UNIVERSITY OF CALIFORNIA
College of Engineering
Department of Electrical Engineering and Computer Sciences
Last modified on October 19, 2007.

Elad Alon ***FALL 2007 TERM PROJECT PHASE I*** *EECS 141*

1. Design of a 32x32-bit SRAM– Background

Memory arrays are an essential building block of all digital systems. In this semester's project, we will design an SRAM array that contains 32 32-bit words. In order to support operation as a FIFO, the memory is addressed by a 5-bit address (whose decimal value ranges from 0 to 24) that is added to a 3-bit offset (whose decimal value ranges from 0 to 7).

1.1. High level structure

A block diagram of the SRAM you will be designing is shown in Figure 1.

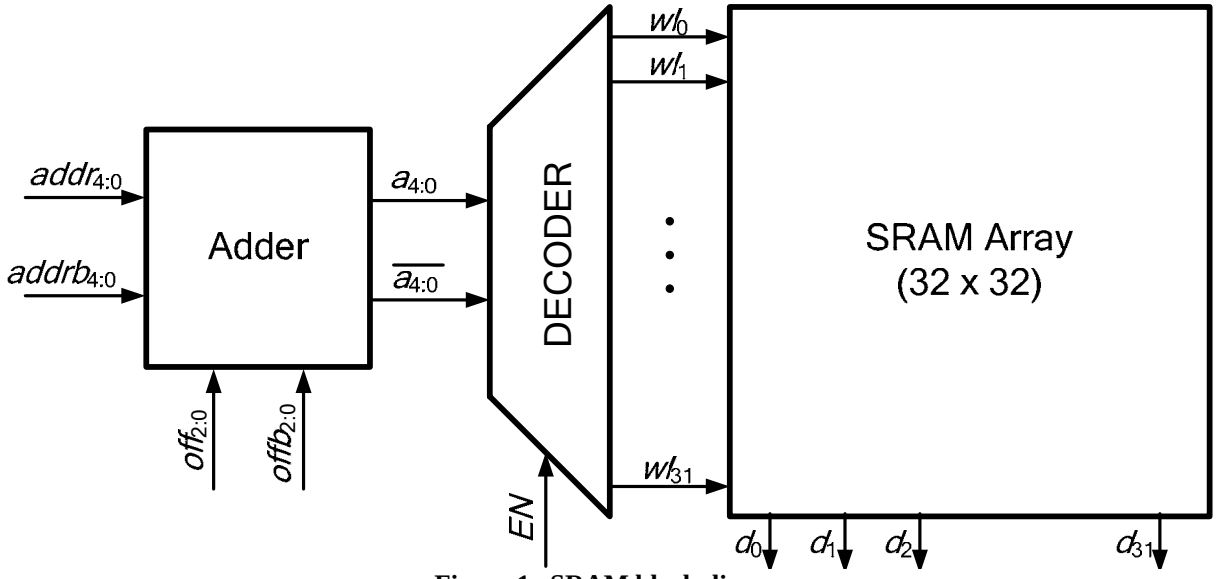

Figure 1. SRAM block diagram.

There are three major blocks to be designed:

- Adder: The adder adds a 3 bit offset address $off_{2:0}$ to a 5 bit relative address $addr_{5:0}$ to calculate the effective address $a_{4:0}$.
- Address decoder: The address decoder takes in the 5 address lines $a_{4:0}$ produced by the adder, and decodes them to generate 32 wordlines $wl_{31:0}$ for the SRAM array.
- SRAM array: Consists of an array of 32 x 32 bit SRAM cells.

In addition to these blocks, the array also contains circuitry that allows data to be written into the array, and for precharging the bitlines to V_{DD} before the read operation; these circuits are not shown in figure.

2. Implementation and Constraints

The goal of this project is to design a functional, compact, fast, energy-efficient SRAM for use in a high-performance or mobile microprocessor. The project will be completed in THREE phases by teams of two students. The first two phases of the project will consist of well-defined tasks (similar to the homeworks), while the final (longest) phase of the project will be much more open-ended.

PHASE 1: Cell Characterization and Decoder Design (due Thursday, Nov 1, at 5pm)

Cell Characterization:

In the first phase of the project, you are provided with a pre-designed SRAM cell. Characterize the cell stability by using Cadence to obtain an extracted netlist and HSPICE to perform simulations to get the read and write margins.

