# Circuits that Remember: Read/Write Memory

## Laboratory Outline

When building circuits, it is common to arrive at circumstances where you would like a circuit to remember if an event (or trigger) has occurred. Memory circuits allow us to achieve this function by storing the voltage present at the input whenever they are triggered by a controlling signal, they then retain the stored voltage until the next control signal. Two major types of random access memories (RAMs) exist today, dynamic (DRAM) and static (SRAM), where “random access” means we can read or write whenever our control signal is set. This lab examines a basic circuit implementation of a DRAM.

### Learning Objectives

- To gain practical experience in building and using digital circuits.
- To improve oscilloscope and circuit analysis skills.
- To understand the circuit level challenges associated with RAM.

### Prerequisites

- Experience with inverters, capacitors, square-wave oscillators, and MOSFETs.
- Experience from lab with use of the function generator including setting “high-Z” mode.

Notes:

Below is a high-level block diagram of the circuit we will be building; it shows the general function of each the sub-circuits in the complete circuit we will be building. Note that the memory itself is only a small component of this circuit, the majority of what we are building is needed to support the correct functionality of the small, but scalable, capacitive memory. These components will be introduced during this lab.

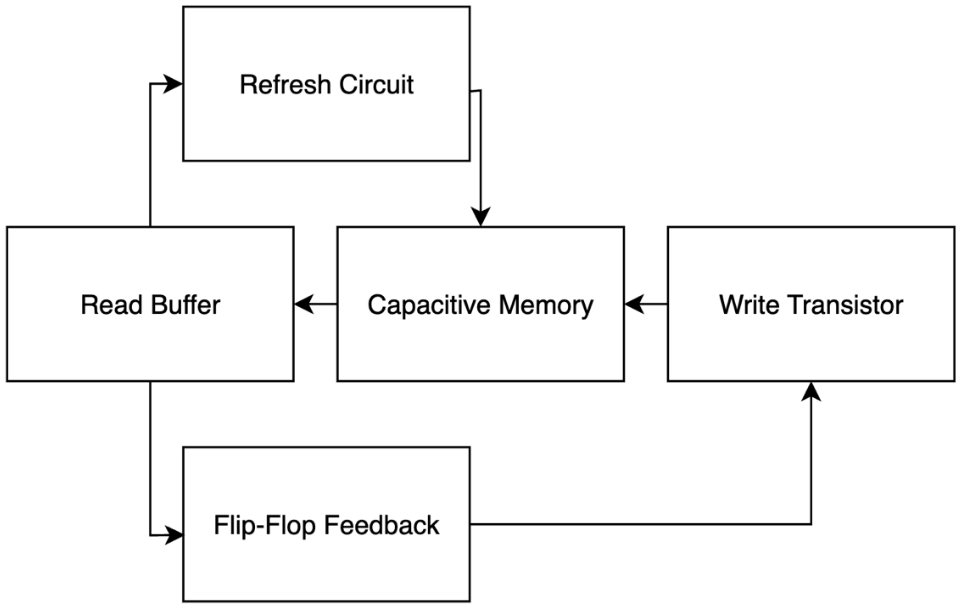


**Figure 0**: Initial Block Diagram

Notes:

## DRAM Introduction

In simplest terms, dynamic RAMs are very straightforward, all we need is a capacitor to store the logical value we want to retain. However, this circuit is not stable, meaning that overtime the charge stored will **always** leak out of any connecting circuity. This property is fundament to capacitors.

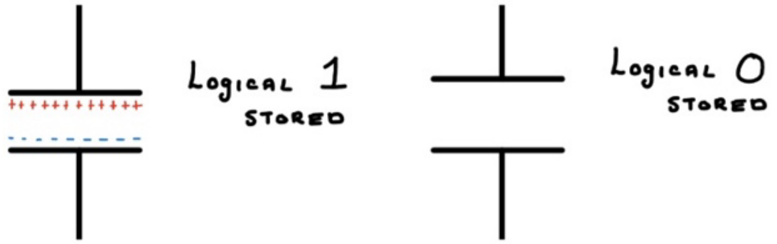


**Figure 1**: How capacitive memory stores information.

As we do not want to use the limited charge on the capacitor to drive any downstream circuit components (thus losing its charge) we can *"buffer"* the capacitor with two inverters. So long as the capacitor maintains enough charge to be higher than threshold voltage of our inverter, it will maintain the correct output.

