Computer Architecture

Week 5: Memory

Fenerbahçe Üniversitesi

Professor & TAs

Prof: Dr. Vecdi Emre Levent

Office: 311

Email: emre.levent@fbu.edu.tr

Assistant: Arş. Gör. Uğur Özbalkan

Office: 311

Email: ugur.ozbalkan@fbu.edu.tr

Course Plan

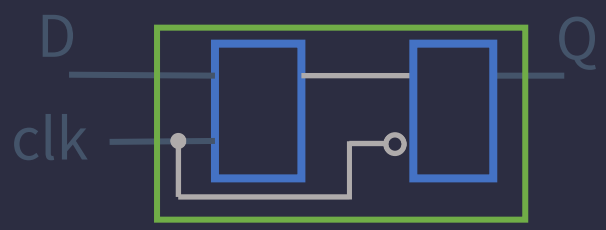
- Memory

Goals for today

Memory

- CPU: Register Files (i.e. Memory w/in the CPU)
- Scaling Memory: Tri-state devices
- Cache: SRAM (Static RAM—random access memory)
- Memory: DRAM (Dynamic RAM)

Last time: How do we store one bit

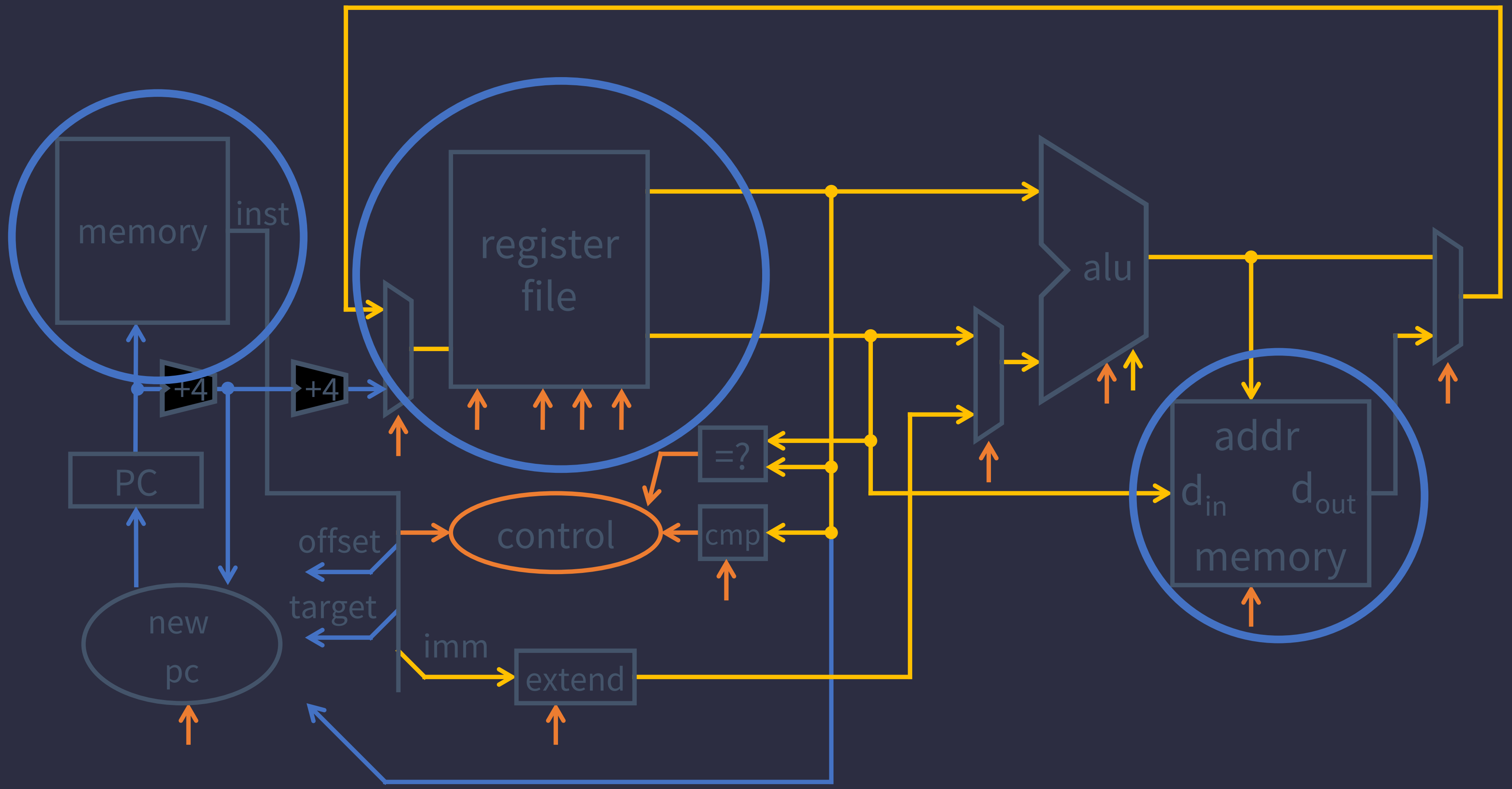


D Flip Flop stores 1 bit

Goal for today

How do we store results from ALU computations?

Big Picture: Building a Processor



A Single cycle processor

Goal for today

How do we store results from ALU computations?

How do we use stored results in subsequent operations?

