Front Page

UNIVERSITY OF OSLO
The Faculty of Mathematics and Natural Sciences
Written examination IN3170/4170
Spring 2024
Duaration: 12-Jun-2022 09:00 to 13:00
Permitted aids:
Everything on paper and approved calculator
It is important that you read this front page before you start.

General information:

- All non-electronic sources of information are allowed, as well as an approved calculator. If you reproduce a text from books, articles, etc., a reference to these sources must be provided to avoid suspicions of plagiarism. This also applies if a text is translated from other languages.
- A PDF file of the lecture's compendium, as well as a PDF with all lecture slides of Spring 2024 are available as a resource-link at the bottom of the page.

Information about hand drawings (Scantron):
The answer for question 4 is a hand drawing. You use the handed out sketch sheets. It is possible to use several sheets per assignment. See instructions on how to fill in the sketch sheets in the link below the assignment overview.
You are not supposed to submit hand drawings for any other questions than question 4. *You will NOT be given extra time to fill in the information boxes on the sketch sheets (assignment codes, candidate number, etc.)*

Lecture slides available as PDF:
See the link under the task bar! There you will find the lecture script (issue 0.15 beta) as a PDF.

File Uploads:
Make sure to have all files uploaded **BEFORE** the exam time runs out! This concerns all the MATLAB functions and scripts in the tasks. It's useless if you create the files, but they remain on the computer and are not uploaded! So check in time!

Information on MATLAB functions and scripts:
Some tasks ask for MATLAB functions or MATLAB scripts. These are MATLAB executables stored in files with the ending '.m'. The file name is specified in the task. Do not name your file anything else! You will need to upload these files as your result. They need to be correctly written and executable in MATLAB! MATLAB is installed on your machine, so you can verify your code. **Note the directory MATLAB is working in**, indicated at the top underneath the tool bar: that is where any new files will be stored and where MATLAB will be looking for the m-files to execute them. You may change that directory to a convenient location before you start working. Use the 'New->function/script' button in the tool bar to make new m-files.

MATLAB functions have to start with a function header of syntax '[outputArg1 outputArg2]=function FunctionName(inputArg1, inputArg2)', where the number of output and input arguments can vary. The function header is specified in the task. Do not use any other function header! The function needs to contain a command that assignes a final expression to all the output arguments before it ends. For this exam functions will usually contain just a series of assignments where a mathematical expression is assigned to a variable. Functions shall be used in tasks where one needs to find a general expression assigning a value to variables in dependency of other variables.

Example in a file test1.m:

```
function [y]=test1(x)
y=2*x
end
```

MATLAB scripts do not have a function header and are just a series of MATLAB commands that are executed when the name of the file is typed into a MATLAB window. For this exam also scripts will usually contain just a series of assignments where a mathematical expression or a value is assigned to a variable. Scripts shall be used in tasks with numerical endresults.

Example in a file test2.m where the final result shall be in 'y':

```
x=3
y=2*x
```

The syntax of some potentially useful functions: log(x) is $\ln(x)$, log10(x) is $\log_{10}(x)$, exp(x) is e^x, ... The imaginary number i is 'j' in MATLAB. A line starting with '%' is a comment.

Questions Overview

Note that most questions give up to 1 point while two questions can give up to 2 points as indicated in the overview below. It is a very good idea to read through this overview once more at the end and to make sure that no question has simply been forgotten.

Part I: Models

question 1: Strong or Weak Inversion, Conduction or Saturation?
(multiple choice, 1 point)

Part II: Linear Circuit Analysis

question 2: Transfer Function
(MATLAB-function, file upload, 1 point)

Part III: Digital Circuits

question 3: Gate Delay
(multiple choice/ordering, 1 point)
question 4: FSM PID-Controller
(hand drawing, 2 points)
question 5: Von Neumann Bottleneck
(free text, 1 point)

Part IV: Analog Circuits

question 6: Find Lambda*L from CS
(MATLAB-script for numerical result, file upload, 1 point)
question 7: SF Transfer Function
(MATLAB-script for numerical result, file upload, 2 points)
question 8: **Current Mirror I/O Voltages**
(MATLAB-function, file upload, 1 point)

1 Strong or Weak Inversion, Conduction or Saturation? (1 point)

A nFET at room temperature (i.e. $U_T = 26\text{mV}$) has the following process- and design parameters:

$\mu C_{ox} = 200\frac{\mu\text{A}}{\text{V}^2}$
$\frac{W}{L} = 3$
$n = 1.2$
$V_t = 0.4\text{V}$
$V_S = V_B = 0\text{V}$ (i.e. the source is at ground as well as the bulk)

If you assume that it is operated in saturation and that $\lambda = 0$, for which of these drain currents I_D will it be in weak inversion?