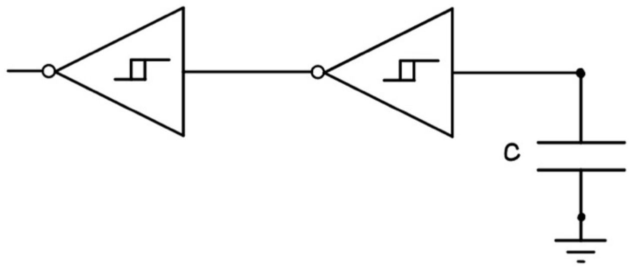


**Figure 2**: Buffering the capacitor in order to drive downstream circuits. This is a "read" buffer.

Notes:

While this address us not driving downstream circuits with our capacitor, the charge will still **leak**. As such we need to periodically restore the charge on the capacitor when it is in the logical “1” state and ensure it does not charge in the logical “0” state.  Let’s first model that leakage with a resistor, note that this resistor is not real, it just allows us to understand the behavior of the capacitor.

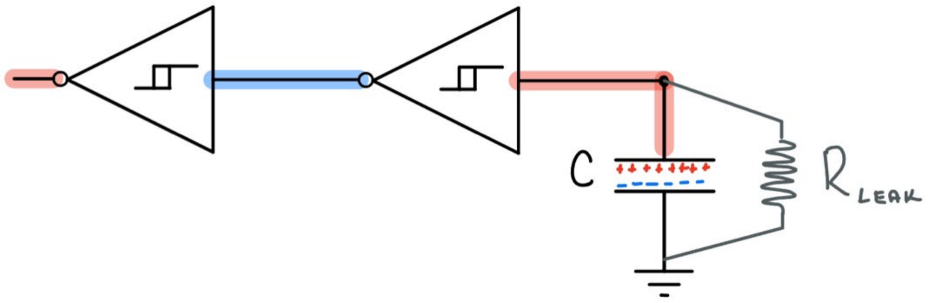


**Figure 3**: Capacitor “leakage” model with buffering inverters.

**Question 1:** How long will this circuit retain the correct logical behavior (meaning the output of the 2nd inverter is high), please leave your answer in. terms $R, C, V_S$ (source voltage), and $V_c$ (voltage across the capacitor). Where the negative threshold ($V_n$) of the inverter is $0.5 \times V_S$. NOTE: $V_c = V_S \times e^{-\frac{t}{RC}}$. You should solve for $t^*$ when $V_c(t^*) = V_n$ where $t^*$ is the discharging time.

Notes:

Now that we know the circuit only maintains the correct output for a finite amount of time, we need to use a restoration circuit that refreshes the charge on the capacitor. To do this we will use an nMOS transistor to connect to a square wave oscillator. A square wave oscillator is needed to ensure correct behavior when we "write" to our capacitor. To prevent the capacitor from charging the square wave when it's low, we will add a diode to stop the current flow. Our circuit now looks like Figure 4.

