SAMTEL
Engineering The Future

DESIGN CHALLENGE

SAMTEL is a fast-growing technology firm located in Cambridge, MA. When you choose to work with SAMTEL, you get access to cutting-edge technology and world-class engineers. SAMTEL's business model is unique: we only employ students at MIT who are currently enrolled in 6.012. This "special-sauce" is what gives SAMTEL its qualitative edge: the students are smart, driven, ingenious, problem-solvers, and economical (in fact, most employees at SAMTEL forgo all salary, and even pay to have the opportunity to work and learn at SAMTEL). When you bring a challenge to SAMTEL, we promise results in a timely manner. Give us two weeks, and we guarantee a solution to your hardest problem.

Designed by Max Shulaker

Client: redacted
Start date: Friday, April 21, 2023
Delivery date: Friday, May 12, 2023
Delivery location: Supervisor

Introduction:

SAMTEL employee:

The client is a major company in the electronic industry, with significant market share in high performance servers and data centers. Our client fabricates the chips that go inside of these massive data centers. Due to their ingenious and highly optimized transistor design, their chips are currently the fastest chips on the market (at the expense of also having the most power-intensive chip on the market). While high-performance data center computer chips continue to yield return to the client, the client is aiming to position themselves to enter the low-power application space. In fact, a previous consulting agreement between SAMTEL and this client identified the "Internet-of-Things" (IoT) as a major market opportunity. This recommendation was validated by subsequent studies from a myriad of sources:

- Bain predicts that by 2025, annual revenues for IoT vendors selling comprehensive solutions of hardware and software could exceed $750 billion.

The global Internet of Things (IoT) market is projected to grow from $662.21 billion in 2023 to $3,352.97 billion by 2030, at a CAGR of 26.1

- General Electric predicts investments in IoT will exceed $60 trillion during the next 10 years.
- HIS forecasts that the IoT market will grow from ≥40 billion devices in 2022 to ≥100 billion devices in 2030.

Our client's vision for the IoT is miniature embedded sensing nodes distributed across the globe, constantly capturing, monitoring, computing on, and transmitting data. These sensing nodes will be used for a wide range of applications: from monitoring weather patterns and environmental concerns (such as pollution), to predicting health epidemics and natural disasters. These sensing nodes will also be ubiquitous: from inside jet engines and skyscrapers, to implanted within the human body.

Client Proposal:

While the wide range of applications for the IoT represent a large potential market to our client, it simultaneously represents a major challenge. Our client feels it is infeasible for them to make specialized hardware and products for every possible application. As a result, our client has decided to position themselves as the main supplier for the electronics underpinning IoT applications. By suppling general-purpose hardware which supports Edge Computing and Machine Learning for IoT applications, our client will be able to focus on only perfecting a single product line, while still being applicable and makesales to the many different application spaces provided by the IoT.

To become the leader in IoT hardware, our client wishes to develop the most energy-efficient general-purpose microprocessor. Regardless of the application, nearly every IoT system will still require a microprocessor, and the key metric for these IoT systems is their energy-efficiency. Therefore, if our client can bring the most energy-efficient general-purpose

microprocessor to the market, we predict they will reap significant benefits.

There are several avenues the client was interested in pursuing to achieve this goal. They were considering (1) building better transistors, (2) improving the microprocessor architectural design, and (3) improving the software. While improved software would yield benefits, every application requires its own software optimization, and therefore a single solution is not viable. Moreover, our research at SAMTEL indicates most companies already have engineers who produce near-optimal software, and therefore further software optimization will not substantially improve system performance. Similarly, modifying the microprocessor architecture is not an ideal solution, as any specialization in the architecture will make the microprocessor less general. Therefore, SAMTEL has advised the client to try to improve the underlying transistors, as we predict the transistors can still be improved a significant amount, and improving transistors does not impact the generality of the microprocessor.

Project Goal:

SAMTEL has selected an ARM microprocessor for this project (www.arm.com/products/processors). The ARM microprocessor is an industry-standard, and is already used in many IoT applications. The ARM microprocessor architecture is also already optimized for low power and energy-efficient operation. However, our client currently fabricates transistors for high performance and high speed computers, where efficiency is not as great a concern. ***Therefore, your task is to modify our client's current transistor design, in order to realize the most energy-efficient ARM microprocessor possible.***

Project Details:

-Start with the current transistor made by the client.

-You must modify their starting transistor, in order to realize an ARM microprocessor that consumes the lowest energy while operating at a minimum of 200 MHz.