Register File

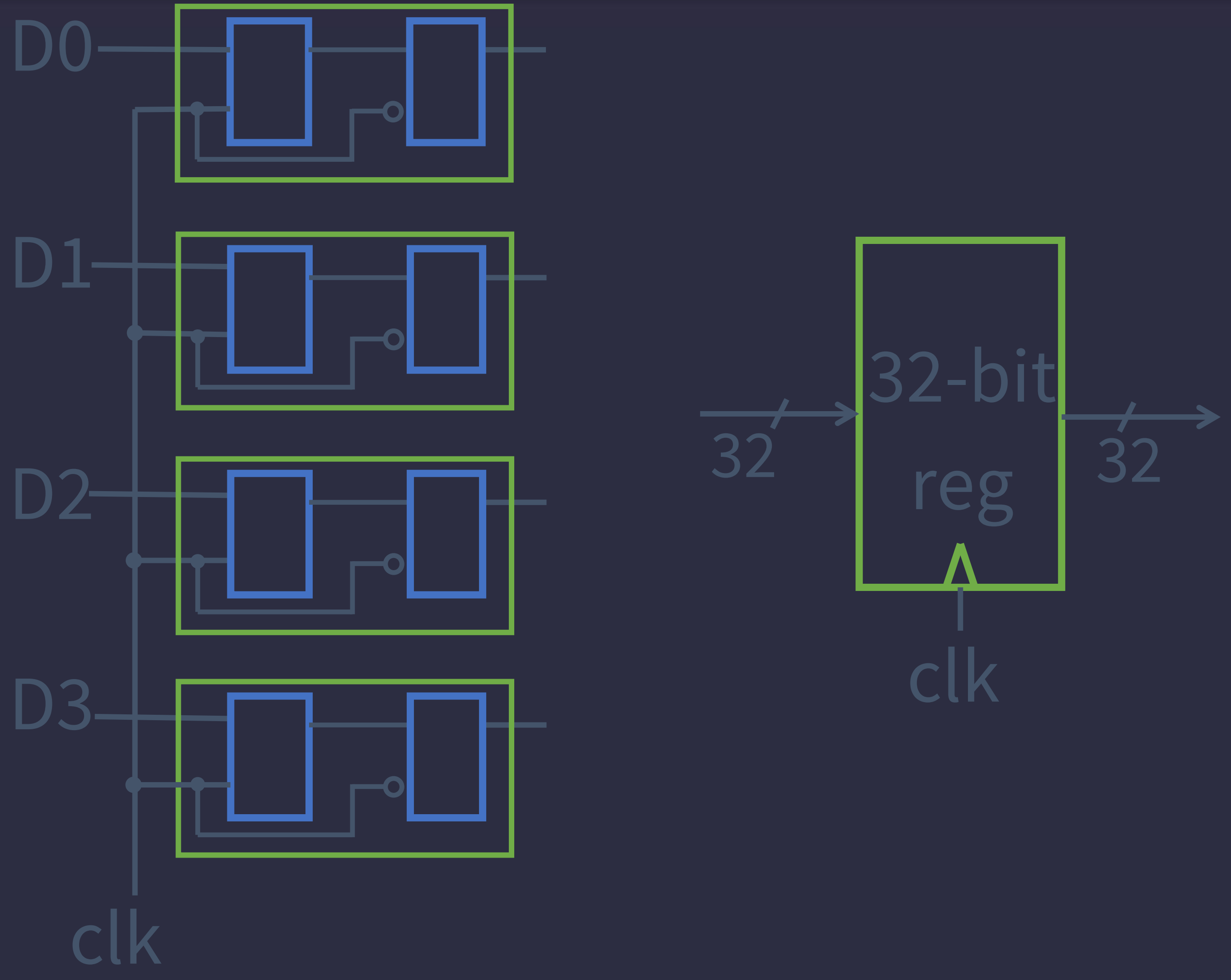
How does a Register File work? How do we design it?

Register File

- N read/write registers
- Indexed by register number

Register File

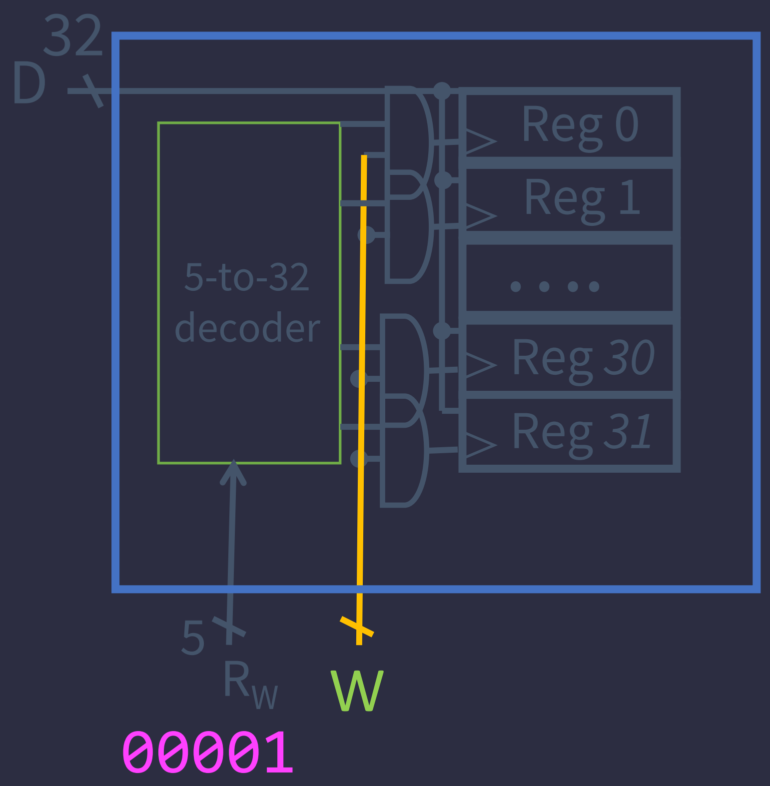
- D flip-flops in parallel
- shared clock
- extra clocked inputs: write_enable, reset, …



Register File

- N read/write registers
- Indexed by register number

```
Addi x1, x0, 10
```



How to write to ***one*** register in the register file?