Select one or more alternatives

- 200pA
- 2nA
- 20nA
- 200nA
- $2\mu\text{A}$
- $20\mu\text{A}$

If you assume that it is operated with a gate voltage of $V_G = 0.7\text{V}$, (and $\lambda > 0$) for which of these drain currents I_D can the transistor be in the conduction region of operation?

Select one or more alternatives:

- 100pA
- 1nA
- 10nA
- 100nA
- $1\mu\text{A}$
- $10\mu\text{A}$

Maximum marks: 1

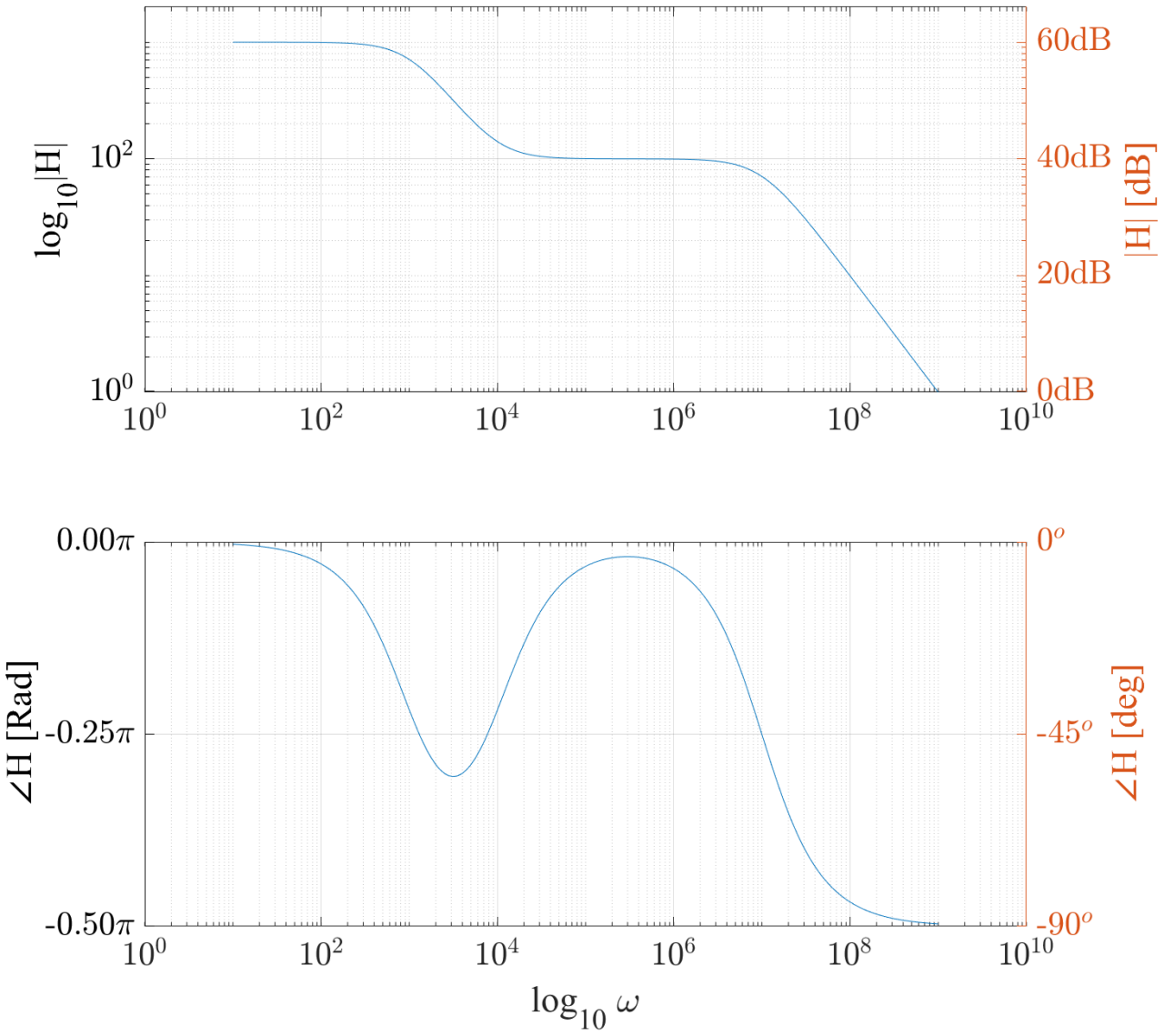
2 Transfer Function (1 point)

What are the transfer functions corresponding to these two Bode plots below? Implement them in a MATLAB function as functions of $s = j\omega$ with header