-To optimize your transistor, you will perform the following two steps:

1) In Sentaurus, set and extract the key metrics which dictate the transistor performance (V_{DD} (operating voltage of the circuit), I_{ON} (maximum drive current), I_{OFF} (leakage current when device should be off), and C_{GATE} (gate capacitance).

2) In MATLAB put the above key metrics into the ARM microprocessor simulator (which is provided for you), which will simulate an ARM microprocessor assuming a transistor with your metrics, and will output the speed and energy of the final design.

Transistor changes:

-A minimum of 3, and a maximum of 8 changes can be made to the transistor. Symmetric changes (e.g., changing the doping values of the source and drain) count as a single change.

-Due to the fabrication limitations of the client, no feature can have a dimension smaller than 4 nanometers.

-The V_{DD} has to be between 400 mV and 4 V.

-The height of the metal gate, source, and drain electrodes cannot be modified.

-Due to the fabrication limitations of the client, the total length of entire transistor is fixed at 175 nanometers.

-Due to the fabrication limitations of the client, doping values cannot exceed $1E20/cm^3$

Deliverables, Grading, and Timeline:

Final Report (6-10 pages):
On the first page, summarize the findings:
1) have a drawing of your final optimized transistor design
2) list your 3-8 changes made
3) provide the energy and speed for your ARM microprocessor, using your optimized transistor.

On the rest of the pages, detail how you approached the project, why you made the changes you made, whether the changes resulted in expected outputs (for instance, did some variables have a smaller or larger effect than you expected?), any calculations you performed (for calculating capacitances or making approximations, *etc.*), and any surprising and interesting results you discovered along the way. Total number of pages (including the first page) must be between 6-10 pages.

4) code (email all code to your supervisor, and rename .cmd files as .txt files)

Project Implementation:

Sentaurus Details:

1. Starting Sentaurus:

- Log into Athena
- type in: setup sentaurus
- change directories to your directory where you want to do the project:
    - make SAMTEL directory: mkdir SAMTEL
    - change to that directory: cd SAMTEL/
    - open sde: sde &