- Need a decoder

Register File

Register File

- N read/write registers
- Indexed by register number

Implementation:

- D flip flops to store bits
- Decoder for each write port
- Mux for each read port

Tradeoffs

Register File tradeoffs

+ Very fast (a few gate delays for both read and write)

+ Adding extra ports is straightforward

– Doesn't scale

e.g. 32Mb register file with 32 bit registers

Need 32x 1M-to-1 multiplexor

and 32x 20-to-1M decoder

How many logic gates/transistors?

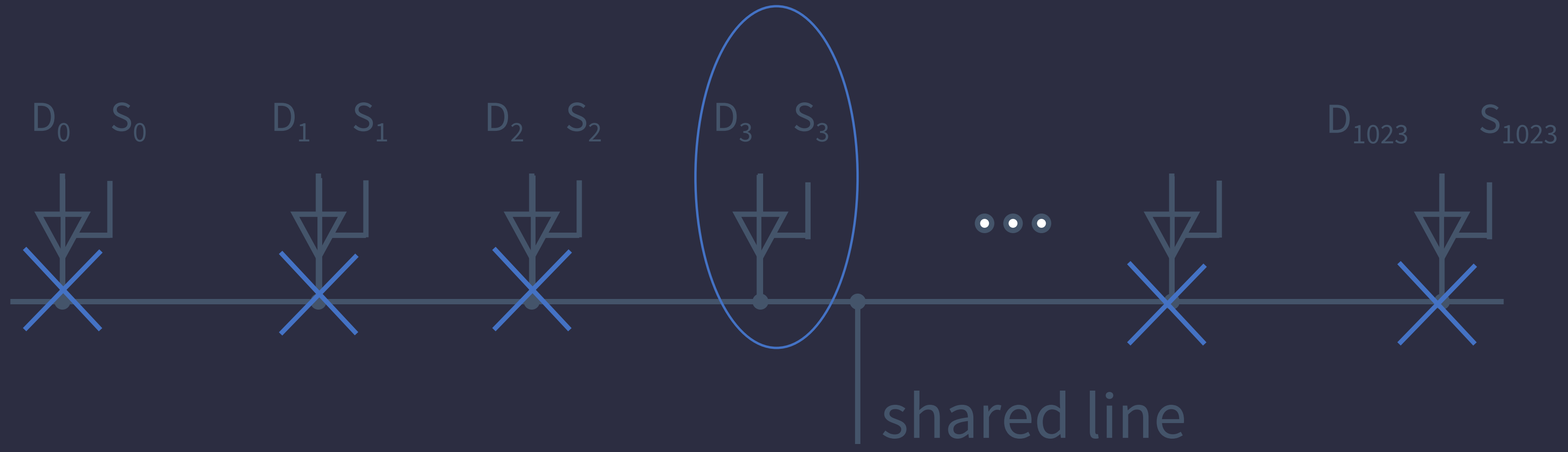
Next Goal

How do we scale/build larger memories?

Building Large Memories

Need a shared bus (or shared bit line)

- Many FlipFlops/outputs/etc. connected to single wire
- Only one output *drives* the bus at a time



- How do we build such a device?

Tri-State Devices

Tri-State Buffers

- If enabled (E=1), then Q = D
- Otherwise, Q is not connected (z = high impedance)

E	D	Q
0	0	z
0	1	z
1	0	0
1	1	1

Tri-State Buffers

- If enabled (E=1), then Q = D
- Otherwise, Q is not connected (z = high impedance)