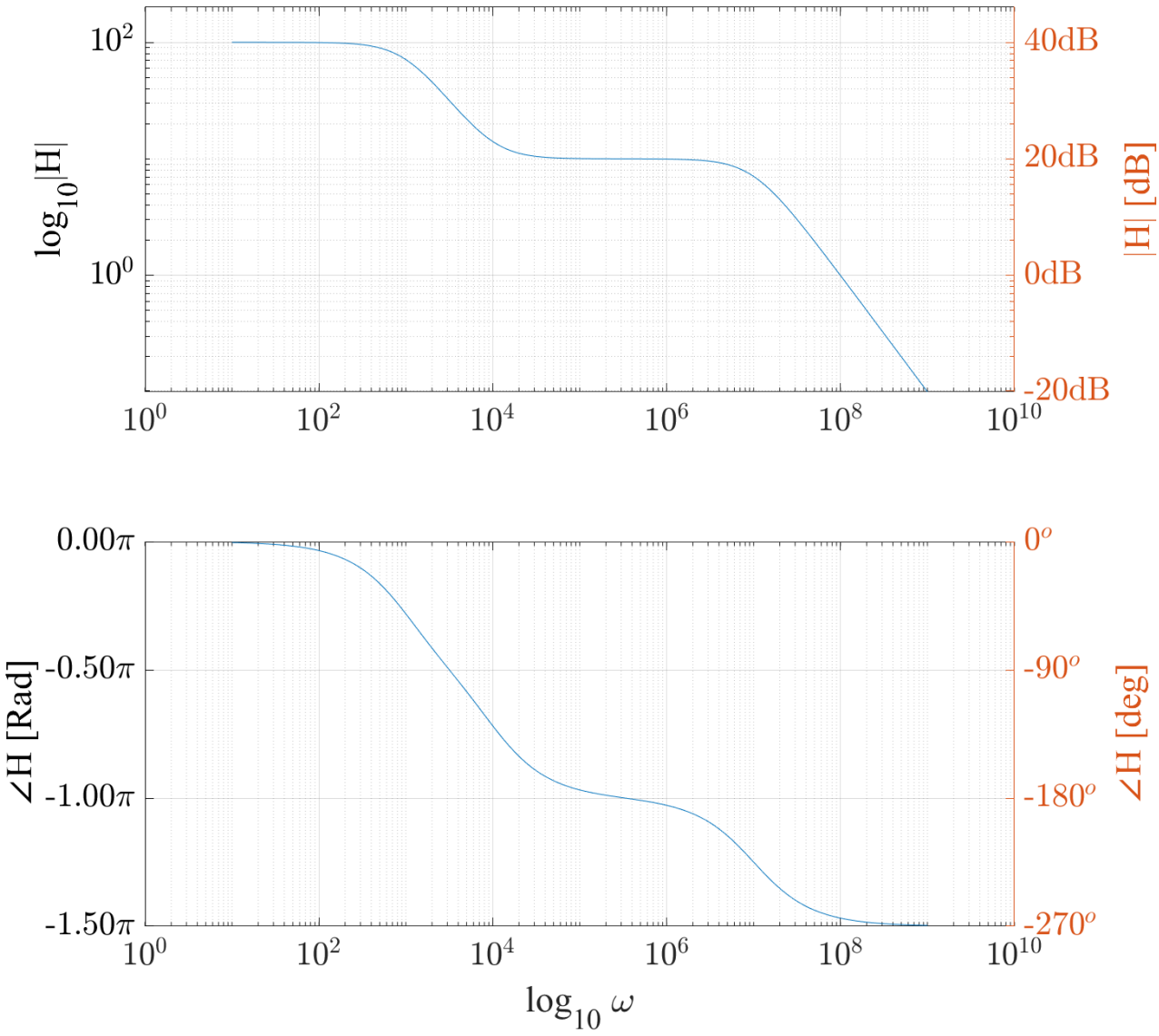
[H1, H2]=TransferFunctions(s) in file TransferFunctions.m and upload the file.

You may safely assume that any poles or zeros occure only exactly at frequencies that are multiples of 10, and that the DC gain is a multiple of 10 as well.

here is the Bode plot for H1:



and here is the Bode plot for H2:



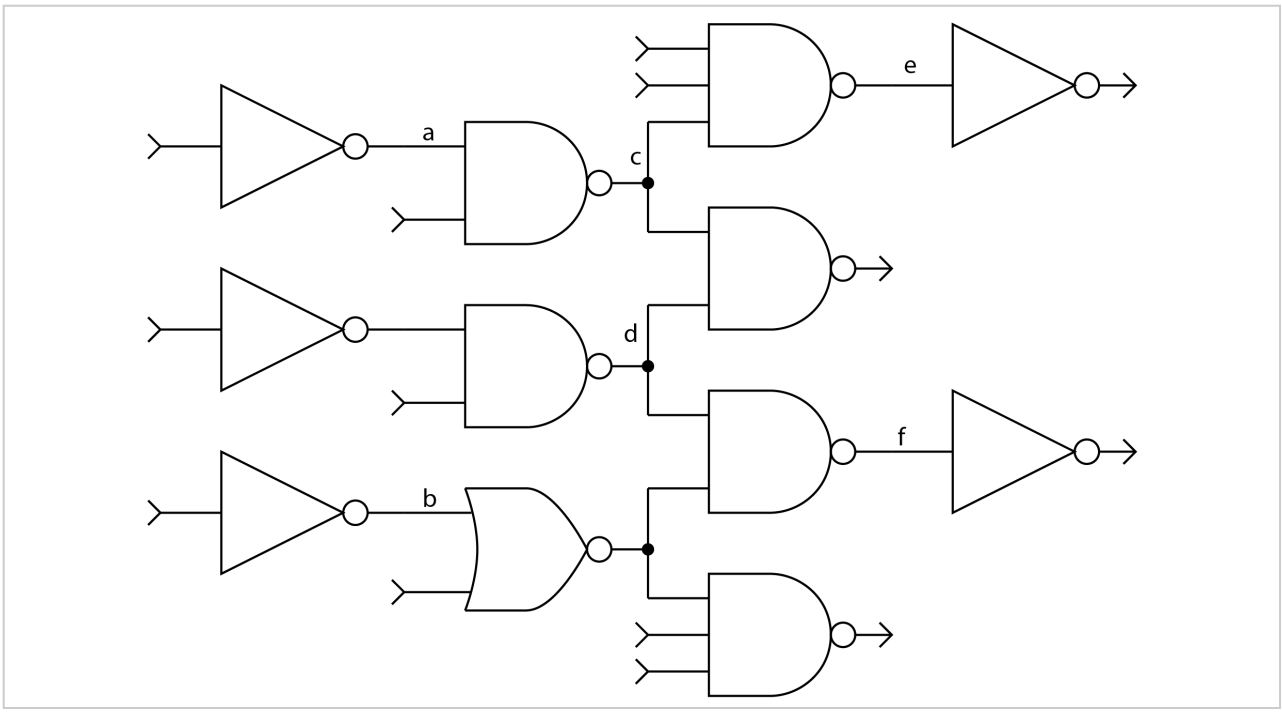
Upload your file here. Maximum one file.

All file types are allowed. Maximum file size is **2 GB**

Select file to upload

Maximum marks: 1

3 Gate Delay (1 point)



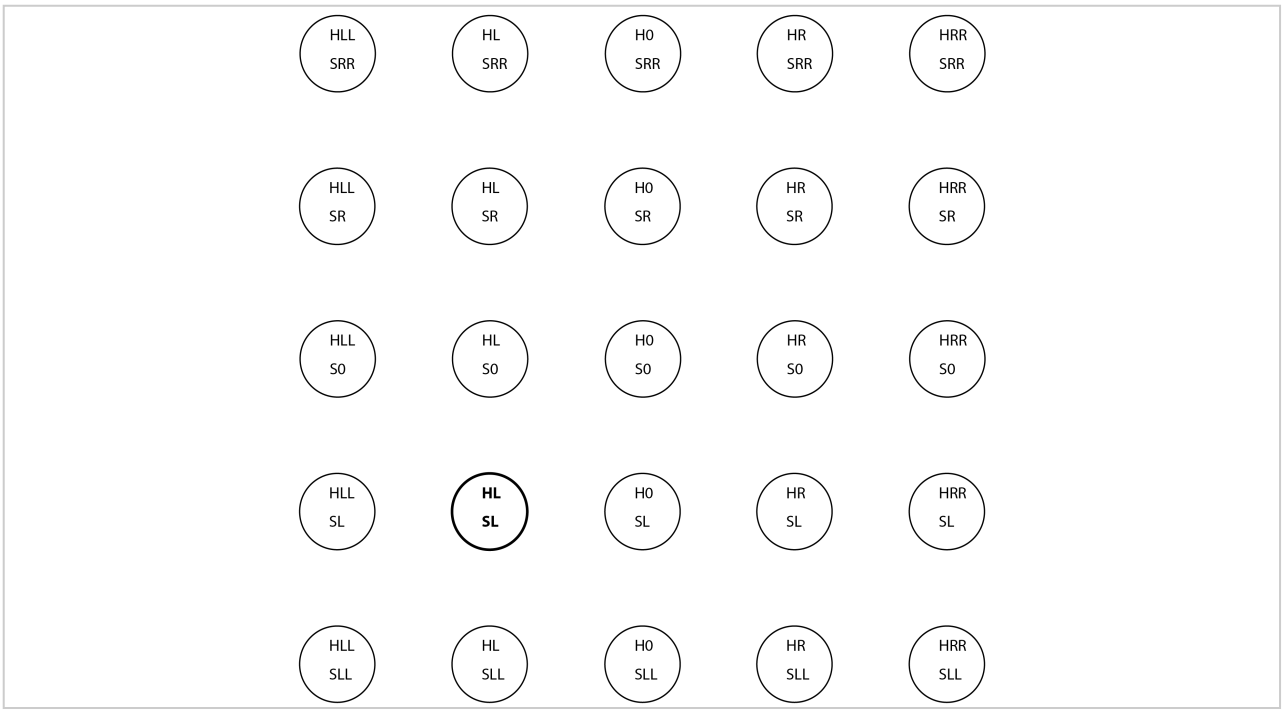
The gates in the illustration are designed to have all the same balanced equivalent on-resistance R_{ON} as an inverter. Balanced means that R_{ON} is the same both for pulling the output up and down. All transistors used in the gates have the same length L, but they may have different widths W to make R_{ON} the same as for an inverter and to make R_{ON} balanced. The mobility of nFETs in this process is 2.5 times higher than for pFETs, so for an inverter the pFET's width W is 2.5 times wider than the nFET's. Let us assume that the total capacitance in the nodes is only composed of the input capacitance C_I of the logic gates and that that capacitance is entirely proportional to the FET gate area it is connected to. Please order the nodes a) to f) marked in the illustration according to their worst case signal delay, i.e. the worst accumulated propagation delay for the signal path from any of the inputs. Assign exactly one position in the ranking to all inputs. Should there be an equal delay between mutiple nodes, their order relative to each other does not matter.