2. Customize transistor: (if you want to run the standard initial transistor, skip this step):

Below is the starting file, which fully describes everything about the standard initial transistor. This can be copy-and-paste directly in the sde, and will generate the transistor. You just have to save the file, and tell it to build the mesh (which is already defined). You should make changes to the following code to change the design of your transistor. The following code will also be posted online for you to access.

```
***assign geometries***
*silicon channel:
(sdegeo:create-rectangle (position 0 0 0.0 )  (position 0.035 0.035 0.0 ) "Silicon" "channel" )
*gate oxide:
(sdegeo:create-rectangle (position 0 0 0 )  (position 0.035 -0.005 0 ) "SiO2" "gate_oxide" )
*Gate metal electrode:
(sdegeo:create-rectangle (position 0 -0.005 0 )  (position 0.035 -0.05 0 ) "Aluminum" "gate_electrode" )
*spacer oxide between metal gate and metal source:
(sdegeo:create-rectangle (position 0 0 0 )  (position -0.035 -0.05 0 ) "SiO2" "spacerL" )
*spacer oxide between metal gate and metal drain:
(sdegeo:create-rectangle (position 0.035 0 0 )  (position 0.07 -0.05 0 ) "SiO2" "spacerR" )
*Drain metal electrode:
(sdegeo:create-rectangle (position 0.07 0 0 )  (position 0.105 -0.05 0.0 ) "Aluminum" "drain_electrode" )
*Source metal electrode:
(sdegeo:create-rectangle (position -0.07 0 0 )  (position -0.035 -0.05 0 ) "Aluminum" "source_electrode" )
*Body metal electrode:
(sdegeo:create-rectangle (position -0.07 0.6 0 )  (position 0.105 0.5 0 ) "Aluminum" "body_electrode" )
*silicon under source metal:
(sdegeo:create-rectangle (position -.07 .035 0 )  (position -0.035 0 0 ) "Silicon" "source_n" )
*silicon between silicon under source metal and silicon channel, called source extension:
(sdegeo:create-rectangle (position -.035 .035 0 )  (position 0 0 0 ) "Silicon" "source_n_ext" )
*silicon between silicon under drain metal and silicon channel, called drain extension:
(sdegeo:create-rectangle (position .035 .035 0 )  (position .07 0 0 ) "Silicon" "drain_n_ext" )
*silicon under drain metal:
(sdegeo:create-rectangle (position .07 .035 0 )  (position .105 0 0 ) "Silicon" "drain_n" )
*body silicon:
(sdegeo:create-rectangle (position -.07 .5 0 )  (position .105 .035 0 ) "Silicon" "body" )



***assign doping***
*channel doping:
```

```
(sdedr:define-constant-profile "constant_channel_doping" "BoronActiveConcentration" 3e18)
(sdedr:define-constant-profile-region "constant_channel_doping_placement" "constant_channel_doping" "channel")
*drain extension doping:
(sdedr:define-constant-profile "constant_drain_ext_doping" "PhosphorusActiveConcentration" 5e+19)
(sdedr:define-constant-profile-region "constant_drain_ext_doping_placement" "constant_drain_ext_doping"
"drain_n_ext")
*drain doping:
(sdedr:define-constant-profile "constant_drain_doping" "PhosphorusActiveConcentration" 5e+19)
(sdedr:define-constant-profile-region "constant_drain_doping_placement" "constant_drain_doping" "drain_n")
*source extension doping:
(sdedr:define-constant-profile "constant_source_ext_doping" "PhosphorusActiveConcentration" 5e+19)
(sdedr:define-constant-profile-region "constant_source_ext_doping_placement" "constant_source_ext_doping"
"source_n_ext")
*source doping:
(sdedr:define-constant-profile "constant_source_doping" "PhosphorusActiveConcentration" 5e+19)
(sdedr:define-constant-profile-region "constant_source_doping_placement" "constant_source_doping" "source_n")
*body doping:
 (sdedr:define-constant-profile "constant_body_doping" "BoronActiveConcentration" 5e+19)
(sdedr:define-constant-profile-region "constant_body_doping_placement" "constant_body_doping" "body")



***define contacts***
(sdegeo:define-contact-set "Gate_contact" 4  (color:rgb 1 0 0 ) "##" )
(sdegeo:define-contact-set "Body_contact" 4  (color:rgb 1 1 0 ) "##" )
(sdegeo:define-contact-set "Source_contact" 4  (color:rgb 1 0 1 ) "##" )
(sdegeo:define-contact-set "Drain_contact" 4  (color:rgb 1 1 1 ) "##" )

***assign location of contacts
(sdegeo:set-current-contact-set "Gate_contact")
(sdegeo:set-contact-boundary-edges (list (car (find-body-id (position 0.02 -0.04 0)))) "Gate_contact")
(sdegeo:set-current-contact-set "Source_contact")
(sdegeo:set-contact-boundary-edges (list (car (find-body-id (position -.06 -0.04 0)))) "Source_contact")
(sdegeo:set-current-contact-set "Drain_contact")
(sdegeo:set-contact-boundary-edges (list (car (find-body-id (position 0.1 -0.04 0)))) "Drain_contact")
(sdegeo:set-current-contact-set "Body_contact")
(sdegeo:set-contact-boundary-edges (list (car (find-body-id (position 0.02 0.55 0)))) "Body_contact")


***make mesh***
*define and set top mesh, everything above the surface of the silicon:
(sdedr:define-refeval-window "RefWin_1" "Rectangle"  (position -0.07 0 0) (position 0.105 -0.05 0))
(sdedr:define-refinement-size "RefinementDefinition_1" 0.002 0.001 0.005 0.002 )
(sdedr:define-refinement-placement "RefinementPlacement_1" "RefinementDefinition_1" "RefWin_1" )
*define and set the bottom mesh, everything below the surface of the silicon:
(sdedr:define-refeval-window "RefWin_2" "Rectangle"  (position -0.07 .6 0) (position 0.105 0 0))
(sdedr:define-multibox-size "MultiboxDefinition_1" 0.05 0.03 .03 .0002 1 1.35 )
(sdedr:define-multibox-placement "MultiboxPlacement_1" "MultiboxDefinition_1" "RefWin_2" )
*define and set the mesh over the channel:
(sdedr:define-refinement-size "RefinementDefinition_1" 0.002 .005 .001 0.002 )
(sdedr:define-refinement-region "RefinementPlacement_3" "RefinementDefinition_1" "channel" )


***save file here***
*MANUALLY SAVE THE FILE WITH A FILENAME HERE BEFORE DOING MESHING!!!
```