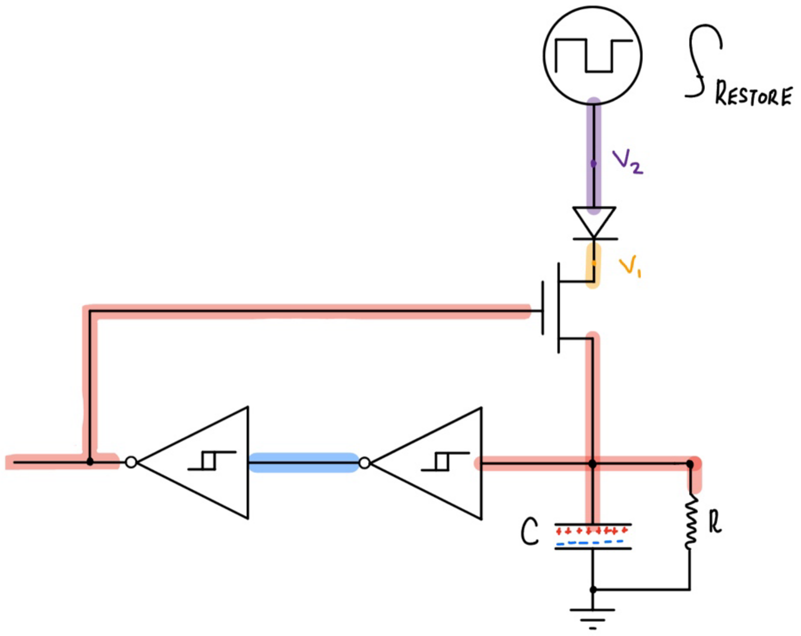
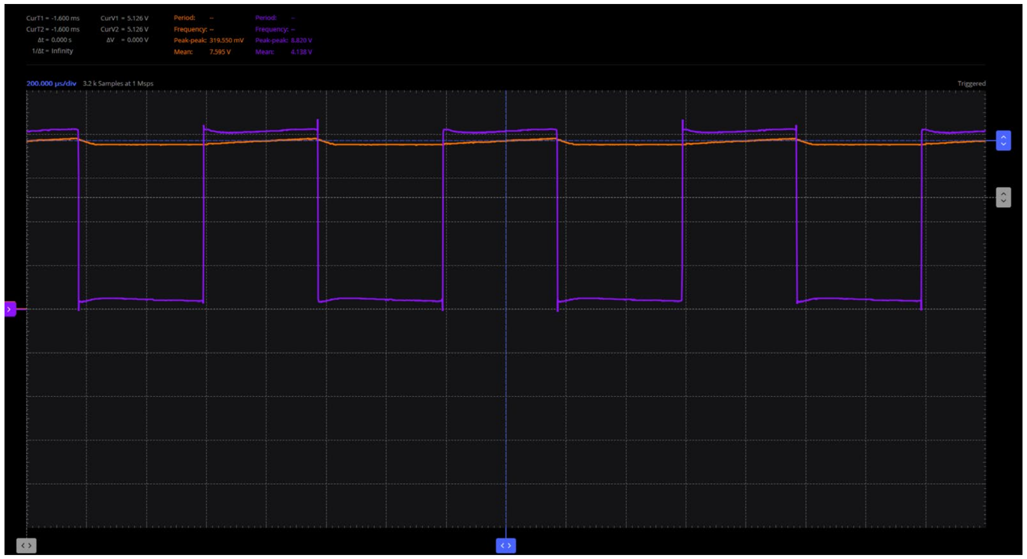


**Figure 4**: Capacitor DRAM with restore circuit and leakage model (refresh circuit)

Let’s pause for a moment to look at circuit on an oscilloscope to see if we can understand it’s behavior:



```
CurT1 = -1.600 ms    CurV1 = 5.126 V    Period:     --           Period:     --
CurT2 = -1.600 ms    CurV2 = 5.126 V    Frequency: --            Frequency: --
   Δt = 0.000 s         ΔV = 0.000 V    Peak-peak: 319.550 mV    Peak-peak: 8.820 V
 1/Δt = Infinity                        Mean:       7.595 V      Mean:       4.138 V

200.000 µs/div   3.2 k Samples at 1 Msps                                  Triggered
```

**Figure 5**: $V_1$ and $V_2$ when measured on an oscilloscope when a logical 1 is stored.

The $V_1$ node voltage is stable at a value slightly less than $V_S$ even as the square wave oscillator goes low. When the $V_2$ node is high it can charge the capacitor, however it must maintain a voltage drop across the diode. The voltage across the capacitor is stable at a value less than $V_S$ by approximately the voltage drop of the diode, although the nMOST also effects this somewhat.