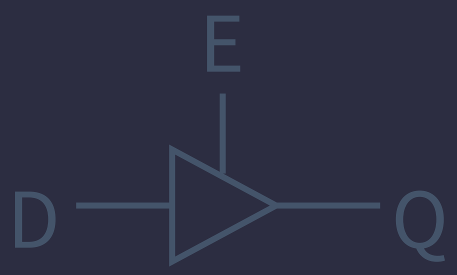
E	D	Q
0	0	z
0	1	z
1	0	0
1	1	1

Tri-State Devices

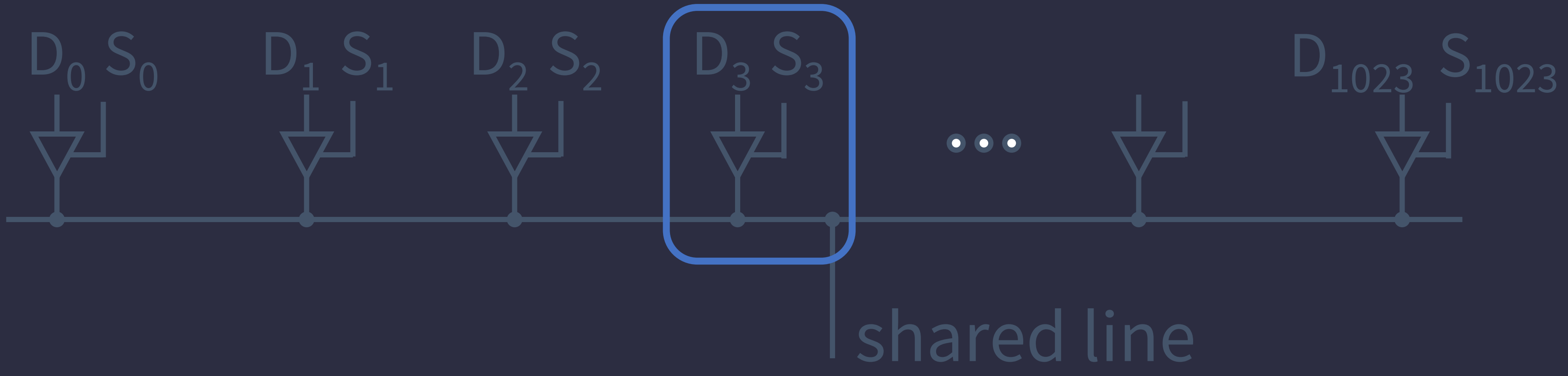
Tri-State Buffers

- If enabled (E=1), then Q = D
- Otherwise, Q is not connected (z = high impedance)

E	D	Q
0	0	z
0	1	z
1	0	0
1	1	1



Shared Bus



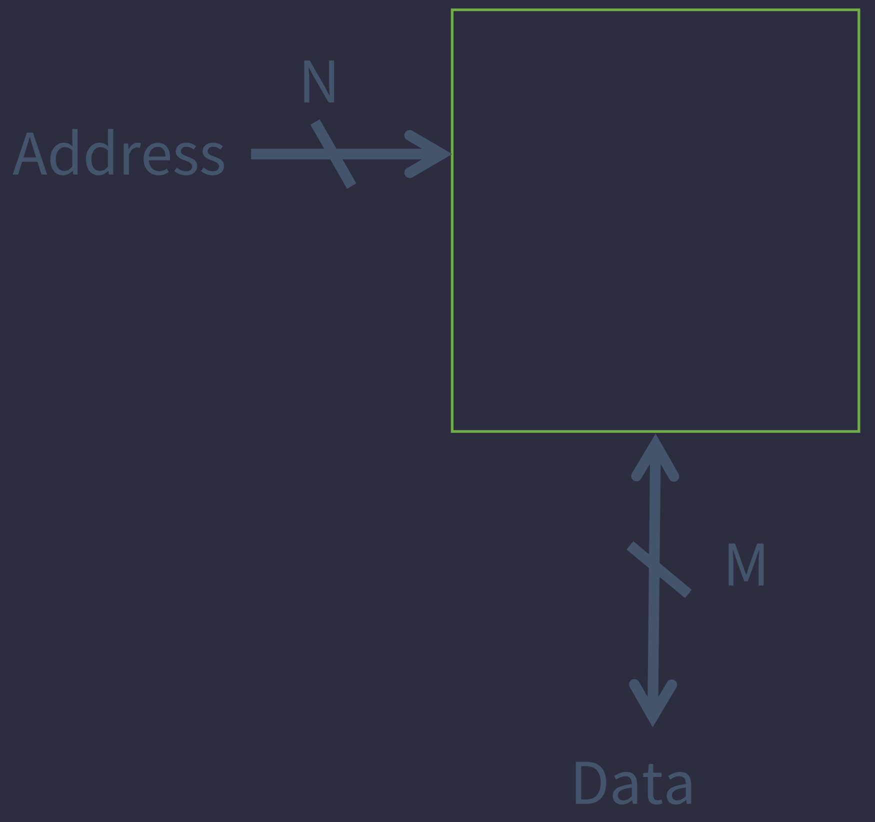
Next Goal

How do we build large memories?

Use similar designs as Tri-state Buffers to connect multiple registers to output line. Only one register will drive output line.

Memory

- Storage Cells + bus
- Inputs: Address, Data (for writes)
- Outputs: Data (for reads)
- Also need R/W signal (not shown)