To obtain the SRAM cell:

- In Cadence, create a library "sram" linked to the TSMC 0.24um technology (see lab 2).
- Create a layout cellview "sram_cell". Do not close the cellview.
- Create a schematic cellview "sram_cell". Do not close the cellview.
- Create a symbol cellview "sram_cell". Do not close the cellview.
- Now in an x-terminal, go to the directory ~/ee141/sram/sram_cell/ and type the following commands:

  cp ~ee141/project/sram_cell/schematic/* schematic
  cp ~ee141/project/sram_cell/layout/* layout
  cp ~ee141/project/sram_cell/symbol/* symbol

  Reply "y" to all "overwrite" prompts.
- Go back to Cadence and close all open cellviews. Now reopen the symbol, schematic and layout views of sram_cell. You should see the SRAM cell design.

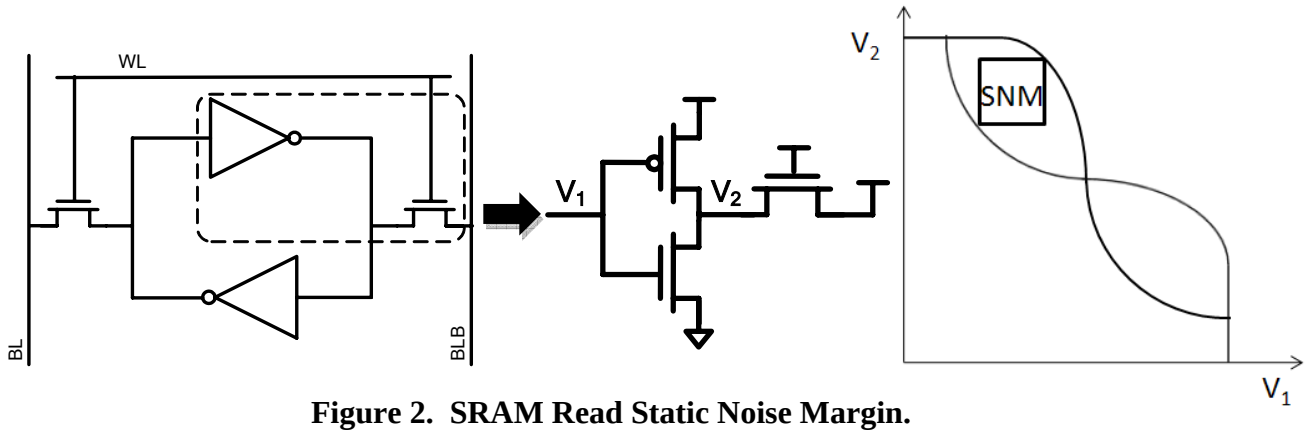

Figure 2. SRAM Read Static Noise Margin.

Recalling that the wordline and bitlines are held at V_{DD} during a read, Figure 2 shows how to extract the read static noise margin (SNM) of the cell. First, the feedback from the cross coupled inverters is broken. Next, the VTC of the "inverter" formed by half of the SRAM cell is found by sweeping V_1 (the inverter's input) from 0 to V_{DD} and measuring

V_2 (the inverter's output). This plot is then used to construct the "butterfly plot" that is representative of the two halves of the cell driving each other. The read SNM is the side length of the maximum possible square that can fit inside of the butterfly plot. You do not have to calculate the size of this maximum square, but you should submit the butterfly plot (generated using HSPICE) that graphically indicates the SNM. You should also measure the worst-case voltage rise in the SRAM cell during a read (i.e., the value of V_2 when V_1 is at V_{DD}) and provide that value in your report.