Assign exactly one ranking to one node. (CAREFUL: You are not allowed to assign two rankings to the same node!)

	a	b	c	d	e	f
fastest/shortest delay	○	○	○	○	○	○
2nd	○	○	○	○	○	○
3rd	○	○	○	○	○	○
4th	○	○	○	○	○	○
5th	○	○	○	○	○	○
slowest/longest delay	○	○	○	○	○	○

Maximum marks: 0.96

4 FSM PID-Controller (2 points)



Design a small part of a finite state machine (FSM) for an auto pilot for a ship that has to hold a heading, for example due north. In the following you will only have to supply the state transitions in the state transition graph for the one highlighted state bubble in the collection of state bubbles in the illustration.

Sensor input: The system has a sensor that indicates deviations from the intended course/heading as binary signals that we'll call $H0$,
HL, HLL, HR, and HRR (the H stands for 'heading'). **Exactly one of these binary sensor signals will be 1 at a time.** All others will be zero.
If the course is correct with less than $\pm 5^o$ deviation, then $H0 = 1$.
$HL = 1$ indicates that the course deviates by more than 5^o but less than 10^o to the left and $HR = 1$ indicates a deviation of more than 5^o but less than 10^o to the right.
$HLL = 1$ indicates that the course deviates more than 10^o to the left and $HRR = 1$ indicates a deviation of more than 10^o to the right.

Control output: Likewise we have five control signals that turn the ship's rudder that we'll call $S0$, SL, SLL, SR, and SRR (the S stands for 'steer'). **Exactly one of them at a time is equal to 1**! So it's OK to write in your state 'bubbles' $SL = 1$ or just SL (see illustration!) as the output signal produced by that state, for example, and all other outputs are implicitly set to zero.
$S0 = 1$ means that the rudder is set to steer straight ahead.
$SL = 1$ indicates that the rudder is turned to correct the course towards the left, halfway to the most extreme setting, and $HR = 1$ correspondingly to the right.
$SLL = 1$ indicates that the rudder is turned to correct the course to the left in its most extreme setting, i.e. the ship turnes more rapidly than when $SL = 1$, and $HRR = 1$ correspondingly to the right.

Now current and wind and imprecision in the rudder design may make it so that just leaving the rudder in its straight position will not do the job to keep the ship on a straight path. It may be pushed slowly or rapidly of course and course corrections may become necessary.

Autopilot algorithm: Course corrections (i.e. the next rudder setting) will depend on the rudder setting right now and the sensor input as follows:

a) If the heading (according to the precision of the sensor) has changed since the last clock cycle, the rudder is turned more towards the opposite direction of that change, and it is turned one 'notch' per 'notch' of that change (limited by the maximal positions of the rudder). So for example if the heading in the previous clock cycle has been HL and the sensor now indicates HR, then the course has changed towards the right by two 'notches'. So if the rudder has been at $S0$ it should now be turned to SLL. **These changes of the rudder position are accumulated together with what rule b) indicates.**

b) **In addition**, if the heading HAS BEEN off on the left, no matter whether HL or HLL, add **one** 'notch' stearing to the right (limited by the maximum rudder setting), and if the heading HAS BEEN off to the right, add one 'notch' stearing to the left (limited by the maximum). So for example if the heading has been HR and the rudder has been at SR it should be turned another 'notch' to stearing left, i.e. to $S0$. **Note that rule a) could add to that.**

The FSM that controls the rudder needs 25 states. Simply having 5 states for the 5 possible outputs won't do: the states also need to keep track of the last sensor reading to be able to follow the algorithm. The 25 states can be identified by the output to the rudder and the last sensor reading, as depicted in the illustration.