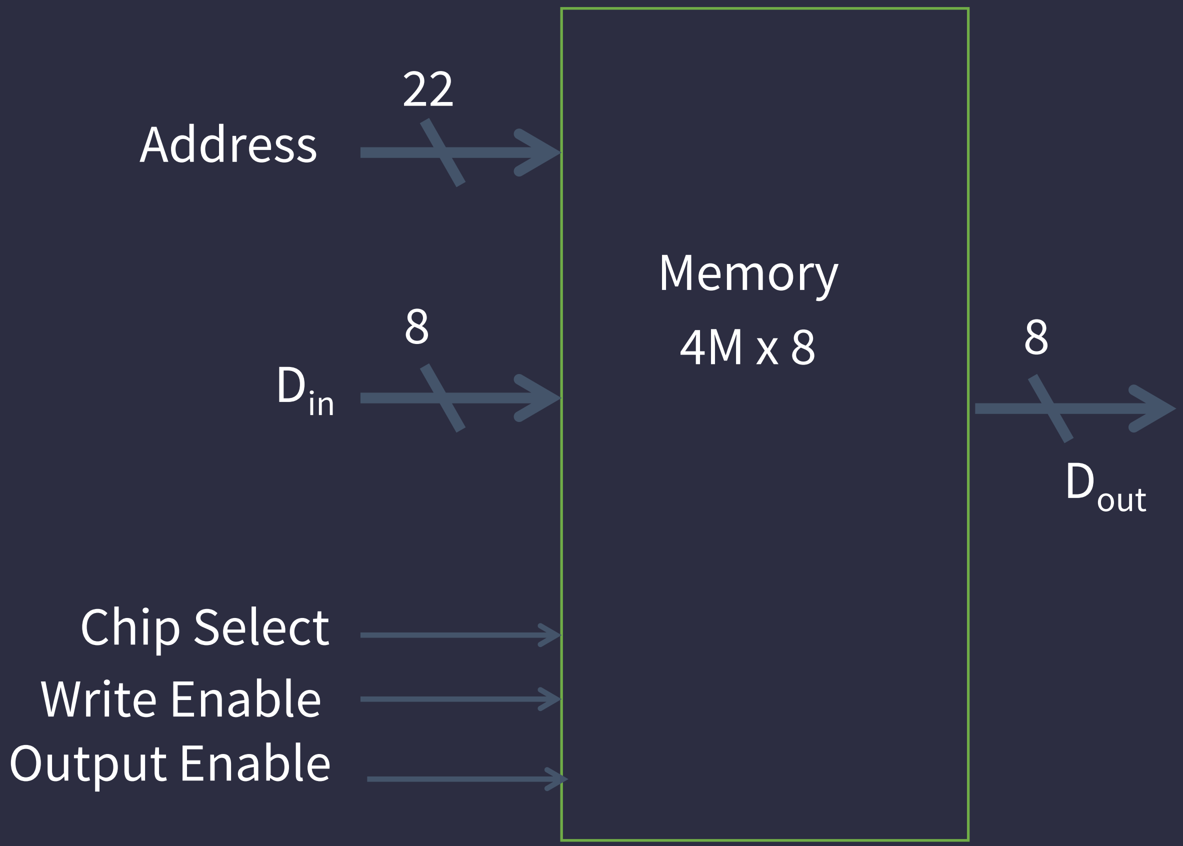
- N address bits → 2^N words total
- M data bits → each word M bits

Memory

- Storage Cells + bus
- Decoder selects a word line
- R/W selector determines access type
- Word line is then coupled to the data lines

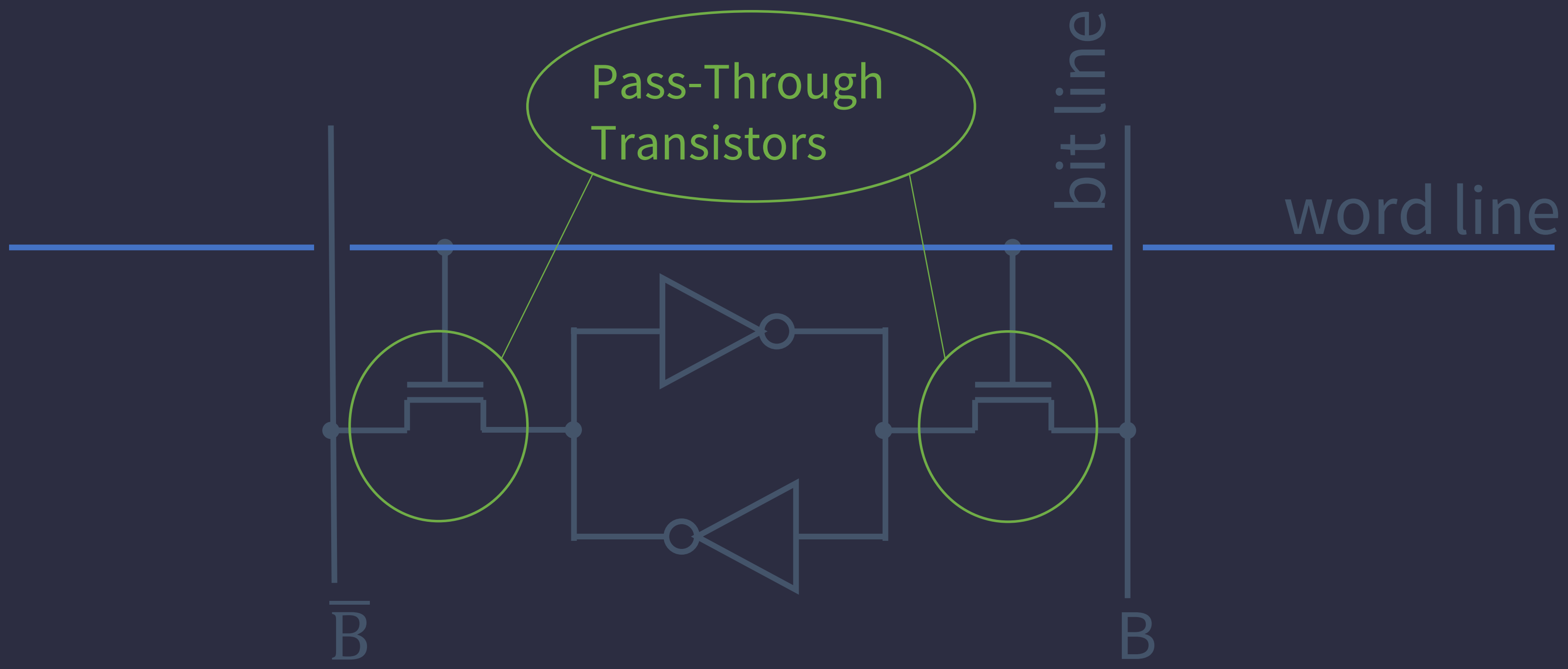
Memory

- Storage Cells + bus
- Decoder selects a word line
- R/W selector determines access type
- Word line is then coupled to the data lines



