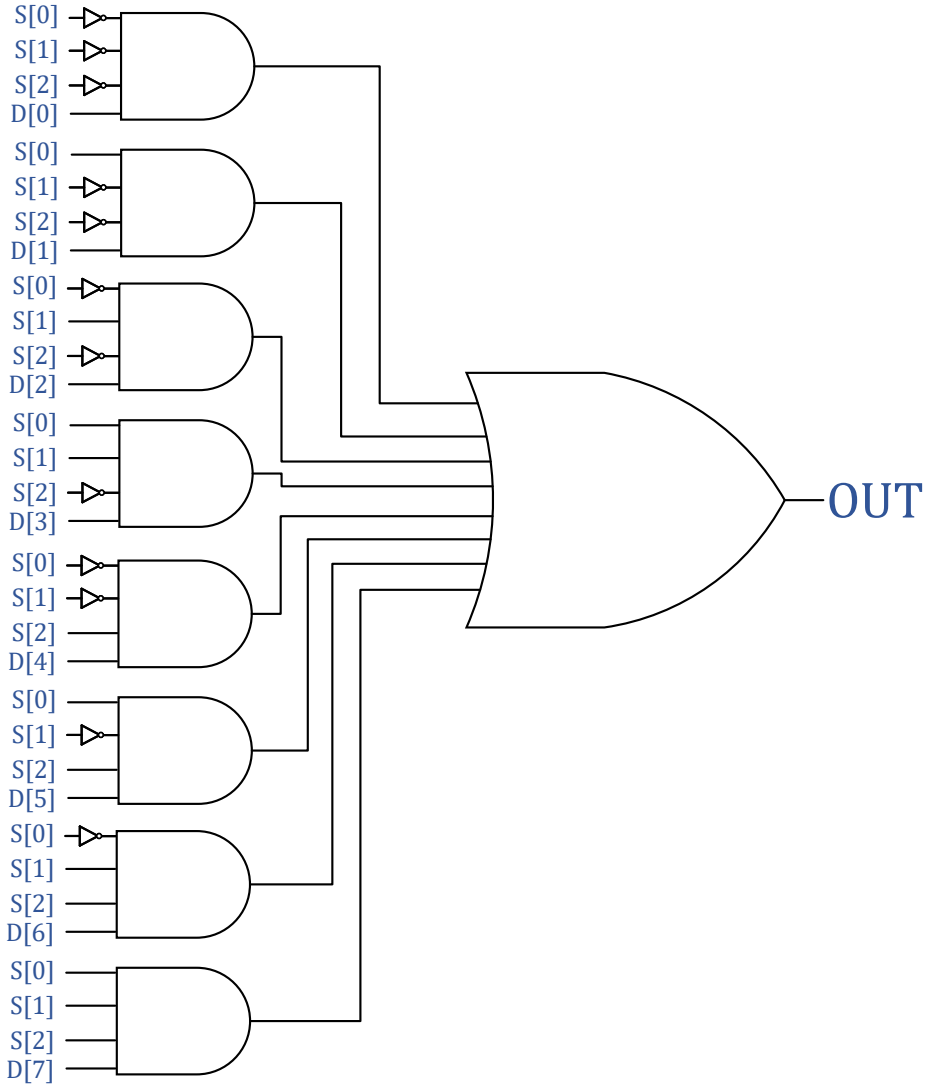


4 Multiplexer (MUX)

Draw the following schematics for an 8-input (8:1) MUX.