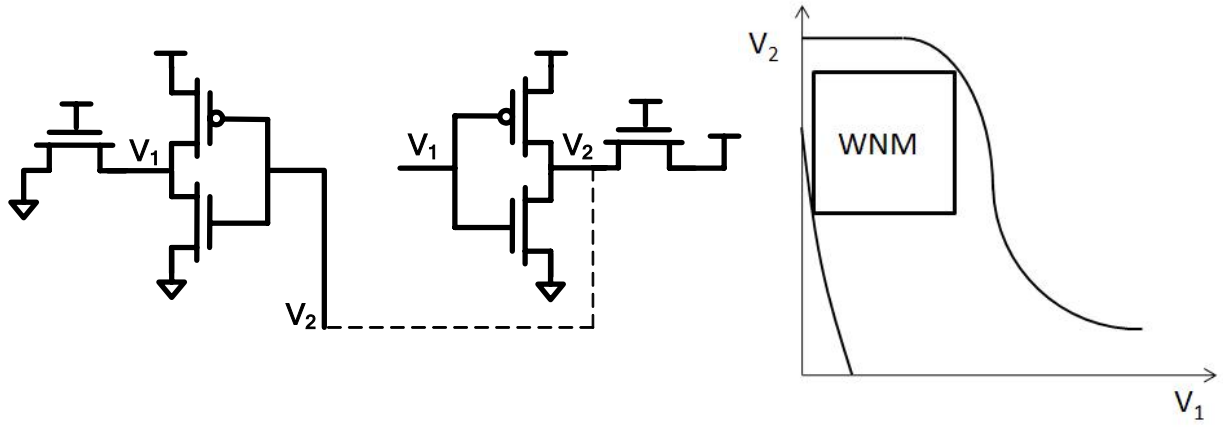

Figure 3. Write Noise Margin.

During a write, VDD is applied to the worldine, and the value to be written into the memory cell is driven onto the bitlines. Thus, Figure 3 shows how to extract the write noise margin (WNM) of the cell. Again, the feedback from the cross coupled inverters is broken, and the VTC of the "inverters" are measured. Note however that in this case, the VTCs of the two halves of the SRAM are no longer the same (since one of the bitlines is driven to 0V, and the other to V_{DD}). These VTCs are used to create a butterfly plot, and the WNM is the side length of the largest square that can fit inside of the butterfly plot. You do not have to calculate the WNM, but you should generate the butterfly plot (again using HSPICE) and graphically indicate the WNM. You should however measure the worst-case cell voltage during a write (which is found from measuring V_1 when V_2 is at 0V).

Layout of SRAM Array:

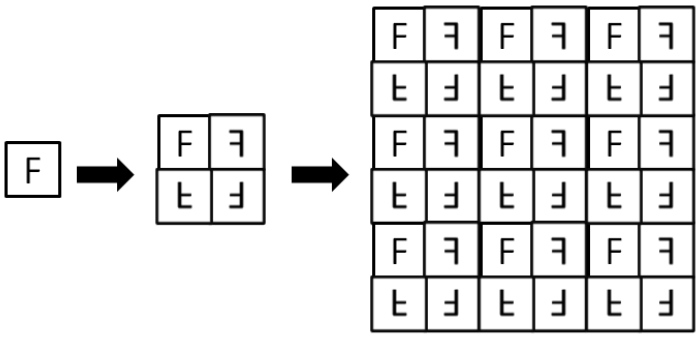

Figure 4. Arraying Procedure

Figure 4 shows how the provided SRAM cell can be arrayed to minimize area. Each adjacent cell is flipped across the X or Y axis. For phase 1 of the project, you will need to array one row of SRAM cells and use the extracted layout to estimate the capacitive loading on each wordline.

Decoder Design:

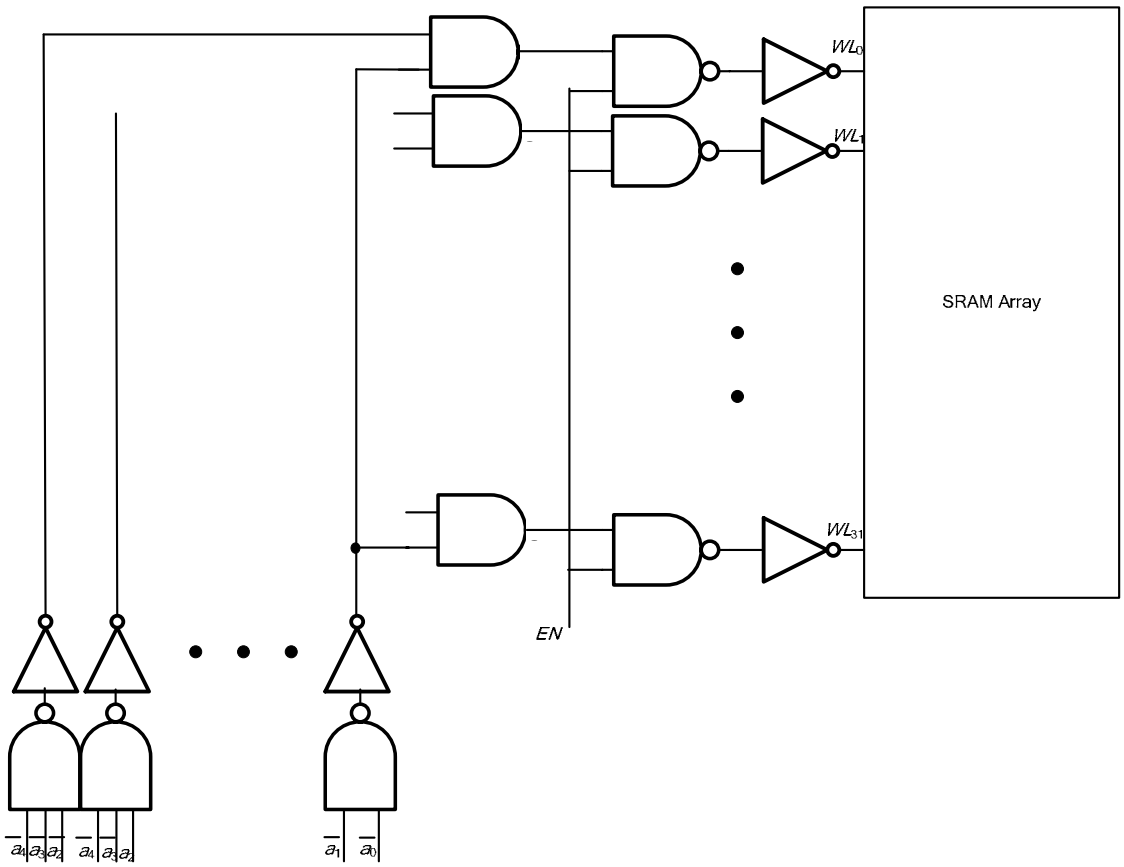

Figure 3. Decoder.

The next stage is to design the 5-to-32 memory decoder. In phase 1 of the project, our goal will be to minimize the delay of the decoder from the address inputs transitioning to the wordline rising. The decoding is performed in two phases: predecoding of 3 or 2 input bits, and then final decoding of 2 bits. The enable signal EN is active high and enables the decoder outputs – this is required in order to be able to precharge the bitlines once a read or write has been completed. For phase 1, you can assume that the EN signal will go high at the same time as the address inputs transition.

The predecoder drives the final wordline decoders together with a wire whose length equals the height of the memory array. The logic structure and the number of logic levels in Figure 2 are not fixed – e.g., you can exchange NANDs with NORs, or add/remove inverters.

You can assume that both the true and complement versions of the addresses are available to you, but the input loading of each of the signals is constrained to be less than 5fF. The output loading of the decoder is determined from the wordline loading of the SRAM cell and by the wire capacitance. You can ignore the wire resistance, but you should estimate the wire capacitance on the wordline by hand (using Tables 4-2 and 4-3 from the book). In order to get a more precise value for the wire capacitance, you will need to layout and extract one row of the SRAM array, as well as the final stages of the decoder (i.e., the per-row AND gates) that generate the wordline for that row. Note that the decoder pitch in layout must match the SRAM cell height, and that for this phase of the project you do not need to layout any of the predecode gates.

In your report, you should include your schematics and layout, your calculation of the worst-case delay from one address input to the wordline rising, and a simulation of this worst-case delay.

Note that before you enter the decoder into Cadence, you should manually design the decoder (on paper) for the minimum delay using the method of logical effort. As part of your report for phase 1, you will need to turn in a one page document that includes your sizing calculations and a diagram of the decoder with all transistor sizes annotated.

PHASE 2: Adder Design (due Tuesday, November 13, at 5pm)

In the second phase, you will design the adder that sums together the 5-bit relative address with the 3-bit offset. More details will be provided in the project phase 2 handout.

PHASE 3: Optimization and Assembly (due Wednesday, November 28, at 10am)

In the last phase, you will assemble the SRAM array along with adder and the decoder, and have the opportunity to choose a set of optimizations to apply to improve the overall performance, area, and/or power of your design. More details will be provided in the project phase 3 handout.

PROJECT POSTERS are due Thursday, November 29, 3:30-5:00pm.