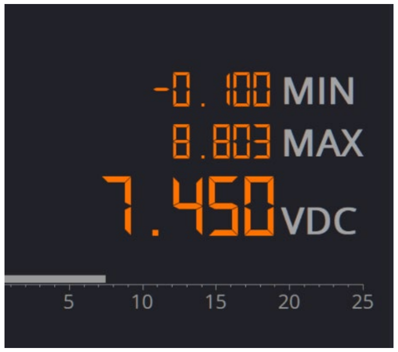
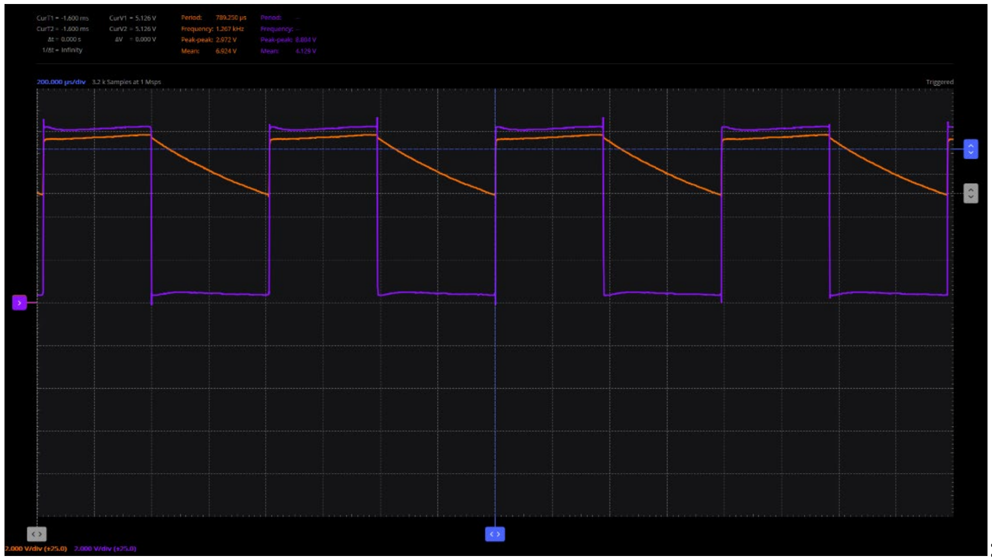


**Figure 6**: $V_C$ when storing a logical 1.

Notes:

Let’s reverse this and see the behavior when a logical 0 is stored:



:

**Figure 7**: $V_1$ and $V_2$ when measured on an oscilloscope when a logical 0 is stored.

Now the node at $V_1$ is slowing discharging when the anode going to zero volts, instead of remaining stable as we previously saw. What is occurring here is related to the *parasitic* capacitances that both diodes and FETs exhibit in a more-accurate model. Figure 7 this effect is shown in a circuit diagram. These “ghost” components are not physically inserted into the breadboard, they are intrinsic to the device and exist as part of the improved model regardless of what is inserted into the circuit. Because capacitors retain charge, the parasitic capacitance will slowly discharge through parasitic resistance until recharged by the square wave.

This may influence the logical 0 value you see depending on the refresh frequency and size of the capacitors you use. The key to ensure the correct behavior of the DRAM circuit is that this logical 0 voltage remain below the $V_{p,th}$ of our inverters.

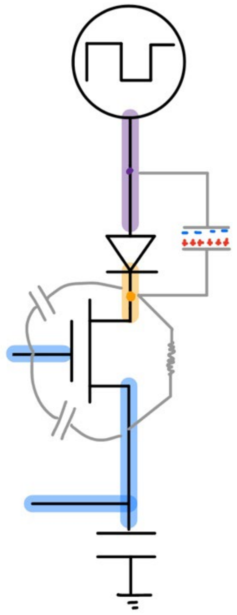
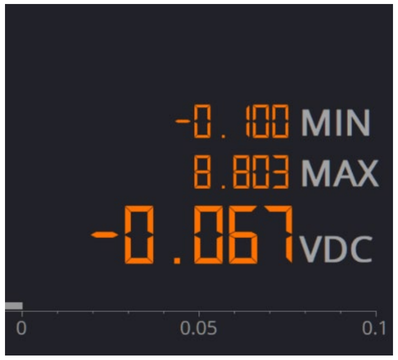