***manually build mesh)
*mesh->build mesh

save file

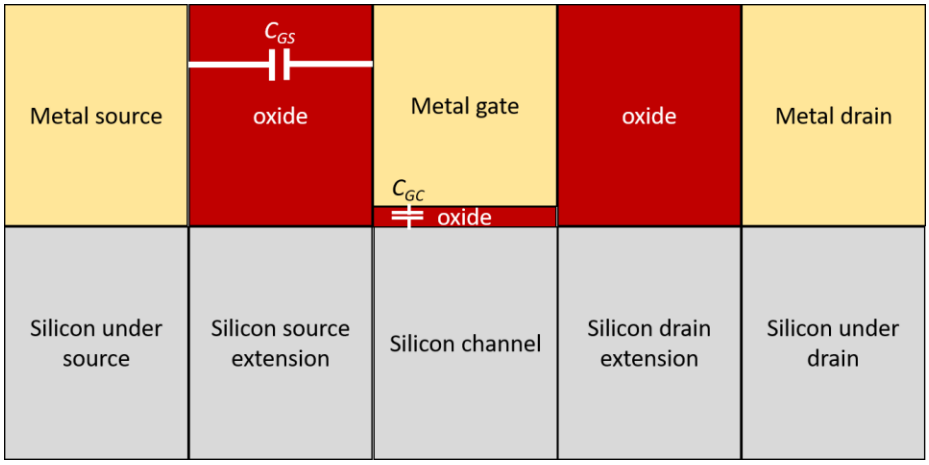
3. Simulate transistor:

- The following file will run the simulation of the transistor. The following code will also be posted online for you to access (as a .txt, rename it yourself as a .cmd).
- Ensure you make the filename match whatever you saved your transistor model under.

```
File {
* The File section defines the input and output files of the simulation
* Input Files
Grid = "FILENAME_msh.tdr" *fix the FILENAME
Current = "FILENAME_def.plt"
Plot = "FILENAME_def.tdr"
Output = "FILENAME_def.log"
}
Electrode {
{ Name="Drain_contact" Voltage=1.8 } * CHANGE THIS TO SET VDD
{ Name="Source_contact" Voltage=0.0 }
{ Name="Gate_contact" Voltage=0.0 }
{ Name="Body_contact" Voltage=0.0 }
}
Physics {
Mobility( DopingDep HighFieldSat Enormal )
EffectiveIntrinsicDensity ( OldSlotBoom )
}
Plot {
eDensity hDensity eCurrent hCurrent
Potential SpaceCharge ElectricField
eMobility hMobility eVelocity hVelocity
Doping DonorConcentration AcceptorConcentration
}
Math {
Extrapolate
RelErrControl *on by default
Iterations=50
Notdamped=100
}
Solve {
Coupled(Iterations=100){ Poisson }
Coupled{ Poisson Electron Hole }
*-Bias Cathode to target bias
Quasistationary(
InitialStep=0.001 Increment=1.1
MinStep=1e-5 MaxStep=0.05
Goal{ Name="Gate_contact" Voltage=1.8 } *CHANGE THIS TO SET VDD
){ Coupled{ Poisson Electron Hole }}
}
```

4. Extract Transistor Values:

- The key metrics of a transistor are the V_{DD}, I_{ON}, I_{OFF}, and C_{GATE}.
    - V_{DD} is a variable you directly control.
    - To extract I_{ON} and I_{OFF}:
        - open svisual
        - plot the gate voltage vs. drain total current
        - extract the I_{ON} and I_{OFF} (you can either export the data and get it from excel, or under "tool" use the "probe" and directly measure them from the plot.
        - For C_{GATE}, the two capacitances that dominate are the two parallel-plate capacitors between the gate and source electrodes (C_{GS}), and between the gate and channel (C_{GC}). Calculate these are parallel plate capacitors, following the diagram below (when calculating C_{GS}, ignore the thickness of the gate oxide). For the ARM simulator, you will need to input the C_{GATE} as a relative percentage of the C_{GATE} of the standard initial transistor. For instance, after optimizing your transistor, if your C_{GATE} is 20% smaller than the C_{GATE} of the standard initial transistor, you would use 0.8 as your new relative C_{GATE}.



ARM Analysis Details:

In the provided MATLAB ARM core simulator, you input the values extracted from Sentaurus. The output is the energy and speed the ARM core would operate at, given your specific transistor specifications. See the online manual for further details.

As your supervisor at SAMTEL, I recommend first being comfortable with this simulator, in order to develop some insight into how changes at the transistor-level impact system-level performance. However, remember that you can put in arbitrary values into the simulator – but you maybe not be able to achieve those values in Sentaurus!

Standard Initial Transistor:

The standard initial transistor has been supplied by our client, and is shown below. It is a 32 nm technology node transistor – still an advanced and relevant technology that is in production today in many chip foundries.