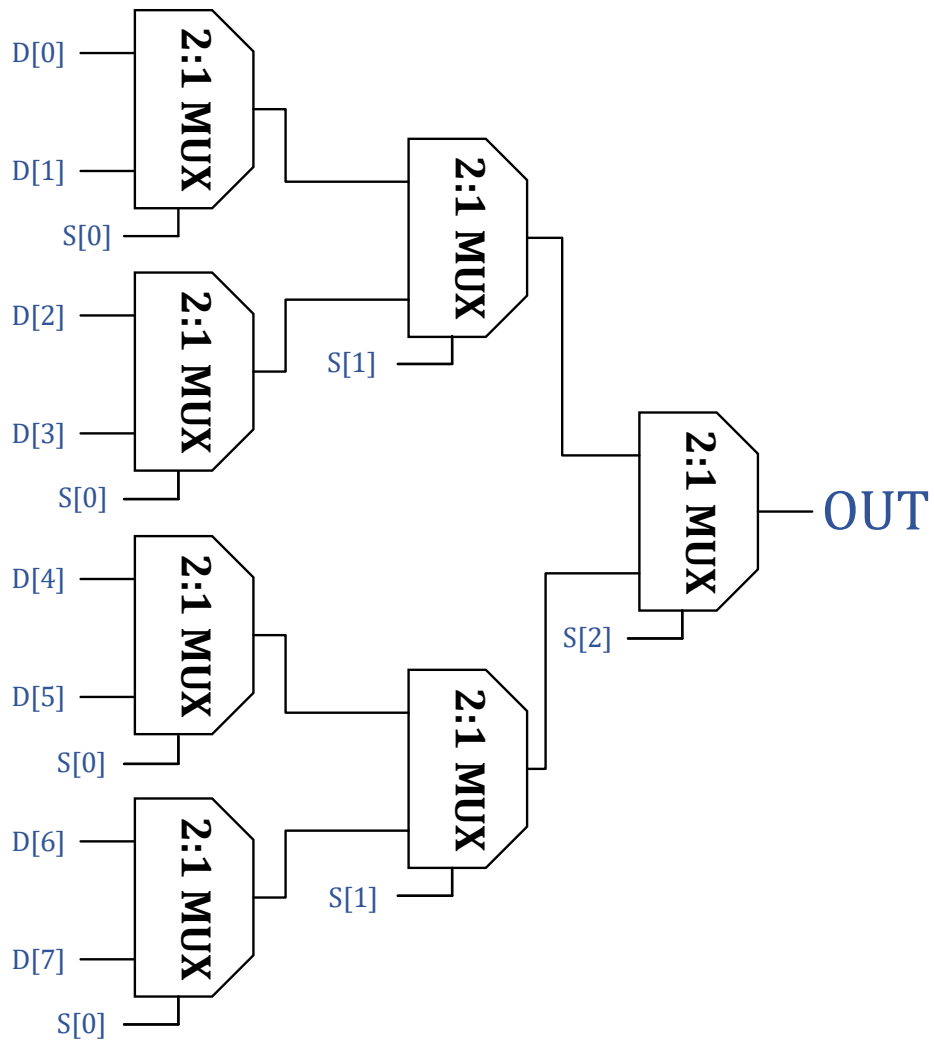
- Gate level: as a combination of basic AND, OR, NOT gates. Use as few gates as possible.

Solution.



- Module level: as a combination of 2-input (2:1) MUXes. Use as few 2-input MUXes as possible.

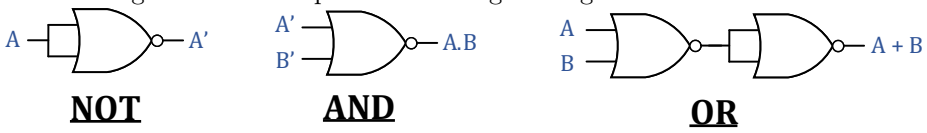
Solution.



5 Logical Completeness

The set of {AND, OR, and NOT} gates is logically complete. We can build a circuit to carry out the specification of any truth table we wish, without using any other kind of gate. From Lecture 4, you know that the NOR gate by itself is also logically complete. Prove that you can build a circuit to carry out the specification of any truth table, by using only NOR gates.

Solution.
To provide that the NOR gate is logically complete, it would be sufficient to show that it is possible to build the logically complete set of {AND, OR, and NOT} gates using only NOR gates. The figures below show how each of these gates can be implemented using NOR gates.



NOT **AND** **OR**

6 Boolean Logic and Truth Tables

In this question we ask you to derive the boolean equations for two 4-input logic functions, X and Y. Please use the truth table below to answer the following three questions.

Inputs				**Outputs**	
A_3	A_2	A_1	A_0	X	Y
0	0	0	0	1	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	1	0
0	1	0	0	1	0
0	1	0	1	1	1
0	1	1	0	1	0
0	1	1	1	0	0
1	0	0	0	1	0
1	0	0	1	1	0
1	0	1	0	1	1
1	0	1	1	1	0
1	1	0	0	1	0
1	1	0	1	1	0
1	1	1	0	0	0
1	1	1	1	0	0

(a) The output X is *one* when the input does **not** contain 3 consecutive 1's in the word A_3, A_2, A_1, A_0. The output X is *zero*, otherwise. **Fill in the truth table above** and use the *product of sums* form to **write the corresponding boolean equation** for X. (*No simplification needed.*)

$X = (A_3 + \overline{A_2} + \overline{A_1} + \overline{A_0}) \cdot (\overline{A_3} + \overline{A_2} + \overline{A_1} + A_0) \cdot (\overline{A_3} + \overline{A_2} + \overline{A_1} + \overline{A_0})$

(b) The output Y is *one* when no two adjacent bits in the word A_3, A_2, A_1, A_0 are the same (e.g., if A_2 is 0 then A_3 and A_1 cannot be 0). The output Y is *zero*, otherwise (e.g., 0000). **Fill in the truth table above** and use the *sum of products* form to **write the corresponding boolean equation** for Y. (*No simplification needed.*)

$Y = \overline{A_3}A_2\overline{A_1}A_0 + A_3\overline{A_2}A_1\overline{A_0}$

(c) Please represent the circuit of Y using *only* 2-input XOR and AND gates.