University of Pennsylvania
Department of Electrical and System Engineering
Circuit-Level Modeling, Design, and Optimization for Digital Systems

ESE370, Fall 2019 — Project 2: Memory Design — Saturday, Nov. 9

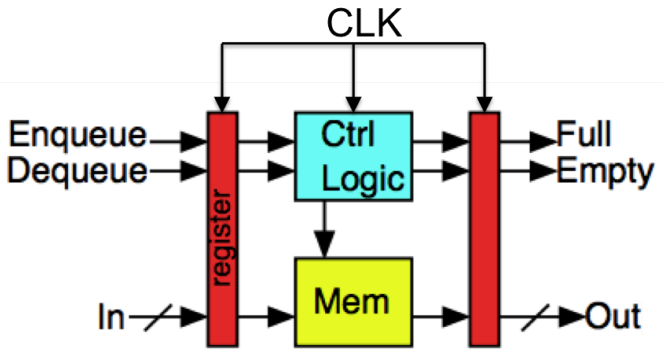
Milestone Due: Wednesday, November 20, 11:59PM
Due: Monday, December 2, 11:59PM

Project Pairs: You can work in groups of no more than 2 for this project. Each team must turn in one report with the clear contributions of each partner clearly delineated. Everyone is responsible for understanding the design and report in its entirety, and the instructor reserves the right to interview the students to verify this.

Extra Credit: The two reports with the lowest and second lowest reported average energy metric (see Design Metrics section), will receive 10 and 5 extra credit points respectively. You must correctly measure, calculate and report the metric to be eligible for the extra credit. Additionally one report will be awarded best writeup and will receive 5 extra credit points.

Design Problem: Design a synchronous FIFO that can both enqueue and dequeue an item on each cycle.

- Target technology is the High Performance 22nm process (`/home1/e/ese370/ptm/22nm_HP.pm`)
- Design should be synchronous, with dequeue operations returning a value (if there is one) on the following cycle.



- We will concretely focus on a FIFO with a maximum capacity of 16, holding 4b-wide data elements.
- Inputs: Clock, In<3:0>, Enqueue, Dequeue
- Outputs: Out<3:0>, Full, Empty
- Full and Empty represent the state of the FIFO at the end of a cycle.
- Dequeue requests are ignored on cycles when Empty is asserted.
- Enqueue requests are ignored on cycles when Full is asserted.

- As a result (a) simultaneous Enqueue and Dequeue on a cycle when Empty is asserted will result in no output and a FIFO with one value (the enqueued one), and (b) simultaneous Enqueue and Dequeue on a cycle when Full is asserted will result in no enqueue and a FIFO with one fewer item than its full capacity.
- Nonetheless, if a value is enqueue into an empty FIFO on one cycle, it should be possible to dequeue it on the following cycle.

Design Metrics:

- *area*: Sum the total transistor width for the entire design.
- *memory cell area*: Sum the total transistor width for the repeated memory cell in the memory core.
- *enqueue energy*: Measure the energy enqueuing 0x1111 into a word that previously held 0x0000 (and vice-versa). Report the larger value.
- *dequeue energy*: Measure the energy for a dequeue of a word with 0x0000 and a word with 0x1111. Report the larger value.
- *enqueue/dequeue energy*: Measure the energy for a cycle that both enqueues and dequeues values. For test case data, use worst-case scenarios as identified for isolated enqueue and dequeue.
- *standby energy*: Measure the energy of a cycle on which no enqueues or dequeues occur.
- *average energy*: Measure the average energy of your queue with the following operation distribution: 15% enqueue&dequeue, 10% enqueue, 10% dequeue, 65% standby operation.

Design: Your primary objective is energy minimization at 500 MHz operation. In other words one cycle with a dequeue and enqueue operation should be less than 2ns. Your secondary objective is energy minimization.

- $V_{dd} \leq 1.0$V.
- A design that only uses complete registers or latches is **not** acceptable.
- Memory cells should be static.
- You select decoder design, driver design, output sensing/buffer, clocking, control timing.
- You may **not** use resistors in the design.
- You get no additional credit for operating your clock faster than 500 MHz.
- You should be trying to minimize all energy components, so the design will be low energy regardless of the mix of operations. Nonetheless, if you need a concrete distribution of operations to help make some decisions, consider: 15% enqueue&dequeue, 10% enqueue, 10% dequeue, 65% no operation.