**Figure 8**: Parasitic model of the refresh path with logical 0 stored.



-0.100 MIN
8.803 MAX
-0.067 VDC

**Figure 8:** $V_C$ when storing a logical 0.

Notes:

Notes:

Great, we now have a memory circuit that is able to retain itself in the face of leakage and that we know functions, we should always be be able to read the current value of the bit that is stored by looking at the output of the 2nd inverter. However this is not the entire story, we still need to be able to write to our capcitor to change its state. To do this, we will start by added a 2nd capactor at the output of our first inverter. Given the nature of inverters, this capcitor should always store the opposite bit value of our initial capcitor.

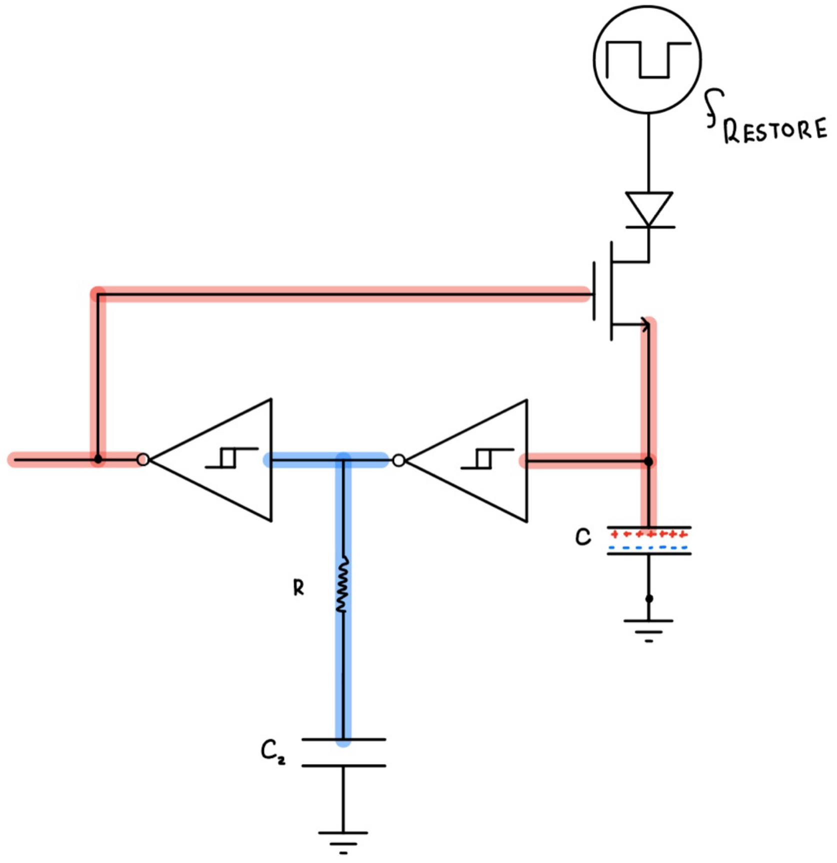


**Figure 9**: DRAM memory with inverted output stored in capacitor.

Notes:

Like our original capacitor memory, we do not want to drive any downstream circuitry with the capacitor's limited charge. So, we will again use buffers.