Finally, here is your task:
Draw all possible state transition arrows from **the one** state (SL, HL) that implement the algorithm as described. You only have to draw that state and its outgoing arrows, as well as all its possible 'descendants'. Mark the arrows with the condition of the input signal for the transition.
Hint: ask yourself how many arrows there need to be. Also ask yourself in which subgroup of states each of those arrows can end up in.

Maximum marks: 2

5 Von Neumann Bottleneck (1 point)

Explain the 'von Neumann bottleneck'. Use no more than 5 sentences.

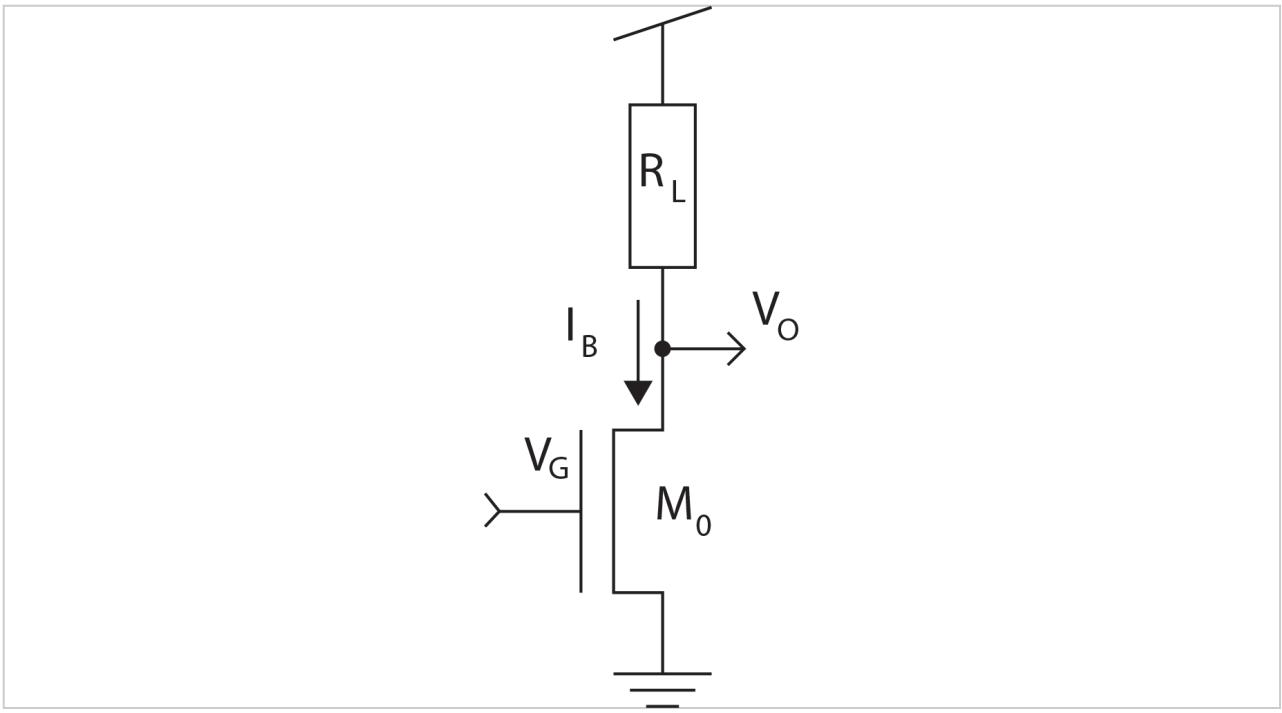
Fill in your answer here

Format

Words: 0

Maximum marks: 1

6 Find Lambda*L from CS (1 point)



Given a CS stage (see illustration) where $\mu C_{ox} = 300\frac{\mu A}{V^2}$, $V_t = 600\text{mV}$, $n = 1$
$W = 1200\text{nm}$, $L = 600\text{nm}$,
$V_S = 0\text{V}$, $V_G = 450\text{mV}$, $R_L = 210\text{M}\Omega$, $\text{Vdd} = 1.8\text{V}$, $A = 20\frac{\text{V}}{\text{V}}$, derive g_m, r_{ds}, and the process parameter $\lambda * L$.

Assume thet the transistor operates in saturation.

Submit you solution in a matlab script that assigns a numeric (!) value to variables 'gm', rds' and 'lambdaL' when it is executed, so with final statements of the form:

gm= ...
rds=...
lambdaL=...

as well as (optionally) statements leading up to that final result.

Hint: to derive parameter g_m and r_{ds} in dependency of I_D it is appropriate to use the value for I_D without channel length modulation effect.

Save it in file 'parameterLambdaL.m' and upload that file!