SRAM Cell

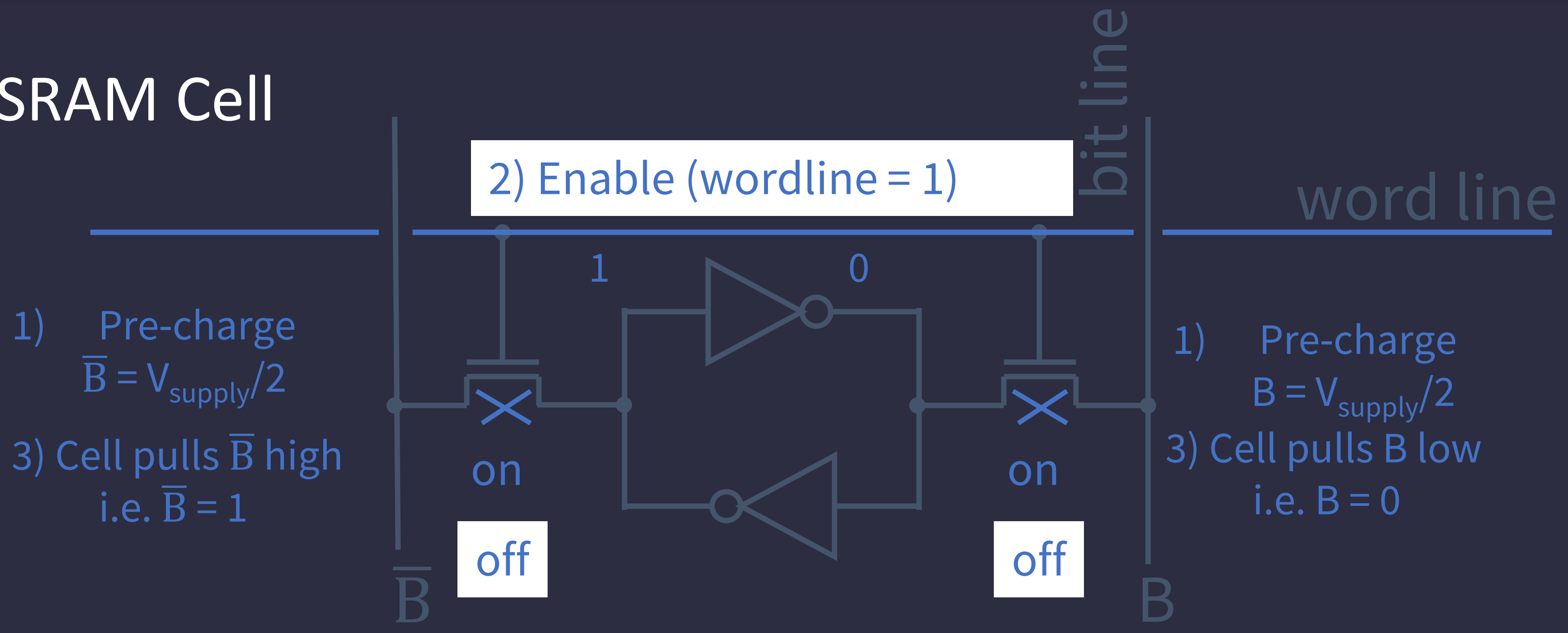
Typical SRAM Cell



Each cell stores one bit, and requires 4 – 8 transistors (6 is typical)

SRAM Cell

Typical SRAM Cell



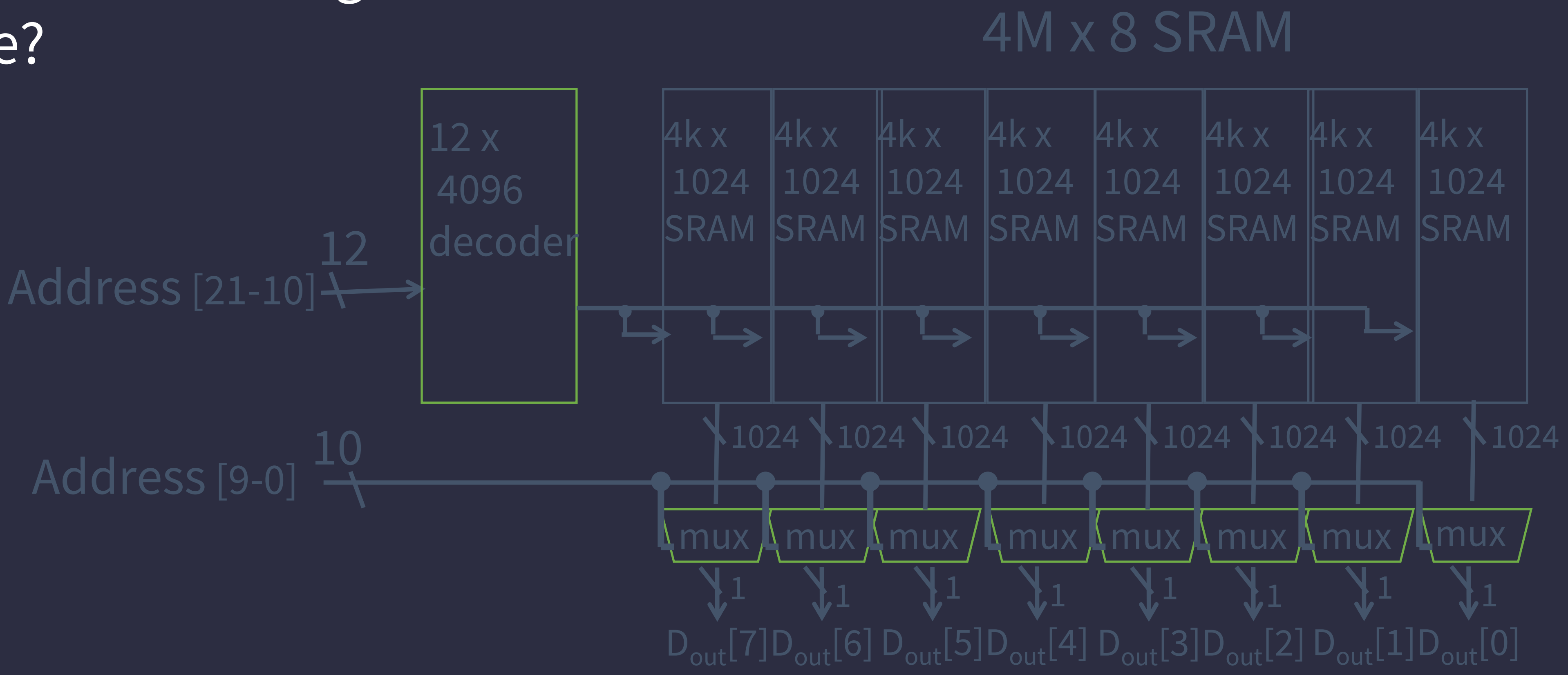
Each cell stores one bit, and requires 4 – 8 transistors (6 is typical)