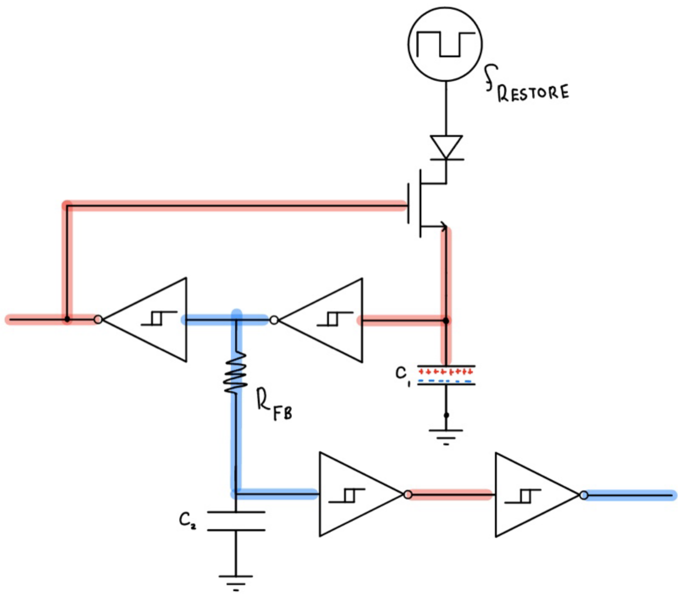


**Figure 10**: DRAM memory with inverted output stored in capacitor. (Feedback flip-flop).

**Question 2:** Since recharging the capacitor is a key component to ensure the correct function of the DRAM, why would we not want a very large capacitor to store our value? HINT: Think about how the RC time constant affects the read/write speeds.

We could now connect the output of our new capacitor's buffers to our first capacitor behind an ideal switch. When this switch was toggled, our capacitor would change discharge all it stored charged into the connection to ground resulting in a change of state.

Notes:

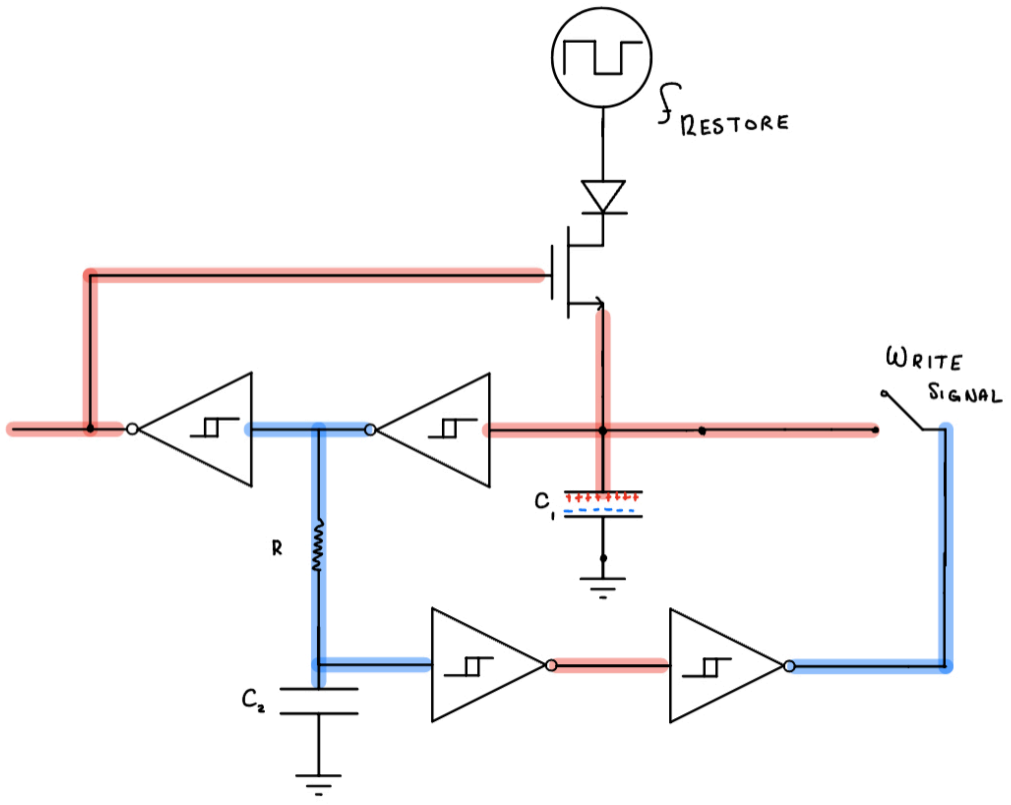


**Figure 11**: DRAM memory with ideal write switch.

Notes:

Unfortunately, ideal switches do not exist, so our switch will be made of transistors. However, using a single transistor presents a problem: the internal PN body junction of the transistor forms a reverse diode (again something we include in an improved model of the nMOS transistor) that allows current to flow from source to drain. To counter act this we must use *two* transistors, where these diodes counteract one another. By facing different directions, the diodes do not allow current flow. This is shown below. Remember, these diodes are parasitic and are not actually inserted into the circuit…they are just how the nMOS behaves in a more-accurate circuit model.