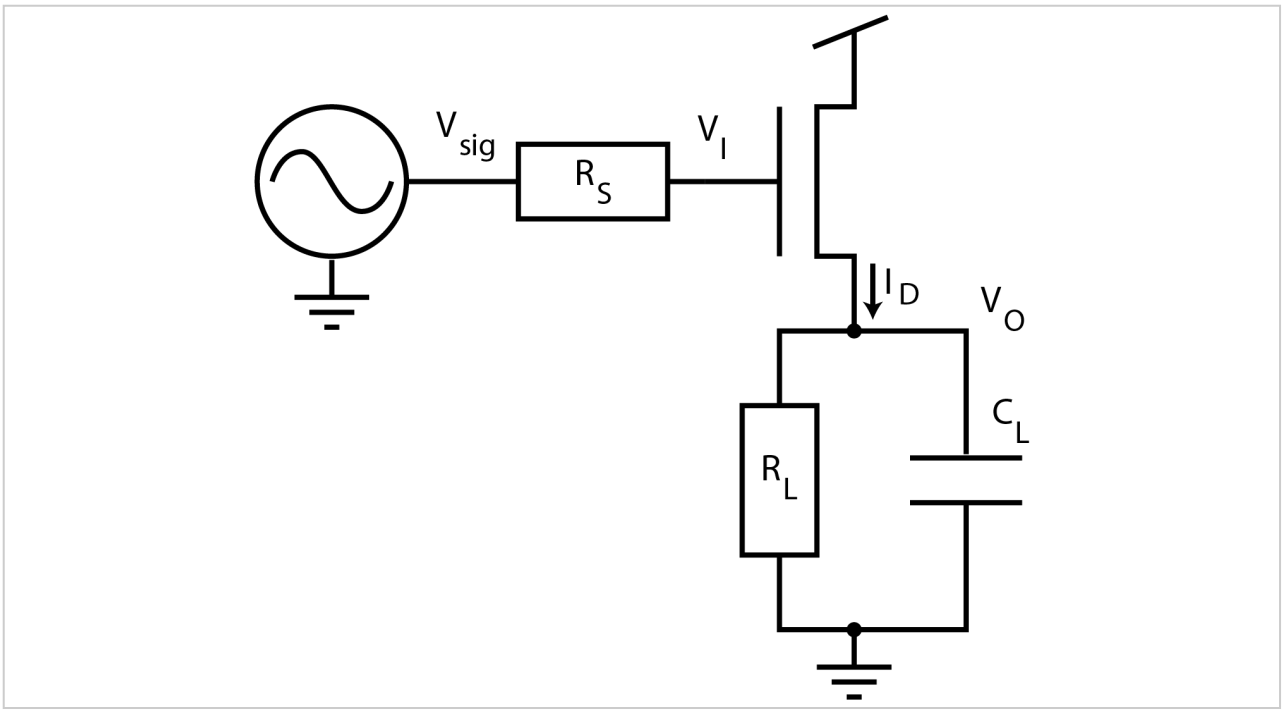
Upload your file here. Maximum one file.

All file types are allowed. Maximum file size is **2 GB**

Select file to upload

Maximum marks: 1

7 SF Transfer Function (2 points)



Find the correct load resistance R_L, the total transconductance G_M, the output resistance R_O looking into the transistor source (so not the total resistance R_L^* seen at the output node) and the two poles p_1, p_2 and the zero $z_1$of a source follower (SF), aka common drain stage (see illustration!), given the following parameters:

$\mu C_{OX} = 320 \frac{\mu A}{V^2}$
$\frac{W}{L} = 1$
$n = 1$
$\lambda = 0.05$
$V_t = 1.2\text{V}$
Vdd=1.8V
$V_I = 1.2\text{V}$
$U_T = 26\text{mV}$
$I_D = 50\text{nA}$
$C_{GS} = 10\text{fF}$
$C_{GD} = 1\text{fF}$
$C_L = 100\text{fF}$
$R_S = 100\text{k}\Omega$
C_{BS}, C_{BD} are negligibly small.

You may assume that the transistor is in weak inversion and that the poles will be real. The solution for quadratic equations may come in handy, i.e. $x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$ are the two solutions for $0 = ax^2 + bx + c$. Use the exact solution for a general transconductance amplifier from chapter 5 (!) applied to a source follower. Also check the new table 9.2 in the compendium 0.15 beta of the appropriate capacitors to be used for a CD/SF.