Recommendations:

- Even before you worry about energy minimization, you need to assure that you can achieve correct operation.
- Pay careful attention to the timing of controls for writes. What do you need to guarantee to make sure you only write into the intended word cells on a write cycle? What test cases will you need to validate these properties?
- **Use hierarchy in your schematics.** You should have a single memory cell design that you instantiate. You should also organize cells into word rows. You may also want a separate organization into columns for reasons noted below. You should also use input/output Vdd and Gnd pins in **every** sub-block so you can isolate energy measurements.
- A common trick for regular design like this is to extract the critical path to avoid having to simulate the entire design during sizing and delay optimization. Can you build a design that has a single row and column rather than all row×column memory cells? The parasitic loading of other cells is important, so you want to capture that when you extract your critical path (hence the need to capture a row and a column; similar reasoning should also apply to address decoders).
- **Use unit testing.** Make sure your memory cell, decoder, buffers work in isolation before you integrate larger structures.
- Think about what sequence of inputs and multiple operations would test a correct read/write.
- To generate a sequence of logic values, you can use a VPWL (Piece-Wise Linear) source in electric. This produces a SPICE PWL. Please see HW7 and the ngspice manual for more information.
- We are not necessarily looking for innovation in the core memory cell circuit, but it will need to be sized for the technology and adapted for your usage.
- Cell sizing may be asymmetric to address the different needs of reads and writes.

Milestone: (Due 11/20) Turnin as a single PDF.

- Estimate bit line capacitance for target design point. Report and use in the simulations below to demonstrate cell operation.
- Include schematics for column driver and **sized** memory cell. You may use PWL controls and ideal voltage references for this phase of the design. PWL controls and ideal voltage references are temporary scaffolding; you will implement the controls and references in transistors for the final design.
- Identify the set of cases you should test in order to validate correct operation of your memory cell on both writes and reads.
- Demonstrate that you can write to the memory cell. Include schematics for setup and simulation waveforms as necessary.
- Demonstrate that you can read from the memory cell without losing the value. What test cases do you need to demonstrate this? Include schematics for setup and simulation waveforms as necessary.

- Identify the constraints on write timing in the full FIFO design (not just the cell) to guarantee correct operation, and identify how you will verify this (what test setup? what test cases?).

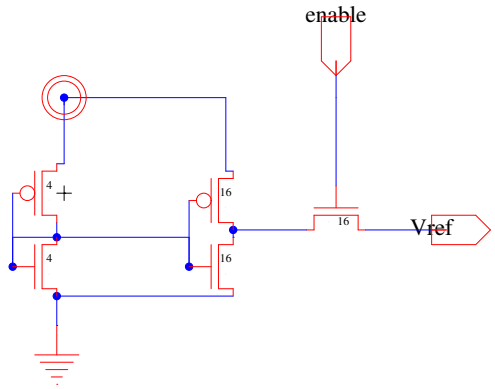
You want to make sure you know how the SRAM core needs to be controlled before building logic to generate the controls, and you want to have an operational memory core.

Report: (Due 12/2) Your report should be a single, stand-alone document (including stand-alone from your milestone report) and should include:

- Schematics for the design – make sure transistor sizing annotations are easily readable in the diagram you include in the report.
- For both the enqueue and the dequeue operations show:
  - Waveform showing timing of key signals in the operation. Identify the delay of each component that makes up the critical path.
- Text description of the memory operation and design choices. Identify and describe the operation of each of the sub-components. Your description should make it easy for us to understand how and why your design works.
  - Explain the timing requirements of the memory for correct operation and to meet the 500 MHz cycle time and how they are met.
  - Explain rationale for sizing of memory cell—why are the transistors sized as you did, and how did you arrive at this conclusion? Show simulations as necessary.
- Breakdown of energy contributions by function: (a) memory (including addressing, bit-line charging, restoration, write drivers), (b) logic (including storing and incrementing head/tail pointers, interpreting enqueue/dequeue requests), and (c) clocking and timing generation (including switching and leakage)
- Description of how you optimized the energy of each sub-component and the ensemble, including the tradeoffs you needed to make. Show experimental simulation results as necessary.
- Description of how you validated correctness of the design. This will include a description of your test cases.
- Summary of the design metrics to two significant figures. Include supporting evidence in the form of equations and/or simulation results. All metrics reported must have supporting simulation results.
- Include a table summarizing all design metrics.
- Please include a statement on your final submission:

> I, *your-name-here*, certify that I have complied with the University of Pennsylvania's Code of Academic Integrity in completing this project.

Voltage Midpoint Reference Generator



Here's a voltage reference generator that may be useful for your memory design. The inverter with input and output tied together settles to the midpoint of the transfer curve where the input and output are equal. The second inverter is driven to the same point. It serves to isolate the output from the reference generated by the first one. The pass gate allows you to connect or disconnect this reference voltage to a node.

You may need to resize the driving buffer and pass transistor. Sizes shown are just to give concrete examples.