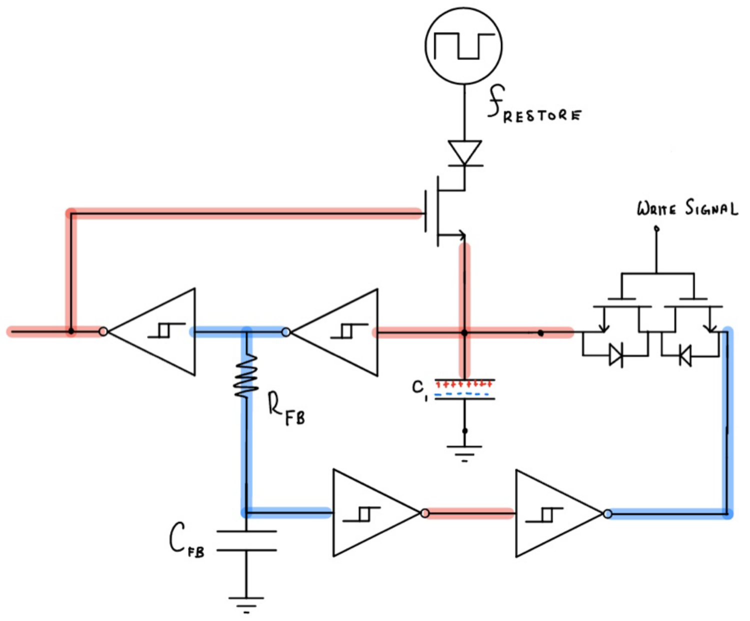


**Figure 12**: Practical DRAM memory with real write switch (a *write* transistor).

**Question 3:** What would occur if we held down the write signal? Recall your work on square wave oscillators. As a follow up, what is the minimum amount of time needed between state transitions in this circuit using the write signal, leave your answer in terms of variables present in the diagram. **HINT**: Which capacitor is limiting the transition?

Notes:

Let's revisit our block diagram from before. In real memory circuits our capacitive memory can be made of many capacitors in parallel, each storing 1 bit. However, the circuity needed to refresh all those capacitors does not need to be repeated, apart from the increase in the number of parallel wires. This regular structure allows us to produce very large memories very quickly, once we understand the basic principle behind build a 1-bit memory.