In each phases 1 and 2 of the project you will turn in a short report. A longer report, together with a poster presentation is due on November 28.

Physical and electrical specifications:

2.1. TECHNOLOGY: The design is to be implemented in a 0.25 µm CMOS process with 5 metal layers. You should only use up to 4 metal layers for the SRAM design. The SPICE model is in the g25.mod file we have been using all semester. At a supply voltage of 2.5V, for hand calculations you can assume that C_G = 1.72fF/µm, C_D = 1.58fF/µm, and R_{sqn} = 17.5 kΩ/□.

2.2. POWER SUPPLY: You are free to choose any supply voltage and logic swing up to 2.5V. Make sure that you use the appropriate model when you perform any hand analysis.

2.3. PERFORMANCE METRIC: The propagation delay of your memory is defined as the time interval between the 50% point of the inputs and the 50% point of the worst-case output signal. Make sure you pick the worst-case condition and state EXPLICITLY in your report what that condition is!

2.4. POWER: Since everyone's design will be capable of running at different frequencies, we will measure power consumption with a 50MHz clock. Further instructions on how to run simulations to measure power will be provided in Phase 3.

2.5. AREA: The area is defined as the smallest rectangular box that can be drawn around the design. Since the SRAM must interface with the rest of the chip, all inputs and outputs must be accessible from the boundary of the block.

2.6. NAMING CONVENTIONS: The input operands of the memory are named *addr*<4:0>, and *off<3:0>* – the complements of these inputs (addrb<4:0> and offb<3:0>) are available as well. Wordline and bitline signals are labeled as *wl*<31:0>, *bl*<31:0>, and *blb*<31:0>. The output data is *d*<31:0>.

2.7. REGISTERS: You don't need to use any registers in this design.

2.8. CLOCKS: You can use as many clocks as you would like for this design. You will need at least one clock to enable/disable the decoder ouputs and precharge the bitlines. Remember that the load capacitance of any clock you use should be included in the energy analysis.

2.9. V_{OH}, V_{OL}, NOISE MARGINS: You are free to choose your logic swing in the decoder. The noise margins should be at least 10% of the supply voltage.

2.10. RISE AND FALL TIMES: All input signals have rise and fall times of 200 ps. The rise and fall times of the output signals (10% to 90%) should not exceed 400ps (unless otherwise noted).

2.11. LOAD CAPACITANCE: Each output from your SRAM must drive a 20fF load.

2.12. INPUT CAPACITANCE: The maximum capacitance you can place on each polarity of the address and offset inputs is 5fF. For example, addr<0> can drive a maximum of 5fF, and addrb<0> can also drive up to 5fF.

3. Simulation

You should always begin your designs by hand analyzing the circuits. Having done this, you should also use HSPICE to simulate the design and prove that it functions correctly. Keep in mind that you will need to determine the input pattern that causes the worst-case propagation delay or energy consumption.

4. Report

The quality of your report is as important as the quality of your design. Be sure to provide all relevant information and eliminate unnecessary material. **Organization, conciseness, and completeness are of paramount importance.** Do not repeat information we already know. Use the templates provided on the web page (Word and PDF formats). Make sure to fill in the cover-page and use the correct units. Turn in the reports for each phase in the homework drop box.
In addition, mail an electronic version of your final report and the poster as a Word or PDF file to ee141-project@bwrc.eecs.berkeley.edu. You will also be asked to provide your final netlist.

4.1 Report for Phase 1

The total report should not contain more than five pages. You are not allowed to add any other sheets. The organization of the report should be based on the following outline:

Cover page: Names, summary of simulated SRAM cell parameters, wordline capacitance, and propagation delay.
Page 1: Simulation of the static noise margins.
Page 2: SRAM row layout and schematic.
Page 3: Annotated schematic and the layout of one row of the decoder.
Page 4: Capacitance estimates. Description of the sizing procedure.

Remember, a good report is like a good circuit: it should perform its function (convey information) in the smallest possible area with the least delay and energy (to the reader) possible.

The quality of the report is an important (major) part of the grade!

The total project grade is divided into three parts, the reports from the first two phases, and the final report/poster. The first two parts of the project are worth ~25% of the project grade apiece, while the final report/poster is worth 50%.

For each of the three phases, the grade will be divided as follows:

30% Approach and correctness
30% Results
35% Report
5% Creativity