Submit the solution as a MATLAB script in file 'SF.m' which needs to assigne solutions to variables RL, GM, RO, p1, p2, and z1 (for respectively R_L, G_M, R_O, p_1, p_2, and z_1)

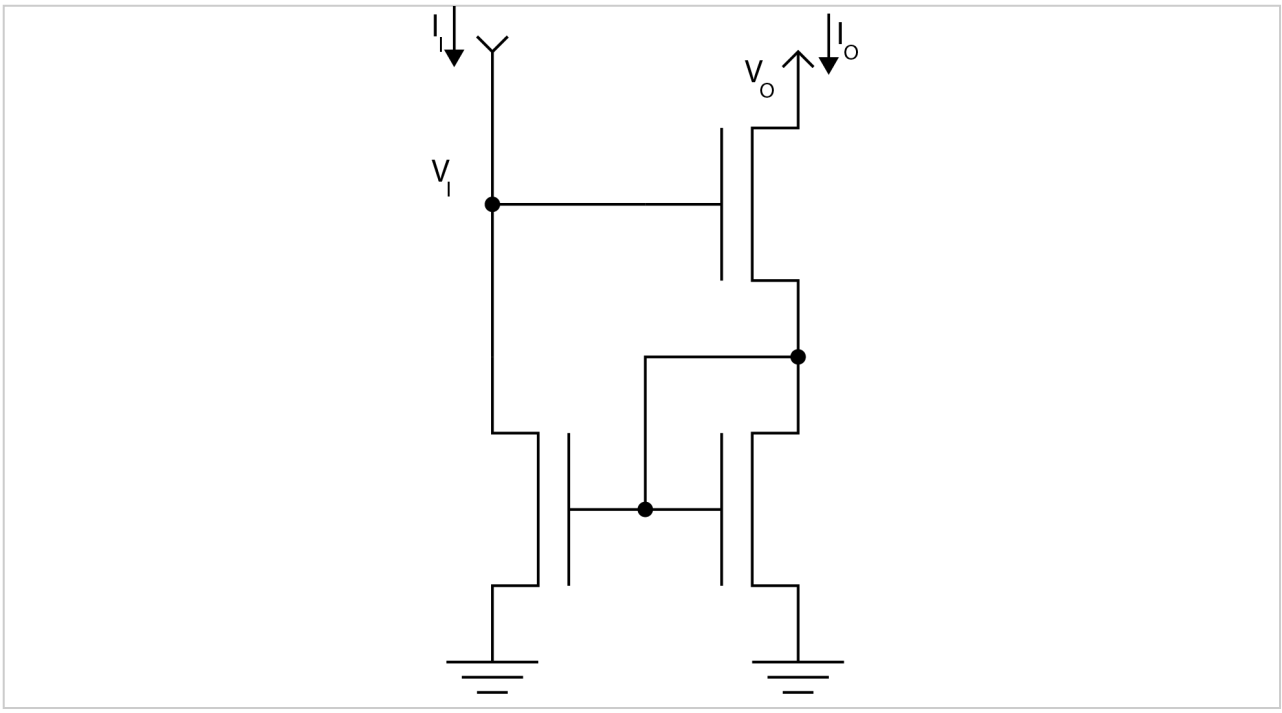
Upload your file here. Maximum one file.

All file types are allowed. Maximum file size is **2 GB**

Select file to upload

Maximum marks: 2

8 Wilson Current Mirror I/O Voltages (1 point)



The illustration shows a so called Willson current mirror. it works just like a simple two transistor current mirror, but achieves a much better output resistance, making it a better current source. So just like for a simple current mirror, if all transistors are in saturation and the channnel shortening effect is insignificant, then the output current I_O matches the input current I_I exactly.

Make a matlab function that describes the condition for the voltages at the input V_I and output V_O such that the output current I_O is the same as the input current I_I (i.e. $I_I = I_O$) while **all three transistors are in saturation**. Please neglect channel length modulation and back gate effects (i.e.assume that $n = 1$ and $\lambda = 0$).

a) V_I and V_O may either need to be an exact value or they may need to be bigger than a minimum value. To indicate whether the following results for these two voltages indicate an exact or minimal value, assign a '1' or a '0' to variables 'VOexactNOTmin' and 'VIexactNOTmin', where '1' means that the value that your function returns is an exact value and '0' means that it is a minimal value.

b) Derive expressions for the two voltages for the **general case**, given a saturation voltage V_{sat} (variable name 'Vsat' in Matlab) that is the minimum voltage accross the drain and source terminals for a transistor to be in saturation, and V_{GS} (variable name 'Vgs' in Matlab) which is the gate to source voltage required for one transistor in saturation to conduct $I_I = I_O$. Add thus two expressions 'VO=...' and 'VI=...' into the function where you may only use 'Vgs' and 'Vsat' on the righthand side and possibly natural constants (as constant numbers or variables that are assigned a constant number).

Use this function header in matlab file '**WillsonVoltages.m**' that you then upload:

```
function [VO, VI, VOexactNOTmin, VIexactNOTmin]=WillsonVoltages(Vgs,Vsat)
```

Upload your file here. Maximum one file.

All file types are allowed. Maximum file size is **2 GB**

Select file to upload

Maximum marks: 1