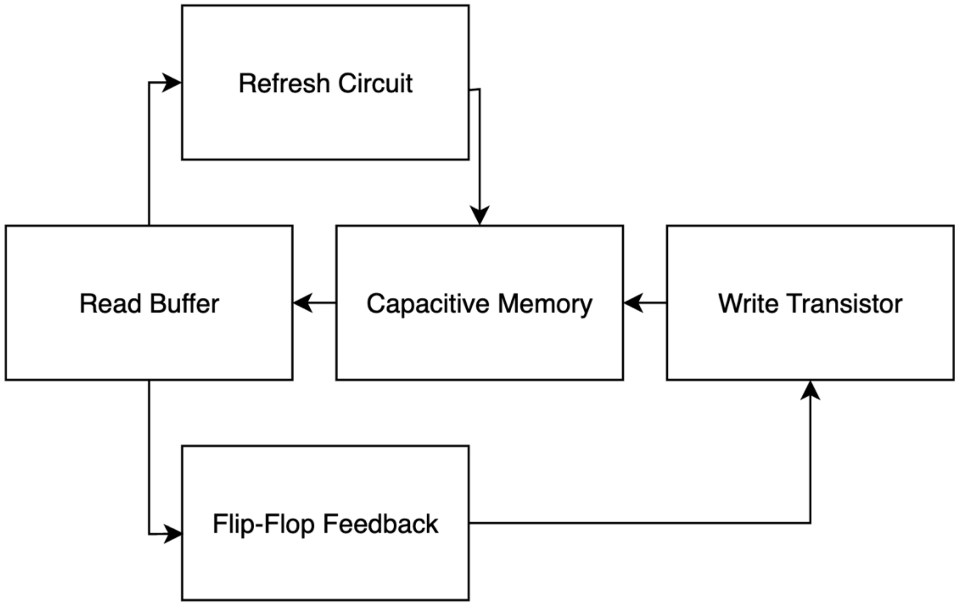


**Figure 13**: Practical DRAM memory.

Notes:

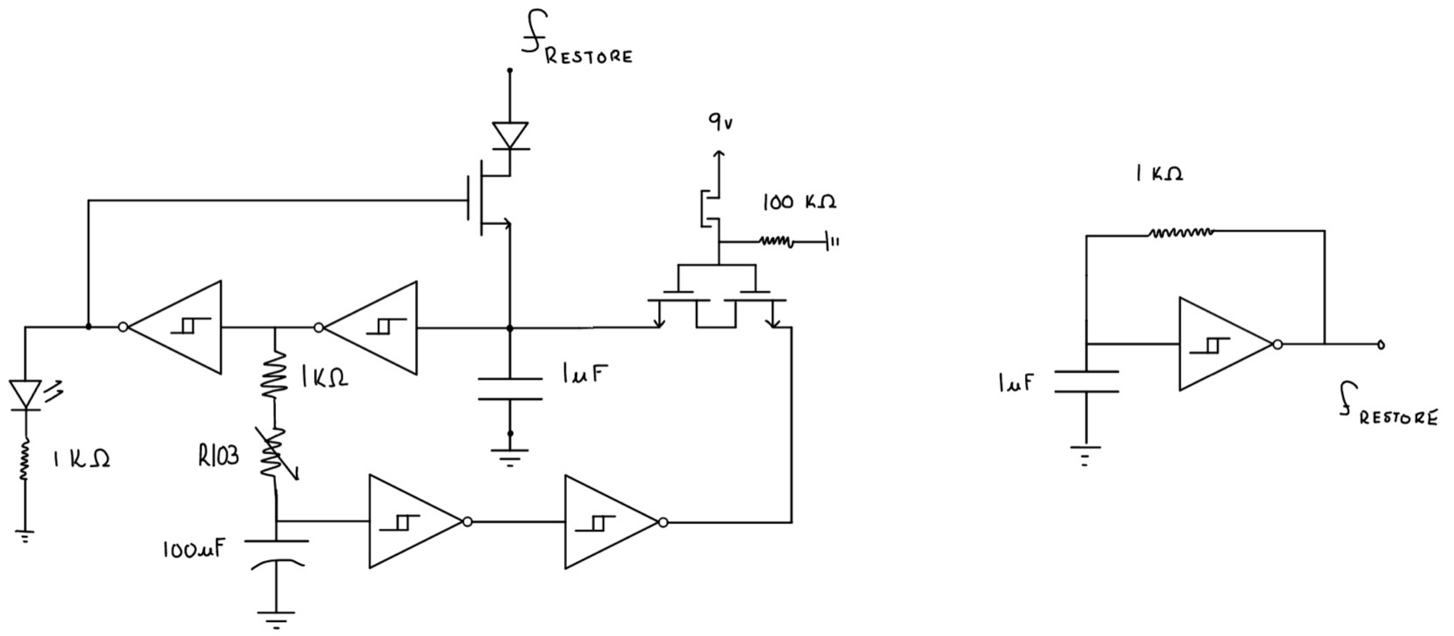


**Figure 14**: A complete memory circuit with read/write capability.

**Question 4:** Build the completed memory circuit and record a video of the oscilloscope as you toggle the charging and discharging of the $100\ uF$ capacitor.

**Challenging Circuit: This circuit build is difficult and can require the extensive use of an oscilloscope to build correctly. Please exercise all the good circuit building practices you have learned in ECE110!**