Read:

- pre-charge B and $\overline{B}$ to $V_{supply}/2$
- pull word line high
- cell pulls B or $\overline{B}$ low, sense amp detects voltage difference

SRAM

E.g. How do we design a ***4M x 8*** SRAM Module?



SRAM Summary

SRAM

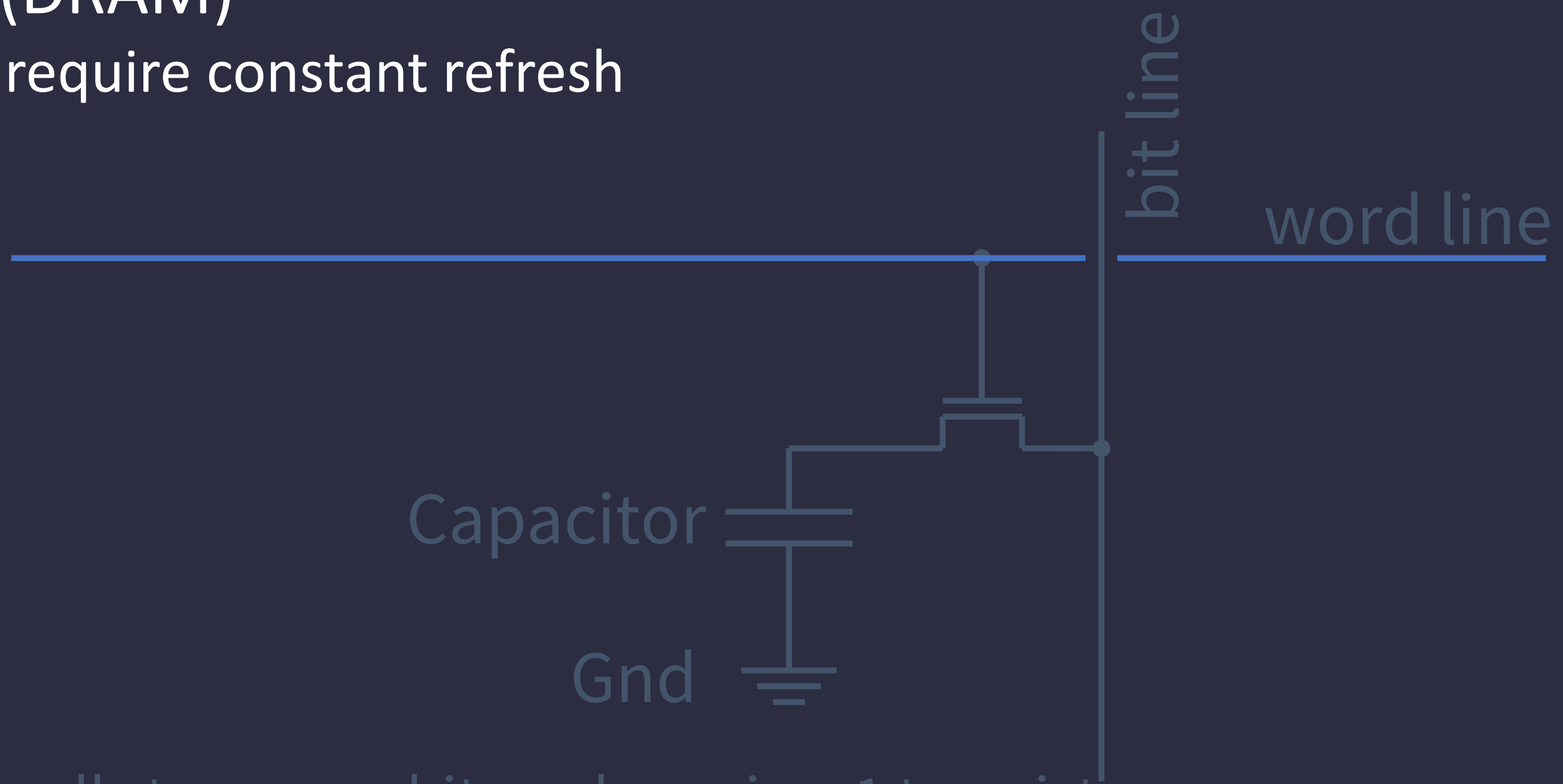
- A few transistors (~6) per cell
- Used for working memory (caches)

- But for even higher density…

Dynamic RAM: DRAM

Dynamic-RAM (DRAM)

- Data values require constant refresh

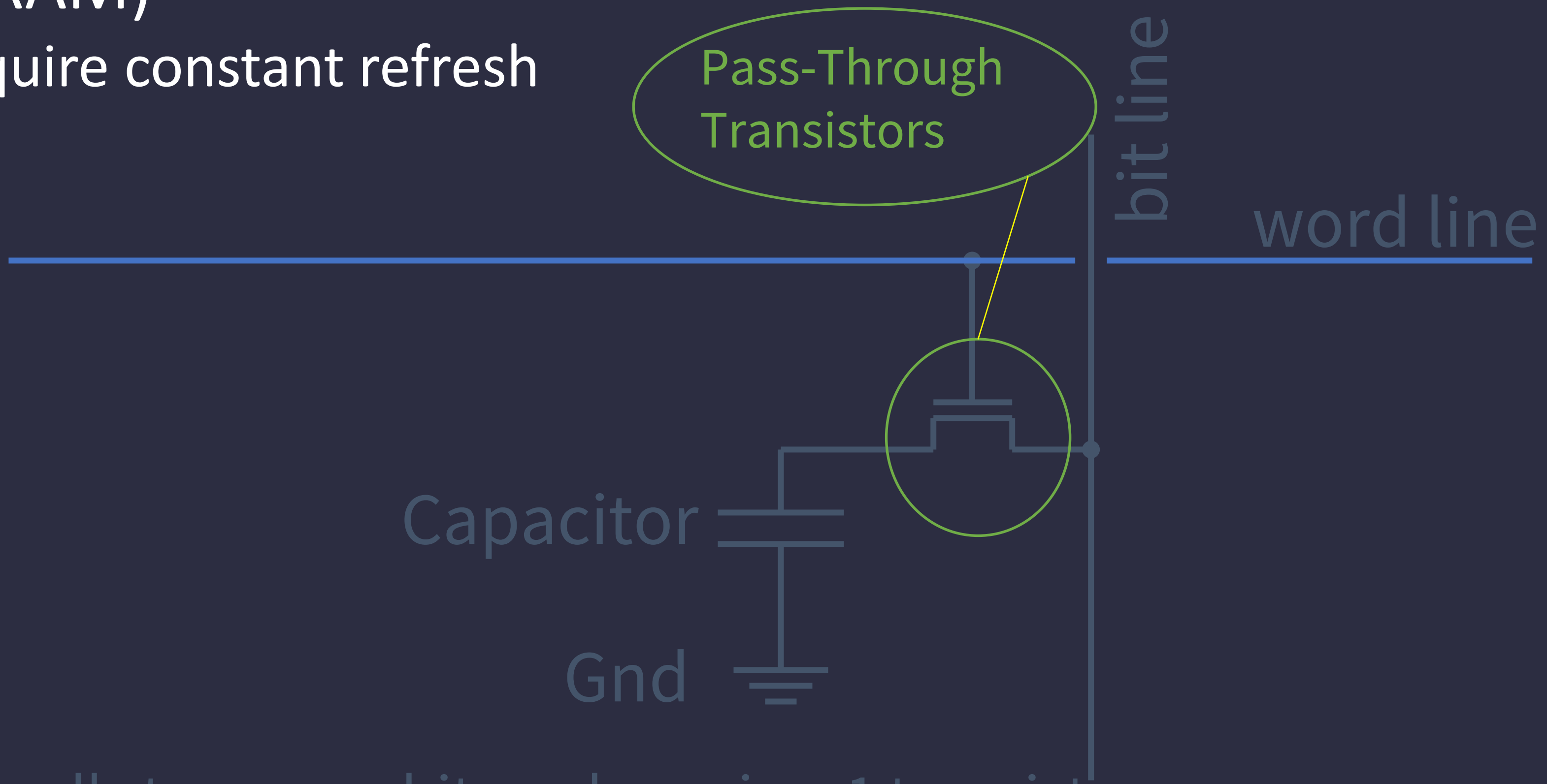


Each cell stores one bit, and requires 1 transistors

Dynamic RAM: DRAM

Dynamic-RAM (DRAM)

- Data values require constant refresh



Each cell stores one bit, and requires 1 transistors

DRAM vs. SRAM

Single transistor vs. many gates

- Denser, cheaper ($30/1GB vs. $30/2MB)
- But more complicated, and has analog sensing

Also needs refresh

- Read and write back…
- …every few milliseconds
- Organized in 2D grid, so can do rows at a time
- Chip can do refresh internally

Hence… slower and energy inefficient

Memory

Register File tradeoffs

+ Very fast (a few gate delays for both read and write)
+ Adding extra ports is straightforward
– Expensive, doesn't scale
– Volatile

Volatile Memory alternatives: SRAM, DRAM, …

– Slower
+ Cheaper, and scales well
– Volatile

Non-Volatile Memory (NV-RAM): Flash, EEPROM, …

+ Scales well
– Limited lifetime; degrades after 100000 to 1M writes